All-or-Nothing Generalized Assignment with Application to Scheduling Advertising Campaigns

Ron Adany[1], Moran Feldman[2], Elad Haramaty[2], Rohit Khandekar[3], Baruch Schieber[4], Roy Schwartz[5], Hadas Shachnai[2], and Tami Tamir[6]

[1] Computer Science Department, Bar-Ilan University, Ramat-Gan 52900, Israel. adanyr@cs.biu.ac.il.
[2] Computer Science Department, Technion, Haifa 32000, Israel. E-mail: {moranfe,eladh,hadas}@cs.technion.ac.il.
[3] Knight Capital Group, Jersey City, NJ 07310. E-mail: rkhandekar@gmail.com
[4] IBM T.J. Watson Research Center, Yorktown Heights, NY 10598. E-mail: sbar@us.ibm.com
[5] Microsoft Research, One Microsoft Way, Redmond, WA 98052. E-mail: roysch@microsoft.com
[6] School of Computer science, The Interdisciplinary Center, Herzliya, Israel. E-mail: tami@idc.ac.il.

Abstract. We study a variant of the *generalized assignment problem* (GAP) which we label *all-or-nothing* GAP *(*AGAP*)*. We are given a set of items, partitioned into n groups, and a set of m bins. Each item ℓ has size $s_\ell > 0$, and utility $a_{\ell j} \geq 0$ if packed in bin j. Each bin can accommodate at most one item from each group, and the total size of the items in a bin cannot exceed its capacity. A group of items is *satisfied* if all of its items are packed. The goal is to find a feasible packing of a subset of the items in the bins such that the total utility from satisfied groups is maximized. We motivate the study of AGAP by pointing out a central application in scheduling advertising campaigns.

Our main result is an $O(1)$-approximation algorithm for AGAP instances arising in practice, where each group consists of at most $m/2$ items. Our algorithm uses a novel reduction of AGAP to maximizing submodular function subject to a matroid constraint. For AGAP instances with fixed number of bins, we develop a randomized *polynomial time approximation scheme (PTAS)*, relying on a non-trivial LP relaxation of the problem.

We present a $(3 + \varepsilon)$-approximation as well as PTASs for other special cases of AGAP, where the utility of any item does not depend on the bin in which it is packed. Finally, we derive hardness results for the different variants of AGAP studied in the paper.

1 Introduction

Personalization of advertisements (ads) allows commercial entities to aim their ads at specific audiences, thus ensuring that each target audience receives its

specialized content in the desired format. Recent media research reports [14, 13] show that global spending on TV ads exceeded $323B in 2011, and an average viewer watched TV for 153 hours per month, with the average viewing time consistently increasing. Based on these trends and on advances in cable TV technology, personalized TV ads are expected to increase revenues for TV media companies and for mobile operators [4, 7, 17]. The proliferation of alternative media screens, such as cell-phones and tablets, generate new venues for personalized campaigns targeted to specific viewers, based on their interests, affinity to the advertised content, and location. In fact, ads personalization is already extensively used on the Internet, e.g., in Google AdWords [6]. Our study is motivated by a central application in personalized ad campaigns scheduling, introduced to us by SintecMedia [19].

An *advertising campaign* is a series of advertisement messages that share a single idea and theme which make up an integrated marketing communication. Given a large set of campaigns that can be potentially delivered to the media audience, a service provider attempts to fully deliver a subset of campaigns that maximizes the total revenue, while satisfying constraints on the placement of ads that belong to the same campaign, as well as possible placement constraints among conflicting campaigns. In particular, to increase the number of viewers exposed to an ad campaign, one constraint is that each commercial break contains no more than a single ad from this campaign.[7] Also, each ad has a given length (=size), which remains the same, regardless of the commercial break in which it is placed. This generic assignment problem defines a family of all-or-nothing variants of the *generalized assignment problem* (GAP).

Let $[k]$ denote $\{1, \ldots, k\}$ for an integer k. In *all-or-nothing* GAP (or AGAP), we are given a set of m bins, where bin $j \in [m]$ has capacity c_j, and a set of N items partitioned into n groups $G_1, \ldots, G_n$. Each group $i \in [n]$, consists of k_i items, for some $k_i \geq 1$, such that $\sum_i k_i = N$. Each item $\ell \in [N]$ has a size $s_\ell > 0$ and a non-negative utility $a_{\ell j}$ if packed in bin $j \in [m]$. An item can be packed in at most one bin, and each bin can accommodate at most one item from each group. Given a packing of a subset of items, we say that a group G_i is *satisfied* if all items in G_i are packed. The goal is to pack a subset of items in the bins so that the total utility of satisfied groups is maximized. Formally, we define a packing to be a function $p : [N] \to [m] \cup \{\perp\}$. If $p(\ell) = j \in [m]$ for $\ell \in [N]$, we say that item ℓ is packed in bin j. If $p(\ell) = \perp$, we say that item ℓ is not packed. A packing is *admissible* if $\sum_{\ell \in p^{-1}(j)} s_\ell \leq c_j$ for all $j \in [m]$, and $|p^{-1}(j) \cap G_i| \leq 1$ for all $j \in [m]$ and $i \in [n]$. Given a packing p, let $S_p = \left\{i \in [n] \mid G_i \subseteq \cup_{j \in [m]} \{p^{-1}(j)\}\right\}$ denote the set of groups satisfied by p. The goal in AGAP is to find an admissible packing p that maximizes the utility: $\sum_{i \in S_p} \sum_{\ell \in G_i} a_{\ell p(\ell)}$.

