# Truthful Assignment without Money

Shaddin Dughmi*
Stanford University
shaddin@cs.stanford.edu

Arpita Ghosh
Yahoo! Research
arpita@yahoo-inc.com

April 16, 2011

**Abstract**

We study the design of truthful mechanisms that do not use payments for the generalized assignment problem (GAP) and its variants. An instance of the GAP consists of a bipartite graph with jobs on one side and machines on the other. Machines have capacities and edges have values and sizes; the goal is to construct a welfare maximizing feasible assignment. In our model of private valuations, motivated by impossibility results, the value and sizes on all job-machine pairs are public information; however, whether an edge *exists* or not in the bipartite graph is a job's private information. That is, the selfish agents in our model are the jobs, and their private information is their edge set. We want to design mechanisms that are truthful without money (henceforth *strategyproof*), and produce assignments whose welfare is a good approximation to the optimal omniscient welfare.

We study several variants of the GAP starting with matching. For the unweighted version, we give an optimal strategyproof mechanism. For maximum weight bipartite matching, we show that no strategyproof mechanism, deterministic or randomized, can be optimal, and present a 2-approximate strategyproof mechanism along with a matching lowerbound. Next we study knapsack-like problems, which, unlike matching, are NP-hard. For these problems, we develop a general LP-based technique that extends the ideas of Lavi and Swamy [14] to reduce designing a truthful approximate mechanism without money to designing such a mechanism for the *fractional* version of the problem. We design strategyproof approximate mechanisms for the fractional relaxations of multiple knapsack, size-invariant GAP, and value-invariant GAP, and use this technique to obtain, respectively, 2, 4 and 4-approximate strategyproof mechanisms for these problems. We then design an $O(\log n)$-approximate strategyproof mechanism for the GAP by reducing, with logarithmic loss in the approximation, to our solution for the value-invariant GAP. Our technique may be of independent interest for designing truthful mechanisms without money for other LP-based problems.

*Supported by a Yahoo! Research internship, a Siebel Foundation Scholarship, and NSF grant CCF-0448664.

# 1 Introduction

The design of truthful mechanisms, where selfish utility maximizing agents have no incentive to lie about their true preferences, has been studied in innumerable settings. The vast majority of these mechanisms, however, assume the existence of money— a carefully designed payment scheme incentivizes agents to report their preferences truthfully. However, there are settings where monetary transfers are not feasible, either because of ethical or legal issues [16], or because of practical issues with enforcing and collecting payments [17]. This observation has led to a growing literature on designing mechanisms that incentivize agents to report their true preferences *without* using payments: for convenience, we refer to such mechanisms as *strategyproof*.

In this paper, we focus on the design of strategyproof mechanisms for *assignment* problems. An instance of an assignment problem consists of a bipartite graph with items, or jobs, on one side, and bins, or machines, on the other; associated with each bin is a capacity, and with each edge a value and a size. A feasible assignment is a (partial) mapping from items to bins, where no bin's capacity is exceeded by the sizes of the items assigned to it. The goal is to compute a feasible assignment that maximizes *welfare*: the sum of the values of the jobs for machines they are assigned to.

The most general version of this problem, where both the size and value of a job can differ for different machines, is referred to as the generalized assignment problem (GAP). A number of well-known algorithmic assignment problems are special cases of the GAP. For instance, the problem with just one bin is the knapsack problem, and the problem with unit-sized items and bins is the maximum weight bipartite matching problem. Assignment problems are ubiquitous, and have been extensively studied due to their vast applicability, both from an algorithmic and mechanism design perspective. However, studying these problems in a setting *without money*, as we do, adds additional difficulties, since properties like monotonicity and weak monotonicity (see [15, 3, 13, 16]) no longer suffice for truthfulness. Moreover, the popular VCG mechanism, as well as the maximal in range paradigm, require payments for truthfulness.

There are a number of settings where part of the input to an assignment problem is held by selfish agents, and the problem must be solved without the use of money. Unfortunately, as we discuss in § 2.3, not much can be done if jobs hold their real-number values for the various machines private; this is consistent with many impossibility results from social choice theory. In this paper, we therefore focus on a restricted, yet natural, setting that admits interesting results— for each pair $(i, j)$, it is public knowledge that the value of job $i$ for machine $j$ is either $v_{ij}$ or 0, but job $i$ holds private which of those is the case for each $j$. This models situations where the private data encodes a *compatibility relation* between jobs and machines; the public $v_{ij}$ values arise in situations where the value derived from an assignment materializes over a public channel (for instance via a verifiable financial transaction). As a result, a job cannot hide its true value for any machine it anticipates being assigned to, although it can misreport a non-zero value as 0, a lie that will not be discovered since the job will not be allocated to this machine. We will see that, if the mechanism is not chosen carefully, such strategic manipulations can be beneficial to a selfish agent even in very simple instances of the GAP. We also note that this is a *multi-parameter* problem, where each job holds one bit of private information for each of the bins.

There are several natural settings that correspond to our model of private values. Suppose, for instance, a group of people are to split up a collection of tasks. Each task requires different skills, and how well each person can perform a task is public knowledge. Each task also has different time and location constraints, and whether or not the constraint is feasible for a person is only known privately to her. This problem is an instance of weighted bipartite matching; while

finding an optimal solution is computationally easy, we are interested in algorithms that also ensure strategyproofness. In another example, consider a resource allocation problem, such as scheduling jobs on a collection of non-identical machines. The value of running a job on a machine, as well as the time it takes, is public knowledge. However, each job requires specific hardware and software that is available on only some of the machines. Moreover, only the owner of the job knows which machines are compatible with the job. Here, the algorithm used to assign jobs to machines must ensure that the jobs do not have an incentive to lie about which machines are compatible.

As in [17], we are interested in strategyproof mechanisms that achieve a good *approximation* to the welfare of the optimal omniscient solution. The need for approximations arises for two reasons in our work: first, the GAP is NP-hard, as are several of its special cases; *i.e.*, approximation is necessitated by computational intractability. The second reason is much more interesting— unlike in settings with payments, solving the allocation problem optimally does not necessarily lead to a truthful mechanism, and we need to sacrifice approximation in order to obtain truthfulness. We will see both factors playing a role as we seek strategyproof mechanisms that are good approximations to the various special cases of GAP. For example, we will see that no strategy proof mechanism for maximum-weight matching can be optimal, and we must sacrifice an approximation factor of 2 for truthfulness. In contrast, a (non-polynomial time) algorithm that returns an optimal solution to the multiple knapsack problem while breaking ties consistently is strategyproof; however, since we are interested in polynomial-time strategyproof algorithms, we must resort to an approximately optimal mechanism that is both truthful and polynomial-time implementable.

Why would an agent benefit from lying in assignment problems when there are no payments? Consider, for instance, the weighted bipartite matching problem (§3.2). Consider the following instance: job $a_1$ has edges with weights $1+\epsilon$ and 1 to machines $b_1$ and $b_2$, and job $a_2$ has an edge to $b_1$ with weight 1. An algorithm that simply chooses the maximum weight matching according to the reports incentivizes job $a_1$ to simply claim that the second edge does not exist: in the first case, the assignment chosen is $(a_1, b_2), (a_2, b_1)$, whereas in the second case, the assignment chosen is $(a_1, b_1)$, which suits $a_1$ better, with value $1 + \epsilon$. This example makes it clear that that the optimal (and obvious) algorithms are not necessarily truthful— not surprisingly, a carefully designed algorithm is essential to ensure that no agent has an incentive to lie, exactly as in mechanism design with money.

### 1.1 Our Results

We study the design of approximation mechanisms that are truthful without money for several variants of the GAP. We begin in § 3 with *matching*, which can be solved optimally in polynomial time from a purely computational perspective. We show that for the maximum matching problem, where all edge values are equal, simply returning the optimal solution while breaking ties consistently leads to a strategyproof mechanism. However, when a job's value depends on the machine, as in weighted bipartite matching, no deterministic strategyproof mechanism can achieve an approximation better than 2; we provide such a mechanism.

Next, we examine knapsack-like variants of the GAP. Instead of specially tailored combinatorial algorithms for each variant, we extend the techniques in [14] to reduce designing a truthful mechanism without money to designing such a mechanism for the *fractional* version of the problem: if the strategyproof mechanism for the fractional version yields an $\alpha$ approximation to the optimal fractional solution, and the corresponding LP has integrality gap $\beta$, we derive a strategyproof randomized mechanism for the original problem with approximation ratio $\alpha \cdot \beta$. This technique applies

to a large class of packing problems, and may of independent interest.

The GAP has integrality gap 2, so a fractional strategyproof mechanism with approximation ratio $\alpha$ yields a $2\alpha$ strategyproof (in expectation) mechanism for each of the knapsack-like variants of the GAP that we study. In § 4.2, we show, using network flows, that solving the fractional version of the multiple knapsack problem (MKP) optimally, while breaking ties consistently independent of the reported edges, gives an optimal strategyproof (fractional) mechanism. For size-invariant GAP (SIGAP), there is no optimal truthful (fractional) mechanism without money— in §4.3, we design a strategyproof, 2-approximate greedy algorithm for fractional SIGAP. Using our extension of [14] gives, respectively, 2 and 4-approximate strategyproof mechanisms for MKP and SIGAP. In §4.4, we sketch the construction of a 4-approximate strategyproof mechanism for value-invariant GAP (VIGAP), as well as a $O(\log n)$-approximate strategyproof mechanism for the GAP.

We point out that without the polynomial time restriction, there exist optimal strategyproof mechanisms for all variants of GAP where a node has the same value for each of its neighbors. That is, for maximum matching, MKP and VIGAP, simply solving the problem optimally, while breaking ties consistently independent of the private values (edges), leads to a truthful-without-money mechanism. For these problems, it is only computational intractability which causes us to lose an approximation factor. This is in contrast to the variants where a node has different values for different edges such as maximum weight matching and its generalizations. There, as we show in Theorem 3.3, strategyproofness and optimality cannot be achieved simultaneously.

