Brief Announcement: Approximation Algorithms for Preemptive Resource Allocation

Kanthi Sarpatwar
IBM T. J. Watson Research Center
Yorktown Heights, New York
sarpatwa@us.ibm.com

Baruch Schieber
IBM T. J. Watson Research Center
Yorktown Heights, New York
sbar@us.ibm.com

Hadas Shachnai
Technion, Computer Science Dept.
Haifa, Israel
hadas@cs.technion.ac.il

ABSTRACT

Cloud services require the allocation of scarce resources to multiple user requests (jobs) in a setting that facilitates preemption and migration, while respecting resource and timing constraints. This gives rise to the following variants of the classical *resource allocation problem* (RAP). The input is a set J of jobs and a set M of homogeneous hosts, each has available amount of some resource. A job is associated with a release time, a due date, a weight and a given length, as well as its resource requirement. A *feasible* schedule is an allocation of the resource to a subset of the jobs, satisfying the job release times/due dates as well as the resource constraints.

We consider two essential objectives: *throughput maximization* (MaxT), which seeks a maximum weight subset of the jobs that can be feasibly scheduled on the hosts in M , and *resource minimization* (MinR), that is finding the minimum number of (homogeneous) hosts needed to feasibly schedule all jobs. Both problems are known to be NP-hard. We address these fundamental problems, which have been studied previously in the *non-preemptive* model, and develop novel techniques for tackling them. In the full version of the paper, we present an $\Omega(1)$ -approximation algorithm for MaxT, assuming there is sufficient slack for each job to be completed in its time window. For MinR we study a more general setting with d resources and derive an $O(\log d)$ -approximation for any fixed $d \geq 1$, under the assumption that time windows are not too small. This assumption can be removed leading to a slightly worse ratio of $O(\log d \log^* T)$, where T is the maximum due date of any job.

CCS CONCEPTS

• **Mathematics of computing** → **Approximation algorithms;**

KEYWORDS

Preemptive resource allocation, throughput maximization, resource minimization, real-time scheduling, approximation algorithms

ACM Reference Format:

Kanthi Sarpatwar, Baruch Schieber, and Hadas Shachnai. 2018. Brief Announcement: Approximation Algorithms for Preemptive Resource Allocation. In *Proceedings of 30th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA'18)*. ACM, New York, NY, USA, 3 pages. <https://doi.org/10.1145/3210377.3210666>

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
SPAA'18, July 16–18, 2018, Vienna, Austria
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-5799-9/18/07.
<https://doi.org/10.1145/3210377.3210666>

1 BACKGROUND

A key challenge in cloud services is the allocation of available resources, such as processor cycles, storage space, and network bandwidth to user requests (jobs), so that each job is completed within its time window. The use of *virtual machines*, which can be assigned at different times to different hosts, enables non-consecutive allocation of resources to jobs, and also implies the possible migration of a job to another host throughout its execution.

This modern setting defines the following variants of the classic *resource allocation problem* (RAP) [4, 7, 25]. The input is a set of jobs J that need to be scheduled on a set of hosts M , where each host has a limited (w.l.o.g., unit) amount of one or more resources. Each job $j \in J$ has a length, p_j , along with a required amount of the resource, s_j , and a time window in which it can be processed. The goal is to schedule jobs to the hosts within their time windows such that, at any given time, the total amount of resource required by the jobs assigned to any host does not exceed the available resource. A schedule may use preemption and job migration.

In the *maximum throughput* (MaxT) variant, we are given a set of homogeneous hosts M and a set of jobs J , such that each job j has a profit w_j and attributes (p_j, s_j, r_j, d_j) ; r_j and d_j are the release times and due dates of j , respectively. We seek a subset $S \subseteq J$ of jobs of maximum profit $\sum_{j \in S} w_j$ that can be feasibly scheduled on M .