We note that AGAP is NP-hard already when the number of bins is fixed. Such instances capture campaign scheduling in a given time interval (of a few hours) during the day. We further consider the following special cases of AGAP, which are of practical interest. In *all-or-nothing group packing*, each group G_i

[7] Indeed, overexposure of ads belonging to the same campaign in one break may cause lack of interest, thus harming the success of the campaign.

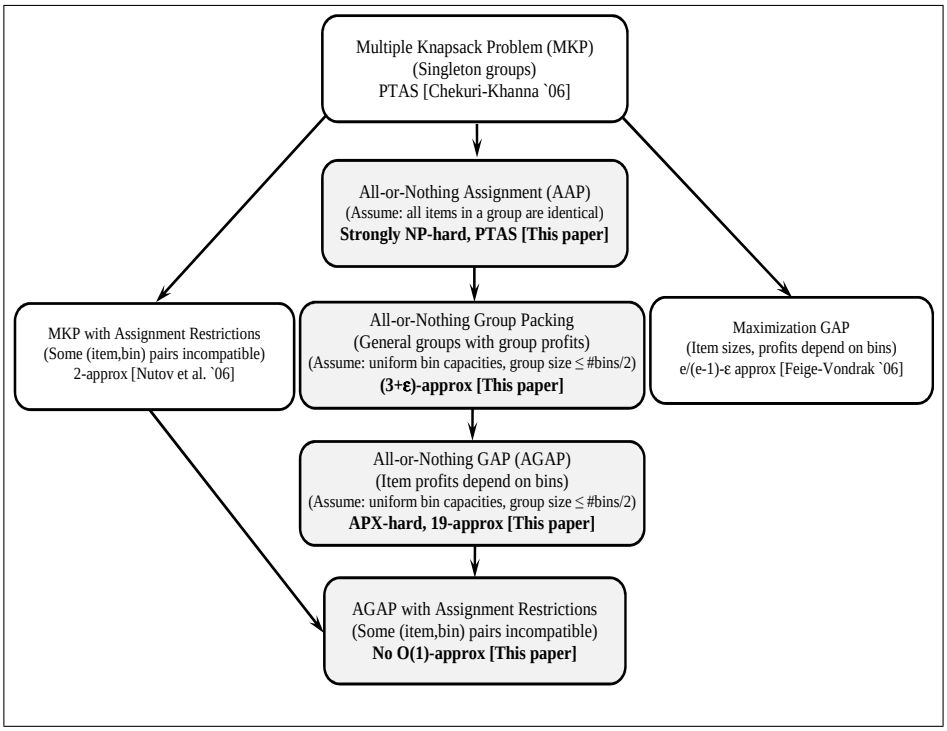

Fig. 1: Summary of our approximation and hardness results and comparison with related problems. An arrow from problem A to B indicates that A is a special case of B.

has a profit $P_i > 0$ if all items are packed, and 0 otherwise. Thus, item utilities do not depend on the bins. In the *all-or-nothing assignment problem* (AAP), all items in G_i have the same size, $s_i > 0$, and same utility $a_i \geq 0$, across all bins.

Note that the special case of AGAP where all groups consist of a *single* item yields an instance of classic GAP, where each item has the same size across the bins. The special case of AAP where all groups consist of a *single* item yields an instance of the multiple knapsack problem. Clearly, AGAP is harder to solve than these two problems. One reason is that, due to the *all-or-nothing* requirement, we cannot eliminate large items of small utilities, since these items may be essential for satisfying a set of most profitable groups. Moreover, even if the satisfied groups are known *a-priori*, since items of the same group cannot be placed in one bin, common techniques for classical packing, such as rounding and enumeration, cannot be applied.

1.1 Our Results

Figure 1 summarizes our contributions for different variants of AGAP and their relations to each other. Even relatively special instances of AAP are NP-hard. Furthermore, in the full paper [1], we show that with slight extensions, AGAP

becomes hard to approximate within any bounded ratio. Thus, we focus in this paper on deriving approximation algorithms for AGAP and the above special cases.

Given an algorithm $\mathcal{A}$, let $\mathcal{A}(I), OPT(I)$ denote the utility of $\mathcal{A}$ and an optimal solution for a problem instance I, respectively. For $\rho \geq 1$, we say that $\mathcal{A}$ is a ρ-approximation algorithm if, for any instance I, $\frac{OPT(I)}{\mathcal{A}(I)} \leq \rho$.

