Brief Announcement: Flexible Resource Allocation for Clouds and All-Optical Networks*

Dmitriy Katz
IBM T.J. Watson Research
Center, Yorktown Heights
NY 10598, USA
katzrog@us.ibm.com

Baruch Schieber
IBM T.J. Watson Research
Center, Yorktown Heights
NY 10598, USA
sbar@us.ibm.com

Hadas Shachnai[†]
Dept. of Computer Science
Technion
Haifa 3200003, Israel
hadas@cs.technion.ac.il

ABSTRACT

Motivated by the cloud computing paradigm, and by key optimization problems in all-optical networks, we study two variants of the classic job interval scheduling problem, where a reusable resource is allocated to competing job intervals in a *flexible* manner. Each job, J_i , requires the use of up to $r_{\max}(i)$ units of the resource, with a profit of $p_i \geq 1$ accrued for each allocated unit. The goal is to feasibly schedule a subset of the jobs so as to maximize the total profit. The resource can be allocated either in *contiguous* or *non-contiguous* blocks. These problems can be viewed as flexible variants of the well known *storage allocation* and *bandwidth allocation* problems.

We show that the contiguous version is strongly NP-hard, already for instances where all jobs have the same profit and the same maximum resource requirement. We derive the best possible positive result for such instances, namely, a *polynomial time approximation scheme* (PTAS). We further show that the contiguous variant admits a $(\frac{5}{4} + \varepsilon)$ -approximation algorithm, for any fixed $\varepsilon > 0$, on instances whose job intervals form a *proper* interval graph. At the heart of the algorithm lies a non-standard parameterization of the approximation ratio itself. For the non-contiguous case, we uncover an interesting relation to the *paging* problem that leads to a simple $O(n \log n)$ algorithm for *uniform* profit instances of n jobs.

Keywords

Flexgrid all-optical networks; cloud computing; scheduling; paging; flexible storage allocation; bandwidth allocation

1. BACKGROUND

Interval scheduling has a wide range of applications in computer science and in operations research (see, e.g., [8]). We study two variants of the problem, where a reusable

[†]This work was partly carried out during a visit to DIMACS supported by the National Science Foundation under grant number CCF-1144502.

*A detailed version of this paper can be found in [7].

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 '16 July 11-13, 2016, Pacific Grove, CA, USA

© 2016 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-4210-0/16/07.

DOI: <http://dx.doi.org/10.1145/2935764.2935806>

resource is allocated to competing job intervals in a *flexible* manner. The input to our problems consists of a set $\mathcal{J}$ of n jobs, where each job $J_i \in \mathcal{J}$ is associated with a half open real interval $[s_i, e_i)$, along which it requires the use of a limited resource. The amount of resource allocated to J_i can take any value up to $r_{\max}(i)$, and the profit accrued for each allocated unit is $p_i \geq 1$. The total amount of resource allocated at any time cannot exceed the available amount, $W > 0$.

In the *flexible bandwidth allocation problem* (FBAP), we seek a feasible allocation which maximizes the total profit. In the *flexible storage allocation problem* (FSAP), we add the requirement that the allocation to each job is a *contiguous* block of the resource.

Motivation. Scheduling problems of this ilk arise naturally in scenarios where activities require a portion of an available resource with some revenue associated with the allocated amount. In the cloud computing paradigm the resource can be servers in a large computing cluster, storage capacity, or bandwidth allocated to time-critical jobs (see, e.g. [6]).

In *flexgrid* all-optical networks, the resource is the available spectrum of light that is divided into frequency intervals of variable width (see, e.g., [5]). Several high-speed signals connecting different source-destination pairs may share a link, provided they are assigned disjoint sub-spectra (see, e.g., [10]). Given a path network and a set of connection requests, represented by intervals, each associated with a profit per allocated spectrum unit and a maximum bandwidth requirement, we need to feasibly allocate frequencies to the requests, with the goal of maximizing the total profit. In the FBAP variant the sub-spectra allocated to each request need not be contiguous, while in the FSAP variant each request requires a contiguous frequency spectrum.

