Instance-Sensitive Robustness Guarantees for Sequencing with Unknown Packing and Covering Constraints*

[Extended Abstract]

Nicole Megow†
Institut für Mathematik
Technische Universität Berlin
Straße des 17. Juni 136
10623 Berlin, Germany
nmegow@math.tu-berlin.de

Julián Mestre‡
School of Information Technologies
SIT Building, J12
The University of Sydney
NSW 2006, Australia
mestre@it.usyd.edu.au

ABSTRACT

Sequencing problems with an unknown covering or packing constraint appear in various applications, e.g., in real-time computing environments with uncertain run-time availability. A sequence is called α-robust when, for any possible constraint, the maximal or minimal prefix of the sequence that satisfies the constraint is at most a factor α from an optimal packing or covering. It is known that the covering problem always admits a 4-robust solution, and there are instances for which this factor is tight. For the packing variant no such constant robustness factor is possible in general.

In this work we address the fact that many problem instances may allow for a much better robustness guarantee than the pathological worst case instances. We aim for more meaningful, instance-sensitive performance guarantees. We present an algorithm that constructs for each instance a solution with a robustness factor arbitrarily close to optimal. This implies nearly optimal solutions for previously studied problems such as the universal knapsack problem and for universal scheduling on an unreliable machine. The crucial ingredient and main result is a nearly exact feasibility test for dual-value sequencing with a given target function. We show that deciding exact feasibility is strongly $\mathcal{NP}$-hard, and thus, our test is best possible, unless $\mathcal{P}$=$\mathcal{NP}$.

We hope that the idea of instance-sensitive performance guarantees inspires to revisit other optimization problems and design algorithm tailored to perform well for each individual instance.

*Supported by the German Science Foundation (DFG) with grant ME 3825/2.

†Supported by DFG grant ME 3825/1.

‡Supported by an ARC grant DE130101664.

ITCS'13, January 9–12, 2012, Berkeley, California, USA.

Categories and Subject Descriptors

F.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and ProblemsSequencing and scheduling; F.1.2 [**Modes of Computation**]: Online computation

General Terms

Algorithms

Keywords

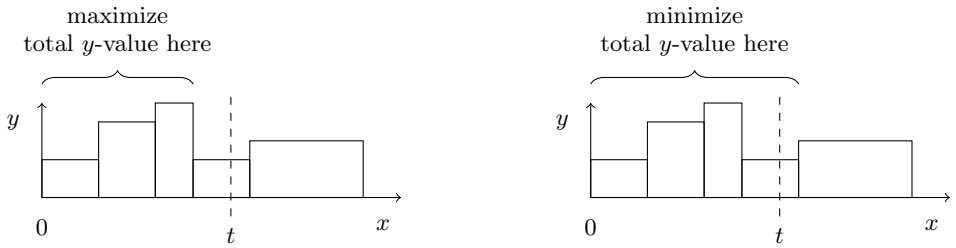
sequencing, instance-sensitive worst-case guarantee, robustness factor, universal solution, knapsack

1. INTRODUCTION

We study two complementary variants of a general *dual-value sequencing problem*, one with an unknown covering constraint, the other one with an unknown packing constraint. In both cases, we want to find a universal sequence of good quality for all realizations of the unknown constraint. Such problems find applications in real-time computing environments with uncertain run-time availability such as medical diagnosis systems, automated trading systems, and game-playing programs [1]. In particular, dual-value sequencing with an unknown covering constraint has been identified as a core subproblem when *scheduling on an unreliable machine* [7] and in the context of *online flow-time scheduling* [2].

In a geometric interpretation of the dual-value sequencing problem (see Figure 1), we are asked to linearly arrange a given set of rectangles defined by positive values x_j and y_j along the positive x-axis. In the covering variant, a solution is called α-robust, for some $\alpha \geq 1$, if, for all values $t > 0$, the total sum of y-values of rectangles covering the interval $[0, t]$ is within a factor α of the least possible total y-value of such a cover. In the packing variant, a solution is called α-robust, for some $\alpha \leq 1$, if the total sum of y-values of rectangles that can be packed into the interval $[0, t]$ is within a factor α of a maximum packing. In both cases we call α the *robustness factor* of the solution.