In [1] we show that AGAP with non-identical bins is hard to approximate within any constant ratio, even if the utility of an item is identical across the bins. Thus, in deriving our results for AGAP, we assume the bins are of uniform capacities. Our main result (in Section 2) is a $(19+\varepsilon)$-approximation algorithm for AGAP instances arising in practice, where each group consists of at most $m/2$ items.

Interestingly, AGAP with a fixed number of bins admits a randomized PTAS (see Section 3.1). In Section 3.2 we show that, for the special case where all items have unit sizes, an $\frac{e}{e-1}$-approximation can be obtained by reduction to submodular maximization with knapsack constraint. In Section 3.3 we give a $(3+\varepsilon)$-approximation algorithm for All-or-Nothing Group Packing. This ratio can be improved to $(2+\varepsilon)$ if group sizes are relatively small. The details of these results are omitted due to lack of space and are given in the full paper [1].

In [1] we also present PTASs for two subclasses of instances of AAP. The first is the subclass of instances with unit-sized items, the second is the subclass of instances in which item sizes are drawn from a divisible sequence,[8] and group cardinalities can take the values $k_1, \ldots, k_r$, for some constant $r \geq 1$. Such instances arise in our campaign scheduling application. Indeed, the most common lengths for TV ads are $15, 30$ and 60 seconds [21, 12]. Also, there are standard sizes of $150, 300$ and 600 pixels for web-banners on the Internet [20].

Finally, hardness results for the different all-or-nothing variants of GAP studied are also given in [1].

Technical Contribution: Our approximation algorithm for AGAP (in Section 2) uses a novel reduction of AGAP to maximizing submodular function subject to matroid constraint. At the heart of our reduction lies the fact that the sequence of sizes of large groups can be discretized to yield a logarithmic number of size categories. Thus, we can guarantee that the set of fractionally packed groups, in the initial Maximization Phase of the algorithm, has a total size at most m. Our reduction to submodular maximization encodes this knapsack constraint as a matroid constraint, by considering feasible vectors $(n_1, \ldots, n_H)$, where n_h gives the number of groups taken from size category h, for $1 \leq h \leq H$. These vectors (which are *implicitly* enumerated in polynomial time) are used for defining the matroid constraint.

Our definition of the submodular set function, $f(S)$ (see Section 2), which finds *fractional* packing of items, in fact guarantees that the rounding that we use for group sizes (to integral powers of $1+\varepsilon$, for some $\varepsilon > 0$), causes only small harm to the approximation ratio. This allows also to define a non-standard polynomial time implementation of an algorithm of [2], for maximizing a submodular

[8] A sequence $d_1 < d_2 < \cdots < d_z$ is a *divisible* if d_{i-1} divides d_i for all $1 < i \leq z$.

function under matroid constraint. More precisely, while the universe for our submodular function f is of exponential size, we show that f can be computed in polynomial time.

Our randomized approximation scheme for AGAP instances with constant number of bins (in Section 3.1) is based on a non-trivial LP relaxation of the problem. While the resulting LP has polynomial size when the number of bins is fixed, solving it in polynomial time for general instances (where the number of variables is exponentially large) requires sophisticated use of separation oracles, which is of independent interest. The fractional solution obtained for the LP is rounded by using an approximation technique of [9, 8] for maximizing a submodular function subject to fixed number of knapsack constraints.

1.2 Related Work

All-or-nothing GAP generalizes several classical problems, including GAP (with same sizes across the bins), the *multiple knapsack problem* (MKP), *multiple knapsack with assignment restrictions* (MKAR) [15], and the *generalized multi assignment problem*. In this section we briefly summarize the state of the art for these problems.

As mentioned above, the special case where all groups consist of a *single* item yields an instance of GAP, where each item takes a single size over all bins. GAP is known to be APX-hard already in this case, even if there are only two possible item sizes, each item can take two possible profits, and all bin capacities are identical [3]. The best approximation ratio obtained for GAP is $\frac{e}{e-1} - \varepsilon$ [5].

In minimum GAP (see, e.g., [10]), there are m machines and n jobs. Each machine i is available for T_i time units, and each job has a processing time (size), and a cost of being assigned to a machine. The goal is to schedule all the jobs at minimum total cost, where each job needs to be assigned to a single machine. The paper [18] gives an algorithm which minimizes the total cost, using a schedule where each machine i completes within $2T_i$ time units, $1 \leq i \leq m$.

The generalized multi-assignment problem extends minimum GAP to include multiple assignment constraints. Job processing times and the costs depend on the machine to which they are assigned, the objective is to minimize the costs, and all the jobs must be assigned. This problem was discussed in [16], where Lagrangian dual-based branch-and-bound algorithms were used for obtaining an exact solution for the problem.[9]

We are not aware of earlier work on AGAP or *all-or-nothing* variants of other packing problems.

