MATHEMATICS OF OPERATIONS RESEARCH
Vol. 39, No. 3, August 2014, pp. 722–736
ISSN 0364-765X (print) | ISSN 1526-5471 (online)

http://dx.doi.org/10.1287/moor.2013.0625
© 2014 INFORMS

Truthful Generalized Assignments via Stable Matching

Ning Chen
Division of Mathematical Sciences, Nanyang Technological University, Singapore 637371, ningc@ntu.edu.sg

Nick Gravin
Microsoft Research New England, Cambridge, Massachusetts 02142, ngravin@microsoft.com

Pinyan Lu
Microsoft Research Asia, Beijing 100080, China, pinyanl@microsoft.com

In the generalized assignment problem (GAP), a set of jobs seek to be assigned to a set of machines. For every job-machine pair, there are a specific value and an accommodated capacity for the assignment. The objective is to find an assignment that maximizes the total sum of values given that the capacity constraint of every machine is satisfied.

The GAP is a classic optimization problem and has been studied extensively from the algorithmic perspective. Dughmi and Ghosh [Dughmi S, Ghosh A (2010) Truthful assignment without money. *ACM Conf. Electronic Commerce* (ACM, New York), 325–334.] proposed the game theoretical framework in which every job is owned by a selfish agent who aims to maximize the value of his own assignment. They gave a logarithmic approximation truthful in expectation mechanism and left open the problem whether there exists a truthful mechanism with a constant approximation factor. In this paper, we give an affirmative answer to this question and provide a constant approximation mechanism that enjoys a stronger incentive property of universal truthfulness than that of truthfulness in expectation.

Our mechanism is inspired by stable matching, which is a fundamental solution concept in the context of matching marketplaces. The mechanism uses a stable matching algorithm as a critical component and adopts other approaches like random sampling.

Keywords: generalized assignment; mechanism design; stable matching
MSC2000 subject classification: Primary: 91B26; secondary: 91B68
OR/MS subject classification: Primary: games/group decisions: bidding/auctions
History: Received September 14, 2012; revised May 9, 2013. Published online in *Articles in Advance* October 23, 2013.

1. Introduction.

Consider a scenario in which there are a set of jobs and a set of machines. Each job i has a value v_{ik} for machine k and occupies a corresponding capacity c_{ik}. Further, each machine k has a sharp capacity constraint C_k, which gives an upper bound on the total capacity of jobs that can be feasibly scheduled on the machine. This defines a natural optimization question, called the *generalized assignment problem* (GAP), where the objective is to find a feasible assignment that maximizes the total sum of values given the capacity constraints. The GAP models resource allocation problems in which different agents (i.e., jobs) compete for limited resources and has numerous applications in computer science, operation research, and economics.

From the theoretical point of view, the GAP model generalizes a number of classic optimization problems and receives considerable attention in the literature. It is easy to see that the problem is NP-hard because it includes in particular the knapsack problem as a special case. Shmoys and Tardos [31] and Chekuri and Khanna [10] gave a 2-approximation algorithm for the GAP. Chekuri and Khanna [10] also developed a polynomial time approximation scheme for a special case in which every job has the same value and capacity for all machines. Fleischer et al. [18] gave an $e/(e-1)$-approximation for the GAP based on linear program rounding. The ratio was later improved to $e/(e-1)-\epsilon$ for a very small ϵ by Feige and Vondrak [17]. On the negative side, Chakrabarty and Goel [9] showed that the problem does not admit any approximation ratio better than $11/10$.

In all aforementioned approximation analyses, there is one important factor that is not taken into account: incentives. In many applications, jobs are owned by self-interested agents, who would like to receive as much benefit as possible. To achieve such an objective, agents may misreport some of their private information to the assignment designer. *Algorithmic mechanism design*, initiated by the seminal work of Nisan and Ronen [27], takes incentive preferences of self-interested agents into account for a multitude of algorithmic challenges. The objective in the field of algorithmic mechanism design is to find algorithms that create incentives for the agents to reveal their private information (such algorithms are called *truthful* or incentive compatible mechanisms) and yield as good solutions as possible. Note that a specific feature in the considered GAP model is that there are no money transfers in an assignment.

In order to design truthful mechanisms, we should first decide on *what private information is held by the agents*. A natural assumption is that all the values v_{ik} are private information. Unfortunately, we cannot expect to have any mechanism with a good approximation in such a setting because each agent can always bid a sufficiently large value to achieve his desired assignment (because there is no charged payment as a medium of compensation). However, in some scenarios, valuations (either the exact values or estimations) are publicly known. For instance, in a job market, a company lists a number of part-time job positions and their corresponding salaries (e.g., search for "part-time job" at http://www.e4s.co.uk/), which are precisely valuations for job hunters. As another example, in Bayesian mechanism design (e.g., Myerson [26], Hartline and Roughgarden [24], Hartline and Lucier [23]), there are publicly known probability distributions over the valuations. In some applications, agents are uncertain about the exact values but instead rely only on the knowledge of the public distributions (see, e.g., Feige and Tennenholtz [16] and Chiesa et al. [11] for more discussion). The agent's expected value in such settings is the mean of the respective distribution and is a fixed known value.

The preference of an agent in GAP is usually determined not only by valuations but also by some other factors, e.g., compatibility. In the above job market example, a job candidate only considers those positions to which he will be able to commute. To capture the feature of compatibility, Dughmi and Ghosh [15] and Bochet et al. [7] assume that there is a subset S_i for each job i that describes the set of compatible machines. That is, the job is willing to be assigned only to a machine in S_i (the preference of the jobs within S_i is determined by valuations). The set S_i is the private information of agent i. In a mechanism, each agent/job i submits a subset of machines S_i'; then the mechanism decides on an assignment given the public valuations and capacity constraints. The objective of the mechanism designer is to maximize the total valuation in the assignment.

The model considered in Dughmi and Ghosh [15] and Bochet et al. [7] captures the feature that in some applications agents can only specify their interested outcomes (but not preferences). As another example, consider assigning teaching assistants (or tutors) to courses in a department. Before the assignment, the department may first look at the performances (e.g., scores) of candidate students for these courses, and usually high scores are desired in the assignment. Likewise, a student may also prefer courses in which he has a high score because then he is more confident in leading tutorials. In practice, a candidate student may only indicate which courses he is willing to teach, and the final assignment is determined by the department. See more discussions on the motivation of the model in Dughmi and Ghosh [15] and Bochet et al. [7].

Dughmi and Ghosh [15] first studied this algorithmic mechanism design problem in GAP and gave a truthful in expectation mechanism that approximates by a logarithmic factor the optimal solution in the complete information setting. They left open the question whether there exists a constant approximation truthful mechanism. We solve the problem with an affirmative answer. Our mechanism is universally truthful (i.e., distributed over deterministic truthful mechanisms), which is stronger than that of the truthfulness in expectation (i.e., truthful bidding maximizes expected utility).

THEOREM (MAIN). *There is a polynomial time–constant approximation universally truthful mechanism for the generalized assignment problem.*

An important feature of the GAP model and our mechanism is that there are no payments. Indeed, in a large number of important domains, e.g., political elections, organ transplants, and resource allocations, monetary transfers are undesirable or strictly prohibited. Mechanism design in these settings, tracing back to the well-studied social choice theory that maps agents' preferences to a set of alternatives, allows no payment and therefore is more challenging. In particular, the strong impossibility theorems by Arrow [3], Gibbard [20], and Satterthwaite [30] imply that in some settings no nontrivial truthful mechanisms exist. Because of those impossibility results, exploring domains in which there exist truthful mechanisms that generate desirable solutions is one of the main questions in social choice.

Approximation, a powerful approach from theoretical computer science, provides an elegant tool to analyze designed mechanisms. In particular, the existence of constant approximation mechanisms can be considered as a balance between interests of the centralized mechanism designer and individual self-interested agents. Our result implies that in the GAP resource allocation model, a reasonable loss in the social welfare suffices to ensure a truthful bidding environment.

1.1. Techniques.

The mechanism design for the GAP is in a multiparameter domain. Our main truthful mechanism is inspired by stable matching, which is introduced by Gale and Shapley [19] and is a fundamental solution concept in the context of matching marketplaces. In the setup, there are a set of men and a set of women, each with a preference ranking over members of the other side; a matching between the men and women is *stable* if there is no man-woman pair who both strictly prefer each other to their current partners. From the strategic point of view, although Roth's impossibility result (Theorem 4.4, Roth and Sotomayor [29]) implies that it does not admit any truthful design for both men and women to claim their preferences truthfully, the men-optimal stable matching algorithm (i.e., men make proposals in the deferred acceptance algorithm of Gale and Shapley) is indeed truthful for all men. The same incentive result holds in many-to-one matching models (i.e., one side of the market can have multiple assignments) if matches are proposed from the unit demand side of the market (Theorem 5.16, Roth and Sotomayor [29]).

In order to apply stable matching theory, we need to define preferences for the two parties, i.e., jobs and machines in GAP. In particular, since every machine with a knapsack constraint can take multiple jobs, one has to also specify the preference of the machine over the subsets of jobs, not only over individual jobs. Given carefully designed preferences of jobs and machines, we run a similar deferred acceptance stable matching algorithm (jobs make proposals) to generate a feasible assignment. The mechanism, although shown to be truthful for some special cases (Theorem 1), for the general case of GAP, surprisingly (compared to the incentive result described above for many-to-one matchings) is not truthful (Example 1). In other words, *stability no longer guarantees truthfulness.*

To derive a truthful mechanism with a constant approximation, we build our mechanism on the stable matching algorithm as a critical component and use the idea of *random sampling*, which turns out to be a powerful approach in mechanism design and used in, e.g., digital goods auctions (Goldberg et al. [21]), combinatorial auctions (Dobzinski [14]), and budget feasible mechanisms (Bei et al. [6]). We choose, in expectation, half of the jobs to form a test set T and run the stable matching algorithm on T. The returned stable assignment $\mathcal{A}^T$ gives a close estimate to the structure of the optimal solution with a high probability. Then using the sampled outcome $\mathcal{A}^T$ as a guidance, we compute a real assignment $\mathcal{A}$ for the rest of the jobs. Stable matching also plays a crucial role in the analysis of the mechanism. In particular, for every unassigned job-machine pair in a stable assignment, since at least one of them prefers its current assignment, the contribution of either the job or the machine will compensate the loss of the pair. Based on this idea, we establish an upper bound on the optimal solution using a stable assignment $\mathcal{A}^*$ returned by the stable matching algorithm running on *all* jobs. By using a probabilistic analysis, we compare the stable assignments $\mathcal{A}$, $\mathcal{A}^T$, and $\mathcal{A}^*$ and show that they are all close to each other, which yields the desired constant approximation.