In the *resource minimization* (MinR) variant, we assume that each job j has a resource requirement vector $\bar{s}_j \in [0, 1]^d$ as one of the attributes, where $d \geq 1$ is the number of available resources. A schedule that assigns a set of jobs $S_{i,t}$ to a host $i \in M$ at time t is feasible if $\sum_{j \in S_{i,t}} \bar{s}_j \leq \bar{1}^d$. Given a set of jobs J with attributes $(p_j, \bar{s}_j, r_j, d_j)$, we seek a set of (homogeneous) hosts M of minimum cardinality such that all of the jobs can be scheduled feasibly on M .

Indeed, a classic RAP instance can be viewed as MaxT with a *single* host. In this aspect, our setting is better suited for cloud services, where resources are distributed (also geographically) among multiple hosts. Further, MaxT allows non-consecutive allocation of the resource to each job. This renders significant structural differences between MaxT and RAP, as discussed below.

Another classical problem of interest in our setting is *vector packing* (VP), in which a set of d -dimensional items needs to be feasibly packed into a minimum number of d -dimensional bins of unit size in each dimension (i.e., the vector sum of all the items packed into each bin is less than $\bar{1}^d$). VP is the special case of MinR with unit processing times, where for all jobs $j \in J$, $r_j = 0$, $d_j = 1$ and $p_j = 1$.

We provide a first step in the study of *preemptive* RAP. In the full version of this paper, we present approximation algorithms for general inputs of MaxT and MinR, with better results for some special cases.

2 OUR CONTRIBUTIONS

2.1 Throughput Maximization

The non-preemptive version of MaxT, classical RAP, was studied earlier. To see the structural differences between the two problems, consider their natural linear programming relaxations. In the case of RAP, it is sufficient to have a single indicator variable x_{jt} for each job j and time slot t to represent its start time. This allows the application of a natural randomized rounding algorithm, where job j is scheduled to start at time t with probability x_{jt} . On the other hand, in MaxT, a job can be preempted several times; therefore, each job requires multiple indicator variables. Further, these variables must be rounded in an *all-or-nothing* fashion, i.e., either we schedule all parts of a job or none of them. Our approach to handle this situation is to, somewhat counter-intuitively, “dumb down” the linear program by not committing the jobs to a particular schedule; instead, we choose a subset of jobs that satisfy certain knapsack constraints and construct the actual schedule in a subsequent phase.

We first consider a *laminar* variant of the problem, where the time windows for the jobs are chosen from a laminar family of intervals. This setting includes several important special cases, such as (i) all jobs are released at $t = 0$ but have different due dates, or (ii) jobs are released at different times, but all must be completed by a given due date.

Our algorithm for the laminar case relies on a combination of a *packing* phase and a *scheduling* phase. While the first phase ensures that the output solution has high profit, the second phase guarantees its feasibility. To facilitate a successful completion of the selected jobs, we formulate a set of conditions that must be satisfied in the packing phase. Both phases make use of the structural properties of a *laminar* family of intervals. In the packing phase, we apply our rounding procedure (for the LP solution) to the tree representation of the intervals. We further use this tree in the scheduling phase, to feasibly assign the resource to the selected jobs in a bottom-up fashion. We first handle instances satisfying some slackness condition; that is, for some constant $\lambda \in (0, 1)$, $p_j \leq \lambda(d_j - r_j + 1)$ for any job j . Thus, the time windows for processing the jobs are large enough. Such an assumption is quite reasonable in scenarios arising in our applications. We show that the two-phase algorithm yields a constant approximation for laminar instances.

Using a simple transformation of an arbitrary instance to laminar, we extend the result to general instances. Let $m = |M|$ be the number of hosts.

THEOREM 1. *For any $\lambda < \frac{1}{4} - \frac{1}{2(m+2)}$, there exists a $\frac{1}{8} - \lambda \left(\frac{1}{2} + \frac{1}{m} \right)$ -approximation algorithm for MaxT with slackness parameter λ .*

We further show that the dependence of the slackness parameter λ on m can be eliminated, i.e., MaxT admits a constant approximation for instances with *any* constant slackness parameter $\lambda \in (0, \frac{1}{4})$.