For the packing variant there are instances that do not admit solutions with constant robustness factor. A simple example is a two-item instance with one small square and one huge square. For the covering variant, Epstein *et al.* [7]



(a) packing constraint (b) covering constraint

Figure 1: Geometric interpretation of the dual-value sequencing problem.

designed an algorithm that returns a 4-robust solution, and showed that there are instances with optimal robustness factor arbitrarily close to 4. Notice that these are absolute bounds over *all* possible instances. Clearly many instances admit a much better robustness factor than the overall worst case. For example, the simple instance for the packing variant requires items with very large ratio of maximum and minimum weight. Also the instance with optimal robustness factor close to 4 in the covering version [7] is highly structured, it uses items with exponentially large values, and is arguably very unlikely to occur in practice. To address such limitations in the meaningfulness of a general performance guarantee, we propose to find solutions with the best robustness factor on an instance-by-instance basis. For a given input instance (a set of items), we are interested in the robustness factor achievable for this particular instance instead of over all possible instances. We call our approach *instance-sensitive robustness guarantees*.

Our main contributions are polynomial time approximation schemes that find for any instance a solution with nearly optimal robustness factor for the packing and the covering variants of the dual-value sequencing problem.

Our approach of approximating an (unknown) instance-sensitive guarantee allows us to circumvent pathological instances while still maintaining a worst-case perspective. It is worth noting that this approach can be applied to other optimization problems where a (large) absolute approximation or competitive ratio has been found to be tight for a certain subclass of instances. As our work shows, it may still be possible to get better solutions by focusing on instances-sensitive guarantees. It is worth noting that this approach has been pursued, albeit not in a systematic way, in other contexts such as low-distortion embeddings of metrics into geometric spaces, e.g., in [6], subset selection problems [14], and earliest arrival network flows [9]. We hope that our work stimulates further research on instance-sensitive worst-case guarantees for other combinatorial optimization problems.

Overview of our results.

We give nearly optimal instance-sensitive performance guarantees for dual-value sequencing with an unknown packing and covering constraint. Our main contribution is a polynomial time approximation scheme (PTAS) that constructs for any instance and any $\epsilon > 0$ a solution with a robustness factor within $1+\epsilon$ of the optimal factor for the input instance. For the covering problem, this contrasts a previous algorithm that achieves a factor of 4 for all instances [7]. For the packing variant, this is, to the best of our knowledge, the first positive result for general instances. Our algorithm also computes the unknown optimal robustness factor for a given instance up to any desired accuracy. These results imply directly instance-dependent (nearly) optimal solutions for universal scheduling on an unreliable machine to minimize $\sum w_j f(C_j)$, for non-negative, non-decreasing cost functions f, and for determining universal or incremental knapsack solutions.

The crucial ingredient of our algorithm, and the main result of the paper, is an approximate feasibility test for *dual-value sequencing with a given target function*. For a given instance J and a function $Z : \mathbb{R}^+ \to \mathbb{R}^+$, the task is to decide, if there is a sequence of rectangles such that, for any $t > 0$, the total sum of y-values of rectangles whose x-values cover (can be packed in) the interval $[0, t]$ is at most (least) $Z(t)$. Our *approximate* feasibility test outputs, for a given target function $Z(t)$ and a precision parameter $\epsilon > 0$, a sequence that achieves for any covering/packing constraint t a total y-value which is not more than a factor $1 + \epsilon$ above/below $Z(t)$, or reports that there is no sequence with a value above/below $Z(t)$ for all t. We believe that this result is of independent interest. On the negative side, we show that deciding exact feasibility for dual-value sequencing with an *arbitrary* target function is strongly $\mathcal{NP}$-hard.

The high-level idea behind our approximate feasibility test is as follows. First, the target curve $Z(t)$ is replaced with a simpler step function $\tilde{Z}$ that approximates it. Then, dynamic programming is used to assign some items with "large" y-value to each step. The remaining gaps are filled with "small" items by allowing a carefully limited fractional assignment in which such items can be split across several steps. Finally, this fractional solution is converted into a proper sequence at the expense of a small multiplicative violation of the target curve. A major hurdle in carrying out this plan is the large number of steps in $\tilde{Z}$ which prohibits to guess the placement of all large items in one shot. Instead, each DP state stores the large items in only a constant number of steps; global consistency is achieved through a careful recursive definition of the state values.