1.2. Related work. There was a recent surge in the study of *approximate mechanism design without money* since the pioneer work of Procaccia and Tennenholtz [28]. The resource allocation problems that have been studied in this framework include those of facility location (Procaccia and Tennenholtz [28], Lu et al. [25], Alon et al. [1]); kidney exchange (Ashlagi et al. [4], Caragiannis et al. [8]); voting (Alon et al. [2]); and job scheduling (Dughmi and Ghosh [15]). All of these works, as well as ours, consider designing truthful mechanisms that approximate optimal solutions and use no monetary transfers.

Bochet et al. [7] studied an assignment problem in a bipartite marketplace: every vertex is held by an agent, and the set of edges incident to a vertex is considered as private information. Similar to Dughmi and Ghosh [15] and our work, in the model studied in Bochet et al. [7], valuations on the edges are public as well (which is simply one) and agents can hide some edges to the mechanism. However, Bochet et al. [7] used a "single peaked preference" for each agent that depends solely on the quantity of matched edges. Thus, the utility function of every agent depends only on a single parameter (the number of edges matched), whereas in our model we consider a more diverse preference in terms of the distinct valuations.

Baiou and Balinski [5] and Dean et al. [13] studied a similar job-machine assignment problem in which both jobs and machines have strict preferences over the other side and provided efficient algorithms to compute job-optimal stable assignments. However, the settings studied in Baiou and Balinski [5] and Dean et al. [13] are quite different from ours: a job can be assigned to multiple machines (Baiou and Balinski [5]) and a machine can be overcapacitated (Dean et al. [13]).

2. Preliminaries.

Consider an instance of the generalized assignment problem (GAP) in which we are given a set of jobs and a set of machines. Throughout the paper, we will denote a generic job by i and a generic machine by k. For each job i and machine k, there is a value $v_{ik} \geq 0$ and a capacity $c_{ik} \geq 0$. Further, each machine k has a capacity constraint C_k. An output of the GAP is a feasible *assignment* $\mathcal{A}$ between the jobs and machines, where each job is assigned to at most one machine and the aggregate capacity of the assigned jobs to each machine is within the machine's capacity constraint.

Given a feasible assignment $\mathcal{A}$ between the jobs and machines, let $\mathcal{A}_i$ be the machine that job i is assigned to (denote by $\mathcal{A}_i = \varnothing$ if i is not assigned to any machine) and $\mathcal{A}_k = \{i \mid \mathcal{A}_i = k\}$ be the set of jobs that are assigned to the machine k. To simplify notations, let $v(\mathcal{A}) = \sum_{(i,k)\in\mathcal{A}} v_{ik}$ be the total value of the assignment and $v_i(\mathcal{A}) = v_{i,\mathcal{A}_i}$ be the value of job i in the assignment (let $v_i(\mathcal{A}) = 0$ if $\mathcal{A}_i = \varnothing$). Further, for any subset of jobs S, let $v(S) = \sum_{i\in S} v_i(\mathcal{A})$; in particular, for the set of jobs $\mathcal{A}_k$ assigned to machine k, the total value is given by $v(\mathcal{A}_k) = \sum_{i\in\mathcal{A}_k} v_i(\mathcal{A}) = \sum_{i\in\mathcal{A}_k} v_{ik}$. We define short hand notations for $c(\cdot)$ similarly to all previous $v(\cdot)$.

We now have a natural optimization problem to find a feasible assignment that maximizes the total valuation (i.e., social welfare):

$$\begin{aligned} \max_{\mathcal{A}} \quad & v(\mathcal{A}) \\ \text{s.t.} \quad & c(\mathcal{A}_k) \le C_k, \quad \forall k \end{aligned}$$

We denote by *opt* the value of an optimal solution to this problem.

Assume that each job i is held by an agent and the private information of the agent is the set of compatible machines. The problem can be then described as a bipartite graph G where one side corresponds to agents/jobs, the other side corresponds to machines, and edges represent compatible job-machine pairs. The value v_{ik} and capacity c_{ik} on every edge (i, k) are public. The private information of every agent/job i is the set of edges $S_i = \{(i, k) \in G \mid k\}$ incident to the vertex. All considered job-machine pairs are with respect to the underlying compatible graph G.

Every agent i, as a self-interested individual, has his own objective and would like to get assigned to a machine k with maximum possible value v_{ik}. It is therefore not always the case that agent i reports his true information S_i to the protocol to his best interest. Algorithmic mechanism design takes such incentive issues into account with a focus on managing self-interested behavior of the agents. In our framework, upon receiving a submitted bid S_i' from each agent, which can be any subset of machines,[1] the mechanism decides on an assignment $\mathcal{A}_i \in S_i' \cup \{\varnothing\}$ for each agent as an output so that the overall assignments satisfy the capacity constraints for all machines.

We say a mechanism is *truthful* if it is a dominant strategy for every agent to report his true information S_i. That is, for any submitted bids of other agents, no individual can get a better outcome by reporting a set that is different from his true information. If a mechanism is nondeterministic, we say the mechanism is *universally truthful* if it takes a random distribution over deterministic truthful mechanisms and is *truthful in expectation* if no risk-neutral agent can obtain more utility in expectation by misreporting his private information.

Note that the mechanisms considered in the current paper only determine assignments and do not collect or distribute any payment. This is the reason why some classic designs (e.g., the VCG mechanism Vickrey [32], Clarke [12], Groves [22]) are not applicable. At the cost of being truthful and without payment, we cannot expect a mechanism to have an output that maximizes social welfare (see examples in, e.g., Procaccia and Tennenholtz [28] and Dughmi and Ghosh [15]). Our focus therefore is to design truthful mechanisms (without money) that can be implemented in polynomial time and yield good approximations to the optimal social welfare *opt* (which is the optimal value of the above optimization problem). We say a (randomized) mechanism has an approximation ratio of α if for any GAP instance, the ratio between *opt* and the generated (expected) social welfare is at most α.

3. Stable matching algorithm.

Stable matching is a fundamental solution concept in the context of two-sided matching marketplaces, introduced by Gale and Shapley in their seminal work (Gale and Shapley [19]). A matching between the two parties of a bipartite marketplace is called stable if there is no pair who both strictly prefer each other to their current partners. The deferred acceptance algorithm of Gale and Shapley [19] computes a stable matching.

In order to apply stable matching theory to our problem GAP, one needs to define preferences for the two parties, jobs and machines. For every job i, its preference over machines is pretty straightforward, which is simply according to the preference of the agent, i.e., ranking machines by the value v_{ik}. For every machine k, we may define its preference over jobs in a similar way according to the value v_{ik}. For such preferences, the deferred acceptance algorithm can be implemented in the same way as the greedy algorithm (i.e., match job-machine pairs according to the decreasing order of v_{ik}) where ties are broken arbitrarily. We denote such implementation by GAP-GREEDY. Because every job goes to at most one machine, this defines a many-to-one matching. Thus, by Theorem 5.16 of Roth and Sotomayor [29], we have the following observation:

LEMMA 1. *The algorithm* GAP-GREEDY *is truthful for all jobs for any fixed tie-breaking rule.*

The solution generated by GAP-GREEDY, however, may have an arbitrarily bad total valuation since the preferences completely ignore the capacities of the jobs. To balance the value v_{ik} and capacity c_{ik}, in the rest of this section, we will consider the following preferences:

- Every job i has a strict preference list, denoted by $\mathcal{L}_i$, which is according to the decreasing order of v_{ik}.

[1] If a job is assigned to a machine that is not in S_i, which is possible if the job bids machines that are not in S_i, we assume that the utility of the job is 0 (or any other nonpositive number; this does not affect our universally truthful mechanism).

• For every machine k, its strict preference over individual jobs is according to the decreasing order of the ratio v_{ik}/c_{ik}. That is, the machine prefers jobs that provide a higher value per capacity. Since a machine may take multiple jobs, we have to specify the preference of each machine k over subsets of jobs. For any two feasible subsets of jobs S and S', where $\sum_{i\in S} c_{ik} \le C_k$ and $\sum_{i\in S'} c_{ik} \le C_k$, consider the two vectors $(v_{ik}/c_{ik})_{i\in S}$ and $(v_{ik}/c_{ik})_{i\in S'}$ with decreasing coordinate values and let k prefer the subset that is lexicographically larger with respect to the two vectors. We denote such preference list of every machine by $\mathscr{L}_k$.[2]

We require that the preferences $\mathscr{L}_i$ and $\mathscr{L}_k$ are *strict*. If there are the same values $v_{ik} = v_{ik'}$ or ratios $v_{ik}/c_{ik} = v_{i'k}/c_{i'k}$, we always break ties in favor of the smaller capacity; if the capacities are the same as well, ties are broken in an arbitrary but fixed order. We say a feasible assignment $\mathscr{A}$ is *stable* if it does not contain any blocking pair i and k, where i strictly prefers k to $\mathscr{A}_i$ and k can pick a strictly better subset of jobs in the collection $\mathscr{A}_k \cup \{i\}$ (which is easy to verify).