2.2 Maximizing Utilization

A central goal in allocating scarce resources is to maximize resource utilization. This is the special case of MaxT where the weight of each job is equal to its total resource requirement (or, *area*), i.e., $w_j = s_j p_j$ for all $j \in J$. For such instances, we obtain a constant approximation guarantee, building on an algorithm of Chen, Hassin and Tzur [10].

THEOREM 2. *There is a polynomial time $\Omega(1)$ -approximation for any instance of MaxT where $w_j = s_j p_j$ for all $j \in J$.*

2.3 General MaxT Instances

Our approach for tackling general instances (with no assumption on slackness) is to distinguish between jobs with ‘large’ and ‘small’ resource requirements. For jobs with requirements at least $\delta \in (0, 1)$, we show the existence of an $\Omega(\frac{1}{\log(1/\delta)})$ -approximation. Thus, taking $\delta = \frac{1}{n}$, we obtain a solution S_1 that is an $\Omega(\frac{1}{\log n})$ -approximation for the large jobs. The second solution, S_2 , consists of all the small jobs, whose total sum of requirements is smaller than 1. Selecting the highest profit solution between S_1 and S_2 , we obtain a logarithmic approximation ratio.

THEOREM 3. *There is a polynomial time $\Omega(\frac{1}{\log n})$ -approximation algorithm for MaxT.*

2.4 Resource Minimization

Our approach for solving MinR builds on a formulation of the problem as a configuration LP, inspired by the works of [2, 15]. However, we quickly deviate from these prior approaches, in order to handle the time windows and the extra constraints. Our algorithm involves two main phases: a *maximization phase* and a *residual phase*. A *configuration* is a subset of jobs that can be feasibly assigned to a host at a given time slot t . Given the minimum number of hosts, m , obtained from an optimal solution for the LP, we choose for each t $O(m \log d)$ configurations with probabilities proportional to their LP-values. In this phase, jobs may be allocated the resource only for part of their processing length. In the second phase, we construct a residual instance based on the amount of time each job has been processed. A key challenge is to show that, for any time window $[r_j, d_j]$, the total “area” of jobs left to be scheduled is at most $1/d$ of the original total area. We use this property to solve the residual instance. We obtain the next result.

THEOREM 4. *Given an instance of MinR with slackness parameter $\lambda \in (0, \frac{1}{4})$, fix an $\epsilon \in (0, 1)$. Then there is a polynomial time algorithm that yields an $O(\log d \log^* T)$ approximation ratio with probability at least $1 - \epsilon$.*

3 RELATED WORK

The classical problem of preemptively scheduling a set of jobs with attributes $(p_j, s_j = 1, r_j, d_j)$ on a single machine so as to maximize throughput can be cast as a special case of MaxT with a single host, where each job requires all of the available resource. Lawler [23] showed that in this special case MaxT admits a PTAS, and the problem is polynomially solvable for uniform job weights. For multiple hosts (i.e., $m = |M| > 1$), this special case of MaxT ($s_j = 1$ for all $j \in J$) admits a $\frac{1}{6+\epsilon}$ -approximation, for any fixed $\epsilon > 0$. This follows from a result of Kalyanasundaram and Pruhs [20].

Another special case of MaxT was studied in the context of advertisement placement. The *ad placement* problem, introduced by Adler *et al.* [1], can be viewed as preemptive RAP with a single host, where all jobs have the same release time and due date. Ad placement was widely studied (see, e.g., [13, 14, 17, 21, 22] and the comprehensive survey in [24]). Freund and Naor [17] presented a $(\frac{1}{3} - \epsilon)$ -approximation for the maximum throughput (MaxT) version of the problem.

Fox and Korupula [16] recently studied our preemptive scheduling model, with job attributes (p_j, s_j, r_j, d_j) , under another popular objective, namely, minimizing weighted flow-time. Their work differs from ours in two ways: while they focus on the online versions, we consider our problems in an offline setting. Further, as they note, while the throughput and resource minimization objectives are also commonly considered metrics, their techniques only deal with flow-time. In fact, these objectives are fundamentally different and we need novel algorithms to tackle them.