We note that standard rounding techniques for knapsack problems do not seem to be amenable to our problem: A round-off error of the y-value of an item could be at the same time negligible with respect to a tall step and prohibitive with respect to a short step; similarly it is not clear how to round the x-value without potentially significantly affecting the robustness factor of the instance. For this reason, our algorithm works directly with the input values and avoids any rounding.

Related work.

Our work falls broadly in the field of optimization under uncertainty, in particular *robust optimization* in an *online* environment, and it builds on the concepts of *universal solutions* [7, 13] and *incremental solutions* [15, 16].

The notion of α-robustness was introduced by Hassin and Rubinstein [11] in the context of cardinality-robust solutions for maximum independent set problems. A related knapsack variant with an uncertain *cardinality bound* has been studied recently in [14]. It has been shown that any knapsack instance I admits a $\nu(I)$-robust solution, where $\nu(I)$ is the rank of the knapsack system corresponding to I [11], and it is $\mathcal{NP}$-hard to decide if a given instance admits an improved or even optimal robustness factor [14]. On the positive side, there is an FPTAS that finds solutions with nearly optimal robustness factor [14]. We note that these results do not apply to our problem. Their setting is somewhat simpler than ours since it is really a subset selection problem: Once a feasible set of items has been chosen, the optimal sequence is to order them by decreasing profit.

Other robust variants of the knapsack problem have been studied extensively from various perspectives, see, e.g., [3–5, 19]. These models differ from ours as the uncertainty lies typically in item sizes and/or profits and the task is to find a fixed subset that is feasible in all scenarios. Closest to our work is a size-robust knapsack variant in which only the knapsack capacity underlies uncertainty, which has been studied in [4] from a computational point of view.

In the *universal knapsack problem* we ask for robust solutions in a different sense: Given a set of items with non-negative sizes and profits, the task is to find a permutation that maximizes the total profit of items that fully fit into the knapsack for an unknown capacity. This problem is exactly dual-value sequencing with an unknown packing constraint. The notion of incremental solutions differs from universal solutions in the limited (constant) number of scenarios for the uncertain capacity constraint. The *incremental knapsack problem* has been studied in [10]. The authors give an FPTAS for approximating the optimal robustness factor for the special case of proportional profits.

Universal sequencing on an unreliable machine refers to scheduling problems in which a given set of jobs is to be executed on a machine that may unpredictably change its processing speed or becomes fully unavailable for periods of time. The goal is to find a sequencing of the jobs that performs well under any machine behavior with respect to some cost function. The problem of minimizing $\sum_{j\in J} w_j f(C_j)$, for non-negative, non-decreasing cost functions f, has been studied in [7]. The authors propose an algorithm (with polynomial running time) that constructs a permutation that is $(4+\epsilon)$-robust for all machine behaviors. Interestingly, the solution is the same universal sequence independently of the cost function f. This result is tight, even when the particular cost function is fixed. To obtain this universal sequencing result, Epstein et al. [7] relate the problem of universally scheduling on an unreliable machine to *sequencing to minimize the remaining weight in the system*, which corresponds to the covering variant of dual-value sequencing with a cost function expressed on the reverse sequence. This problem also plays a role in online flow-time scheduling [2].

Other related problems occur in real-time systems where the available computing time is uncertain and the output quality for each job improves as the spent execution time increases. Incremental problem-solving techniques, in particular, *contract algorithms* [1, 18, 20], play an important role. However, these problems differ from ours since multiple copies of a job can be scheduled for different durations and the evaluation function that is based on the computation time that has been spent on each job.

Outline of the paper.

In Section 2 we define the two problems of dual-value sequencing with uncertain covering and with uncertain packing constraints. We also state our main results and their relation to scheduling on an unreliable machine and the universal knapsack problem. In Section 3 we describe the general algorithmic framework for our PTAS for the dual-value sequencing problem. Here we focus on the covering variant. The key routine within the PTAS is an approximate feasibility test for the dual-value sequencing problem with given target function which we provide in Section 4. This key result is best possible since deciding feasibility is $\mathcal{NP}$-hard as we show in Section 5. In Section 6 we argue how the main results can be obtained for the problem with an unknown packing constraint. We conclude with open problems and an outlook in Section 7.

