The Second International Workshop on Dynamic Scheduling Problems
Adam Mickiewicz University in Poznań, June 26-28, 2018

FPTASes for minimizing makespan of deteriorating jobs with non-linear processing times

Nir Halman [*]
Hebrew University of Jerusalem, Jerusalem, Israel

Keywords: time-dependent scheduling, non-linear processing times, FPTAS

1 Introduction

There are n independent jobs that need to be processed by a single machine. Each job J_j has basic processing time $p_j, j = 1, \ldots, n$. The jobs have a given common critical date d, after which they start to deteriorate and a common maximum deteriorating date D ($D > d$) after which they deteriorate no further. The actual processing time $a_j(t)$ of J_j depends on its start time t in the following way:

$$a_j(t) = \begin{cases} p_j, & \text{if } t \leq d, \\ p_j + w_j(\min\{t, D\} - d), & \text{otherwise,} \end{cases}$$

where $w_j \geq 0$ is the deteriorating rate of J_j. If $D < \infty$, the deterioration is called *bounded*, which is the case handled by Kovalyov and Kubiak [5]; otherwise – as considered by Cai et al. [1], Kovalyov and Kubiak [4] it is called *unbounded*. We assume that d, D, and all p_j and w_j are integral for $j = 1, \ldots, n$. If we set $L = \max_j\{p_j, w_j, D\}$ (or $L = \max_j\{p_j, w_j, d\}$ in the unbounded case) to be the largest numerical parameter in the input, then the problem (binary) size is $O(n \log L)$. The problem is to schedule the jobs to minimize the makespan, i.e., the completion time of the last job in the schedule. We shall denote this problem by Det. We shall assume that $\sum_{j=1}^{n} p_j > d$. Otherwise all jobs can start by d and the problem becomes trivial. Furthermore, there is no machine idle time in any optimal schedule. Under these conditions, there is a unique job for any given schedule that starts by d and completes after d. We call such a job *pivotal*. Any job that ends by d is called *early*. Any job that starts after d and ends by (after) D is called *tardy* (respectively, *suspended*).

Cai et al. [1] developed an $O(\frac{n^6}{\epsilon^2} \log^2 L)$ FPTAS for the unbounded case. Kovalyov and Kubiak [4] derived an $O(\frac{n^5}{\epsilon^3} \log^4 L)$ FPTAS for the unbounded case, which is faster than the FPTAS of [1] for $n >> \frac{\log^2 L}{\epsilon}$. Note that the authors erroneously

[*] Speaker, email: halman@huji.ac.il

claim their FPTAS for the bounded case (see [5, Sect. 4]). Kovalyov and Kubiak [5] addressed the bounded case and developed an $O(\frac{n^6}{\epsilon^4}\log^5 L)$ FPTAS.

In this paper, we give a simple $O\left(\frac{n^4}{\epsilon^2}\log^3 L \log\frac{n\log L}{\epsilon}\right)$ FPTAS for the unbounded case. Compared to the FPTAS by Cai et al. [1] it is faster by a factor of $\frac{n^2}{\log L}$, up to log terms (i.e., our algorithm is faster as long as $L < 2^{n^2}$). Compared to the FPTAS of Kovalyov and Kubiak [4], our FPTAS is faster by a factor of $\frac{n\log L}{\epsilon}$, up to log terms. For the bounded case, we give an $O\left(\frac{n^3\log^2 L\log(nL)}{\epsilon^2}\log\frac{n\log(nL)}{\epsilon}\right)$ FPTAS with a speedup of a factor of $\frac{n^3\log^2 L}{\epsilon^2}$, up to log terms.

2 K-approximation sets and functions

In this section, we provide an overview of the technique of *K-approximation sets and functions*, the tool we use to construct an FPTAS for DET.

Related research. Halman et al. [3] have introduced the technique of K-approximation sets and functions, and used it to develop an FPTAS for a certain stochastic inventory control problem. Halman et al. [2] have applied this tool to develop a framework for constructing FPTASes for three rather general classes of stochastic DPs. This technique has been used to yield FPTASes to various optimization problems (cf. [2]).