2 Approximation Algorithm for AGAP

In this section we consider general AGAP instances, where each item ℓ has a size $s_\ell \in (0, 1]$ and arbitrary utilities across the bins. We assume throughout this section that all bins are of the same (unit) capacity. Our approach is based on a version of AGAP, called RELAXED-AGAP, obtained by relaxing the constraint that the total size of items packed in a bin must be at most 1, and by defining

[9] The running time of this algorithm is not guaranteed to be polynomial.

the *utility* of a solution to RELAXED-AGAP slightly differently. We prove that the maximum utility of a solution to RELAXED-AGAP upper bounds the objective value of the optimal AGAP solution. Our algorithm proceeds in two phases.

Maximization Phase: The algorithm approximates the optimal utility of RELAXED-AGAP in polynomial time, by applying a novel reduction to submodular function maximization under matroid constraints. Let S denote the subset of groups assigned by this RELAXED-AGAP solution.

Filling Phase: The algorithm next chooses a subset $S' \subseteq S$ whose utility is at least a constant fraction of the utility of S. Then, the algorithm constructs a feasible solution for AGAP that assigns the groups in S' (not necessarily to the same bins as the RELAXED-AGAP solution) and achieves AGAP value that is at least half of the utility of S', thereby obtaining $O(1)$-approximation for AGAP.

2.1 Maximization Phase

RELAXED-AGAP**:** The input for RELAXED-AGAP is the same as that for AGAP. A feasible RELAXED-AGAP solution is a subset S of the groups whose total size is no more than m (the total size of the bins) and a *valid* assignment p of the items in groups in S to bins; a valid assignment is defined as one in which no two items from the same group are assigned to the same bin. In RELAXED-AGAP, we do not have a constraint regarding the total size of the items assigned to a single bin. Given a solution (S, p) and a bin $j \in [m]$, let $p^{-1}(j) \subseteq [N]$ be the set of items assigned by p to bin j. The utility of a solution (S, p) is the sum of the utility contributions of the bins. The utility contribution of a bin $j \in [m]$ is the maximum value from (fractionally) assigning items in $p^{-1}(j)$ to j satisfying its unit capacity. In other words, we solve for bin j the *fractional knapsack* problem. To define this more formally, we introduce some notation.

Definition 1. *Given a subset $I \subseteq [N]$ of items and a bin j, define $\pi(j, I) = \max_{\boldsymbol{w}} \sum_{\ell \in I} w_\ell a_{\ell j}$, where the maximum is taken over all weight vectors $\boldsymbol{w} \in \Re_+^{|I|}$ that assign weights $w_\ell \in [0, 1]$ to $\ell \in I$, satisfying $\sum_{\ell \in I} w_\ell s_\ell \leq 1$.*

The utility of a solution (S, p) is given by $\sum_{j \in [m]} \pi(j, p^{-1}(j))$. The RELAXED-AGAP is to find a solution with maximum utility.

We can extend Definition 1 to *multisets* as follows.

Definition 2. *A multiset I of $[N]$ can be viewed as a function $I : [N] \to \mathbb{Z}_+$ that maps each $\ell \in [N]$ to a non-negative integer equal to the number of copies of ℓ present in I. Define $\pi(j, I) = \max_{\boldsymbol{w}} \sum_{\ell \in [N]} w_\ell a_{\ell j}$, where the maximum is taken over all weight vectors $\boldsymbol{w} \in \Re_+^N$ that assign weights $w_\ell \in [0, I(\ell)]$ to $\ell \in [N]$ satisfying $\sum_{\ell \in [N]} w_\ell s_\ell \leq 1$.*

It is easy to determine $\boldsymbol{w}$ that maximizes the utility contribution of bin j. Order the items in I as $\ell_1, \ldots, \ell_b$ in a non-increasing order of their ratio of utility to size, i.e., $a_{\ell_1 j}/s_{\ell_1} \geq a_{\ell_2 j}/s_{\ell_2} \geq \cdots \geq a_{\ell_b j}/s_{\ell_b}$. Let d be the maximum index such that $s = \sum_{i=1}^{d} s_{\ell_i} \leq 1$. Set $w_1 = \cdots = w_d = 1$. If $s < 1$ and $d < b$, set $w_{d+1} = (1 - s)/s_{\ell_{d+1}}$. Set the other weights $w_{d+2} = \cdots = w_b = 0$.

Solving RELAXED-AGAP **near-optimally:** Recall that a valid assignment of a subset of items in $[N]$ to bins is one in which no two items from a group get assigned to the same bin. Now define a universe U as follows:

$$U = \{(G, L) \mid L \text{ is a valid assignment of } \textit{all} \text{ items in group } G \text{ to bins } [m]\}$$

A subset $S \subseteq U$ defines a multiset of groups that appear as the first component of the pairs in S. Below, we use $G(S)$ to denote the multiset of such groups. For a subset $S \subseteq U$ and a bin $j \in [m]$, let $I_j = \uplus_{(G,L)\in S} L^{-1}(j)$ be the *multiset* union of sets of items mapped to j over all elements $(G, L) \in S$. Note that I_j can indeed be a multiset since S may contain two elements (G_1, L_1) and (G_2, L_2) with $G_1 = G_2$. Now define $f(S) = \sum_{j\in[m]} \pi(j, I_j)$. The following important but simple claim is proved in [1].