2. FORMAL STATEMENT OF OUR PROBLEMS AND MAIN RESULTS

2.1 Problem definitions

An instance of a dual-value sequencing problem is a set of items $J = \{1, \dots, n\}$ with two values $x_j > 0$ and $y_j \geq 1$. Let $x(J') = \sum_{j\in J'} x_j$ and $y(J') = \sum_{j\in J'} y_j$, for any $J' \subseteq J$. We denote by $X = x(J)$ and $Y = y(J)$. For two functions F and G of t we use the shorthand notation $F \leq G$ to denote $F(t) \leq G(t)$ for all t. For a function F of t and for a scalar α, we use $\alpha \cdot F$ to denote the function mapping t to $\alpha F(t)$.

The problem of *dual-value sequencing with an uncertain covering constraint* is to minimize the cost function of accumulated y-value for any covering constraint on the x-values. More precisely, we define for a given sequence π the cost function:

$$Y^\pi(t) = \min_{j:x(\pi(\{1,\dots,j\}))\geq t} y(\pi(\{1,\dots,j\})).$$

A point-wise lower bound is $Y^*(t) = \min_\pi Y^\pi(t)$. In general there is no sequence that satisfies $Y^\pi = Y^*$. Instead, our goal is to find a sequence π such that the maximum deviation from Y^* is minimized. A sequence π is *α-robust* if $Y^\pi \leq \alpha \cdot Y^*$. A sequence is *optimal-robust* if it obtains a best possible (or optimal) robustness factor for J:

$$\alpha^*(J) = \min_\pi \max_t \frac{Y^\pi(t)}{Y^*(t)}\,.$$

The variant with *an uncertain packing constraint* is to maximize the cost function of accumulated y-value for any packing constraint on the x-values. For a sequence π, we define

$$\bar{Y}^\pi(t) = \max_{j:x(\pi(\{1,\dots,j\}))\leq t} y(\pi(\{1,\dots,j\})),$$

and let $\bar{Y}^*(t) = \max_\pi \bar{Y}^\pi(t)$ denote the function of upper bounds for all packing constraints. A sequence π is α-robust if $\bar{Y}^\pi \geq \alpha \cdot \bar{Y}^*$. A sequence is optimal-robust if it obtains an

optimal robustness factor for J:

$$\alpha^*(J) = \max_{\pi} \min_{t} \frac{\bar{Y}^{\pi}(t)}{\bar{Y}^*(t)}.$$

2.2 Main results

We efficiently find a nearly optimal-robust sequence for both the covering and packing variants.

THEOREM 1. *Given an instance J of the dual-value sequencing problem, there are polynomial time approximation schemes that compute, for any $\epsilon > 0$, a $(1+\epsilon)\cdot\alpha^*(J)$-robust covering solution and a $(1-\epsilon)\cdot\alpha^*(J)$-robust packing solution, respectively. The algorithms also compute the exact value of $\alpha^*(J)$ up to an accuracy factor of $1+\epsilon$.*

Dual-value sequencing with an uncertain covering constraint was identified as a key subproblem of universal sequencing on an unreliable machine with the objective of minimizing $\sum_{j\in J} w_j f(C_j)$ for non-decreasing, non-negative cost functions f [7]. In particular, there is a one-to-one correspondence of performance guarantees for any instance [7, Lemma 1]. Thus, our new algorithm implies directly the following nearly optimal instance-sensitive results.

COROLLARY 1. *There is a PTAS for approximating instance-wise the best possible competitive ratio for universal sequencing to minimize $\sum_{j\in J} w_j f(C_j)$ for all machine behaviors and all non-decreasing, nonnegative functions f simultaneously.*

Dual-value sequencing with uncertain packing constraint is precisely the universal knapsack problem with x-values as item sizes and y-values as profits.

COROLLARY 2. *There is a PTAS for approximating the optimal robustness factor for the universal knapsack problem.*

3. PTAS FOR DUAL-VALUE SEQUENCING WITH COVERING CONSTRAINT

For a given dual-value sequencing instance J, we know neither the optimal robustness factor $\alpha^*(J)$ nor the function Y^*. In fact, even computing a single point of Y^* is a knapsack problem by itself and its space complexity (number of different values) of Y^* can be exponential on the number of items [17]. Therefore, we use a simplified approximation of Y^* depending on the parameter $\epsilon > 0$ which we compute as follows. Let t_i^* be the smallest value such that $Y^*(t_i^*) \geq (1+\epsilon')^i$. Using an FPTAS for knapsack [12], we can find in polynomial time values $t_i \in [t_i^*, t_{i+1}^*)$. We define a new curve $Z^*(t) = (1+\epsilon')^{i+1}$ for $t \in [t_{i-1}, t_i)$, which satisfies $Y^* \leq Z^* \leq (1+\epsilon')^2 \cdot Y^*$.