Notation. For any pair of integers $A \le B$, let $\{A,\ldots,B\}$ denote the set of integers between A and B. For a function $\varphi : \{A,\ldots,B\}\to\mathbb{R}$ that is not identically zero we denote $\varphi^{\min} := \min_{A\le x\le B}\{|\varphi(x)| : \varphi(x)\neq 0\}$, and $\varphi^{\max} := \max_{A\le x\le B}\{|\varphi(x)|\}$. Let t_φ be the time needed to calculate $\varphi(x)$, for any x.

K-approximation sets and functions. Let $K \ge 1$ and $\varphi,\tilde{\varphi} : \{A,\ldots,B\}\to\mathbb{R}^+$ be arbitrary functions. We say that $\tilde{\varphi}$ is a *K-approximation function of φ* if $\varphi(x) \le \tilde{\varphi}(x) \le K\varphi(x)$ for all $x = A,\ldots,B$. The following property (cf. [2, Prop. 5.1]) provides a set of general computational rules of K-approximation functions.

Property 1. ([2, Proposition 5.1]) *Let $K_i \ge 1$, let $\varphi_i,\tilde{\varphi}_i : \{A,\ldots,B\}\to\mathbb{R}^+$ and let $\tilde{\varphi}_i$ be a K_i-approximation of φ_i for $i = 1,2$. Let $\psi_1 : \{A',\ldots,B'\}\to\{A,\ldots,B\}$ be an arbitrary function and $\alpha,\beta\in\mathbb{R}^+$ be arbitrary positive reals. Then the following properties hold: (i) Linearity of approximation: $\alpha+\beta\tilde{\varphi}_1$ is a K_1-approximation function of $\alpha+\beta\varphi_1$; (ii) Summation of approximation: $\tilde{\varphi}_1+\tilde{\varphi}_2$ is a $\max\{K_1,K_2\}$-approximation function of $\varphi_1+\varphi_2$; (iii) Composition of approximation: $\tilde{\varphi}_1(\psi_1)$ is a K_1-approximation of $\varphi(\psi_1)$; (iv) Minimization of approximation: $\min\{\tilde{\varphi}_1,\tilde{\varphi}_2\}$ is a $\max\{K_1,K_2\}$-approximation of $\min\{\varphi_1,\varphi_2\}$; (v) Approximation of approximation: if $\varphi_2=\tilde{\varphi}_1$, then $\tilde{\varphi}_2$ is a K_1K_2-approximation function of φ_1.*

As DET has a non-increasing monotone structure over intervals of integers, we concentrate on definitions for K-approximation sets and functions specialized to non-increasing functions over intervals of integers (cf. [2]).

Definition 2. ([2, Definition 4.4]) *Let $\varphi : \{A, \ldots, B\} \to \mathbb{R}$ be a non-increasing function. For any subset $W \subseteq \{A, \ldots, B\}$ satisfying $A, B \in W$, the* approximation of φ induced by W *is the function $\hat{\varphi}(x) = \varphi(\max_{y \in W}\{y \leq x\}),\ \forall x \in \{A, \ldots, B\}$.*

Definition 3. ([2, Definition 4.2 and Proposition 4.5]) *Let $K \geq 1$ and let $\varphi : \{A, \ldots, B\} \to \mathbb{R}^+$ be a non-increasing function. Let $W \subseteq \{A, \ldots, B\}$ be such that $A, B \in W$ and let $\hat{\varphi}$ be the approximation of φ induced by W. We say that W is a* K-approximation set of φ *if for every $x \in \{A, \ldots, B\}$, we have $\hat{\varphi}(x) \leq K\varphi(x)$.*

The above two definitions tell us that the approximation of φ induced by a K-approximation set of φ is a K-approximation function of φ.

Proposition 4. ([2, Proposition 4.6]) *Let $\varphi : \{A, \ldots, B\} \to \mathbb{R}^+$ be a non-increasing function. Then for every $K > 1$, it is possible to compute a K-approximation set of φ of size $O(\log_K \frac{\varphi^{\max}}{\varphi^{\min}})$ in $O(t_\varphi(1 + \log_K \frac{\varphi^{\max}}{\varphi^{\min}}) \log(B - A))$ time.*

A procedure for the construction of a K-approximation function for function $\varphi : \{A, \ldots, B\} \to \mathbb{R}^+$ is stated as Algorithm 1, giving a pseudo-code of function COMPRESS which returns a non-increasing K-approximation of φ.