Claim 1 *The function $f(S)$ is non-decreasing and submodular.*

To identify subsets $S \subset U$ that define feasible RELAXED-AGAP solutions, we need two constraints.

Constraint 1. The subset S does not contain two elements (G_1, L_1) and (G_2, L_2) such that $G_1 = G_2$.

Constraint 2. The total size of the groups in $G(S)$, counted with multiplicities, is at most m, i.e., $\sum_{(G,L)\in S} \sum_{\ell\in G} s_\ell \leq m$.

Constraint 1 is easy to handle since it is simply the independence constraint in a partition matroid. Unfortunately, Constraint 2, which is essentially a knapsack constraint, is not easy to handle over the exponential-sized universe U.

Handling Constraint 2 approximately in polynomial time: To this end, we split the groups into a logarithmic number of classes. Fix $\epsilon > 0$. Class 0 contains all groups G such that $s(G) := \sum_{\ell\in G} s_\ell \leq \varepsilon m/n$. For $h \geq 1$, class h contains all groups G with $s(G) \in (\varepsilon m/n \cdot (1+\varepsilon)^{h-1}, \varepsilon m/n \cdot (1+\varepsilon)^h]$. We use $\mathcal{C}_h$ to denote class h. Since $s(G) \leq m$ for all groups G, there are only $H = O(1/\varepsilon \cdot \log(n/\varepsilon))$ non-empty classes. We enforce an upper bound of m on the total size of groups in $G(S)$ by enforcing an upper bound on the total size of groups in $G(S)$ from each class separately. We call a vector $(y_1, \ldots, y_H) \in \mathbb{Z}_+^H$ of non-negative integers *legal* if $\sum_{h=1}^H y_h \leq H(1+1/\varepsilon)$. Note that the number of legal vectors is $O(\binom{H(1+1/\varepsilon)}{H}) = O(2^{H(1+1/\varepsilon)})$, which is polynomial in m and n.

Lemma 2. *For any $S \subseteq U$ satisfying Constraint 2, there exists a legal vector $(y_1, \ldots, y_H)$ such that for all $h \in [H]$, the number of groups in $G(S)$, counted with multiplicities, that are in $\mathcal{C}_h$ is at most $\hat{y}_h := \lfloor y_h n/(H(1+\varepsilon)^{h-1})\rfloor$.*

This lemma implies, in particular, that the optimum solution to AGAP satisfies the above property as well. With this motivation, we define $U_h = \{(G, L) \in U \mid G \in \mathcal{C}_h\}$ and define a new constraint as follows.

Constraint $2'$ for a fixed legal vector $(y_1, \ldots, y_H)$. For each $1 \leq h \leq H$, the number of groups in $G(S)$, counted with multiplicities, that are in $\mathcal{C}_h$ is at most $\hat{y}_h$ as defined in Lemma 2.

Lemma 3. *Fix a legal vector $(y_1, \ldots, y_H)$. The collection of all $S \subseteq U$ satisfying Constraint 1 and Constraint 2′ for this vector defines a* laminar *matroid $M(y_1, \ldots, y_H)$ over U. Furthermore, any independent set $S \subseteq U$ in this matroid satisfies $\sum_{(G,L)\in S} \sum_{\ell \in G} s_\ell \leq m((1+\varepsilon)^2 + \varepsilon)$.*

Given a legal vector $(y_1, \ldots, y_H)$, consider the SUBMOD-MATROID problem of maximizing the non-decreasing submodular function $f(S)$ over all independent sets in the matroid $M(y_1, \ldots, y_H)$. Recall that Nemhauser et al. [11] proved that a greedy algorithm that starts with an empty set and iteratively adds "most profitable" element to it while maintaining independence, as long as possible, is a 1/2-approximation. Each iteration can be implemented in polynomial time as follows. Given a current solution S and a group G, the problem of finding the assignment L that increases the utility f relative to S by the maximum amount can be cast as a bipartite matching problem. To see this, create a bipartite graph with elements in G as vertices on the left-hand-side and bins as vertices on the right-hand-side. For $\ell \in G$ and a bin j, add an edge (ℓ, j) with weight equal to the amount by which contribution of bin j would increase if ℓ is added to bin j. This quantity, in turn, can be computed by solving a fractional knapsack problem on bin j. The maximum weight assignment corresponds to the maximum-weight matching in this graph.

In the maximization phase, we enumerate over all (polynomially many) legal vectors and compute a 1/2-approximate solution to the corresponding SUBMOD-MATROID problem. In the end, we pick the maximum valued solution over all such solutions.