### 1.2 Related Work

Assignment problems have been studied extensively in the algorithms literature. Shmoys and Tardos [18] presented a 2-approximation for a minimization version of the GAP, and Chekuri and Khanna [10] observed that a 2-approximation to the maximization version – the version considered in this paper – is implicit in [18]. Moreover, it was shown in [10] that the multiple knapsack problem – a special case of the GAP – admits a PTAS[1], yet most generalizations of MKP – including the GAP– are APX hard. Fleischer et al [12] obtained a $\frac{e}{e-1}$ approximation for the GAP, and showed that this is optimal for a slight generalization of the GAP. However, Feige and Vondrak [11] then showed that the GAP admits a constant approximation slightly better than $\frac{e}{e-1}$, and this is the best currently known.

A number of results for the mechanism design version of assignment problems are known, although these are all in settings with money. In all of these results, the items hold their values private, and the rest of the instance is public. A 2-approximate truthful-in-expectation mechanism follows immediately from the framework of Lavi and Swamy [14]. Moreover, Briest et al [7] devised a truthful FPTAS for the knapsack problem, as well as a truthful PTAS for VIGAP when the number of bins is fixed. Recently, Azar and Gamzu [5] obtained a truthful 11-approximate mechanism for a variant of MKP, and Chekuri and Gamzu obtained a $2 + \epsilon$ approximation for a variant of VIGAP. We note that all the above mechanisms use money, and moreover the mechanisms in [7, 5, 9] consider an incomparable setting to ours: our model is multi-parameter whereas theirs is single-parameter, but we consider a binary private value for each item and bin as opposed to an arbitrary real number.

[1]However, we note that this PTAS is not applicable to the generalization of MKP that we consider, where assignments are constrained by a bipartite graph over items and bins. It follows from [10, Theorem 3.2] that this is APX-hard.

Mechanisms without money have a rich history in the social choice literature; for a survey, see [16]. Interest in approximate mechanisms without money has been sparked by the recent work of Procaccia and Tennenholtz [17], which introduces the idea of using approximation to enable truthfulness in settings where solving optimally does not admit truthfulness without money. Approximate mechanisms without money have been developed for facility location [17, 1], and selecting influential nodes in a social graph [2]. In very recent work, Ashlagi et al. [4] study strategyproof mechanisms for matching motivated by kidney exchange. While we also study (bipartite) matching as a special case of the GAP, their model is very different from ours: each agent owns a set of vertices in a graph, and reports the existence of vertices to maximize the number of her vertices matched by the mechanism, plus the number that she can match amongst her hidden vertices and the vertices unmatched by the mechanism. Also related is the work of [8], which studies a very general combinatorial assignment problem without money, and designs a mechanism which sacrifices efficiency, as well as weakens the notion of incentive compatibility, to achieve fairness.

# 2 Model

We describe the optimization version of the Generalized Assignment Problem (GAP), as well as its various special cases that we consider, in §2.1. We then review truthfulness in §2.2, and discuss the limitations of truthfulness without money for the GAP in §2.3. These limitations motivate our model of private valuations, which we then introduce in §2.4.

## 2.1 The Generalized Assignment Problem

In the GAP, there are $n$ jobs and $m$ machines. We denote the set of jobs by $[n] = \{1, \ldots, n\}$, and the set of machines by $[m] = \{1, \ldots, m\}$. Machine $j$ has capacity $c_j \in \mathbb{R}^+$. For each job $i$ and machine $j$, we associate a value $v_{ij} \in \mathbb{R}^+$ and a size $s_{ij} \in \mathbb{R}^+$. An *assignment* is a function $x : [n] \to [m] \bigcup \{*\}$ partially mapping jobs to machines, where $*$ indicates that a job is left unassigned. We use $x(i)$ to denote the machine (or $*$) that job $i$ is assigned to. Moreover, the binary variable $x_{ij}$ indicates whether $x(i) = j$. A feasible assignment may allocate to machine $j$ a set of jobs of total size at most $c_j$. The GAP can be written as an integer program with decision variables $\{x_{ij}\}_{ij}$; the LP relaxation obtained by relaxing the constraint $x_{ij} \in \{0, 1\}$ is given below.

$$\begin{array}{lll} \text{maximize} & \sum_{i,j} v_{ij} x_{ij} & \\ \text{subject to} & \sum_{j=1}^{m} x_{ij} \leq 1, & \text{for } i = 1, \ldots, n. \\ & \sum_{i=1}^{n} s_{ij} x_{ij} \leq c_j, & \text{for } j = 1, \ldots, m. \\ & 0 \leq x_{ij} \leq 1 & \end{array} \qquad \text{(GAP LP)}$$

The above LP is known to have an integrality gap of 2, and the rounding can be done in polynomial time [18, 10].

As detailed in §2.4, we consider a setting where the private data is a bipartite graph specifying job-machine compatibility. That is, the private data are not the values $v_{ij}$ or the sizes $s_{ij}$, but rather the *existence* of the edge $(i, j)$. Note that this does not change the GAP from an algorithmic point of view, since one can encode the compatibility relation in the values $\{v_{ij}\}_{ij}$. We define GAP[E] for a bipartite graph $E \subseteq [n] \times [m]$ as the problem of computing the welfare-maximizing assignment using only edges in $E$. The LP relaxation of GAP[E] is given below.

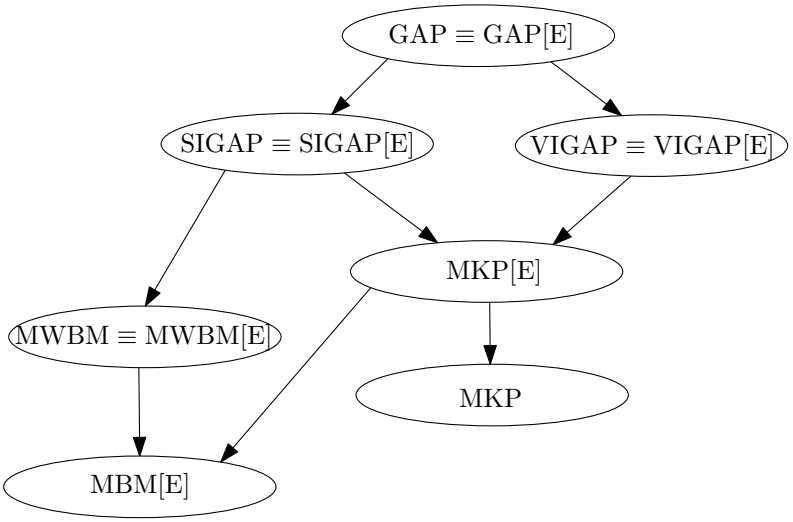


Figure 1: Relationships between Assignment Problems
Arrow indicates the second problem is a special case of the first.
The ≡ symbol indicates computationally equivalent problems.

$$\begin{array}{lll} \text{maximize} & \sum_{i,j} v_{ij} x_{ij} & \\ \text{subject to} & \sum_{j=1}^{m} x_{ij} \leq 1, & \text{for } i = 1, \ldots, n. \\ & \sum_{i=1}^{n} s_{ij} x_{ij} \leq c_j, & \text{for } j = 1, \ldots, m. \\ & 0 \leq x_{ij} \leq 1 & \\ & x_{ij} = 0, & \text{for } (i,j) \notin E. \end{array} \qquad \text{(GAP[E] LP)}$$

We distinguish several variants of the GAP and their respective bipartite-graph versions. The *Size-Invariant Generalized Assignment Problem* (henceforth SIGAP) is the problem where the size of a job $i$ does not depend on the machine – we denote this size by $s_i$. Similarly, the *Value-Invariant Generalized Assignment Problem* (henceforth VIGAP) is the problem where the value of a job $i$ does not depend on the machine – we denote the value by $v_i$. The *Multiple Knapsack Problem* (henceforth MKP) is the problem where neither size nor value depend on the machine. The *knapsack problem* (henceforth KP) is MKP with $m = 1$.

In addition to knapsack-type problems like those dicussed above, the GAP also generalizes bipartite matching problems. The *maximum weight bipartite matching problem* (henceforth MWBM) is the problem where all capacities and sizes are 1. The *maximum bipartite matching problem* (henceforth MBM) is the special case of MWBM where all values are 1. The latter is only interesting when constrained by a graph; that is, when we consider MBM[E] for some $E \subseteq [n] \times [m]$.

When discussing a special case of GAP, say MKP, we refer to the bipartite-graph constrained version as MKP[E]. Moreover, we refer to (GAP LP) and (GAP[E] LP) as (MKP LP) and (MKP[E] LP), respectively. We also use similar notation for other special cases of the GAP.

Figure 1 illustrates the ordering of the various assignment problems by generality. We summarize the known algorithmic results, both upper and lower bounds, in Table 1.

| Problem | Upper Bound | Lower Bound | Integrality Gap |
|---|---|---|---|
| GAP/ GAP[E] | $\frac{e}{e-1} - \delta$ | APX-hard | 2 |
| SIGAP/ SIGAP[E] | $\frac{e}{e-1} - \delta$ | APX-hard | 2 |
| VIGAP/ VIGAP[E] | $\frac{e}{e-1} - \delta$ | APX-hard | 2 |
| MKP[E] | $\frac{e}{e-1} - \delta$ | APX-hard | 2 |
| MKP | PTAS | Strongly NP-hard | 2 |
| MWBM/ MWBM[E] | 1 | 1 | 1 |
| MBM[E] | 1 | 1 | 1 |

Table 1: Computational Upper and Lower Bounds
$\delta$ is a small, positive, fixed real number.
The integrality gap is given for the standard LP relaxation.