Algorithm 1 FUNCTION COMPRESS($\varphi, \{A, \ldots, B\}, K$)

1: find a K-approximation set W of φ over domain $\{A, \ldots, B\}$
2: **return** the approximation of φ induced by W as an array $\{(x, \tilde{\varphi}) \mid x \in W\}$ sorted in increasing order of x

By applying the calculus of approximation and the discussion above, we get the following result (see also [2, Proposition 4.5]).

Proposition 5. *Let $K_1, K_2 \geq 1$ be real numbers and let $\varphi : \{A, \ldots, B\} \to \mathbb{R}^+$ be a non-increasing function. Let $\bar{\varphi}$ be a non-increasing K_2-approximation function of φ. Then* COMPRESS*$(\bar{\varphi}, \{A, \ldots, B\}, K_1)$ returns in $O(t_\varphi(1 + \log_{K_1} \frac{\varphi^{\max}}{\varphi^{\min}}) \log(B - A))$ time a non-increasing step function $\tilde{\varphi}$ with $O(\log_{K_1} \frac{\varphi^{\max}}{\varphi^{\min}})$ steps that K_1K_2-approximates φ, and of which the query time is $t_{\tilde{\varphi}} = O(\log\log_{K_1} \frac{\varphi^{\max}}{\varphi^{\min}})$.*

3 A DP formulation

Let DET(k, s), with $d \leq s \leq d + p_k$, denote a DET problem in which the pivotal job is J_k and the start time of the earliest tardy job is s. Kovalyov and Kubiak [4] noticed that there exists $1 \leq k^* \leq n$ and s^*, where $s_V^{k^*} := \max\{d + 1, p_{k^*}\} \leq$

$s^* \le s_U^{k^*} := d + p_{k^*}$, such that any optimal solution to $\textsc{Det}(k^*, s^*)$ is an optimal solution to Det. Thus, Det reduces to solving $N^* := \sum_{k=1}^{n}(s_U^k - s_V^K + 1)$ problems $\textsc{Det}(k,s),\ s = s_V^K, \ldots, s_U^K,\ k = 1, \ldots, n$. Note that $N^* \sim nd = O(nL)$ is pseudo-polynomial in the input size, so solving (or even approximating) all N^* $\textsc{Det}(k,s)$ problems and taking the best solution results in a pseudo-polynomial algorithm. Therefore, Kovalyov and Kubiak [4] turn to approximating only a number of $\textsc{Det}(k,s)$ problems that is logarithmic in N^*.

Theorem 6. ([4, Lemma 2]) *The best solution in the family of* $(1+\frac{\epsilon}{3})$*-approximate solutions for* $\textsc{Det}(k, d + (\max\{d+1, p_k\} - d)(1+\frac{\epsilon}{3})^{i_k}),\ k = 1, \ldots, n,$ $i_k = 1, \ldots, 1 + \frac{3\log p_k}{\epsilon}$*, is a* $(1+\epsilon)$*-approximate solution for* Det.

Thus, by Theorem 6, all we need to do in order to obtain an FPTAS for Det is to design an FPTAS for $\textsc{Det}(k,s)$. Kovalyov and Kubiak [4, Theorem 1] design an FPTAS for $\textsc{Det}(k,s)$ with running time $O(n^4 \frac{\log^3 L}{\epsilon^2})$, resulting in an FPTAS for Det with running time $O(n^5 \frac{\log^4 L}{\epsilon^3})$.