We consider the following deferred acceptance algorithm, called SM-DA-ALG.

```
SM-DA-ALG
1. Initialize all jobs to be free and set $\mathscr{A} = \varnothing$.
2. while there is a free job $i$ that did not propose to all machines in $\mathscr{L}_i$
   • Among all pairs $(i, k)$ where $i$ is free and $k$ is the highest unproposed machine in $\mathscr{L}_i$, pick one with
     the maximum $v_{ik}/c_{ik}$ and let $i$ propose to $k$.
   • Let $S = \varnothing$.
   • For each $i' \in \mathscr{A}_k \cup \{i\}$ in the preference order of $\mathscr{L}_k$
     o if $S \cup \{i'\}$ is feasible for $k$, let $S \leftarrow S \cup \{i'\}$.
   • If $i \in S$, let $\mathscr{A}_k \cup \{i\} \setminus S$ be free and update assignment $\mathscr{A}_k \leftarrow S$.
   • Else (we must have $S = \mathscr{A}_k$), let $i$ remain free.
3. Output $\mathscr{A}$.
```

When there are ties in the selection of the largest ratio, i.e., $v_{ik}/c_{ik} = v_{i'k'}/c_{i'k'}$, to implement the algorithm, we use the same tie-breaking rule as described above for $\mathscr{L}_i$ and $\mathscr{L}_k$. Given this tie-breaking rule and the strict preferences of $\mathscr{L}_i$ and $\mathscr{L}_k$, the output of the algorithm is uniquely determined. Since every job proposes to every machine at most once, the algorithm is guaranteed to terminate. Note that in the first step of the "while" loop, picking a pair with the largest ratio v_{ik}/c_{ik}, is necessary in order to derive the following Theorem 1.

We note that SM-DA-ALG is not truthful for the general GAP, as the following example shows.

EXAMPLE 1. There are four jobs $\{1, 2, 3, 4\}$ and three machines $\{x, y, z\}$. The values, capacities, and preferences of compatible pairs are given by the following table:

	Job 1	Job 2	Job 3	Job 4
Job				
Value	$v_{1x} = 1, v_{1y} = 0.5$	$v_{2x} = 1, v_{2z} = 0.5$	$v_{3x} = 10, v_{3z} = 20$	$v_{4x} = 5, v_{4y} = 0.1$
Capacity	$c_{1x} = 0.5, c_{1y} = 1$	$c_{2x} = 0.5, c_{2z} = 1$	$c_{3x} = 1, c_{3z} = 100$	$c_{4x} = 1, c_{4y} = 1$
Preference	$\mathscr{L}_1$: $x > y$	$\mathscr{L}_2$: $x > z$	$\mathscr{L}_3$: $z > x$	$\mathscr{L}_4$: $x > y$

	Machine x	Machine y	Machine z
Machine			
Capacity	$C_x = 1$	$C_y = 1$	$C_z = 100$
Preference	$\mathscr{L}_x$: $\{3\} > \{4\} > \{1 = 2\}$	$\mathscr{L}_y$: $\{1\} > \{4\}$	$\mathscr{L}_z$: $\{2\} > \{3\}$

If all agents report their true information, jobs 1 and 2 propose to x, and job 3 proposes to z; then job 4 proposes to x, which kicks jobs 1 and 2 off to y and z, respectively. Job 3 is kicked off by 2 then and next proposes to x, which kicks job 4 off. Hence, job 4 does not get any assignment eventually. However, if job 4 hides x and only reports machine y, then it will get y by the algorithm. Therefore, SM-DA-ALG is not truthful.

[2] Alternatively, one can define the preference over feasible subsets of jobs by simply comparing the aggregate values of the jobs in the subsets. Such a preference rule, although capturing the real preference of social welfare, is not implementable as for any given unfeasible set S; finding a feasible subset $S' \subset S$ that has the largest total value is equivalent to solving a knapsack problem, which is known to be NP-hard. Our preference $\mathscr{L}_k$, however, still balances the values and capacities in terms of their ratios and is easy to implement in the following stable matching algorithm and the main mechanism.

Although SM-DA-ALG is not truthful for the general GAP, for the special cases job/machine value/capacity invariant (defined below), it is truthful. Indeed, the combination between SM-DA-ALG and GAP-GREEDY gives a universally truthful mechanism with an approximation ratio of 4 for these special cases. (Note that Dughmi and Ghosh [15] gave a 4-approximation truthful in expectation mechanisms for the job value/capacity invariant by an LP-based approach.)

THEOREM 1. *Consider the following mechanism: Run either* GAP-GREEDY *or* SM-DA-ALG *with equal probability. The mechanism is universally truthful with an approximation ratio of* 4 *for each of the following cases:*
1. (*Job value invariant*) $v_{ik} = v_{ik'}$ *for all machines* k, k' *and any job* i.
2. (*Job capacity invariant*) $c_{ik} = c_{ik'}$ *for all machines* k, k' *and any job* i.
3. (*Machine value invariant*) $v_{ik} = v_{i'k}$ *for all jobs* i, i' *and any machine* k.
4. (*Machine capacity invariant*) $c_{ik} = c_{i'k}$ *for all jobs* i, i' *and any machine* k.

To see the idea behind our mechanism, consider the following two illustrative examples:

• There are one machine with capacity 1 and two jobs with values and sizes ($v_1 = 2\epsilon, c_1 = \epsilon$) and ($v_2 = 1, c_2 = 1$), respectively.

• There are one machine with capacity 1 and a job with value and size ($v_1 = 1, c_1 = 1$). In addition, there are many identical jobs with value and size ($v_2 = 1 - \epsilon, c_2 = \epsilon$) each.

It can be seen that the two greedy algorithms, deciding allocations according to the decreasing order of v_{ik} (i.e., GAP-GREEDY) and v_{ik}/c_{ik} (denoted by GAP-GREEDY$_{\text{ratio}}$), respectively, yield an optimal solution and work arbitrarily badly (and the other way around) for the above two examples, respectively. Indeed, a randomization between these two greedy algorithms gives a constant approximation for any instance. However, it can be seen that GAP-GREEDY$_{\text{ratio}}$ is not truthful. (For instance, there are a job and two machines with $v_1 = c_1 = 1$ and $v_2 = 0.5$ and $c_2 = \epsilon$. Then the job would only claim the first machine in order to achieve an assignment of value 1.) We therefore need to find an alternative to replace GAP-GREEDY$_{\text{ratio}}$.

Our stable matching algorithm, SM-DA-ALG, ensures truthfulness and captures the feature of GAP-GREEDY$_{\text{ratio}}$ because of the preferences $\mathcal{L}_k$ defined for the machines. We next give the formal proof of Theorem 1.

PROOF OF THEOREM 1. Since GAP-GREEDY is truthful, it suffices to show that SM-DA-ALG is truthful as well under the assumptions stated in the theorem. For all four invariant settings, the process of the SM-DA-ALG has the following nice property: Once i and k are matched, the assignment will never be broken.

We first consider the job value invariant setting and consider the first pair, say i and k, picked by the algorithm. According to the algorithm, for every job i, its first proposed machine has the largest value v_{ik}. Because in the job value invariant setting every job is indifferent between machines and in the algorithm we always select a pair with the largest ratio, the first picked pair i and k have the largest ratio v_{ik}/c_{ik} among all pairs. Thus, job i has the highest rank in the preference $\mathcal{L}_k$ of machine k. Hence, and after i is assigned to k, the assignment will never be broken. We can then apply the same argument iteratively for all assigned pairs. The argument for the job capacity invariant case is similar.

We note that the machine capacity invariant case can be reduced to the standard many-to-one matching problem because in this case every machine has a fixed amount of slots to allocate. In the machine value invariant case, all jobs have identical preferences over machines. Hence, the algorithm runs in such a way that all jobs try to fill machines one by one, according to the increasing order of their capacities. As before, we have that every assignment will never be broken.

Given the above property, we can see that the assignment returned by the algorithm is stable and truthful. Indeed, given that the assignments are never broken, no job would want to get assigned to a machine without a real edge between them. On the other hand, since each job i makes proposals according to the decreasing order of its preference $\mathcal{L}_i$, hiding an edge can only make the assignment worse.

Next we will prove that the approximation ratio of the mechanism is 4. We denote by $\mathcal{A}^{(1)}$, $\mathcal{A}^{(2)}$ the assignments we get from GAP-GREEDY and SM-DA-ALG, respectively, and by $\mathcal{A}^{\text{opt}}$ the optimal assignment.

The expected value of the mechanism is given by $\frac{1}{2}v(\mathcal{A}^{(1)}) + \frac{1}{2}v(\mathcal{A}^{(2)})$. We want to show that

$$4 \cdot \left(\frac{1}{2}v(\mathcal{A}^{(1)}) + \frac{1}{2}v(\mathcal{A}^{(2)})\right) = 2v(\mathcal{A}^{(1)}) + 2v(\mathcal{A}^{(2)}) \geq v(\mathcal{A}^{\text{opt}}).$$

We denote by S the set of jobs i for which $v_i(\mathcal{A}^{\text{opt}}) > v_i(\mathcal{A}^{(1)})$ and $v_i(\mathcal{A}^{\text{opt}}) > v_i(\mathcal{A}^{(2)})$. Then we get

$$\sum_i v_i(\mathcal{A}^{(1)}) + \sum_i v_i(\mathcal{A}^{(2)}) \geq \sum_{i \notin S} v_i(\mathcal{A}^{\text{opt}}).$$

Further, we have $v(\mathcal{A}^{\text{opt}}) = \sum_{i\notin S} v_i(\mathcal{A}^{\text{opt}}) + \sum_{i\in S} v_i(\mathcal{A}^{\text{opt}}) = \sum_{i\notin S} v_i(\mathcal{A}^{\text{opt}}) + \sum_k v(\mathcal{A}_k^{\text{opt}} \cap S)$.

Now consider any job $i_0 \in S$; let $\mathcal{A}_{i_0}^{\text{opt}} = k$. By the definition of S, we know that in both assignments $\mathcal{A}^{(1)}$ and $\mathcal{A}^{(2)}$, job i_0 prefers machine k to its actual assignments. Therefore, for $\mathcal{A}^{(1)}$, by the greedy rule, we have $v(\mathcal{A}_k^{(1)}) \geq v_{ik} = v_i(\mathcal{A}^{\text{opt}})$ for any $i \in \mathcal{A}_k^{\text{opt}} \cap S$.