### 2.2 Truthfulness

We consider the setting where jobs are selfish agents, and their private types encode information about their value for being assigned on different machines. Other information, such as $n$,$m$, $\{s_{ij}\}_{ij}$ and $\{c_j\}_{j=1}^m$ is considered public[2].

We assume that player $i$ has a type $v_i = \{v_{ij}\}_{j=1}^m$, specifying his value for the different machines. We assume the possible types of player $i$ are restricted to some public set $\mathcal{V}_i \subseteq \mathbb{R}^m$, and use $\mathcal{V}$ to denote $\mathcal{V}_1 \times \ldots \mathcal{V}_n$. For example, when working in the multiple knapsack problem, we require that for each job $i$ there is a real number $v_i$ such that $v_{ij} = v_i$ for all $j$. Moreover, as we will see in §2.3, no interesting results without money are possible if possible values are unbounded. Therefore, our positive results will assume a restricted, discrete set of possible types described in §2.4.

A *mechanism without money* for the GAP is simply an algorithm that takes in all the problem data, public and private, and outputs an assignment of the jobs to the machines. We allow our mechanisms to be randomized. We use $\mathcal{A}(I, v)$ to denote the output of mechanism $\mathcal{A}$ on public data $I$ and private data $v \in \mathcal{V}$. Each mechanism $\mathcal{A}$ and instance of the public data $I$ induces a *social choice rule* : a function $\mathcal{A}(I, *)$ mapping private valuations to assignments.

We now state truthfulness without money generally.

**Definition 2.1.** *Fix a mechanism without money $\mathcal{A}$, let $x$ denote the assignment $\mathcal{A}(I, v)$, and let $x'$ denote the assignment $\mathcal{A}(I, v_{-i} \bigcup v_i')$ when player $i$ changes his report to $v_i'$. Mechanism $\mathcal{A}$ is truthful if and only if the following always holds for for each $I$, $v \in \mathcal{V}$, $i$, and $v_i' \in \mathcal{V}_i$.*

$$\sum_{j=1}^{m} v_{ij} x_{ij} \geq \sum_{j=1}^{m} v_{ij} x'_{ij}. \tag{1}$$

[2]It is conceivable that the job sizes $s_{ij}$ are private information as well. However, if jobs can lie about their sizes, we need to define the utility to a job when it is fractionally assigned— a job can either derive fractional utility from a fractional assignment, or zero utility if it is not fully assigned to a machine. Both models are reasonable; for the first, a nontrivial proof shows that no reasonable approximation can be obtained by any randomized strategyproof mechanism. For the second model , where jobs derive no utility from partial assignments, it turns out that jobs have no incentive to lie about their sizes in any of the algorithms we design.

*Similarly, a randomized mechanism $\mathcal{A}$ is truthful in expectation without money if and only if*

$$\mathbf{E}[\sum_{j=1}^{m} v_{ij}x_{ij}] \geq \mathbf{E}[\sum_{j=1}^{m} v_{ij}x'_{ij}]. \tag{2}$$

In other words, a mechanism $\mathcal{A}$ is truthful if a job never benefits from misreporting its private data. A randomized mechanism $\mathcal{A}$ is truthful in expectation if no (risk-neutral) job has an incentive to misreport its private data.

### 2.3 Limits of Truthfulness without Money

Here, we will justify considering a discrete valuation model by observing that, assuming general valuations, no interesting results are possible. Indeed, we assume a restricted setting: the knapsack problem, with a knapsack of capacity 1, and jobs of size 1. If $\mathcal{V}_i = \mathbb{R}^+$ for each $i$, then it is easy to see that this is equivalent to the classical problem of a single item auction [19]. It is well known that no non-trivial guarantees are possible for a single-item auction if the mechanism is required to be truthful-in-expectation without using money. In fact, it is easy to see that no mechanism can outperform the trivial one which allocates the item (in our case, the entire capacity of the knapsack) uniformly at random, achieving an approximation ratio of $n$.

### 2.4 The Private Graph Valuation Model

Given that no nontrivial upperbounds are possible when players can arbitrarily misrepresent their values, we consider a restricted model of the valuations: one of a discrete nature as is characteristic of many problems for which truthfulness without money is possible. We assume job $i$ has a value of $\delta_{ij}v_{ij}$ for being assigned to machine $j$, where $v_{ij}$ is public and $\delta_{ij} \in \{0, 1\}$ is private. In other words, jobs may not lie about their potential value $v_{ij}$ (that is, $v_{ij}$ are publicly known or *verifiable*), yet they may lie about which machines they are *compatible* with. This compatibility relation is encoded via a bipartite graph on the jobs and machines. Each job's private data is the set of its outgoing edges, i.e. the machines with which it is compatible. As we discuss in §1, this situation arises in many natural settings.

An *instance* of the *GAP on a private bipartite graph* is a pair $(I, E)$, where $I$ is a tuple $(\{v_{ij}\}_{ij}, \{s_{ij}\}_{ij}, \{c_j\}_j)$ of public information, and $E \subseteq [n] \times [m]$ is private information solicited from the jobs. $E$ is a set of *edges* summarizing the compatibility of jobs and machines, where job $i$'s private data is the set $E_i \subseteq E$ of edges in $E$ incident on $i$. A job $i$ receives value $v_{ij}$ from being assigned to machine $j$ only if $(i, j) \in E$, else it receives value 0. Our goal is to maximize welfare via a mechanism that, without using money, incentivizes $i$ to report her set $E_i$ of edges truthfully.

We can now restate truthfulness as it applies to our model. We use $\mathcal{A}(I, E)$ to denote the assignment computed by mechanism $\mathcal{A}$ on instance $(I, E)$. Moreover, for an edge set $E$ we use $E_i \subseteq E$ to denote the set of edges with one endpoint at job $i$, and use $E_{-i}$ to denote $E \setminus E_i$.

**Definition 2.2.** *For a mechanism without money $\mathcal{A}$, let $x$ denote allocation $\mathcal{A}(I, E)$, and let $x'$ denote allocation $\mathcal{A}(I, E_{-i} \bigcup E'_i)$. $\mathcal{A}$ is truthful if and only if the following holds for each $I$, $E$, $i$, and $E'_i$.*

$$\sum_{j:(i,j)\in E_i} v_{ij}x_{ij} \geq \sum_{j:(i,j)\in E_i} v_{ij}x'_{ij} \tag{3}$$

*Similarly, a randomized mechanism $\mathcal{A}$ is truthful in expectation if and only if*

$$\mathbf{E}[\sum_{j:(i,j)\in E_i} v_{ij}x_{ij}] \geq \mathbf{E}[\sum_{j:(i,j)\in E_i} v_{ij}x'_{ij}] \tag{4}$$

Note that the summation on both sides of the inequality is over the set of *true* edges $E_i$: that is, $\mathcal{A}$ is truthful [in expectation] if when a job misreports its incident edges, its [expected] utility from the new assignment $x'$ does not increase on its true edges $E_i$.

# 3 Combinatorial Mechanisms for Matching

We will show that optimally solving the maximum matching problem, with some careful tiebreaking, yields a strategyproof polynomial-time mechanism. For maximum weight matching, we show matching upper and lower bounds of 2 for truthful mechanisms without money, and a constant lowerbound for truthful-in-expectation mechanisms.

## 3.1 Warmup: Maximum Bipartite Matching

We consider the Maximum Bipartite Matching problem, constrained by a private bipartite graph $E$. We observe that simply finding the maximum matching, using consistent tiebreaking, immediately gives a strategyproof mechanism.

**Proposition 3.1.** *Fix a total order $\prec$ on matchings in the complete bipartite graph. For a set of edges $E$, let $M(E)$ denote the set of matchings on edge set $E$. Let $\mathcal{A}$ be the mechanism that, on input $(I, E)$, finds the $\prec$-minimal matching in the set* $\text{argmax}_{x\in M(E)} \sum_{ij} x_{ij}$. *Then $\mathcal{A}$ is truthful.*

*Proof.* Assume for a contradiction that $\mathcal{A}$ is not truthful. Then, there exists $I$ and $E = (E_{-i}, E_i)$ and $E'_i$ violating (3). Let $x = \mathcal{A}(I, E)$ and $x' = \mathcal{A}(I, E')$, where $E' = E_{-i} \bigcup E'_i$. Job $i$ is not matched by an edge in $E_i$ in $x$, yet is matched by an edge in $E_i$ in $x'$. Since $\mathcal{A}$ only uses reported edges, $i$ is not matched at all in $x$, yet is matched by an edge $e \in E_i \bigcap E'_i$ in $x'$. This implies that both $x$ and $x'$ are in $M(E) \bigcap M(E')$. Observe that $\sum_{ij} x_{ij} = \sum_{ij} x'_{ij}$, otherwise either $x$ is not optimal in $M(E)$, or $x'$ is not optimal in $M(E')$, contradicting the definition of $\mathcal{A}$. Therefore, both $x$ and $x'$ are optimal in both $M(E)$ and $M(E')$. Recalling that algorithm $\mathcal{A}$ breaks ties consistently, this yields a contradiction, as needed. □

Therefore, it suffices to define $\prec$ so that the $\prec$-minimal maximum-matching on $E$ can be computed in polynomial time. This gives the following proposition.