Improving the approximation to $(e-1)/e$: Instead of the greedy algorithm of Nemhauser et al. [11], we can also use the $\frac{e}{e-1}$-approximate Continuous Greedy Algorithm of Calinescu et al. [2]. Some care is needed to show that this algorithm can indeed to implemented in polynomial time in our setting. We omit the details due to lack of space.

In summary, we find a set $S^* \subseteq U$ such that (1) each group appears at most once in $G(S^*)$, (2) the total size of the groups in $G(S^*)$ is at most $m((1+\varepsilon)^2+\varepsilon) \leq m(1+4\varepsilon)$ (if $\varepsilon \leq 1$), and (3) $f(S^*)$ is at least 1/2 (or $(e-1)/e$ if we use the algorithm of Calinescu et al. [2]) of the maximum value achieved by such sets.

2.2 Filling Phase

We show how to choose a subset of the groups in $G(S^*)$ and a feasible assignment of the items in these groups such that the utility of these assignments is a constant fraction of $f(S^*)$. In the description we use parameters $u, v > 0$, whose value will be optimized later.

Lemma 4. *Assume $v \geq 4$, $v(1+4\varepsilon) < u$ and $k_{\max} := \max_i k_i \leq m/2$. In polynomial time, we can compute a subset of groups $F \subseteq G(S^*)$ and a feasible assignment of their items to the bins, forming a feasible solution to* AGAP *with value at least $f(S^*) \cdot \min\{1/u, \frac{1}{2}(1/v(1+4\varepsilon) - 1/u)\}$.*

Recall that $f(S^*) = \sum_j \pi(j, I_j)$, where I_j is a set of items mapped to bin j over all $(G, L) \in S^*$. Since S^* satisfies Constraint 1, we do not have two

elements $(G, L_1), (G, L_2) \in S^*$ for any G. We now subdivide the value $f(S^*)$ into the groups $G \in G(S^*)$, naturally, as follows. Suppose that $\pi(j, I_j)$ is achieved by a weight-vector $\boldsymbol{w}(j)$. Fix any such optimum weight vector $\boldsymbol{w}^*(j)$ for each j. These vectors, when combined, give a weight vector $\boldsymbol{w}^* \in \Re_+^N$, assigning a unique weight w_ℓ^* to each $\ell \in [N]$. We define the *contribution* of a group $G \in G(S^*)$ to $f(S^*)$ as $\sigma^*(G) = \sum_{\ell \in G} w_\ell^* a_{\ell L(\ell)}$ where $(G, L) \in S^*$.

Proof of Lemma 4. If there is a group $G \in G(S^*)$ with $\sigma^*(G) \geq f(S^*)/u$, we output $F = \{G\}$ with the best assignment of items in G to bins (computed using maximum matching, as described in the previous section) as solution. Clearly, the utility of this solution is at least $f(S^*)/u$.

Suppose that no such group exists. In this case we consider the groups $G \in G(S^*)$ in non-increasing order of $\sigma^*(G)/s(G)$. Choose the longest prefix in this order whose total size is at most m/v. Let $T \subset S^*$ be the solution induced by these groups. We first argue that $T \neq \emptyset$. Note that T can be empty only if the first group G in the above order has size more than m/v. Thus $\sigma^*(G)/(m/v) > \sigma^*(G)/s(G) \geq f^*(S)/(m(1+4\varepsilon))$. The second inequality holds since the total size of groups in $G(S^*)$ is at most $m(1+4\varepsilon)$ and the "density" $\sigma^*(G)/s(G)$ of G is at least the overall density of $G(S)$, which in turn is at least $f^*(S)/(m(1+4\varepsilon))$. This implies that $\sigma^*(G) > f^*(S)/(v(1+4\varepsilon)) > f^*(S)/u$, a contradiction.

The following three steps find a feasible solution to AGAP that consists of the groups in $G(T)$ and whose value is at least $f(T)/2$.

1. Eliminate all zero weights: Let $\boldsymbol{w} \in \Re_+^N$ be the weight vector that determines the value $f(T)$. Note that the weight w_ℓ assigned to some of the items ℓ in groups in $G(T)$ may be zero. We modify the assignment of items in the solution T so that no item would have zero weight. Note that if an item ℓ assigned to bin j in solution S has $w_\ell = 0$, the total size of the items assigned to bin j in S is at least 1. It follows that there are at most $\lfloor m/v \rfloor$ bins that may contain items of zero weight, since the total size of all items assigned in T is no more than m/v.

For each item with zero weight that belongs to a group G_i, there is at least one bin j such that the total size of the items assigned to bin j is less than 1 and no items from group G_i are assigned to bin j. This follows since $|G_i| + \lfloor m/v \rfloor \leq m/2 + \lfloor m/v \rfloor < m$. It follows that this item can be assigned to bin j and be assigned non-zero weight. We can continue this process as long as there are items with zero weight, thereby, eliminating all zero weights.