For $\mathcal{A}^{(2)}$, by the property of stability, for any $i \in \mathcal{A}_k^{\text{opt}} \cap S$, we have (i) $v_i(\mathcal{A}^{\text{opt}}) \leq c_i(\mathcal{A}^{\text{opt}})(v_{i'}(\mathcal{A}^{(2)})/c_{i'}(\mathcal{A}^{(2)}))$ for each $i' \in \mathcal{A}_k^{(2)}$ and (ii) $C_k < c(\mathcal{A}_k^{(2)}) + c_i(\mathcal{A}^{\text{opt}})$, since we cannot add i to $\mathcal{A}_k^{(2)}$. Then we get

$$v_i(\mathcal{A}^{\text{opt}}) \leq c_i(\mathcal{A}^{\text{opt}}) \frac{\sum_{i\in\mathcal{A}_k^{(2)}} v_i(\mathcal{A}^{(2)})}{\sum_{i\in\mathcal{A}_k^{(2)}} c_i(\mathcal{A}^{(2)})} = c_i(\mathcal{A}^{\text{opt}}) \frac{v(\mathcal{A}_k^{(2)})}{c(\mathcal{A}_k^{(2)})}.$$

For the considered job $i_0 \in \mathcal{A}_k^{\text{opt}} \cap S$, we can write

$$v((\mathcal{A}_k^{\text{opt}} \cap S)\backslash\{i_0\}) = \sum_{\substack{i\in\mathcal{A}_k^{\text{opt}}\cap S \\ i\neq i_0}} v_i(\mathcal{A}^{\text{opt}}) \leq c((\mathcal{A}_k^{\text{opt}} \cap S)\backslash\{i_0\}) \frac{v(\mathcal{A}_k^{(2)})}{c(\mathcal{A}_k^{(2)})} \leq v(\mathcal{A}_k^{(2)}),$$

where the last inequality follows because $c((\mathcal{A}_k^{\text{opt}} \cap S)\backslash\{i_0\}) \leq C_k - c_{i_0}(\mathcal{A}^{\text{opt}}) < c(\mathcal{A}_k^{(2)})$. Then we have

$$v(\mathcal{A}_k^{(1)}) + v(\mathcal{A}_k^{(2)}) \geq v((\mathcal{A}_k^{\text{opt}} \cap S)\backslash\{i_0\}) + v_{i_0}(\mathcal{A}^{\text{opt}}) = v(\mathcal{A}_k^{\text{opt}} \cap S).$$

Hence,

$$\sum_k v(\mathcal{A}_k^{(1)}) + \sum_k v(\mathcal{A}_k^{(2)}) \geq \sum_k v(\mathcal{A}_k^{\text{opt}} \cap S).$$

Therefore, we have

$$\begin{aligned} 2v(\mathcal{A}^{(1)}) + 2v(\mathcal{A}^{(2)}) &= \sum_i v_i(\mathcal{A}^{(1)}) + \sum_i v_i(\mathcal{A}^{(2)}) + \sum_k v(\mathcal{A}_k^{(1)}) + \sum_k v(\mathcal{A}_k^{(2)}) \\ &\geq \sum_{i\notin S} v_i(\mathcal{A}^{\text{opt}}) + \sum_k v(\mathcal{A}_k^{\text{opt}} \cap S) \\ &= v(\mathcal{A}^{\text{opt}}). \end{aligned}$$

This completes the proof. □

4. Main mechanism. In this section we use SM-DA-ALG as a critical component to describe a universally truthful mechanism for the general GAP. We first give three truthful mechanisms; then the main mechanism takes a uniform distribution on these mechanisms. Throughout this section, $\lambda > 2$ is an integer and $\mu > 0$ is a constant, both fed as parameters to the mechanisms; their values can be taken appropriately with some conditions described at the end of the analysis.

We first present two simple deterministic mechanisms, which try to assign those pairs with large values. The first one is designated to deal with the pairs that have "large" capacities with respect to the capacity of the corresponding machine.

```
GAP-MECHANISM-1($\lambda$)
1. Remove all pairs $(i, k)$ with capacity $c_{ik} < (1/\lambda)C_k$.
2. For the remaining pairs, output an assignment by the GAP-GREEDY algorithm with the restriction that
   each machine can take at most one job.
```

The following procedure divides the capacity of each machine evenly into λ slots and assigns at most one job to each slot.

```
GAP-MECHANISM-2($\lambda$)
1. Remove all pairs $(i, k)$ with capacity $c_{ik} > (1/\lambda)C_k$.
2. For the remaining pairs, output an assignment by the GAP-GREEDY algorithm with the restriction that
   each machine can take at most $\lambda$ jobs.
```

Next we will describe the key mechanism, which will apply the stable matching algorithm SM-DA-ALG as a critical component. In fact, we will use a slightly different version of SM-DA-ALG: For every machine k and a given number $C'_k < C_k$, we define as follows a *virtual capacity* constraint C'_k that provides an additional restriction on the feasible set: Given a feasible assignment $i_1, i_2, \ldots, i_l$ on machine k with preferences $i_1 \succ i_2 \succ \cdots \succ i_l$ on $\mathcal{L}_k$, we require $\sum_{j=1}^{l-1} c_{i_jk} \le C'_k$ and $\sum_{j=1}^{l} c_{i_jk} \le C_k$ (note that it is allowed that $\sum_{j=1}^{l} c_{i_jk} > C'_k$). That is, although the total capacity of the assigned job on machine k can be larger than C'_k, the removal of the least preferred one must ensure that the total capacity is within the virtual capacity. Running SM-DA-ALG with virtual capacity C'_k means that in the process of the algorithm, the assigned jobs on machine k always satisfy the virtual capacity constraint.

```
GAP-MECHANISM-3($\lambda, \mu$)
1. Remove all pairs $(i, k)$ with capacity $c_{ik} > (1/\lambda)C_k$.
2. Select each job independently at random with probability $\frac{1}{2}$ into group $T$.
   • For each job $i \in T$, let $\mathcal{L}_i$ be its preference list over machines.
   • For each machine $k$, let $\mathcal{L}_k$ be its preference list over jobs in $T$.
   • Run the SM-DA-ALG with virtual capacity $((\lambda - 1)/\lambda)C_k$ for each machine $k$.
   • Denote the generated assignment by $\mathcal{A}^T$.
3. For each machine $k$, define the threshold value $t_k = \mu \cdot (v(A_k^T)/C_k)$.
4. Let $R$ be the remaining jobs that are not selected in $T$ and set $\mathcal{A} = \varnothing$.
5. For each job $i \in R$ in a given fixed order
   • Let $k = \arg\max_k \{v_{ik} \mid c_{ik} + c(\mathcal{A}_k) \le C_k$ and $v_{ik}/c_{ik} \ge t_k\}$.
   • If $k$ defined above exists, let $\mathcal{A}_i = k$; otherwise, let $\mathcal{A}_i = \varnothing$.
6. Output $\mathcal{A}$.
```

In the mechanism, we first sample, in expectation, half of the jobs in T and then on this set T run the stable matching algorithm SM-DA-ALG. The assignment $\mathcal{A}^T$ from T gives a good estimate to the structure of the optimal solution with a high probability. We therefore use the density $v(A_k^T)/C_k$, multiplied by a given constant μ, as a threshold t_k for the jobs that can be assigned to machine k. (Note that the jobs in T are then discarded from step 4 and will not be assigned to any machine at the end of the mechanism; randomness ensures that these jobs are still willing to participate in the mechanism.) For the remaining jobs that are not in T, we explicitly remove those pairs (i, k) with a ratio less than the threshold; i.e., $v_{ik}/c_{ik} < t_k$. This ensures that the value-to-size ratios of all jobs that are finally assigned on the machine are large enough. The assignment of these jobs simulates an online greedy mechanism: When a job arrives, the mechanism assigns it to the best possible machine, given the capacity constraints on the machine.

To make the whole idea of stable matching and random sampling in GAP-MECHANISM-3 work, we need the condition that there is a constant probability of computing a reasonable estimate of the threshold for a constant fraction of the machines. This condition fails if most of the values in the optimal solution are generated by a few jobs. In such a case, at least one of the other two mechanisms is guaranteed to yield a good solution.

Our main mechanism, as described in the following claim, is a combination of the aforementioned three mechanisms.

THEOREM 2. *Consider the following mechanism* GAP-MECHANISM-MAIN(λ, μ): *Run one of the following three mechanisms* GAP-MECHANISM-1(λ), GAP-MECHANISM-2(λ), *or* GAP-MECHANISM-3(λ, μ) *with uniform probability. Then* GAP-MECHANISM-MAIN *is a universally truthful mechanism with a constant approximation ratio for the* GAP *problem.*

The proof of the above main theorem are given in the following subsections.

4.1. Truthfulness.

PROPOSITION 1. *The mechanisms* GAP-MECHANISM-1(λ) *and* GAP-MECHANISM-2(λ) *are deterministic truthful mechanisms, and* GAP-MECHANISM-3(λ, μ) *is a universally truthful mechanism.*

To prove the claim, we first argue that GAP-MECHANISM-1(λ) and GAP-MECHANISM-2(λ) are deterministic truthful mechanisms and then show that GAP-MECHANISM-3(λ, μ) is a universally truthful mechanism.

CLAIM 1. GAP-MECHANISM-1(λ) *is a truthful mechanism. Further, for any instance where all pairs (i, k) satisfy $c_{ik} \ge (1/\lambda)C_k$, its approximation ratio is at most 2λ.*

Proof. For truthfulness, reporting pairs with $c_{ik} < (1/\lambda)C_k$ will not change anything in the mechanism. Therefore, we may restrict ourselves to the setting only with pairs $c_{ik} \geq (1/\lambda)C_k$. Then by Lemma 1, it is truthful for all jobs.