**Proposition 3.2.** *There is a polynomial-time, without-money mechanism for maximum bipartite matching that is optimal, and truthful in the private graph model.*

*Proof.* We represent each matching as a binary vector $(x_{11}, x_{12}, \ldots, x_{21}, x_{22}, \ldots, x_{nm})$ in $\{0,1\}^{nm}$, and let $\prec$ be the lexicographic order on these vectors. Then we proceed to find the $\prec$-minimal maximum matching as follows. We compute the size $OPT$ of the maximum matching (this can be done in polynomial time). We then process edges in the order they appear in the vector representation, while maintaining a working set of edges $X$ initialized to $E$. When processing an edge $e$, we check if removing $e$ decreases the size of the maximum matching in $X$ by solving the problem on edges $X \setminus e$. If so we keep $e$ in $X$, else we discard $e$ by setting $X = X \setminus e$. When finished, $X$ is a maximum matching; it is easy to see that $X$ is $\prec$-minimal among all maximum matchings. □

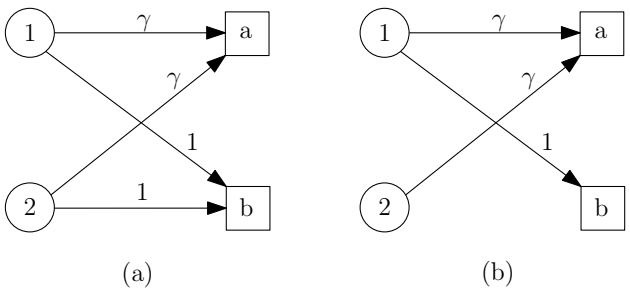


Figure 2: Maximum Weight Matching Lowerbound
Circles represent jobs, and squares represent machines.

### 3.2 Maximum Weight Bipartite Matching

Next, we consider the MWBM problem, constrained by a private bipartite graph $E$. Unlike MBM, we show constant lower bounds on the approximation ratio of truthful and truthful in expectation mechanisms.

**Theorem 3.3.** *No deterministic truthful mechanism without money for MWBM has approximation ratio better than* 2. *Moreover, no truthful-in-expectation randomized mechanism gives better than a* $\frac{2}{2\sqrt{2}-1} \approx 1.0938$ *approximation.*

*Proof.* First, consider a deterministic mechanism $\mathcal{A}$. Assume for a contradiction that $\mathcal{A}$ attains an approximation $\alpha < 2$. Consider Figure 2(a), where $\gamma > 1$. Both jobs 1 and 2 prefer machine $a$ to machine $b$. If we let $\gamma$ be sufficiently close to 1, $\mathcal{A}$ cannot leave either job unassigned. Without loss of generality, we assume job 1 is assigned to $a$ and job 2 is assigned to $b$. Now consider job 2 changing his bid as in Figure 2(b). $\mathcal{A}$ cannot assign job 2 at all under these new bids, as that would violate truthfulness. Therefore, when given the bids in Figure 2(b), the welfare of the solution is at most $\gamma$, whereas the optimal is $\gamma + 1$. Letting $\gamma$ be sufficiently close to 1 gives the contradiction.

Now, we consider an arbitrary truthful-in-expectation mechanism $\mathcal{A}$ on Figure 2(a). At least one of the jobs must be assigned to the preferred machine $a$ with probability no more than 1/2; without loss of generality this is job 2. Job 2 derives value at most $(\gamma + 1)/2$. Now, consider job 2 changing his bid as in Figure 2(b). By truthfulness, now $\mathcal{A}$ can assign job 2 with probability at most $p = (\gamma + 1)/2\gamma$. Therefore, the welfare of the assignment returned on the bids of Figure 2(b) is at most $\gamma + 1 - (1 - p) \cdot 1 = \gamma + p$. However, the optimum is still $\gamma + 1$, so the approximation ratio is at least

$$\frac{\gamma + 1}{\gamma + p} = \frac{\gamma + 1}{\gamma + (\gamma + 1)/2\gamma}$$

Using elementary calculus, we can choose $\gamma$ to maximize this expression and complete the proof. □

Therefore, we cannot hope for better than a constant factor approximation (specifically a PTAS, randomized or deterministic, is not possible). We will show a factor 2 deterministic truthful mechanism, matching Theorem 3.3.

Consider the greedy Algorithm 1. Notice that step (1) *does not depend* on the reported edges $E$.

**Algorithm 1** Mechanism for MWBM on Private Bipartite Graph

1: Order pairs $(i, j) \in [n] \times [m]$ in decreasing order of $v_{ij}$, breaking ties arbitrarily.
2: Let $X = \emptyset$.
3: **for all** $e \in E$ in the order defined above **do**
4: **if** $X \bigcup \{e\}$ is a matching **then**
5: Let $X = X \bigcup \{e\}$
6: **end if**
7: **end for**
8: **return** $X$

**Theorem 3.4.** *Algorithm 1 is a polynomial-time, 2 approximate, no-money truthful mechanism for maximum weight bipartite matching in the private graph model.*

*Proof.* The approximation ratio immediately follows from a standard charging argument against the optimal solution.

For truthfulness, consider a job $i$ misrepresenting his true edges $E_i$ as $E'_i$. Let $E' = E_{-i} \bigcup E'_i$. Let $X$ be the matching returned by the algorithm on reports $E$, and let $X'$ be the matching returned on reports $E'$. If $X = X'$, then $i$ does not improve his value. Assume $X \neq X'$, and let $e' \in E'$ be the first edge in $X' \setminus X$ according to the order of step (1). Since the algorithm processes edges in the bid-independent order of step (1), it is easy to see that $e' \in E' \setminus E = E'_i \setminus E_i$. Thus, $i$ is matched to $e' \neq E_i$, an edge from which he derives no value, when he reports $E'_i$. This completes the proof. □

# 4 LP-Based Mechanisms for Knapsack type Problems

The bipartite matching problems studied in the previous section can be solved in polynomial time; there, the need for approximation is a result purely of the requirement of strategyproofness. We now investigate knapsack-like variants of the GAP — unlike matching, these problems are NP-hard (in fact, they are APX-hard). The mechanisms we design have the following common structure: first, we design a truthful mechanism without money for the *fractional* LP relaxation of the problem. Then, as in [14], we use a randomized procedure to obtain a feasible integral assignment from the fractional solution— this composition leads to a truthful-in-expectation mechanism without money. We introduce this technique for designing truthful mechanisms without money in §4.1, and then use it to design mechanisms without money for knapsack type assignment problems in §4.2-§2.1. Some of our analyses will use notions from network flow theory, which we recap in Appendix B.

## 4.1 A Reduction to Fractional Truthfulness

The construction of Lavi and Swamy allows us to reduce constructing a truthful-in-expectation mechanism (with money) for the integral problem to constructing a truthful mechanism for a fractional version of the problem. We redevelop their construction, in a form convenient for our purposes, in Appendix A. While the truthful mechanism they construct for the fractional *welfare maximization problem* (defined in Appendix A) simply solves the problem optimally and uses VCG payments, we observe that this need not be the case. Indeed, this is crucial for our purposes; for

some of the assignment problems we are interested in, the optimal algorithm for the fractional problem requires non-zero payments for truthfulness. Instead, we sacrifice optimality in the fractional solution to get a truthful fractional mechanism *with zero payments*. Then we use the fractional mechanism to get a scaled down truthful-in-expectation mechanism – also with zero payments – for the combinatorial problem as in Theorem A.5.

By examining the proof of Theorem A.5, we notice that the assumption that the fractional mechanism solves the LP exactly is not used. In fact, an arbitrary truthful mechanism $M_{frac}$ for the fractional problem can be converted to a truthful-in-expectation mechanism $M_{exp}$ for the combinatorial problem. If $M_{frac}$ is a $\beta$-approximation algorithm for the fractional problem, then $M_{exp}$ is an $\alpha \cdot \beta$ approximation algorithm for the combinatorial problem, where $\alpha$ is the integrality gap of the LP relaxation. Moreover, by examing the proof of Theorem A.5, we notice that the payment scheme $p_{exp}$ of mechanism $M_{exp}$ is simply a scaled down copy of the payment scheme $p_{frac}$ of $M_{frac}$. In particular, if $M_{frac}$ is a truthful mechanism without money for the fractional problem, then $M_{exp}$ is a truthful-in-expectation mechanism without money for the combinatorial problem. We sum up these observations in the following Lemma.

**Lemma 4.1.** *Assume the fractional welfare maximization problem over polytope $P$ and valuation class $\mathcal{V}$ (as defined in Appendix A) admits a $\beta$-approximate mechanism that is truthful without money. Moreover, assume $P$ satisfies the conditions of Lemma A.4 with integrality gap $\alpha$. Then there exists an efficient truthful-in-expectation $\alpha \cdot \beta$-approximate mechanism for the welfare maximization problem over $P$ and $\mathcal{V}$ that does not use money.*

In other words, we reduce the problem of designing a truthful-in-expectation mechanism without money to that of designing such a mechanism for its fractional relaxation. This will prove particularly useful, since arguing about truthfulness of a continuous fractional assignment algorithm is more tractable than designing a combinatorial algorithm directly. Moreover, since the integrality gap of (GAP LP) is 2, an $\alpha$-approximate mechanism for a fractional assignment problem gives a $2\alpha$-approximate mechanism for the integral problem. (Recall that the algorithm of [18, 10] shows an integrality gap of 2 for GAP as needed for Lemma A.4.)

**Corollary 4.2.** *Consider any special case of the GAP. If the fractional version of the problem admits a $\beta$-approximate mechanism that is truthful without money, then there exists a $2\beta$-approximate truthful-in-expectation mechanism for the combinatorial problem without money.*

A note is in order on the LP relaxations of the GAP and various cases used in this reduction. Observe that the LP's for the variants of the GAP on a bipartite graph *do not* fit the framework of [14]. This is because the valuation – i.e. the edges – are encoded explicitly in the polytope and not in the objective. However, this is not a problem for us, since the equivalent GAP LP *does* fit the framework. Therefore, after getting a fractional solution to, say, MKP[E], we can simply re-interpret it as a fractional solution to (GAP LP) and perform the reduction of Lavi and Swamy.