To develop our FPTAS, we first show how to compute the makespan of a schedule, with pivotal job J_k and the earliest tardy job starting at s, using a set of simple recursive equations similar to the ones given in [4, page 291]. The equations will be developed for the indexing of jobs as follows. Let us index all jobs, except for the pivotal job J_0, according to Smith's rule so that $\frac{p_1}{w_1} \le \ldots \le \frac{p_{n-1}}{w_{n-1}}$ (Smith [7]). Kubiak and van de Velde [6] observe that there exists an optimal schedule, where early jobs are sequenced arbitrarily followed by the pivotal job, which in turn is followed by tardy and suspended jobs, if any. The former are sequenced in increasing order of their indices, the latter's order is arbitrary. We formulate our recursive equations for $\textsc{Det}(k,s)$ as follows. Let $M = \sum_{j=0}^{n-1} p_j + (D-d)\sum_{j=1}^{n-1} w_j = O(nL^2)$ be an upper bound on the makespan of any schedule with no idle time in the bounded case, and let $M = O(nL^n)$ be such an upper bound in the unbounded case. Denote collectively the following definitions as (1):

$$
\begin{array}{ll}
f_0(y) = s,\ g_0(y) = s - d, & y = 0, \ldots, s - p_0, \\
f_j(y) = \begin{cases} f_{j-1}(y) + p_j + w_j g_{j-1}(y), & \text{if } 0 \le y < p_j, \\ \min\{f_{j-1}(y - p_j), f_{j-1}(y) + p_j + w_j g_{j-1}(y)\}, & \text{if } y = p_j, \ldots, s - p_0, \end{cases} & \\
g_j(y) = \min\{f_j(y), D\} - d, & y = 0, \ldots, s - p_0.
\end{array}
$$

The problem is a knapsack-type problem, where the items considered to be placed in the knapsack are the jobs, the knapsack size is the time $s - p_0$ to schedule early jobs, item i size is p_i, the cost of placing an item in the knapsack is zero (it is an early job), and the cost of not placing an item in the knapsack is the time to process it as either a tardy or a suspended job. Then, $f_j(y)$ is the makespan of an optimal schedule of jobs $0, \ldots, j$, where at most y time is allocated to early jobs.

If job j is scheduled early, then it does not change the makespan, but it decreases the time allocated for the remaining early jobs, i.e., $f_j(y) = f_{j-1}(y - p_j)$. Then, $g_{j-1}(y)$ is the delay of job j when its starting time is the makespan of an optimal schedule of jobs $0, \ldots, j-1$ with at most y time allocated to early jobs. In this way, if job j is scheduled as either tardy or suspended, then it finishes at $f_j(y) = f_{j-1}(y) + p_j + w_j g_{j-1}(y)$. Note that the makespan of $\textsc{Det}(k, s)$ equals $f_{n-1}(s - p_0)$.

4 An FPTAS for the bounded case

Now, based upon (1), we are ready to state and analyze the FPTAS for $\textsc{Det}(k, s)$.

Algorithm 2 Function $\textsc{SolveDet}(k, s, \epsilon)$

1: Let $K \leftarrow \sqrt[n-1]{1+\epsilon}$, $\tilde{f}_0(\cdot) \equiv s$
2: **for** $j = 1$ to $n-1$ **do**
3: $\quad \bar{f}_j(y) \leftarrow \begin{cases} \tilde{f}_{j-1}(y) + p_j + w_j(\min\{\tilde{f}_{j-1}(y), D\} - d), & \text{if } 0 \le y < p_j, \\ \min\{\tilde{f}_{j-1}(y - p_j), \tilde{f}_{j-1}(y) + p_j + w_j(\min\{\tilde{f}_{j-1}(y), D\} - d)\}, & \text{if } y \ge p_j, \end{cases}$
4: $\quad \tilde{f}_j(\cdot) \leftarrow \textsc{Compress}(\bar{f}_j(\cdot), \{0, \ldots, s - p_0\}, K)$ $\quad$ /* $\bar{f}_j(\cdot)$ as defined in line 3 */
5: **return** $\tilde{f}_{n-1}(s - p_0)$

Theorem 7. *Function* $\textsc{SolveDet}(k, s, \epsilon)$ *computes a* $(1+\epsilon)$*-approximation for* $\textsc{Det}(k, s)$ *in* $O\left(\frac{n^2}{\epsilon} \log L \log(nL) \log \frac{n \log(nL)}{\epsilon}\right)$ *time.*

Proof. We note first that all calls to $\textsc{Compress}$ are well defined, because $\bar{f}_j(\cdot)$ are non-increasing functions.