Consider the optimal assignment $\mathcal{A}^{\text{opt}}$ of jobs to machines. Note that for each machine k we have $|\mathcal{A}_k^{\text{opt}}| \leq \lambda$ (we recall that $\mathcal{A}_k^{\text{opt}}$ is the set of jobs that machine k is assigned to). Let $\mathcal{A}$ be the assignment generated by our mechanism. Let us consider a pair (i, k) in $\mathcal{A}_k^{\text{opt}}$ with the highest value v_{ik}. Let $v_i(\mathcal{A})$ and $v_i(\mathcal{A}^{\text{opt}})$ be the values of the assignments of job i in $\mathcal{A}$ and $\mathcal{A}^{\text{opt}}$, respectively. In the assignment $\mathcal{A}$, there are two possibilities for job i:

- either $v_i(\mathcal{A}) \geq v_i(\mathcal{A}^{\text{opt}})$
- or $v_i(\mathcal{A}) < v_i(\mathcal{A}^{\text{opt}})$. For the latter case, by the greedy rule, machine k should be assigned in $\mathcal{A}$ to a job with a value greater than or equal to v_{ik}.

Hence, we have

$$2v(\mathcal{A}) = \sum_k \sum_{i\in\mathcal{A}_k} v_{ik} + \sum_k v(\mathcal{A}_k) \geq \sum_k \max_{i\in\mathcal{A}_k^{\text{opt}}} v_{ik} \geq \frac{1}{\lambda} v(\mathcal{A}^{\text{opt}}).$$

Therefore, $2\lambda \cdot v(\mathcal{A}) \geq v(\mathcal{A}^{\text{opt}})$ and the claim follows. □

Claim 2. gap-mechanism-2(λ) *is a truthful mechanism.*

Proof. Observe that reporting pairs with $c_{ik} > (1/\lambda)C_k$ will not change anything in the mechanism. Therefore, we may restrict ourselves to the setting only with pairs $c_{ik} \leq (1/\lambda)C_k$. Note that the value λ, i.e., the number of jobs that each machine can take, is independent of the bids of all jobs. Thus, Lemma 1 implies that all jobs are truthful. □

Claim 3. gap-mechanism-3(λ, μ) *is a universally truthful mechanism.*

Proof. Although gap-mechanism-3(λ, μ) is not deterministic, it does not use any reported information from jobs to generate the testing group T. Once T is generated, the mechanism performs deterministically. Therefore, it runs on a distribution over deterministic mechanisms. Further, notice that every job in the testing group T derives utility zero. Therefore, given the fixed set T, none of the jobs in T can benefit from reporting untruthfully. On the other hand, for each remaining job $i \in R$, reporting truthfully maximizes its utility since jobs are processed one by one in a fixed given order and every time we search for the best possible assignment for the processed job. (Note that the threshold value t_k is computed according to the outcome of $\mathcal{A}^T$ from the testing group T; thus it is independent of the bids of any job in R.) □

Since all three mechanisms are (universally) truthful, gap-mechanism-main is a universally truthful mechanism as well.

4.2. Stability.

Recall that we say an assignment $\mathcal{A}$ is *stable* if it does not contain any blocking pair i and k, where i strictly prefers k to $\mathcal{A}_i$ and k can pick a strictly better subset of jobs in the collection $\mathcal{A}_k \cup \{i\}$.

Proposition 2. *The assignment* $\mathcal{A}^T$ *returned by the mechanism* gap-mechanism-3(λ, μ) *is stable with respect to all jobs in* T *and all machines with virtual capacity* $((\lambda - 1)/\lambda)C_k$.

Proof. Assume the contrary, that i and k form a blocking pair, where i strictly prefers k to $\mathcal{A}_i$ and k can get a better assignment from jobs in $\mathcal{A}_k \cup \{i\}$ (with respect to the virtual capacity constraint). Since every job proposes according to its preference list $\mathcal{L}_i$, we know that at a certain step job i has proposed to k during the execution of the algorithm. Then either i got rejected right away or got assigned but rejected later (because of proposals from other jobs). For both cases, from that moment until the end of the algorithm, because the capacity of all considered pairs is at most $(1/\lambda)C_k$, we know that (i) the total capacity of the assigned jobs on k is strictly larger than the virtual capacity $((\lambda - 1)/\lambda)C_k$ (and, of course, less than or equal to the real capacity C_k) and (ii) k prefers every assigned job to i. These two facts imply that k cannot get a better assignment from $\mathcal{A}_k \cup \{i\}$, a contradiction. □

We note that the stability result established in the above claim is only with respect to the virtual capacity constraint, without which the sm-da-alg may not compute a stable assignment. Stability plays a critical role in the approximation analysis of the main mechanism. In particular, in a solution generated by the stable matching algorithm, if a job i is not assigned to a machine k because the property of stability and the definitions of the preferences of i and k, we know that the value v_{ik} can be "charged" to either the machine k or the job i: if the machine k is fully occupied, then the total value of the machine is large enough since all assigned jobs have a higher rank in the preference; if not, the job i must be assigned to a machine with a larger value.

4.3. Approximation ratio. We first establish the following technical lemma.

Lemma 2. *Let $\delta_1 \le 1/36$ and $a_1 \ge a_2 \ge \cdots \ge a_l$ be positive real numbers, such that the sum $a = a_1 + a_2 + \cdots + a_l$ satisfies $\delta_1 a > a_1$. We select each number $a_1, \ldots, a_l$ independently at random with probability 1/2 each and let b to be the random variable equal to the sum of the selected numbers. Then*

$$\mathbf{Pr}\left(\frac{1}{3}a < b < \frac{2}{3}a\right) \ge \frac{3}{4}.$$

Proof. Let us consider for each $1 \le j \le l$, the random variables X_j with $\mathbf{Pr}(X_j = 0) = \mathbf{Pr}(X_j = a_j) = 0.5$. Let $X = \sum_{i=j}^{l} X_j$; we have $b = X$. Then the expectation $\mathbf{E}(X) = a/2$ and variance

$$\sigma^2 = \mathbf{Var}(X) = \sum_{j=1}^{l} \mathbf{Var}(X_j) = \frac{1}{4}\sum_{j=1}^{l} a_j^2.$$

Applying Chebyshev's inequality, we get

$$\mathbf{Pr}\left(\left|X - \frac{a}{2}\right| \ge 2\sigma\right) \le \frac{1}{4}.$$

In order to conclude the proof of the lemma, it remains to show that $2\sigma \le a/6$, which is equivalent to showing that

$$36 \cdot (a_1^2 + a_2^2 + \cdots + a_l^2) \le (a_1 + a_2 + \cdots + a_l)^2.$$

Since $a \ge a_1/\delta_1 \ge a_j/\delta_1$ for every $1 \le j \le l$, we have

$$(a_1 + a_2 + \cdots + a_l)^2 = \sum_{j=1}^{l} a_j \cdot a \ge \sum_{j=1}^{l} \frac{a_j^2}{\delta_1} \ge 36 \sum_{j=1}^{l} a_j^2.$$

Therefore, the lemma follows. □

Let *OPT* be the optimal allocation. We denote by $\mathcal{A}^{(1)}$, $\mathcal{A}^{(2)}$, and $\mathcal{A}^{(3)}$ the allocations obtained in the mechanisms GAP-MECHANISM-1(λ), GAP-MECHANISM-2(λ), and GAP-MECHANISM-3(λ, μ), respectively. The expected value of the mechanism GAP-MECHANISM-MAIN is therefore $\frac{1}{3}(v(\mathcal{A}^{(1)}) + v(\mathcal{A}^{(2)}) + v(\mathcal{A}^{(3)}))$.

We divide all pairs into two groups: a group containing the "large" pairs (i, k) with $c_{ik} > C_k/\lambda$ and a group containing the remaining "small" pairs. Let OPT^{small} and OPT^{large} denote the optimal allocations for the settings restricted on the pairs only in the "small" group and "large" group, respectively. Then we have

$$v(OPT) \le v(OPT^{\text{small}}) + v(OPT^{\text{large}}).$$

By Claim 1, we know that the total value derived from GAP-MECHANISM-1(λ) satisfies $v(\mathcal{A}^{(1)}) \ge v(OPT^{\text{large}})/(2\lambda)$. It remains to handle OPT^{small}. Instead of dealing with OPT^{small} directly, we consider the allocation $\mathcal{A}^*$ defined according to step 4 of GAP-MECHANISM-3(λ, μ) for *all* jobs with respect to the pairs with capacities less than or equal to C_k/λ. Formally, $\mathcal{A}^*$ is defined as follows:

1. Remove all pairs (i, k) with capacity $c_{ik} > C_k/\lambda$.
2. For each job i, let $\mathcal{L}_i$ be his preference list over machines.
3. For each machine k, let $\mathcal{L}_k$ be its preference list over jobs.
4. Run the SM-DA-ALG with virtual capacity $((\lambda - 1)/\lambda)C_k$ for each machine k.
5. Denote the generated assignment by $\mathcal{A}^*$.

Similarly to the proof of Proposition 2, $\mathcal{A}^*$ is a stable assignment with respect to virtual capacity $((\lambda - 1)/\lambda)C_k$. The following claim implies that $v(\mathcal{A}^*)$ is a constant approximation to $v(OPT^{\text{small}})$.

Claim 4. $(2 + 1/(\lambda - 1)) \cdot v(\mathcal{A}^*) \ge v(OPT^{\text{small}})$.

Proof. To simplify the notations, let $\mathcal{A}^{\text{opt}}$ denote the assignment OPT^{small}. Since $\mathcal{A}^*$ is a stable assignment, for each pair $(i, k) \in \mathcal{A}^{\text{opt}} \setminus \mathcal{A}^*$, we have either (i) $v_i(\mathcal{A}^{\text{opt}}) = v_{ik} \le v_i(\mathcal{A}^*)$ or (ii) $v_{ik}/c_{ik} \le v_{i'k}/c_{i'k}$ for every $i' \in \mathcal{A}_k^*$, and $c(\mathcal{A}_k^*) \ge C_k((\lambda - 1)/\lambda)$. The latter implies that $v_{ik}/c_{ik} \le v(\mathcal{A}_k^*)/c(\mathcal{A}_k^*) \le (v(\mathcal{A}_k^*)/C_k)\lambda/(\lambda - 1)$.