## 4.2 The Multiple Knapsack Problem

We consider the multiple knapsack problem on a private bipartite graph $E$. First, we make the simple observation that, if we ignore computational constraints, there exists a truthful optimal mechanism for the multiple knapsack problem in the private graph model. As in maximum (unweighted) bipartite matching, simply returning an optimal solution, breaking ties consistently, leads to a strategyproof mechanism.

**Proposition 4.3.** *Consider the without-money mechanism that, on reports $E \subseteq [n] \times [m]$, finds the optimal integral solution to MKP[E] LP, breaking ties consistently via an arbitrary total order $\preceq$ on the set of assignments $([m] \bigcup \{*\})^{[n]}$. This mechanism is truthful in the private graph model.*

*Proof.* Fix a player $i$ with true edges $E_i$, and fix the reported edges $E_{-i}$ of the other players. Let $x$ be the assignment on reports $E = (E_i, E_{-i})$, and let $x'$ be the assignment on reports $E' = (E'_i, E_{-i})$. Assume for a contradiction that truthfulness is violated, in particular that $x(i) = *$, yet $x'(i) = j$ for some machine $j$ where $(i, j) \in E_i$.

Recall that $x$ is the optimal feasible solution for MKP[$E$], and $x'$ is the optimal feasible solution for MKP[$E'$]. Since $i$ is unassigned in $x$, we know that $x$ is feasible for MKP[$E'$] as well. Moreover, since $x'$ assigns $i$ using an edge in $E$, we know $x'$ is feasible for MKP[$E$]. We conclude that each of $x$ and $x'$ is feasible and optimal for MKP[$E$] and MKP[$E'$]. This contradicts the consistent tie-breaking of the mechanism. □

The above implies that, unlike maximum weight matching and its various generalizations, MKP is not fundamentally incompatible with truthfulness without money – at least when ignoring computational constraints. Nevertheless, MKP on a bipartite graph is APX-hard. Therefore, we consider the problem of finding a truthful constant-factor approximation. Though a simple greedy algorithm gives a deterministic, truthful $2 + \epsilon$ approximation, we will instead illustrate our techniques from Section 4.1 – which will also come in handy for other generalizations of the GAP – by designing a randomized, 2-approximate, truthful-in-expectation mechanism for MKP in our model.

By the discussion in §4.1, it suffices to devise a truthful fractional algorithm in the sense of Equation (6) of Appendix A. In this section, we show that solving (MKP[E] LP) optimally, with careful tiebreaking, yields such an algorithm. By Corollary 4.2, this yields a 2-approximate truthful-in-expectation mechanism for MKP in the private graph model.

**Algorithm 2** Fractional Mechanism for MKP on Private Bipartite Graph

**Input:** Public instance of MKP, and reported edges $E$.
**Output:** A feasible optimal solution $x$ for MKP[E] LP

1: Fix an arbitrary order on the edges of the complete bipartite graph $[n] \times [m]$. Let $\prec$ be the lexicographic order on $[\mathbb{R}]^{[n]\times[m]}$ corresponding to the order on edges.
2: Find the optimal solution $x$ to MKP[E] LP, breaking ties according to $\prec$.
3: **return** $x$

Consider Algorithm 2 for the fractional multiple knapsack problem on a bipartite graph. First of all, it is easy to see that Algorithm 2 can be implemented in polynomial time by solving a sequence of linear programs in step (2). In order to show truthfulness, we will use a flow-based interpretation of Algorithm 2. For reported edges $E$, we define graph $G[E]$, seen in Figure 3, as follows. There is a node for each job, and a node for each machine. We connect job $i$ to machine $j$ if $(i, j) \in E$, with weight $w_{(i,j)} = 0$ and capacity $c_{(i,j)} = \infty$. Next, we include a source node $s$, and create an edge $(s, i)$ for each job $i$ with weight $w_{(s,i)} = v_i/s_i$ and capacity $c_{(s,i)} = s_i$. We then create a sink $t$ and create an edge $(j, t)$ for each machine $j$, with weight $w_{(j,t)} = 0$ and capacity $c_{(j,t)} = c_j$. Finally, we connect the sink to the source via an edge $(t, s)$ with $w_{(t,s)} = 0$ and capacity $c_{(t,s)} = \infty$.

Observe that fractional assignments $[0, 1]^{[n]\times[m]}$ are in one to one correspondence with feasible circulations in the complete bipartite graph $G[[n] \times [m]]$. Moreover, feasible solutions of MKP[E]

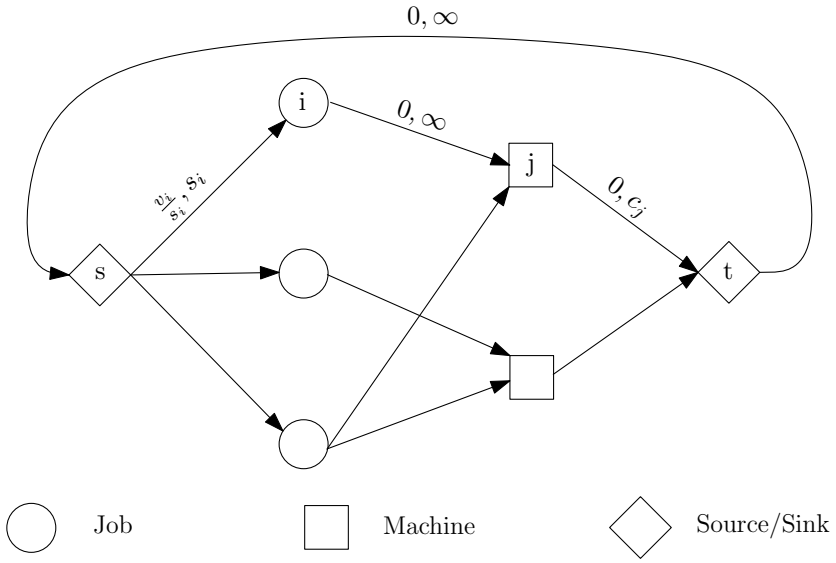


Figure 3: Flow Interpretation of the Multiple Knapsack Problem
Edges are labeled with (weight,capacity) pairs.

LP are in one-to-one correspondence with feasible circulations in $G[E]$. In particular, assignment $x$ feasible for MKP[E] LP maps to the unique feasible circulation $f_x$ on $G[E]$ satisfying $f_x((i,j)) = x_{ij}s_i$ for each job $i$ and machine $j$. Notice, also, that the value (i.e. the welfare) of assignment $x$ is the same as the weight of the circulation $f_x$. Moreover, we define a total order on circulations in $G[[n] \times [m]]$ that corresponds to the lexicographic order $\prec$ defined on $[\mathbb{R}]^{[n]\times[m]}$. We abuse notation and use $\prec$ to refer to both total orders, and let $f_x \prec f_y$ if and only if $x \prec y$. Notice that $\prec$ also orders feasible circulations lexicographically. Therefore, we can interpret Algorithm 2 as finding the $\prec$-minimal maximum-weight feasible circulation in $G[E]$, and then converting it to the corresponding assignment.

Next, we show that Algorithm 2 is a truthful fractional mechanism, that is, a player $i$ cannot benefit by misrepresenting his edges $E_i$ as some $E'_i$. For this, if the algorithm returns assignment $x$ when reported edges are $E$, and assignment $x'$ when reported edges are $E' = E_{-i} \bigcup E'_i$, then we must have $x(E_i) \geq x'(E_i)$ (note that this is because $v_{ij} = v_i$ in MKP). Equivalently, we need to show $\sum_{e\in E_i} f_x(e) \geq \sum_{e\in E_i} f_{x'}(e)$, where we use the convention $f(e) = 0$ when $f$ is a flow on $G[E]$ and $e \notin E$ (and the same for $G[E']$).

We will show that any increase in job $i$'s utility after lying implies that one of $f_x$ or $f_{x'}$ is suboptimal, yielding a contradiction. We will use notions from network flow, developed in Appendix B. We begin with the following lemma.

**Lemma 4.4.** *Let circulation $\Delta$ be the difference between $f_{x'}$ and $f_x$; i.e. $\Delta = f_{x'} - f_x$. If $\sum_{e\in E_i} f_{x'}(e) > \sum_{e\in E_i} f_x(e)$ then $\Delta$ can be conformally decomposed into $\{C, \Delta - C\}$ where: (1) $C$ is a flow cycle, and (2) $C$ sends positive flow on both $(s,i)$ and some $e \in E_i \bigcap E'_i$.*

*Proof.* Let $\left\{C^1, \ldots, C^k\right\}$ be the conformal decomposition of $\Delta$ into cycles as in theorem B.6. It suffices to show that some $C^j$ satisfies conditions (1) and (2).

Since $x'$ sends more flow on edges in $E_i$ than $x$, some $C^j$ enters $i$ through an edge $e' \notin E_i$,

and exits through an edge $e \in E_i$. By conformality and the fact that $x'$ sends no flow on $E_i \setminus E'_i$, we know that $e \in E_i \bigcap E'_i$. Moreover, by conformality and the fact that $x$ sends no flow on edges in $E'_i \setminus E_i$, it is easy to see that $e' \notin E'_i \setminus E$. Therefore, the only remaining possibility is that $e' = (s, i)$. □

This yields truthfulness of the algorithm.