Related Work. The classic interval scheduling problem, where each interval requires all of the resource for its execution, is solvable in $O(n \log n)$ time [1]. The *storage allocation problem* (SAP), which is strongly NP-hard, was first studied in [1, 9]. Bar-Noy *et al.* [1] presented an approximation algorithm that yields a ratio of 7, with subsequent improvements and results for subclasses of instances. The best known result is a $(2 + \varepsilon)$ -approximation algorithm due to [11]. The *bandwidth allocation problem* (BAP) is known to be strongly NP-hard, already for uniform profits [4]. The best known result for BAP is an LP-based $(2 + \varepsilon)$ -approximation algorithm due to Chekuri *et al.* [3]. Both BAP and SAP have been studied also in the *nonuniform* resource case, where the amount of available resource may change over time (see [7] and the references therein).

The *flexible* variants of SAP and BAP were introduced by Shalom *et al.* [13]. The authors study instances where each job interval J_i has a minimum and a maximum resource requirement, satisfying $0 \leq r_{\min}(i) < r_{\max}(i) \leq W$, for which they show that FBAP can be solved in polynomial time. The paper also presents a $\frac{4}{3}$ -approximation algorithm for FSAP instances in which the input graph is proper, and $r_{\min}(i) \leq \lceil \frac{r_{\max}(i)}{2} \rceil$, for all i . The paper [12] shows that FSAP is NP-hard even when $r_{\min}(i) = 0$ for all i , and presents a $(2 + \varepsilon)$ -approximation algorithm for such instances. In the more general case, where each interval has a minimum and a maximum resource requirement, FBAP and FSAP can be approximated within factor 3 and $3 + \varepsilon$, respectively [11].

2. OUR CONTRIBUTION

For FBAP we uncover an interesting relation to the *paging* problem that leads to a simple $O(n \log n)$ algorithm for *uniform* profit instances. This improves the running time of the algorithm for FBAP due to [13], which uses flow techniques.

THEOREM 1. *Algorithm Paging_FBA runs in $O(n \log n)$ time and finds an optimal allocation for FBAP instances with arbitrary $r_{\min}(i) < r_{\max}(i)$, and $p_i = 1$ for all $1 \leq i \leq n$.*

For FSAP we present approximation algorithms for several subclasses, as well as an improved hardness result.

THEOREM 2. *FSAP admits a $(\frac{5}{4} + \varepsilon)$ -approximation algorithm on any instance in which the job intervals form a proper interval graph.*

THEOREM 3. *FSAP is strongly NP-hard even if $r_{\max}(i) = 3$ and $p_i = 1$, for all $1 \leq i \leq n$, and W is not divisible by 3. We derive the best possible positive result for such instances.*

THEOREM 4. *FSAP admits a PTAS on instances where $r_{\max}(i) = \text{Max}$ for some $1 \leq \text{Max} \leq W$, and $p_i = 1$ for all $1 \leq i \leq n$.*

3. GENERALIZED FBAP AND PAGING

Consider a generalized version of FBAP, where each job J_i has also a lower bound $r_{\min}(i)$ on the amount of resource it is allocated. We show that when all jobs have the same profit per allocated resource unit, the problem can be solved by an efficient algorithm based on Belady's well known algorithm for offline paging [2]. We use *coloring* terminology when referring to the assignment of resource to the jobs, and each available resource unit is represented by a distinct color.

We view the available colors as slots in fast memory, and each job (interval) J_i as $r_{\max}(i)$ pairs of requests for pages in the main memory. Each such pair of requests is associated with a distinct page: one request at s_i and one at $e_i - 1$. We now apply Belady's offline paging algorithm: if a page remains in the fast memory between the two times it was requested, then the color that corresponds to its fast memory slot is allocated to the corresponding job. To ensure feasibility, at least $r_{\min}(i)$ colors are allocated to each J_i . The optimality of the paging algorithm implies the optimality of the solution for our FBA instance.