In the assignment $\mathcal{A}^{\text{opt}}$, we denote by X the set of jobs i such that $v_i(\mathcal{A}^{\text{opt}}) \le v_i(\mathcal{A}^*)$ and by Y the remaining jobs. Then we have

$$\begin{aligned} v(\mathcal{A}^{\text{opt}}) &= \sum_{i\in X} v_i(\mathcal{A}^{\text{opt}}) + \sum_k v(\mathcal{A}_k^{\text{opt}} \cap Y) \\ &\le \sum_{i\in X} v_i(\mathcal{A}^*) + \frac{\lambda}{\lambda-1}\sum_k \frac{v(\mathcal{A}_k^*)}{C_k} \sum_{i\in \mathcal{A}_k^{\text{opt}} \cap\, Y} c_{ik} \\ &\le v(\mathcal{A}^*) + \frac{\lambda}{\lambda-1}\sum_k v(\mathcal{A}_k^*) \\ &= \left(1+\frac{\lambda}{\lambda-1}\right) v(\mathcal{A}^*). \end{aligned}$$

Hence, we get $v(\mathcal{A}^{\text{opt}}) \le (2 + 1/(\lambda-1))v(\mathcal{A}^*)$. □

In addition, the stable assignment $\mathcal{A}^*$ enjoys the following useful properties.

Claim 5. *For each job $i \in T$ and machine k, we have $v_i(\mathcal{A}^T) \ge v_i(\mathcal{A}^*)$ and $v(\mathcal{A}_k^T) \le (\lambda/(\lambda-1))v(\mathcal{A}_k^*)$.*

Proof. Our proof of the claim exploits similar ideas from the standard one-to-one stable matching. Note that both assignments $\mathcal{A}^*$ and $\mathcal{A}^T$ are generated by the same algorithm, but $\mathcal{A}^*$ is on a bigger set of jobs. Similar to the proof of Proposition 2, the virtual capacity $((\lambda-1)/\lambda)C_k$ and that all considered pairs have capacities $c_{ik} \le C_k/\lambda$ ensure that at any moment of the algorithm, the set of assigned jobs to any machine is simply the best possible subset selected from the set of all jobs that have ever proposed to the machine up to that moment. In particular, at the end of the algorithm, every machine k will have its most preferred feasible set (given the virtual capacity constraint) chosen from the set of all proposals that k has ever received.

For each job i, denote by $best(i)$ the best possible assignment for i taken over all stable assignments. In what follows we argue that SM-DA-ALG assigns every job i to $best(i)$. Assume otherwise, since every job i makes proposals in decreasing order of $\mathcal{L}_i$, in the course of the algorithm, there must be a job i_0 that has been rejected by $best(i_0)$. We consider the first moment in the algorithm when a job i_0 is rejected from the machine $k = best(i_0)$. At that moment, let S be the set of jobs that are assigned to k. By the above argument, we know that for any $i \in S$, k prefers i to i_0. Next consider a stable assignment $\mathcal{A}'$, where i_0 is assigned to k (by the definition of $best(i_0)$, such assignment exists). Then there exists $i' \in S$ that is not assigned to k in $\mathcal{A}'$; let $\mathcal{A}'_{i'} = k' (\neq k)$. By the definition of $best(i')$, we know that i' weakly prefers $best(i')$ to k'. Since $\mathcal{A}'$ is a stable assignment, i' and k do not form a blocking pair, which implies that i' prefers k' to k. Now we recall that i is the first job rejected from the machine $best(i)$ in SM-DA-ALG; we know that at that moment, i' has not been rejected by $best(i')$. Hence, i' weakly prefers k to $best(i')$. Therefore, on the preference list $\mathcal{L}_{i'}$, we have the following preference order: $k \succeq best(i') \succeq k' \succ k$, which gives a contradiction. Therefore, each job i is assigned to $best(i)$ in the output of SM-DA-ALG.

Note that the above argument does not rely on any specific order of proposals (especially the one defined in Step 1 of SM-DA-ALG). That is, an arbitrary order of proposals, as long as every job proposes to the most preferred machine that it has not yet proposed to, will lead to the same assignment by matching every job i to $best(i)$. Now given that the order in which jobs propose does not matter for the outcome of the algorithm, in the computation of $\mathcal{A}^*$, we may consider the order where jobs in T are settled first and then add the remaining jobs. Thus, for each job $i \in T$ the assignment $\mathcal{A}_i^*$ can only be worse than that in $\mathcal{A}_i^T$, i.e., $v_i(\mathcal{A}^T) \ge v_i(\mathcal{A}^*)$.

In addition, we know that the set of jobs proposed to each machine k in $\mathcal{A}^T$ is a subset in $\mathcal{A}^*$. Thus, by the above argument, the average ratio v_{ik}/c_{ik} of the assigned jobs in $\mathcal{A}_k^*$ is larger than or equal to that in $\mathcal{A}_k^T$. Notice that in the assignment $\mathcal{A}_k^T$, the total capacity used is at most C_k. For the assignment $\mathcal{A}_k^*$, either we use more than $(1-1/\lambda)C_k$ capacity or otherwise (which implies that $\mathcal{A}_k^T \subseteq \mathcal{A}_k^*$). For either case, we have $v(\mathcal{A}_k^T) \le (\lambda/(\lambda-1))v(\mathcal{A}_k^*)$, which completes the proof. □

For the assignment $\mathcal{A}^*$, let us consider a restricted assignment $\tilde{\mathcal{A}}^*$, where we only keep the first λ largest value jobs of $\mathcal{A}^*$ for each machine k. Note that $\tilde{\mathcal{A}}^*$ is a feasible assignment for the setting in GAP-MECHANISM-2 as well. We have the following claim.

Claim 6. $v(\mathcal{A}^{(2)}) \ge \frac{1}{2} v(\tilde{\mathcal{A}}^*)$.

PROOF. We observe that both $\mathcal{A}^{(2)}$ and $\tilde{\mathcal{A}}^*$ use the same set of feasible pairs; i.e., $c_{ik} \le C_k/\lambda$. We may split each machine into λ equal slots, where each slot can take only one job. Thus, $\tilde{\mathcal{A}}^*$ may be viewed as a matching and $\mathcal{A}^{(2)}$ as a maximal matching for the new setting between jobs and slots. Since a maximal matching is a 2-approximation to the maximum matching, which is an upper bound on $v(\tilde{\mathcal{A}}^*)$, we have $2 \cdot v(\mathcal{A}^{(2)}) \ge v(\tilde{\mathcal{A}}^*)$. □

Consider the partition of jobs into the two sets T and R in the GAP-MECHANISM-3(λ, μ). For a fixed machine k, we consider the jobs $\mathcal{A}_k^* \cap T$ and $\mathcal{A}_k^* \cap R$. By the definition of T and R, every job in $\mathcal{A}_k^*$ will be placed in T with probability 1/2. Recall that $\tilde{\mathcal{A}}_k^*$ keeps the top λ value jobs in $\mathcal{A}_k^*$. Depending on the relation between $v(\tilde{\mathcal{A}}_k^*)$ and $v(\mathcal{A}_k^*)$, at least one of the following alternatives has to be true (where the latter follows from Lemma 2):

1. $v(\tilde{\mathcal{A}}_k^*) \ge \delta_1 v(\mathcal{A}_k^*)$,
2. $\mathbf{Pr}_T(\frac{1}{3}v(\mathcal{A}_k^*) \le v(\mathcal{A}_k^* \cap T) \le \frac{2}{3}v(\mathcal{A}_k^*)) \ge \frac{3}{4}$.

Intuitively, if there are many similar jobs in $\mathcal{A}_k^*$, then with a high probability the total value of jobs in $\mathcal{A}_k^* \cap T$ and in $\mathcal{A}_k^* \cap R$ will be close to each other. On the other hand, if the former does not hold, the GAP-MECHANISM-2 gives us a good value for machine k. Let us denote the set of the machines satisfying the condition $v(\tilde{\mathcal{A}}_k^*) \ge \delta_1 v(\mathcal{A}_k^*)$ by G and the remaining set of machines by H. We note that H and G do not depend upon the particular draw of T.

Thus, by Claim 6, GAP-MECHANISM-2 guarantees that we get a constant fraction of the total value of $\mathcal{A}^*$ taken over all machines in G. For every machine $k \in H$, Lemma 2 ensures that

$$\frac{1}{3}v(\mathcal{A}_k^*) \le v(\mathcal{A}_k^* \cap T) \le \frac{2}{3}v(\mathcal{A}_k^*) \tag{1}$$

occurs with a probability of at least 3/4. Consider the collection $\mathcal{D}_k^*$ of "regular" realizations of T for which (1) holds; slightly abusing the notations we let $\mathcal{D}_k^*$ to denote as well the distribution of T drawn from such collection of sets.

The desired constant approximation follows from the following arguments (to be shown later):

- Due to Claim 6, for each machine $k \in G$

$$v(\mathcal{A}^{(2)}) \ge \frac{1}{2} \cdot v(\tilde{\mathcal{A}}^*) \ge \frac{\delta_1}{2} \cdot \sum_{k \in G} v(\mathcal{A}_k^*).$$

- For each fixed machine $k \in H$ and a random draw $T \in \mathcal{D}_k^*$, we show that either the total value of jobs allocated on k in $\mathcal{A}^{(3)}$ is comparable to $v(\mathcal{A}_k^T)$ or the total value of jobs allocated to k in $\mathcal{A}^*$ is comparable to $v(\mathcal{A}_k^T)$. That is,

$$v(\mathcal{A}_k^{(3)}) \ge \delta_2 \cdot v(\mathcal{A}_k^T) \ge \delta_4 \cdot v(\mathcal{A}_k^T)$$

and

$$\sum_{i \in \mathcal{A}_k^* \cap R} v_i(\mathcal{A}^{(3)}) \ge \delta_3 \cdot v(\mathcal{A}_k^*) \ge \delta_4 \cdot v(\mathcal{A}_k^T)$$

for some constants $\delta_2, \delta_3, \delta_4 > 0$ depending on λ and μ.