Our algorithm now does a binary search on the robustness factor by testing feasibility for multiples of the curve Z^*. The crucial ingredient of our approximation scheme is an approximate feasibility test that outputs, for a given target function Z and a precision parameter $\epsilon > 0$, a sequence π such that $Y^{\pi} \leq (1+\epsilon)\cdot Z$, or reports that there is no π^* such that $Y^{\pi^*} \leq Z$. We provide such a test in Section 4, but now we show how to use it to approximate the optimal robustness factor.

THEOREM 2. *Given an instance J, $\epsilon > 0$ and a $(1+\epsilon)$-approximate feasibility test, there is a PTAS that computes a $(1+\epsilon)\cdot\alpha^*(J)$-robust solution for dual-value sequencing with an unknown covering constraint. Furthermore, the algorithm computes a value α with $\alpha^*(J) \leq \alpha \leq (1+\epsilon)\cdot\alpha^*(J)$.*

PROOF. Let $\epsilon' = \epsilon/8$. First, we compute an approximation of $Y^*(t)$ such that $Y^* \leq Z^* \leq (1+\epsilon')^2 \cdot Y^*$ as described above. Then we perform binary search for $\alpha^*(J)$ in the interval $[1, 4]$ up to the desired level of accuracy. In iteration $i \in \{1, 2, \ldots\}$ we test α_i, the mid-point of the current pinning interval, by running the approximate feasibility test with accuracy parameter ϵ' on the target function $Z_i = \alpha_i \cdot Z^*$. After $k = \lceil \log 3/\epsilon' \rceil$ iterations the length of the search interval for $\alpha^*(J)$ has reduced to ϵ'. We consider three cases.

(a) Suppose that all feasibility tests were positive. In this case, $1 \leq \alpha^*(J) \leq \alpha_k \leq 1+\epsilon'$, and thus, the solution sequence returned in last iteration is $(1+\epsilon')\cdot\alpha^*(J)$-robust. Let $\alpha = \alpha_k$.

(b) Suppose that all feasibility tests returned a negative answer. In this case we know that $4-\epsilon' \leq \alpha_k < \alpha^*(J)$. Algorithm DOUBLE in [7] is guaranteed to return a solution with robustness $4+\epsilon'$ in polynomial time. We let this be our final solution and set $\alpha = 4$.

(c) In the remaining case, let i denote the last iteration that returned a positive answer and let i' denote the last iteration with a negative answer. Then $\alpha^*(J) \in [\alpha_{i'}, \alpha_i]$ and $\alpha_i \leq (1+\epsilon')\alpha^*(J)$. Let $\alpha = \alpha_i$. The sequence π returned in iteration i satisfies

$$\begin{aligned} Y^{\pi} &\leq (1+\epsilon')\cdot\alpha\cdot Z^* \leq (1+\epsilon')^2\cdot\alpha^*(J)\cdot Z^* \\ &\leq (1+\epsilon')^4\cdot\alpha^*(J)\cdot Y^*. \end{aligned}$$

In all cases, our choice of ϵ' guarantees that the computed sequence is $(1+\epsilon)\cdot\alpha^*(J)$-robust for all $\epsilon < 4$. Furthermore, the value α approximates the optimal robustness factor arbitrarily well, i.e., $\alpha^*(J) \leq \alpha \leq (1+\epsilon)\cdot\alpha^*(J)$. □

Together with Theorem 3 in Section 4 this implies the covering result stated in Theorem 1.

4. A NEARLY EXACT FEASIBILITY TEST FOR A GIVEN TARGET FUNCTION

In this section we show that there is an approximate feasibility test for the covering variant of the dual-value sequencing problem with a given target curve.

THEOREM 3. *For any fixed $\epsilon > 0$, there is polynomial time approximate feasibility test for the dual-value sequencing problem with a given target curve Z that either reports that there is no π^* such that $Y^{\pi^*} \leq Z$, or it outputs a permutation π such that $Y^{\pi} \leq (1+\epsilon)\cdot Z$.*

In order to describe our algorithm we need to introduce the concept of *fractional sequencing* where we allow items to be split into smaller proportional items; more precisely, an item (x_j, y_j) can be split into items $(x_j^1, y_j^1), \ldots, (x_j^a, y_j^a)$ such that $x_j = x_j^1 + x_j^2 + \cdots + x_j^a$ and $x_j/y_j = x_j^1/y_j^1 = \cdots = x_j^a/y_j^a$. We will use $\hat{\pi}$ to denote a fractional sequencing of a set of items J.