The algorithm, Paging_FBA, is implemented iteratively by scanning the endpoints of the intervals from left to right. When Paging_FBA scans s_i , it first assigns $r_{\min}(i)$ colors to J_i , starting with the available colors. If less than $r_{\min}(i)$

colors are available at s_i , Paging_FBA examines the intervals intersecting J_i at s_i in decreasing order of right endpoints, and feasibly decreases the number of colors assigned to the corresponding jobs and reassigns them to J_i , until J_i is allocated $r_{\min}(i)$ colors. The feasibility of the instance implies that so many colors can be reassigned.

Next, Paging_FBA allocates up to $r_{\max}(i) - r_{\min}(i)$ additional colors to job J_i . If $r_{\max}(i) - r_{\min}(i)$ colors are available at s_i , then they are assigned to J_i . Otherwise, Paging_FBA follows Belady's algorithm to potentially assign additional colors to J_i . Paging_FBA examines the intervals intersecting J_i at s_i , and in case there are such intervals with right endpoint larger than e_i , it feasibly decreases the number of colors assigned to intervals with the largest right endpoints (furthest in the future), and increases the number of colors assigned to J_i , up to $r_{\max}(i)$.

When Paging_FBA scans e_i , the right endpoint of an interval J_i , the colors assigned to J_i are released and become available.

Acknowledgments. We thank Magnús Halldórsson and Viswanath Nagarajan for valuable discussions.

4. REFERENCES

- [1] A. Bar-Noy, R. Bar-Yehuda, A. Freund, J. Naor, and B. Schieber. A unified approach to approximating resource allocation and scheduling. *JACM*, 48(5):735–744, 2000.
- [2] L. A. Belady. A study of replacement algorithms for virtual-storage computer. *IBM Systems Journal*, 5(2):78–101, 1966.
- [3] C. Chekuri, M. Mydlarz, and F. B. Shepherd. Multicommodity demand flow in a tree and packing integer programs. *ACM Transactions on Algorithms*, 3(3), 2007.
- [4] M. Chrobak, G. J. Woeginger, K. Makino, and H. Xu. Caching is hard - even in the fault model. *Algorithmica*, 63(4):781–794, 2012.
- [5] O. Gerstel. Flexible use of spectrum and photonic grooming. In *Photonics in Switching*, 2010.
- [6] N. Jain, I. Menache, J. Naor, and J. Yaniv. A truthful mechanism for value-based scheduling in cloud computing. In *SAGT*, pages 178–189, 2011.
- [7] D. Katz, B. Schieber, and H. Shachnai. Flexible resource allocation for clouds and all-optical networks. <http://arxiv.org/abs/1605.04982>.
- [8] A. W. Kolen, J. K. Lenstra, C. H. Papadimitriou, and F. C. Spieksma. Interval scheduling: A survey. *Naval Research Logistics*, 54(5):530–543, 2007.
- [9] S. Leonardi, A. Marchetti-Spaccamela, and A. Vitaletti. Approximation algorithms for bandwidth and storage allocation problems under real time constraints. In *FSTTCS*, pages 409–420, 2000.
- [10] R. Ramaswami, K. N. Sivarajan, and G. H. Sasaki. *Optical Networks: A Practical Perspective*. Morgan Kaufmann Publisher Inc., San Francisco, 2009.
- [11] H. Shachnai, A. Voloshin, and S. Zaks. Flexible bandwidth assignment with application to optical networks. In *MFCS*, 2014.
- [12] H. Shachnai, A. Voloshin, and S. Zaks. Optimizing bandwidth allocation in flex-grid optical networks with application to scheduling. In *IPDPS*, 2014.
- [13] M. Shalom, P. Wong, and S. Zaks. Profit maximization in flex-grid all-optical networks. In *SIROCCO*, 2013.