- Using the previous lower bounds, we estimate the expected value of GAP-MECHANISM-3

$$\mathbf{E}_T[v(\mathcal{A}^{(3)})] \ge \delta_5 \cdot v(\mathcal{A}^*) - \frac{\delta_4 \lambda}{2(\lambda - 1)} \cdot \sum_{k \in G} v(\mathcal{A}_k^*),$$

for a constant $\delta_5 > 0$ depending on μ; λ; and $\delta_2, \delta_3, \delta_4$.

- We finally derive a lower bound on the expected value of GAP-MECHANISM-MAIN(λ, μ), for a proper choice of the parameters μ; λ; and $\delta_1, \delta_2, \delta_3, \delta_4, \delta_5$:

$$\frac{1}{3} \cdot (v(\mathcal{A}^{(1)}) + v(\mathcal{A}^{(2)}) + \mathbf{E}_T[v(\mathcal{A}^{(3)})]) \ge \delta_6 \cdot v(OPT),$$

for a certain positive constant δ_6.

We next give a detailed proof of these arguments. We first estimate the value obtained by GAP-MECHANISM-3(λ, μ) over machines from H. For each machine $k \in H$, there are following two cases.

1. In GAP-MECHANISM-3(λ, μ), we have rejected a job at machine k due to the capacity constraint, that is, we have rejected a pair (i, k) because of $c_{ik} + c(\mathcal{A}_k) > C_k$. Since GAP-MECHANISM-3($\lambda, \mu$) only considers pairs with capacities less than or equal to C_k/λ, in that case almost all the capacity of k is used. That is,

$\sum_{i\in\mathcal{A}_k^{(3)}} c_{ik} \geq C_k(1-1/\lambda)$. Further, by the rule of the mechanism, for every $i \in \mathcal{A}_k^{(3)}$, we have $v_{ik}/c_{ik} \geq t_k$; i.e., $v_{ik} \geq c_{ik}((\mu \cdot v(\mathcal{A}_k^T))/C_k)$. Let $\delta_2 = \mu(1-1/\lambda) > 0$; therefore, we have

$$v(\mathcal{A}_k^{(3)}) \geq \sum_{i\in\mathcal{A}_k^{(3)}} c_{ik} \frac{\mu \cdot v(\mathcal{A}_k^T)}{C_k} \geq \mu\left(1-\frac{1}{\lambda}\right) v(\mathcal{A}_k^T) = \delta_2 \cdot v(\mathcal{A}_k^T). \tag{2}$$

2. No job who has passed the threshold t_k has been rejected from machine k in GAP-MECHANISM-3(λ,μ). Denote by R^+ the set of jobs in R who have passed the threshold for the corresponding machine in assignment $\mathcal{A}^*$; denote by R^- the remaining jobs in R. Therefore, all jobs in $\mathcal{A}_k^* \cap R^+$ get an assignment at least as good as in $\mathcal{A}^*$. Then for any $i \in \mathcal{A}_k^* \cap R^+$, we have $v_i(\mathcal{A}^{(3)}) \geq v_i(\mathcal{A}^*)$.

Further, for each job $i \in \mathcal{A}_k^* \cap R^-$, we have $v_{ik} < c_{ik} t_k = \mu((v(\mathcal{A}_k^T)c_{ik})/C_k)$. Therefore,

$$v(\mathcal{A}_k^* \cap R^-) \leq \mu \cdot v(\mathcal{A}_k^T) \frac{\sum_{i\in\mathcal{A}_k^*\cap R^-} c_{ik}}{C_k} \leq \mu \cdot v(\mathcal{A}_k^T) \leq \mu \frac{\lambda}{\lambda-1} v(\mathcal{A}_k^*)$$

where the last inequality follows from Claim 5. Thus,

$$\sum_{i\in\mathcal{A}_k^*\cap R} v_i(\mathcal{A}^{(3)}) \geq \sum_{i\in\mathcal{A}_k^*\cap R^+} v_i(\mathcal{A}^{(3)}) \geq v(\mathcal{A}_k^* \cap R) - \mu\frac{\lambda}{\lambda-1} v(\mathcal{A}_k^*).$$

Let $\delta_3 = \frac{1}{3} - \mu\lambda/(\lambda-1)$. Taking appropriate values for λ and μ, we can ensure that $\delta_3 > 0$. For any $T \in \mathcal{D}_k^*$, we have $v(\mathcal{A}_k^* \cap R) \geq \frac{1}{3} v(\mathcal{A}_k^*)$; then we get

$$\sum_{i\in\mathcal{A}_k^*\cap R} v_i(\mathcal{A}^{(3)}) \geq \left(\frac{1}{3} - \mu\frac{\lambda}{\lambda-1}\right) v(\mathcal{A}_k^*) = \delta_3 \cdot v(\mathcal{A}_k^*). \tag{3}$$

Next we estimate the expectation of $v(\mathcal{A}^{(3)})$. Let $\mathcal{D}$ denote the distribution of the mechanism GAP-MECHANISM-3(λ,μ) to generate T.

$$\begin{aligned}
\mathbf{E}_{T\sim\mathcal{D}}[2\cdot v(\mathcal{A}^{(3)})] &= \mathbf{E}_{T\sim\mathcal{D}}\left[\sum_k v(\mathcal{A}_k^{(3)}) + \sum_k \sum_{i\in\mathcal{A}_k^*\cap R} v_i(\mathcal{A}^{(3)})\right] \\
&\geq \sum_{k\in H} \mathbf{E}_{T\sim\mathcal{D}}\left[v(\mathcal{A}_k^{(3)}) + \sum_{i\in\mathcal{A}_k^*\cap R} v_i(\mathcal{A}^{(3)})\right] \\
&\geq \sum_{k\in H} \mathbf{Pr}(T\in\mathcal{D}_k^*) \cdot \mathbf{E}_{T\sim\mathcal{D}_k^*}\left[v(\mathcal{A}_k^{(3)}) + \sum_{i\in\mathcal{A}_k^*\cap R} v_i(\mathcal{A}^{(3)})\right].
\end{aligned}$$

The last inequality follows because for any random variable $\xi(t)$, one has

$$\mathbf{E}_x[\xi(x)] = \mathbf{Pr}(x \mid Z) \cdot \mathbf{E}_{x:x\in Z}[\xi(x)] + \mathbf{Pr}(x \mid \bar{Z}) \cdot \mathbf{E}_{x:x\in\bar{Z}}[\xi(x)].$$

We continue the argument; by applying either (2) or (3), we have

$$\begin{aligned}
\mathbf{E}_{T\sim\mathcal{D}}[2\cdot v(\mathcal{A}^{(3)})] &\geq \sum_{k\in H} \mathbf{Pr}(T\in\mathcal{D}_k^*) \cdot \mathbf{E}_{T\sim\mathcal{D}_k^*}[\min(\delta_2 \cdot v(\mathcal{A}_k^T), \delta_3 \cdot v(\mathcal{A}_k^*))] \\
&\geq \min\left(\delta_2, \frac{(\lambda-1)\delta_3}{\lambda}\right) \cdot \sum_{k\in H} \mathbf{Pr}(T\in\mathcal{D}_k^*) \cdot \mathbf{E}_{T\sim\mathcal{D}_k^*}[v(\mathcal{A}_k^T)]
\end{aligned}$$

where the last inequality follows from Claim 5. Let $\delta_4 = \min(\delta_2, (\lambda-1)\delta_3/\lambda) > 0$.

Recall that

$$\mathbf{E}_{T\sim\mathcal{D}}[v(\mathcal{A}_k^T)] = \mathbf{Pr}(T\in\mathcal{D}_k^*) \cdot \mathbf{E}_{T\sim\mathcal{D}_k^*}[v(\mathcal{A}_k^T)] + \mathbf{Pr}(T\notin\mathcal{D}_k^*) \cdot \mathbf{E}_{T\sim\mathcal{D}\setminus\mathcal{D}_k^*}[v(\mathcal{A}_k^T)].$$

Since $\mathbf{Pr}(T\notin\mathcal{D}_k^*) \leq \frac{1}{4}$, by Claim 5, which says that $v(\mathcal{A}_k^T) \leq (\lambda/(\lambda-1))v(\mathcal{A}_k^*)$, for each $k \in H$ we get

$$\mathbf{Pr}(T\in\mathcal{D}_k^*) \cdot \mathbf{E}_{T\sim\mathcal{D}_k^*}[v(\mathcal{A}_k^T)] \geq \mathbf{E}_{T\sim\mathcal{D}}[v(\mathcal{A}_k^T)] - \frac{1}{4}\frac{\lambda}{\lambda-1} v(\mathcal{A}_k^*). \tag{4}$$

We continue our lower bound on $\mathbf{E}_{T\sim\mathcal{D}}[2\cdot v(\mathcal{A}^{(3)})]$; by applying (4), we have

$$\begin{aligned}\mathbf{E}_{T\sim\mathcal{D}}[2\cdot v(\mathcal{A}^{(3)})] &\geq \delta_4\cdot\sum_{k\in H}\left(\mathbf{E}_{T\sim\mathcal{D}}[v(\mathcal{A}_k^T)]-\frac{\lambda}{4(\lambda-1)}v(\mathcal{A}_k^*)\right)\\ &\geq \delta_4\cdot\sum_{k\in H}\mathbf{E}_{T\sim\mathcal{D}}[v(\mathcal{A}_k^T)]-\delta_4\cdot\frac{\lambda}{4(\lambda-1)}v(\mathcal{A}^*)\\ &\geq \delta_4\cdot\sum_{k}\mathbf{E}_{T\sim\mathcal{D}}[v(\mathcal{A}_k^T)]-\delta_4\cdot\frac{\lambda}{\lambda-1}\sum_{k\in G}v(\mathcal{A}_k^*)-\delta_4\cdot\frac{\lambda}{4(\lambda-1)}v(\mathcal{A}^*).\end{aligned}$$

By using the first property of Claim 5, we have

$$\sum_k\mathbf{E}_{T\sim\mathcal{D}}[v(\mathcal{A}_k^T)]=\mathbf{E}_{T\sim\mathcal{D}}[v(\mathcal{A}^T)]=\mathbf{E}_{T\sim\mathcal{D}}\sum_{i\in T}v_i(\mathcal{A}^T)\geq\mathbf{E}_{T\sim\mathcal{D}}\sum_{i\in T}v_i(\mathcal{A}^*)=\frac{1}{2}\cdot v(\mathcal{A}^*).$$

Then we have the following lower bound:

$$\mathbf{E}_{T\sim\mathcal{D}}[v(\mathcal{A}^{(3)})]\geq\delta_4\cdot\left(\frac{1}{4}-\frac{\lambda}{(8(\lambda-1))}\right)v(\mathcal{A}^*)-\frac{\delta_4\lambda}{2(\lambda-1)}\cdot\sum_{k\in G}v(\mathcal{A}_k^*).\qquad(5)$$

Let $\delta_5=\delta_4\cdot(\frac{1}{4}-\lambda/(8(\lambda-1)))$. Taking an appropriate value for λ, we can ensure that $\delta_4,\delta_5>0$.