We start by analyzing the error bound. We show by induction that $\tilde{f}_j(\cdot)$ is a K^j-approximation of $f_j(\cdot)$, $j = 0, \ldots, n-1$. The base case for $j = 0$ holds true due to the initialization of (1) and Step 1 of Algorithm 2. We assume by induction correctness for $m-1$, i.e., $\tilde{f}_{m-1}(\cdot)$ is a K^{m-1}-approximation of $f_{m-1}(\cdot)$, and prove that $\tilde{f}_m(\cdot)$ is a K^m-approximation of $f_m(\cdot)$. When the value of the index of the **for** loop is m, we get by the induction hypothesis and the calculus of approximation that $\bar{f}_m(y)$ is a K^{m-1}-approximation of $f_m(\cdot)$. Calling $\textsc{Compress}$ in Step 4, by Proposition 5 with parameters set to $K_1 = K$ and $K_2 = K^{m-1}$, we get that $\tilde{f}_j(\cdot)$ is a K^m-approximation of $f_j(\cdot)$ as needed.

Now, we analyze the running time. First, note that no actual computation is involved in Step 3, because we did not fix a value of y yet, but including this step in the algorithm helps us for the analysis. Due to Proposition 5, the running time of Step 4 is $O\big(t_{\tilde{f}_m} \log_K M \log(s - p_0)\big)$. By the definition in Step 3, we get that $t_{\tilde{f}_m} = O(t_{\bar{f}_m}) = O(\log \log_K M)$, where the second equality is due to Proposition 5. Moving to base 2 logarithm, substituting M with nL^2 and $s - p_o$ with L, using the

equation $\log K = \log \sqrt[n-1]{1+\epsilon} = O(\frac{\epsilon}{n})$ and taking into account that there are n iterations, the claimed running time follows. □

Remark. Regarding line 1 in Algorithm 2, we assume (by setting the rounding mode of the floating point unit to "round down", so we get an overestimate) the root operation takes constant time. Alternatively, we can set $K = 1 + \epsilon/2(n-1)$ and proceed as in the proof of [2, Theorem 8.2].

5 An FPTAS for the unbounded case

In this case, the upper bound on the makespan of any schedule with no idle time turns to $M = O(nL^n)$, instead of $M = O(nL^2)$, so the $\log(nL)$ terms in the running time of SolveDet for the bounded case turn to $O(n \log L)$. Therefore, we get for SolveDet the running time of $O\left(\frac{n^3}{\epsilon} \log^2 L \log \frac{n \log L}{\epsilon}\right)$, and the resulting FPTAS for Det runs in $O\left(\frac{n^4}{\epsilon^2} \log^3 L \log \frac{n \log L}{\epsilon}\right)$ time.

Acknowledgement. The author is grateful for partial support from the Israel Science Foundation (Grant 399/17).

References

[1] J-Y. Cai, P. Cai, Y. Zhu, On a scheduling problem of time deteriorating jobs, *Journal of Complexity*, **14** (1998), 190–209.

[2] N. Halman, D. Klabjan, C.-L. Li, J. Orlin, D. Simchi-Levi, Fully polynomial time approximation schemes for stochastic dynamic programs, *SIAM Journal on Discrete Mathematics*, **28** (2014), 1725–1796.

[3] N. Halman, D. Klabjan, M. Mostagir, J. Orlin, D. Simchi-Levi, A fully polynomial time approximation scheme for single-item stochastic inventory control with discrete demand, *Mathematics of Operations Research*, **34** (2009), 674–685.

[4] M. Y. Kovalyov, W. Kubiak, A fully polynomial approximation scheme for minimizing makespan of deteriorating jobs, *Journal of Heuristics*, **3** (1998), 287–297.

[5] M. Y. Kovalyov, W. Kubiak, A generic FPTAS for partition type optimisation problems, *International Journal of Planning and Scheduling*, **1** (2012), 209–233.

[6] W. Kubiak, S. L. van de Velde, Scheduling deteriorating jobs to minimize makespan, *Naval Research Logistics*, **45** (1998), 511–523.

[7] W. E. Smith, Various optimizers for single-stage production, *Naval Research Logistics Quarterly*, **3** (1956), 59–66.