**Lemma 4.5.** *Algorithm 2 is a truthful fractional mechanism*

*Proof.* Fix an instance $(I, E)$, and assume for a contradiction that a player $i$ with true edges $E_i$ benefits by reporting $E'_i$ instead. Let $x$ and $x'$ be the assignments computed by the algorithm on reports $E$ and $E' = E_{-i} \bigcup E'_i$. By assumption, $\sum_{e \in E_i} f_{x'}(e) > \sum_{e \in E_i} f_x(e)$. Let $\Delta$ and $C$ be as in Lemma 4.4. Observe that $C$ does not send flow on any edges in the symmetric difference of $E_i$ and $E'_i$. Therefore, by Lemma B.7 $f_x + C$ is a feasible circulation in $G[E]$, and moreoever $f_{x'} - C$ is a feasible circulation in $G[E']$. If the weight $w(C)$ of circulation $C$ is non-zero, then one of $f_x$ or $f_{x'}$ is non-optimal. Therefore, $w(C) = 0$. Now notice that, by definition of $\prec$, either $f_x + C \prec f_x$ or $f_{x'} - C \prec f_{x'}$. Therefore, one of $f_x$ or $f_{x'}$ is not a $\prec$-minimal optimal solution, yielding the contradiction. □

Combining with Corollary 4.2, we get the following theorem.

**Theorem 4.6.** *There is a polynomial-time, 2-approximate, without-money mechanism for the multiple knapsack problem that is truthful-in-expectation in the private graph model.*

### 4.3 Size-Invariant GAP

We consider the size-invariant generalized assignment problem on a private bipartite graph $E$. Since SIGAP generalizes MWBM, by Theorem 3.3 no deterministic truthful approximation can achieve better than a factor 2 approximation, and moreover no truthful in expectation PTAS is possible. In this section, we devise a 4-approximate without-money mechanism for SIGAP that is truthful-in-expectation in the private graph model. Even though solving SIGAP[E] LP is not fractionally truthful (again, by Theorem 3.3), we show that a simple greedy algorithm is fractionally truthful and yields a 2-approximate solution the LP. Combining this with Corollary 4.2, we get a 4-approximate, without-money mechanism for SIGAP that is truthful in expectation in the private graph model. Consider the following algorithm.

**Algorithm 3** Fractional Mechanism for SIGAP on Private Bipartite Graph

**Input:** Public instance of SIGAP, and solicited private edges $E$.
**Output:** A feasible solution $x$ for SIGAP[E] LP

1: Order $[n]\times[m]$ in decreasing order of value density $d_e$, where $d_{(i,j)} = \frac{v_{ij}}{s_i}$, breaking ties arbitrarily.
2: **for all** $(i, j) \in E$, in the order defined above **do**
3: Fractionally assign as much of job $i$ on machine $j$, until the job is exhausted or the machine is full.
4: **end for**
5: **return** the resulting assignment $x$.

First, we bound the approximation factor of Algorithm 3.

**Lemma 4.7.** *Algorithm 3 returns a 2-approximate solution to SIGAP[E] LP.*

*Proof.* This can be shown by a charging argument, best formalized by constructing a feasible solution to a dual of SIGAP[E] LP of value at most twice the value attained by the algorithm. This dual is shown below, and has decision variables $u \in \mathbb{R}^n$ and $z \in \mathbb{R}^m$:

$$\begin{array}{lll} \text{minimize} & \sum_{i=1}^{n} u_i + \sum_{j=1}^{m} c_j z_j & \\ \text{subject to} & u_i + s_i z_j \geq v_{ij}, & \text{for } (i,j) \in E. \\ & u_i \geq 0, & \text{for } i = 1, \ldots, n. \\ & z_j \geq 0, & \text{for } j = 1, \ldots, m. \end{array} \qquad \text{(SIGAP[E] LPD)}$$

First, we make a simple observation about the algorithm that will be useful in the proof.

**Observation 4.8.** *For a job $i$, edges incident on $i$ are examined in decreasing order of $v_{ij}$ (since size $s_i$ is independent of the machine). For a machine $j$, edges incident on $j$ are examined in decreasing order of $v_{ij}/s_i$.*

We now construct the dual solution $u, z$ in parallel with the execution of the algorithm as follows. Begin with $u = 0$ and $z = 0$. Consider the iteration of Algorithm 3 corresponding to edge $e = (i, j)$. If job $i$ is exhausted on this iteration, set $u_i = v_{ij}$. If the capacity on machine $j$ is exhausted on this iteration, set $z_j = v_{ij}/s_i$. Notice that, in both cases, this satisfies the dual constraint corresponding to edge $(i, j)$. If no assignment is made on this iteration – i.e. either $i$ or $j$ was exhausted in a previous iteration – then we do not update the dual variables. Indeed, there is no need to do so for feasibility: By Observation 4.8, if $i$ is already exhausted then already $u_i \geq v_{ij}$, and if $j$ is already exhausted then already $z_j \geq v_{ij}/s_i$, and either suffices to satisfy the dual constraint for edge $e$.

It remains to bound the value of the dual solution as compared to the primal solution. First, we write twice the value of the primal in a convenient form:

$$2\sum_{i,j} v_{ij} x_{ij} = \sum_i \sum_j v_{ij} x_{ij} + \sum_j \sum_i \frac{v_{ij}}{s_i}(s_i x_{ij})$$

Observe that, by Observation 4.8, $u_i$ lower bounds the value of any edge on which any part of job $i$ is assigned, and $z_j$ lower-bounds the density of any job assigned to machine $j$. Moreover, $u_i$ is non-zero only if $i$ is fully assigned, and $z_j$ is non-zero only if $j$ is full. Therefore, we get

$$\begin{aligned} \sum_i \sum_j v_{ij} x_{ij} + \sum_j \sum_i \frac{v_{ij}}{s_i}(s_i x_{ij}) &\geq \sum_i u_i \sum_j x_{ij} + \sum_j z_j \sum_i (s_i x_{ij}) \\ &= \sum_i u_i + \sum_j z_j c_j \end{aligned}$$

The final term is precisely the value of the dual. Invoking weak LP duality completes the proof. □

It remains to show that Algorithm 3 is fractionally truthful. We begin with an observation.

**Observation 4.9.** *Let $e_1, \ldots, e_{nm}$ be the ordering $[n] \times [m]$ in decreasing order of density $v_{ij}/s_i$. Let $\prec$ denote the lexicographic ordering on $\mathbb{R}^{[n]\times[m]}$. Algorithm 3 returns the $\prec$-maximal feasible solution of SIGAP[E] LP.*

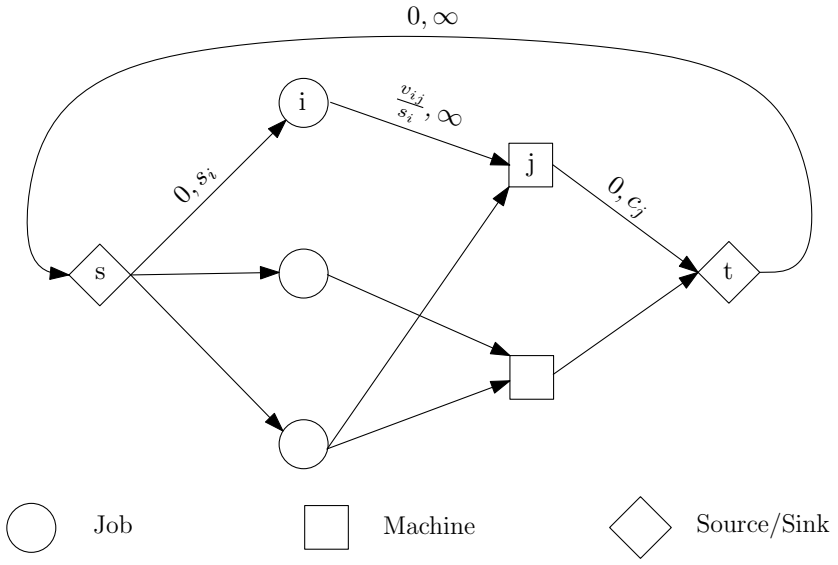


Figure 4: Flow Interpretation of the Size-invariant Generalized Assignment Problem
Edges are labeled with (weight,capacity) pairs.

As in section 4.2, we use a network flow interpretation. We define graph $G[E]$ for $E \subseteq [n] \times [m]$. This construction is similar to that of §4.2, and the graph is shown in Figure 4. As in §4.2, feasible assignments for SIGAP[E] LP are in one-to-one correspondence with feasible circulations in $G[E]$. We can similarly define an order $\prec$ on flows in $G[[n] \times [m]]$ by setting $f_x \prec f_y$ when $x \prec y$. Therefore, we can interpret Algorithm 3 as finding the $\prec$-maximal feasible circulation in $G[E]$. Next, we establish a decomposition lemma similar to, yet more involved than, Lemma 4.4.

**Lemma 4.10.** *Let $x$ be the output assignment of Algorithm 3 with declarations $E$, and let $x'$ be the output assignment with declarations $E' = E_{-i} \bigcup E'_i$. Let circulation $\Delta$ be the difference between $f_{x'}$ and $f_x$; i.e. $\Delta = f_{x'} - f_x$. If $\sum_{e \in E_i} w_e f_{x'}(e) > \sum_{e \in E_i} w_e f_x(e)$ then $\Delta$ can be conformally decomposed into $\{C, \Delta - C\}$ where:*

1. *$C$ is a flow cycle*
2. *$C \prec 0$. In other words, adding $C$ to any other circulation worsens the lexicographic order.*
3. *$C$ enters $i$ through some $e \in E_i \setminus E'_i$, with $C(e) < 0$.*
4. *$C$ exits $i$ through some $e' \in E_i \bigcap E'_i$, with $C(e') > 0$.*
5. *$w_{e'} > w_e$.*

*Proof.* Consider the conformal decomposition of $\Delta$ into cycles $C^1, \ldots, C^k$. It suffices to show that some $C^j$ satisfies the conditions above. By assumption, there is a cycle $C$ in the decomposition with $\sum_{e \in E_i} w_e C(e) > 0$. By conformality, cycle $C$ must exit $i$ through an edge $e' \in E_i \bigcap E'_i$. Moreover $C$ cannot enter through an edge in both graphs $G[E]$ and $G[E']$: if it did, then both $f_x + C$ and $f_{x'} - C$ are feasible in $G[E]$ and $G[E']$ respectively, and thus one of them is not lexigographically maximal, contradicting observation 4.9. Therefore, by conformality, $C$ must remove flow from an

edge $e \in E_i \setminus E'_i$, and add flow to $e' \in E_i \bigcap E'_i$, with $w_{e'} > w_e$. Thus, $C$ satisfies conditions 1, 3, 4, and 5.