The non-preemptive variant of MaxT, known as the *resource allocation problem* (RAP), was introduced by Phillips *et al.* [25], and later studied by many authors (see, e.g., [5–8, 10, 18] and the references therein).¹ Chakaravarthy *et al.* [8] consider a generalization of RAP and obtain a constant approximation based on a primal-dual algorithm.

As mentioned above, MinR generalizes the classic vector packing (VP) problem. The first non-trivial $O(\log d)$ -approximation algorithm for VP was presented by Chekuri and Khanna [9], for any fixed $d \geq 1$. This ratio was improved by Bansal, Caprara and Sviridenko [2] to a randomized algorithm with asymptotic approximation ratio arbitrarily close to $\ln d + 1$. Bansal, Eliás and Khan [3] recently improved this ratio further to $0.807 + \ln(d + 1)$. A “fractional variant” of MinR was considered by Jansen and Porkolab [19], where time was assumed to be continuous. For this problem, in the case of a single host, they obtain a polynomial time approximation scheme (PTAS), by solving a configuration linear program (rounding the LP solution is not necessary because time is continuous in their case).

Resource minimization was considered also in the context of the ad placement problem. In this variant, a set of ads must be scheduled, and the objective is to minimize the total amount of resource (i.e., width of the display area) required to make this possible. Freund and Naor [17] gave a 2-approximation algorithm for the problem, which was later improved by Dawande *et al.* [14] to $\frac{3}{2}$. This implies a 3-approximation for MinR instances with $d = 1$, where all jobs have the same release time and due date. For this special case of MinR we show that the ratio can be improved, namely, the jobs can be feasibly scheduled on $2OPT + 1$ hosts, where OPT is the optimal number of hosts.

Another line of work relates to the non-preemptive version of MinR, where $d = 1$ and the requirement of each job is equal to 1 (see, e.g. [11, 12]); thus, at most one job can be scheduled on each host at any time.

ACKNOWLEDGMENTS

H. Shachnai’s work was conducted during a visit to DIMACS partially supported by the National Science Foundation under grant number CCF-1445755.