Recall the definition of group G and what we already got for $v(\mathcal{A}^{(2)})$ in Claim 6:

$$v(\mathcal{A}^{(2)})\geq\frac{1}{2}\cdot v(\tilde{\mathcal{A}}^*)\geq\frac{\delta_1}{2}\cdot\sum_{k\in G}v(\mathcal{A}_k^*).$$

Taking appropriate values for δ_1 and δ_4 (such that $\delta_1/2\geq(\delta_4\lambda)/(2(\lambda-1))$), we have

$$\mathbf{E}_{T\sim\mathcal{D}}[v(\mathcal{A}^{(3)})]+v(\mathcal{A}^{(2)})\geq\delta_5\cdot v(\mathcal{A}^*).$$

Finally, the expected value of the GAP-MECHANISM-MAIN(λ,μ) satisfies

$$\begin{aligned}\frac{1}{3}\cdot(v(\mathcal{A}^{(1)})+v(\mathcal{A}^{(2)})+\mathbf{E}_{T\sim\mathcal{D}}[v(\mathcal{A}^{(3)})])&\geq\frac{1}{3}\cdot\left(\frac{1}{2\lambda}v(OPT^{\text{large}})+\frac{\delta_5}{2+1/(\lambda-1)}\cdot v(OPT^{\text{small}})\right)\\ &\geq\frac{1}{3}\cdot\min\left(\frac{1}{2\lambda},\frac{\delta_5}{2+1/(\lambda-1)}\right)\cdot v(OPT).\end{aligned}$$

By choosing, e.g., $\lambda=3$, $\mu=\frac{1}{6}$, and $\delta_1=\frac{1}{36}$, it can be seen that all conditions for $\delta_2,\delta_3,\delta_4, and \delta_5$ are satisfied. Hence, GAP-MECHANISM-MAIN gives a constant approximation. This completes the proof of Theorem 2.

5. Concluding remarks.

We give a constant approximation universal truthful mechanism for the generalized assignment problem (GAP). Our mechanism, from a high level viewpoint, is based on and extensively exploits the ideas of stable matching and random sampling. The fairness condition captured by stable matching enables us to manage self-interested behavior of the participating agents and to bound efficiency loss for unmatched pairs. We believe that the idea of stable matching may find applications in other mechanism design problems.

In the generalized assignment problem, one side of the market has knapsack constraints, which is a generalization of the original many-to-one matching model. Specifically, the original deferred acceptance algorithm may not compute a stable assignment for GAP.[3] This illustrates a significant difference between GAP and the original model, where the deferred acceptance algorithm always computes a stable outcome. It is an interesting direction for future work to explore different aspects of stable matchings in GAP, say, existence, efficient computation, solution structure, and economic properties (e.g., incentives).

[3] The main reason of it is that because of sizes and capacities, the preference structure may not satisfy the *substitutability* condition (Roth and Sotomayor [29]). For instance, a machine has capacity of 10 and there are four jobs of sizes 7, 5, 4, 3 and with values 10, 5, 5, 1. When the whole set is available, the machine prefers the first and last job; however, if the first job is removed, the machine prefers the second and third job and the last job is not preferred anymore. This is a major difference between our model with the knapsack constraints and those classic matching models, e.g., Roth and Sotomayor [29], in which substitutability is a critical assumption.

References

[1] Alon N, Feldman M, Procaccia A, Tennenholtz M (2010) Strategyproof approximation of the minimax on networks. *Math. Oper. Res.* 35(3):513–526.
[2] Alon N, Fischer F, Procaccia A, Tennenholtz M (2011) Sum of Us: Strategyproof selection from the selectors. *Conf. Theoret. Aspects of Rationality and Knowledge*, 101–110.
[3] Arrow K (1951) *Social Choice and Individual Values* (John Wiley & Sons, New York).
[4] Ashlagi I, Fischer F, Kash I, Procaccia A (2010) Mix and match. *ACM Conf. Electronic Commerce* (ACM, New York), 305–314.
[5] Baiou M, Balinski M (2002) Erratum: The stable allocation (or ordinal transportation) problem. *Math. Oper. Res.* 27(4):662–680.
[6] Bei X, Chen N, Gravin N, Lu P (2012) Budget feasible mechanism design: From prior-free to Bayesian. *ACM Sympos. Theory Comput.* (ACM, New York), 449–458.
[7] Bochet O, Ilkilic R, Moulin H, Sethuraman J (2012) Balancing supply and demand under bilateral constraints. *Theoret. Econom.* 7(3):395–423
[8] Caragiannis I, Filos-Ratsikas A, Procaccia A (2011) An improved 2-agent kidney exchange mechanism. *Workshop Internet Network Econom.* 37–48.
[9] Chakrabarty D, Goel G (2008) On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and GAP. *IEEE Sympos. Foundations Comput. Sci.* (IEEE, Piscataway, NJ), 687–696.
[10] Chekuri C, Khanna S (2005) A polynomial time approximation scheme for the multiple Knapsack problem. *SIAM J. Comput.* 35(3):713–728.
[11] Chiesa A, Micali S, Zhu Z (2012) Mechanism design with approximate valuations. *Innovations Theoret. Comput. Sci. Conf.* (ACM, New York), 34–38.
[12] Clarke E (1971) Multipart pricing of public goods. *Public Choice* 11(1):17–33.
[13] Dean B, Goemans M, Immorlica N (2006) The unsplittable stable marriage problem. *Fourth IFIP Conf. Theoret. Comput. Sci.*, 65–75.
[14] Dobzinski S (2007) Two randomized mechanisms for combinatorial auctions. *Workshop on Approximation Algorithms Combin. Optim. Problems*, 89–103.
[15] Dughmi S, Ghosh A (2010) Truthful assignment without money. *ACM Conf. Electronic Commerce* (ACM, New York), 325–334.
[16] Feige U, Tennenholtz M (2011) Mechanism design with uncertain inputs: (To err is human, to forgive divine). *ACM Sympos. Theory Comput.* (ACM, New York), 549–558.
[17] Feige U, Vondrak J (2006) Approximation algorithms for allocation problems: Improving the factor of $1 - 1/e$. *IEEE Sympos. Foundations Comput. Sci.* (IEEE, Piscataway, NJ), 667–676.
[18] Fleischer L, Goemans M, Mirrokni V, Sviridenko M (2006) Tight approximation algorithms for maximum general assignment problems. *ACM-SIAM Sympos. Discrete Algorithms* (SIAM, Philadelphia), 611–620.
[19] Gale D, Shapley LS (1962) College admissions and the stability of marriage. *Amer. Math. Monthly* 69(1):9–15.
[20] Gibbard A (1973) Manipulation of voting schemes: A general result. *Econometrica* 41(4):211–215.
[21] Goldberg A, Hartline J, Karlin A, Saks M, Wright A (2006) Competitive auctions. *Games Econom. Behav.* 55(2):242–269.
[22] Groves T (1973) Incentives in teams. *Econometrica* 41(4):617–631.
[23] Hartline J, Lucier B (2010) Bayesian algorithmic mechanism design. *ACM Sympos. Theory Comput.* (ACM, New York), 301–310.
[24] Hartline J, Roughgarden T (2008) Optimal mechanism design and money burning. *ACM Sympos. Theory Comput.* (ACM, New York), 75–84.
[25] Lu P, Sun X, Wang Y, Zhu Z (2010) Asymptotically optimal strategy-proof mechanisms for two-facility games. *ACM Conf. Electronic Commerce* (ACM, New York), 315–324.
[26] Myerson R (1981) Optimal auction design. *Math. Oper. Res.* 6(1):58–73.
[27] Nisan N, Ronen A (1999) Algorithmic mechanism design. *ACM Sympos. Theory Comput.* (ACM, New York), 129–140.
[28] Procaccia A, Tennenholtz M (2009) Approximate mechanism design without money. *ACM Conf. Electronic Commerce* (ACM, New York), 177–186.
[29] Roth A, Sotomayor M (1992) *Two-Sided Matching: A Study in Game-Theoretic Modeling and Analysis* (Cambridge University Press, Cambridge, UK).
[30] Satterthwaite M (1975) Strategy-proofness and arrow's condition: Existence and correspondence theorems for voting procedures and social welfare functions. *J. Econom. Theory* 10(2):187–217.
[31] Shmoys D, Tardos É (1993) An approximation algorithm for the generalized assignment problem. *Math. Programming* 62(3):461–474.
[32] Vickrey W (1961) Counterspeculation, auctions and competitive sealed tenders. *J. Finance* 16(1):8–37.