It remains to establish condition 2. Observe that $f_x + C$ is a feasible circulation on $G[E]$. Since $f_x$ is a lexicographically maximal feasible circulation on $G[E]$, we deduce $C \prec 0$. □

We are now ready to show truthfulness.

**Lemma 4.11.** *Algorithm 3 is a truthful fractional mechanism for SIGAP.*

*Proof.* Assume, for a contradiction, that a player $i$ benefits by reporting $E'_i$ instead of his true edges $E_i$. Let $x$ and $x'$ be the assignments computed by the algorithm on reports $E$ and $E' = E_{-i} \bigcup E'_i$. By assumption, $\sum_{e \in E_i} w_e f_{x'}(e) > \sum_{e \in E_i} w_e f_x(e)$. Let $\Delta$ and $C$ be as in Lemma 4.10.

Recall that $C \prec 0$. Thus, if $f_{x'} - C$ were feasible in $G[E']$ we would be done, as we would contradict Observation 4.9. However, this is not the case, as $f_{x'} - C$ sends positive flow on edge $e \in E_i \setminus E'_i$. We remedy this by simply zeroing out the flow on edge $e = (i, j)$, as follows: Let $D$ be the flow cycle through $s, i, j, t$ such that $f_{x'} - C - D$ sends no flow on $e$. It is clear that $f_{x'} - C - D$ is a feasible circulation on $G[E']$. We claim that still $f_{x'} - C - D \succ f_{x'}$. To see this, notice that by condition 5 of Lemma 4.10, edge $e$ is not the greatest weight edge with non-zero flow in $-C$. Thus, since $-C \succ 0$, it is easy to see that also $-C - D \succ 0$. Therefore $f_{x'} - C - D \succ f_{x'}$, as needed. □

Combining with Corollary 4.2, we get the theorem.

**Theorem 4.12.** *There is a polynomial-time, 4 approximate, without-money mechanism for the size-invariant generalized assignment problem that is truthful-in-expectation in the private graph model.*

## 4.4 VIGAP and GAP

In this section, we overview the results that we obtain for VIGAP and GAP, utilizing the techniques developed in §4.1-4.3. However, since these differ from our results so far only technically, we defer details to the full version of the paper.

Consider the VIGAP. We observe that Proposition 4.3 holds essentially unchanged; that is, solving VIGAP optimally gives a truthful mechanism in our model. Next, we observe that greedy Algorithm 3, when adapted to the fractional VIGAP (namely, density $d_{(i,j)}$ is now defined as $v_i/s_{ij}$), still provides a 2-approximation by essentially the same analysis. A more involved inductive argument is needed to show that this fractional algorithm is truthful; We defer this technical, yet simple, proof to the full version of the paper. We get the following theorem.

**Theorem 4.13.** *There is a polynomial-time, 4 approximate, without-money mechanism for the value-invariant generalized assignment problem that is truthful-in-expectation in the private graph model.*

We now turn to the GAP. First, we make an assumption – to be removed later – that the maximum value $v_{max}$ of an edge in $E$ is publicly known up-front. Under this assumption, we reduce designing a truthful mechanism for GAP to the truthful mechanism for VIGAP with a loss of $O(\log n)$ in the approximation ratio. In particular, we randomly pick $v \in \{v_{\max}, \frac{v_{\max}}{2}, \frac{v_{\max}}{4}, \ldots, \frac{v_{\max}}{O(n^2)}\}$, and define a new value $\widehat{v}_{ij}$ for each item $i$ and bin $j$ as follows: If $v_{ij} \geq v$ then $\widehat{v}_{ij} = v$, else $\widehat{v}_{ij} = 0$ (equivalently, we discard edge $(i, j)$). This gives an instance of VIGAP that we can solve using the

mechanism of Theorem 4.13. It is easy to verify that, in expectation, this reduction results in a loss of at most $O(\log n)$ in the approximation ratio. However, to ensure truthfulness we need to guarantee that edge $(i, j)$ actually results in value exactly $\widehat{v}_{ij}$ if chosen; we do so by positing up-front that, whenever job $i$ is assigned to machine $j$ by the subroutine that solves VIGAP, we cancel $i$'s assignment with probability $1 - \widehat{v}_{ij}/v_{ij}$. It is now easy to see that, under our original assumption that $v_{\max}$ is known up-front, this mechanism is truthful-in-expectation and has an approximation ratio of $O(\log n)$.

We now remove the assumption that $v_{max}$ is public knowledge by appropriately incentivizing the job with the maximum value edge. In particular, after receiving the reported edges $E$, we flip a fair coin. If the coin turns up heads, we assign the job with the maximum value edge on that edge (*i.e.*, to his favorite machine, with value $v_{\max}$), and leave all other jobs unassigned. If the coin turns up tails, we discard the job with the maximum value edge, and proceed with the algorithm described above using this value of $v_{max}$. It is easy to see that this is still an $O(\log n)$ approximation algorithm. Moreover, the job with the maximum value edge can do no better than report his true edges, and no job has incentive to falsely claim a maximum value edge. This gives the following theorem.

**Theorem 4.14.** *There is a polynomial-time, $O(\log n)$ approximate, without-money mechanism for the generalized assignment problem that is truthful-in-expectation in the private graph model.*

We leave open the question of whether there exists a constant factor truthful [in expectation] mechanism without money for the GAP in our model.

## 5 Acknowledgements

We thank Preston McAfee, Serge Plotkin, Ariel Procaccia, Tim Roughgarden, Michael Schwarz, and Mukund Sundararajan for helpful discussions and comments.

## References

[1] Noga Alon, Michal Feldman, Ariel D. Procaccia, and Moshe Tennenholtz. Strategyproof approximation mechanisms for location on networks. *CoRR*, abs/0907.2049, 2009.

[2] Noga Alon, Felix Fischer, Ariel D. Procaccia, and Moshe Tennenholtz. Sum of us: Strategyproof selection from the selectors. *Working paper*, 2009.

[3] A. Archer and É. Tardos. Truthful mechanisms for one-parameter agents. In *FOCS*, pages 482–491, 2001.

[4] Itai Ashlagi, Felix Fischer, Ian Kash, and Ariel D. Procaccia. Mix and match. In *ACM Conference on Electronic Commerce*, 2010.

[5] Yossi Azar and Iftah Gamzu. Truthful unification framework for packing integer programs with choices. In *ICALP (1)*, pages 833–844, 2008.

[6] Dimitri Bertsekas. *Network Optimization: Continuous and Discrete Models*. Athena Scientific, 1998.

[7] Patrick Briest, Piotr Krysta, and Berthold Vöcking. Approximation techniques for utilitarian mechanism design. In *STOC*, 2005.

[8] Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Working paper*, 2009.

[9] Chandra Chekuri and Iftah Gamzu. Truthful mechanisms via greedy iterative packing. In *APPROX-RANDOM*, pages 56–69, 2009.

[10] Chandra Chekuri and Sanjeev Khanna. A PTAS for the multiple knapsack problem. In *SODA*, pages 213–222, 2000.

[11] Uriel Feige and Jan Vondrák. Approximation algorithms for allocation problems: Improving the factor of 1 - 1/e. In *FOCS*, pages 667–676, 2006.

[12] Lisa Fleischer, Michel X. Goemans, Vahab S. Mirrokni, and Maxim Sviridenko. Tight approximation algorithms for maximum general assignment problems. In *SODA*, pages 611–620, 2006.

[13] R. Lavi, A. Mu'alem, and N. Nisan. Towards a characterization of truthful combinatorial auctions. In *FOCS*, pages 574–583, 2003.

[14] Ron Lavi and Chaitanya Swamy. Truthful and near-optimal mechanism design via linear programming. In *FOCS*, 2005.

[15] R. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6:58–73, 1981.

[16] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.

[17] Ariel D. Procaccia and Moshe Tennenholtz. Approximate mechanism design without money. In *ACM Conference on Electronic Commerce*, pages 177–186, 2009.

[18] David B. Shmoys and Éva Tardos. An approximation algorithm for the generalized assignment problem. *Math. Program.*, 62:461–474, 1993.

[19] William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16(1):8–37, 1961.

# A The Construction of Lavi and Swamy

First, we recall some basic concepts from combinatorial optimization, and state the key lemma in the construction of Lavi and Swamy.

**Definition A.1.** *Let $P \subseteq \mathbb{R}^d$ be a polytope. We define the* Integer hull *of $P$, which we denote by $I(P)$, as the convex hull of all integer points in $P$. Equivalently,*

$$I(P) = hull(P \bigcap \mathbb{Z}^d)$$

**Definition A.2.** *Let $P$ be a polytope. The* integrality gap *of $P$ is defined as:*

$$\max_{c \in \mathbb{R}^d} \frac{\max\{c^T x : x \in P\}}{\max\{c^T x : x \in I(P)\}}$$

Note that the integrality gap of a polytope does not depend on any particular set of objectives $c$. In fact, even in problems where the only objectives of interest are of a certain class – say submodular valuations as used in combinatorial auctions, or objectives $c$ in the nonnegative orthant – the construction of Lavi and Swamy can only perform as well as the integrality gap of the polytope as a whole, as defined above.