REFERENCES

- [1] Micah Adler, Phillip B Gibbons, and Yossi Matias. 2002. Scheduling space-sharing for internet advertising. *Journal of Scheduling* 5, 2 (2002), 103–119.
- [2] Nikhil Bansal, Alberto Caprara, and Maxim Sviridenko. 2009. A New Approximation Method for Set Covering Problems, with Applications to Multidimensional Bin Packing. *SIAM J. Comput.* 39, 4 (2009), 1256–1278.
- [3] Nikhil Bansal, Marek Eliás, and Arindam Khan. 2016. Improved approximation for vector bin packing. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms*. 1561–1579.
- [4] Nikhil Bansal, Zachary Friggstad, Rohit Khandekar, and Mohammad R Salavatipour. 2014. A logarithmic approximation for unsplittable flow on line graphs. *ACM Transactions on Algorithms* 10, 1 (2014), 1:1–1:15.
- [5] Amotz Bar-Noy, Reuven Bar-Yehuda, Ari Freund, Joseph Naor, and Baruch Schieber. 2001. A unified approach to approximating resource allocation and scheduling. *J. ACM* 48, 5 (2001), 1069–1090.
- [6] Amotz Bar-Noy, Sudipto Guha, Joseph Naor, and Baruch Schieber. 2001. Approximating the Throughput of Multiple Machines in Real-Time Scheduling. *SIAM J. Comput.* 31, 2 (2001), 331–352.
- [7] Gruia Călinescu, Amit Chakrabarti, Howard J. Karloff, and Yuval Rabani. 2011. An improved approximation algorithm for resource allocation. *ACM Trans. Algorithms* 7, 4 (2011), 48:1–48:7.
- [8] Venkatesan T Chakaravarthy, Anamita R Choudhury, Shalmoli Gupta, Sambuddha Roy, and Yogish Sabharwal. 2014. Improved algorithms for resource allocation under varying capacity. In *Proceedings of the 22nd Annual European Symposium on Algorithms*. Springer, 222–234.
- [9] Chandra Chekuri and Sanjeev Khanna. 2004. On multidimensional packing problems. *SIAM journal on computing* 33, 4 (2004), 837–851.
- [10] Bo Chen, Refael Hassin, and Michal Tzur. 2002. Allocation of bandwidth and storage. *IEEE Transactions* 34, 5 (2002), 501–507.
- [11] Julia Chuzhoy and Paolo Codenotti. 2009. Resource minimization job scheduling. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, Proceedings of the 12th Annual APPROX 2009, and 13th Annual RANDOM*. Springer, 70–83.
- [12] Julia Chuzhoy, Sudipto Guha, Sanjeev Khanna, and Joseph Naor. 2004. Machine minimization for scheduling jobs with interval constraints. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*. 81–90.
- [13] Milind Dawande, Subodha Kumar, and Chelliah Sriskandarajah. 2003. Performance bounds of algorithms for scheduling advertisements on a web page. *Journal of Scheduling* 6, 4 (2003), 373–394.
- [14] Milind Dawande, Subodha Kumar, and Chelliah Sriskandarajah. 2005. Scheduling web advertisements: a note on the minspace problem. *Journal of Scheduling* 8, 1 (2005), 97–106.
- [15] Lisa Fleischer, Michel X. Goemans, Vahab S. Mirrokni, and Maxim Sviridenko. 2011. Tight Approximation Algorithms for Maximum Separable Assignment Problems. *Math. Oper. Res.* 36, 3 (2011), 416–431.
- [16] Kyle Fox and Madhukar Korupolu. 2013. Weighted flowtime on capacitated machines. In *Proceedings of the 24th annual ACM-SIAM symposium on Discrete algorithms*. 129–143.
- [17] Ari Freund and Joseph Naor. 2004. Approximating the advertisement placement problem. *Journal of Scheduling* 7, 5 (2004), 365–374.
- [18] Navendu Jain, Ishai Menache, Joseph Naor, and Jonathan Yaniv. 2015. Near-optimal scheduling mechanisms for deadline-sensitive jobs in large computing clusters. *ACM Transactions on Parallel Computing* 2, 1 (2015), 3.
- [19] Klaus Jansen and Lior Porkolab. 2002. On preemptive resource constrained scheduling: polynomial-time approximation schemes. In *Proceedings of the 9th International Conference on Integer Programming and Combinatorial Optimization*. 329–349.
- [20] Bala Kalyanasundaram and Kirk Pruhs. 2001. Eliminating Migration in Multiprocessor Scheduling. *J. Algorithms* 38, 1 (2001), 2–24.
- [21] Arshia Kaul, Sugandha Aggarwal, Anshu Gupta, Niraj Dayama, Mohan Krishnamoorthy, and PC Jha. 2017. Optimal advertising on a two-dimensional web banner. *International Journal of System Assurance Engineering and Management* (2017), 1–6.
- [22] Subodha Kumar, Milind Dawande, and Vijay Mookerjee. 2007. Optimal scheduling and placement of internet banner advertisements. *IEEE Transactions on Knowledge and Data Engineering* 19, 11 (2007).
- [23] Eugene L Lawler. 1990. A dynamic programming algorithm for preemptive scheduling of a single machine to minimize the number of late jobs. *Annals of Operations Research* 26, 1 (1990), 125–133.
- [24] Shinjini Pandey, Goutam Dutta, and Harit Joshi. 2017. Survey on Revenue Management in Media and Broadcasting. *Interfaces* 47, 3 (2017), 195–213.
- [25] Cynthia A. Phillips, R. N. Uma, and Joel Wein. 2000. Off-line admission control for general scheduling problems. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*. 879–888.

¹RAP is also known as the *bandwidth allocation problem*.