2. Evicting overflowed items: Suppose there are a (respectively, b) bins that are assigned items of total size more than 1 (respectively, more than 1/2 and at most 1). Call these bins 'full' (respectively, 'half full'). Since the total volume of packed items is at most m/v, we have $a + b/2 \leq m/v$. Next, we remove some items from these a full bins to make the assignment feasible. Consider such a bin. We keep in this bin either all the items assigned to it that have weight equal to 1, or the unique item that has weight strictly between 0 and 1, whichever contributes more to $f(T)$. In this step, we lose at most half of the contribution of the full bins to $f(T)$. We further evict all items assigned to the least profitable

$\lfloor (m-a)/2 \rfloor$ non-full bins. In this step, we lose at most half of the contribution of the non-full bins to $f(T)$.

3. Repacking evicted items: We now repack all the evicted items to maintain feasibility of the solution. We first repack evicted items of size at least half. Note that are at most a such items from full bins, and at most b such items from half full bins. These $a+b$ items can be packed into evicted $\lfloor (m-a)/2 \rfloor$ bins by ensuring $a+b \leq \lfloor (m-a)/2 \rfloor$, i.e., $3a+2b < m$. This is indeed true since $v \geq 4$ together with $a + b/2 \leq m/v$ implies $4a + 2b \leq m$.

We are now left only with items whose size is less than half to repack. For each such item from group i, we find a bin that does not contain another item from group i and whose total size is less than half, and insert the item to this bin. Note that, since the size of the item is less than half, the solution remains feasible. Since the total size of the items to be packed is at most m/v, there are at most $\lfloor 2m/v \rfloor$ bins of size at least half. Thus, we are guaranteed to find such a bin in case $m - \lfloor 2m/v \rfloor - k_i \geq 0$, i.e., $k_i \leq \lceil m(1-2/v) \rceil$.

We now bound $f(T)$. Since the contribution of any group to $f(S^*)$ is no more than $f(S^*)/u$, the contribution of the groups in T is at least $f(S^*) \cdot (1/v(1+4\varepsilon) - 1/u)$. Recall that the reduction in $f(T)$ due to the eviction of items is at most half of $f(T)$. Thus the value of the final solution is at least $f(S^*) \cdot \frac{1}{2}(1/v(1+4\varepsilon) - 1/u)$. This completes the proof of the lemma. ■

Now, to bound the overall approximation ratio, we seek the values of u and v satisfying $1/u = \frac{1}{2}(1/(v(1+4\varepsilon)) - 1/u)$. Thus, we set $u = 3v(1+4\varepsilon)$. For $v = 4$ and $u = 12(1+4\varepsilon)$, we get a ratio of $\frac{1}{12(1+3\varepsilon)}$. Since we lost a factor of $1/2$ (or $(e-1)/e$) in the maximization phase, we get an overall $24(1+4\varepsilon)$-approximation (or $12(1+\varepsilon)\frac{e}{e-1}$-approximation).

This proves the following theorem.

Theorem 1. AGAP *admits a polynomial-time* $12(1+\varepsilon)\frac{e}{e-1}$*-approximation for any* $0 < \epsilon < 1$*, provided any group has at most* $k_{\max} \leq m/2$ *items.*

3 Approximating Special Cases of AGAP

In this section we consider several special cases of AGAP. We assume throughout the discussion that the bins have uniform (unit) capacities.

3.1 Approximation Scheme for Constant Number of Bins

We formulate the following LP relaxation for AGAP. For every group G_i, we define $\mathcal{P}_i$ to be the collection of admissible packings of elements of group G_i alone. The relaxation has an indicator variable $x_{i,p}$ for every group G_i and admissible assignment $p \in \mathcal{P}_i$. Beside the constraints of AGAP, we further require the total size of the elements in the fractional solution to be at most $M \in [0, m]$. Note that this LP is a relaxation of AGAP only for $M = m$.

$$
\begin{array}{lll}
\text{(AGAP-LP)} \max & \sum_{i\in[n]} \sum_{p\in\mathcal{P}_i} \left(x_{i,p} \cdot \sum_{\ell\in G_i} a_{\ell p(\ell)}\right) & \\
\text{s.t.} & \sum_{p\in\mathcal{P}_i} x_{i,p} \leq 1 & \forall\, i \in [n] \qquad (1)\\
& \sum_{i\in[n]} \sum_{p\in\mathcal{P}_i \mid \exists\, \ell: p(\ell)=j} x_{i,p} \cdot s_\ell \leq c_j & \forall\, j \in [m] \qquad (2)\\
& \sum_{i\in[n]} \sum_{p\in\mathcal{P}_i} \left(x_{i,p} \cdot \sum_{\ell\in G_i} s_\ell\right) \leq M & \qquad (3)\\
& x_{i,p} \geq 0 & \forall\, i \in [n], p \in \mathcal{P}_i
\end{array}
$$

Constraint (1) requires every group to have at most one assignment. Constraint (2) guarantees that no bin is over-packed. Finally, constraint (3) enforces that the total size of the packed elements does not exceed M.