**Definition A.3.** *We say an algorithm* shows an integrality gap of $\alpha$ *for a polytope $P \subseteq \mathbb{R}^d$ if it takes as input an arbitrary vector $c \in \mathbb{R}^d$, and outputs a point $z \in P \bigcap \mathbb{Z}^d$ with the guarantee that:*

$$cz \geq \frac{1}{\alpha} \max\{cx : x \in P\}$$

The construction of Lavi and Swamy concerns *packing polytopes*. A polytope $P \subseteq \mathbb{R}^d$ is a packing polytope if it is contained in the positive orthant – i.e. $P \subseteq \mathbb{R}+^d$ –, and moreover it is *downwards closed*: if $y \in P$ and $x \in \mathbb{R}+^d$ is such that $x \preceq y$ (component-wise) then $x \in P$.

Now, we come to the key lemma. We consider a packing polytope $P$, and use $P/\alpha$ to denote the scaled down copy of $P$ – i.e. $P/\alpha = \{y/\alpha : y \in P\}$. Using an algorithm showing the integrality gap of $\alpha$ for $P$, we can explicitly construct for every $x \in P/\alpha$ a distribution over integer points of $P$ that evaluates to $x$ in expectation.

**Lemma A.4** ([14])**.** *Let $P$ be a packing polytope of integrality gap at most $\alpha$. Then, for every $x \in P/\alpha$ there exists a distribution $D_x$ over $P \bigcap \mathbb{Z}^d$ such that $E_{y \sim D_x} y = x$. Moreover, if there exists a polynomial-time algorithm $B$ that shows an integrality gap of $\alpha$ for $P$, then, for any $x \in P/\alpha$ we can compute $D_x$ in polynomial time.*

Armed with the above lemma, Lavi and Swamy *reduce* designing a truthful-in-expectation mechanism (with money) to designing a *truthful fractional mechanism* (also with money), to be defined later.

We illustrate the technique of Lavi and Swamy by considering welfare maximization problems in a very general form. Let $P \subseteq \mathbb{R}+^d$ be a packing polytope representing the set of feasible solutions. Moreover, assume that there are $n$ players $[n]$, and the *valuation function* $v_i : \mathbb{R}^d \to \mathbb{R}$ of player $i$ is required to lie in some valid set of linear valuations $\mathcal{V}_i \subseteq \mathbb{R}^{\mathbb{R}^d}$. We denote $\mathcal{V} = \mathcal{V}_1 \times \mathcal{V}_2 \ldots \times \mathcal{V}_n$. The *combinatorial welfare maximization problem* (henceforth CWMP) over $P$ and $\mathcal{V}$ is the problem of finding an integer point in $P$ maximizing the sum of values of the players. This

gives the following LP relaxation, which we refer to as the *fractional welfare maximization problem* (henceforth FWMP):

$$\begin{array}{ll} \text{maximize} & \sum_i v_i(x) \\ \text{subject to} & x \in P \end{array}$$

We define a *truthful-in-expectation mechanism* for the combinatorial welfare maximization problem as a pair $M = (f, p)$ where $f : \mathcal{V} \to P \bigcap \mathbb{Z}^d$ is a *randomized allocation rule*, and $p : \mathcal{V} \to \mathbb{R}^n$ is a *randomized payment scheme*. As usual, we say $M$ is *truthful in expectation* if the following holds for each $\vec{v} \in \mathcal{V}$ and $v_i \in \mathcal{V}_i$

$$\mathbf{E}[v_i(f(v)) - (p(v))_i] \geq \mathbf{E}[v_i(f(v_{-i}, v_i')) - (p(v, v_i'))_i]. \tag{5}$$

Moreover, we define a *fractional mechanism* as a pair $M = (f, p)$ where $f : \mathcal{V} \to P$ is a *fractional allocation rule*, and $p : \mathcal{V} \to \mathbb{R}^n$ is a *payment scheme*. We say $M$ is a *truthful fractional mechanism* if the following holds for each $\vec{v} \in \mathcal{V}$ and $v_i \in \mathcal{V}_i$

$$v_i(f(v)) - (p(v))_i \geq v_i(f(v_{-i}, v_i')) - (p(v, v_i'))_i. \tag{6}$$

It is easy to see that we can use the VCG mechanism to obtain a truthful fractional mechanism $M_{frac}$ for the fractional welfare maximization problem. Moreover, when optimizing objectives in $\mathcal{V}$ over $P$ can be done efficiently, $M_{frac}$ is a polynomial time mechanism. Lavi and Swamy observed that, given an efficient approximation algorithm that shows the integrality gap of $P$ in the sense defined above, one can obtain a polynomial-time, truthful-in-expectation mechanism $M_{exp}$ that looks simply like a scaled down version of $M_{frac}$. This follows from lemma A.4. We state the main theorem of Lavi and Swamy and sketch its proof below.

**Theorem A.5** ([14])**.** *Assume the fractional welfare maximization problem can be solved efficiently for any valuations in $\mathcal{V}$. Moreover, assume $P$ satisfies the conditions of Lemma A.4 with integrality gap $\alpha$ and algorithm $B$ showing the integrality gap. Then there exists an efficient truthful-in-expectation $\alpha$-approximate mechanism for the welfare maximization problem over $P$ and $\mathcal{V}$.*

*Proof Sketch.* Consider the efficient, truthful fractional mechanism $M_{frac} = (f_{frac}, p_{frac})$ constructed using VCG. We will define $M_{exp} = (f_{exp}, p_{exp})$ as follows. We let $p_{exp}(v) = \frac{p(v)}{\alpha}$, and let $f_{exp}(v)$ be drawn from the distribution $D_{f_{frac}(v)/\alpha}$, as defined in Lemma A.4. The random function $f_{exp}$ can be evaluated efficiently using $B$, as in the lemma. Now, using linearity of expectations, the linearity of the valuation functions, and Lemma A.4, it is easy to show truthfulness of $M_{exp}$ by showing that the expected valuation less the payment of a player matches that of $M_{frac}$ up to a scaling factor of $\alpha$.

$$\begin{aligned} \mathbf{E}[v_i(f_{exp}(v')) - (p_{exp}(v'))_i] &= v_i(\mathbf{E}[f_{exp}(v')]) - (p_{exp}(v'))_i \\ &= v_i(\frac{1}{\alpha} \cdot f_{frac}(v')) - \frac{1}{\alpha} \cdot (p_{frac}(v'))_i \\ &= \frac{1}{\alpha} \cdot [v_i(f_{frac}(v')) - (p_{frac}(v'))_i] \end{aligned}$$

Applying this equivalence to both sides of inequality (6) yields truthfulness of $M_{exp}$. □

It is worth noting that a simple modification to $p$ allows us to guarantee individual rationality, while preserving truthfulness-in-expectation. However, this will not be relevant for our purposes, as all our mechanisms will utilize no payments.

# B Network Flow Preliminaries

Here we recall some notions from network flow theory, and define simple concepts that we will need in analysis of our algorithms for MKP and SIGAP.

We recall that a *weighted-capacitated directed graph* is a directed graph $G = (V, E)$, with a weight $w_e \in \mathbb{R}^+$ and capacity $c_e \in \mathbb{R}^+$ on an edge $e$. We define a flow simply as follows.

**Definition B.1.** *A flow is a vector $f \in \mathbb{R}^{E(G)}$.*

We distinguish circulations on $G$ as follows.

**Definition B.2.** *A flow $f \in \mathbb{R}^{E(G)}$ is a* circulation *if it conserves flow on each vertex. In particular, for each $v \in V$ with incoming edges $\Gamma^-(v)$ and outgoing edges $\Gamma^+(v)$, we have:*

$$\sum_{e \in \Gamma^-(v)} f(e) = \sum_{e \in \Gamma^+(v)} f(e)$$

We define the *weight* of circulation $f$ as follows: $w(f) = \sum_{e \in E(G)} w_e f(e)$. Note that we allow a circulation to send negative flow on an edge, as well as overflow the capacity of an edge. A *feasible* circulation is better behaved.

**Definition B.3.** *A vector $f \in \mathbb{R}^{E(G)}$ is a* feasible circulation *if it is a circulation, and moreover it satisfies nonnegativity and capacity constraints. In particular, for each edge $e$ we have:*

$$0 \leq f(e) \leq c_e$$

We distinguish the simplest circulations, or *flow cycles*, and recall that every circulation can be decomposed into cycles that are oriented consistently.

**Definition B.4** (Flow Cycle)**.** *A circulation $C$ on $G$ is a* flow cycle *if there exists a simple undirected cycle $H$ such that $C$ sends non-zero flow only on edges of $H$.*

**Definition B.5** (Conformal Decomposition)**.** *Fix a circulation $f$. We say a set of circulations $\{g^1, \ldots, g^k\}$ is a* conformal decomposition *of $f$ if the following hold*

1. *(Decomposition) $f = \sum_{i=1}^{k} g^i$*
2. *(Conformal) For each $e \in E(G)$, $f(e) \cdot g^i(e) \geq 0$.*

Note that every $g^i$ in the conformal decomposition must send flow in the same direction as $f$ on each edge.

**Theorem B.6** ([6])**.** *Every circulation $f$ can be conformally decomposed into flow cycles.*

Conformal decomposition immediately yields a useful property of feasible circulations.

**Lemma B.7.** *Let $f$ and $f'$ be feasible circulations on $G$. Let $C_1, \ldots, C_k$ be a conformal decomposition into cycles of $f' - f$. Then, for every $L \subseteq \{1, \ldots, k\}$ the circulation $f + \sum_{i \in L} C_i$ is feasible. Equivalently, for every $L \subseteq \{1, \ldots, k\}$ the circulation $f' - \sum_{i \in L} C_i$ is feasible.*

In other words, the cycles of the conformal decomposition of $f' - f$ may be added to $f$ (or subtracted from $f'$) in any order, maintaining feasibility along the way.