Lemma 5. *AGAP-LP can be solved in polynomial time.*

The proof of Lemma 5 is based on finding a separation oracle for the dual LP. We give the details in [1].

We present an approximation scheme for the case where the number of bins is a constant. The algorithm uses AGAP-LP, which in this case is of polynomial size and thus can be solved in polynomial time using standard techniques. The rounding procedure we apply draws many ideas from the rounding procedure suggested in [9, 8] for the problem of maximizing a submodular function subject to a constant number of knapsack constraints. The idea of the rounding procedure is to guess the most valuable groups of the optimal solution and their corresponding assignment in this solution. Note that this can be done efficiently because the number of bins is constant. None of the remaining groups can be valuable on their own, and therefore, we can safely dismiss all such groups containing a large element. This allows us to show, via concentration bounds, that a randomized rounding satisfies the capacity constraints of all bins with high enough probability (recall that all remaining elements are small).

Theorem 2. *There is a randomized polynomial time approximation scheme for* AGAP *with fixed number of bins.*

3.2 Approximation Algorithm for Unit Size Items

In the special case where all items have unit sizes, we give the best possible approximation ratio.

Theorem 3. AGAP *with unit item sizes admits an* $\frac{e}{e-1}$*-approximation.*

3.3 The All-or-Nothing Group Packing Problem

For AGAP instances where each group G_i has a utility $P_i > 0$ if all of its items are packed, and 0 otherwise, we show that AGAP can be approximated within a small constant $\rho \in (2, 3+\varepsilon]$, for some $\varepsilon > 0$. Specifically,

Theorem 4. *There is a* $(\frac{2(\gamma+1)}{\gamma} + \varepsilon)$*-approximation for all-or-nothing group packing, where* $\gamma = \lfloor \frac{m}{k_{\max}} \rfloor$.

References

1. R. Adany, M. Feldman, E. Haramaty, R. Khandekar, B. Schieber, R. Schwartz, H. Shachnai, and T. Tamir. All-or-nothing generalized assignment with application to scheduling advertising campaigns. *Full version*, http://www.cs.technion.ac.il/~hadas/PUB/AGAP_full.pdf, 2012.
2. G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a submodular set function subject to a matroid constraint. *SIAM J. on Computing*, 40(6), 2011.
3. C. Chekuri and S. Khanna. A PTAS for the multiple knapsack problem. *SIAM J. on Computing*, 35(3):713–728, 2006.
4. V. Dureau. Addressable advertising on digital television. In *Proceedings of the 2nd European conference on interactive television: enhancing the experience*, Brighton, UK, March–April 2004.
5. U. Feige and J. Vondrák. Approximation algorithms for allocation problems: Improving the factor of 1-1/e. In *FOCS*, pages 667–676, 2006.
6. Google AdWords. `http://adwords.google.com`.
7. E. M. Kim and S. S. Wildman. A deeper look at the economics of advertiser support for television: the implications of consumption-differentiated viewers and ad addressability. *J. of Media Economics*, 19:55–79, 2006.
8. A. Kulik, H. Shachnai, and T. Tamir. Maximizing submodular set functions subject to multiple linear constraints. To appear in *Mathematics of Operations Research*.
9. A. Kulik, H. Shachnai, and T. Tamir. Maximizing submodular set functions subject to multiple linear constraints. In *SODA*, pages 545–554, 2009.
10. O. E. Kundakcioglu and S. Alizamir. Generalized assignment problem. In C. A. Floudas and P. M. Pardalos, editors, *Encyclopedia of Optimization*, pages 1153–1162. Springer, 2009.
11. G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Math. Programming*, 14:265–294, 1978.
12. Nielsen Media Research. Advertising fact sheet. blog.nielsen.com, September 2010.
13. Nielsen Media Research. The cross-platform report, quarter 1, 2012 – US. blog.nielsen.com, May 2012.
14. Nielsen Media Research. Nielsen's quarterly global adview pulse report. blog.nielsen.com, April 2012.
15. Z. Nutov, I. Beniaminy, and R. Yuster. A (1-1/e)-approximation algorithm for the generalized assignment problem. *Operations Research Letters*, 34(3):283–288, 2006.
16. J. Park, B. Lim, and Y. Lee. A Lagrangian dual-based branch-and-bound algorithm for the generalized multi-assignment problem. *Management Science*, 44:271–282, 1998.
17. K. Pramataris, D. Papakyriakopoulos, G. Lekakos, and N. Mulonopoulos. Personalized Interactive TV Advertising: The iMEDIA Business Model. *Electronic Markets*, 11:1–9, 2001.
18. D. Shmoys and É. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming*, 62(1):461–474, 1993.
19. SintecMedia - On Air. `http://www.sintecmedia.com/OnAir.html`.
20. The Interactive Advertising Bureau (IAB). `http://iab.net`.
21. C. Young. Why TV spot length matters. *Admap*, (497):45–48, September 2008.