Algorithm 1 APPROXIMATE FEASIBILITY TEST(J, Z, ϵ)

1. Simplify Z into a step function $\widetilde{Z}$ such that $\widetilde{Z} \leq (1+\epsilon) \cdot Z$
2. Try to find a restricted fractional sequencing $\hat{\pi}$ such that $Y^{\hat{\pi}} \leq (1+\epsilon) \cdot \widetilde{Z}$
3. **if** $\hat{\pi}$ was found **then**
4. Compute a sequencing π such that $Y^{\pi} \leq Y^{\hat{\pi}}$
5. **return** π
6. **else**
7. **return** "Z is not attainable"

Figure 2: Algorithm to approximately test feasibility

At a very high level, our algorithm consists of three steps: First, the input target curve Z is simplified into a step function $\widetilde{Z}$ where the steps are powers of $(1+\epsilon)$ such that $\widetilde{Z} \leq (1+\epsilon)Z$; second, we attempt to find a fractional sequencing $\hat{\pi}$ such that $Y^{\hat{\pi}} \leq (1+\epsilon) \cdot \widetilde{Z}$ having the property that no "large" item is split and the splitting of the remaining items obeys further restrictions (details of these restrictions are given in Section 4.2); third, from $\hat{\pi}$ we compute a proper (non-fractional) sequencing π such that $Y^{\pi} \leq (1+\epsilon) \cdot Y^{\hat{\pi}}$. If Step 2 fails, we will get a certificate showing that there is no π such that $Y^{\pi} \leq \widetilde{Z}$. Otherwise, the algorithm is guaranteed to output a solution π such that $Y^{\pi} \leq (1+\epsilon)^3 \cdot Z$. The pseudo-code for the algorithm is given in Figure 2

The next three subsections explain how each of these steps is carried out. The final subsection provides the proof of Theorem 3.

4.1 Simplifying the input target curve

As a first step in the development of our algorithm we simplify the target curve Z by turning it into a step function[1]. Let t_i be the smallest value such that $Z(t_i) \geq (1+\epsilon)^i$. We define a new curve $\widetilde{Z}(t) = (1+\epsilon)^i$ for $t \in [t_{i-1}, t_i)$. Clearly $\widetilde{Z} \leq (1+\epsilon) \cdot Z$, as desired. We consider Z to be part of the input, so computing the t_i values is trivial (in contrast to the similar approximation of the unknown curve Y^* in Section 3).

We call $[t_{i-1}, t_i)$ the ith step of $\widetilde{Z}$. Given a (perhaps fractional) sequencing, we say that an item j *is assigned to step* i, if when linearly arranging the items in the interval $[0, x(J)]$ according to the sequencing, the left endpoint of j (viewed as a rectangle) belongs to the ith step.

We say an item j is *large* for step i if $y_j > (1+\epsilon)^i/\ell$, where ℓ is set to ϵ^{-2}. Similarly, we say an item j is *medium* for step i if $y_j \in [(1+\epsilon)^i/\ell^2, (1+\epsilon)^i/\ell]$, and *small* if $y_j < (1+\epsilon)^i/\ell^2$. We denote by L_i, M_i, and S_i the set of large, medium, and small items for step i respectively. For brevity we may omit the particular step when we refer to large, medium, or small items in a high level description whenever the step in question is clear from the context.

4.2 Computing a fractional sequencing

Our aim is to find a fractional sequencing such that (i) at most one new item is split per step and (ii) such an item is not large for that step. By at most one new split item per step, we mean that at most one item can be split for the first time in this step. The total number of items that appear fractionally in the ith step is then bounded by i. We show that if there is a proper (non-fractional) sequencing π such that $Y^{\pi} \leq \widetilde{Z}$ then we can find such a fractional solution $\hat{\pi}$ where $Y^{\hat{\pi}} \leq (1+\epsilon) \cdot \widetilde{Z}$.

To find such a restricted fractional sequencing we develop a dynamic programming algorithm (DP) that assigns large (and some medium) items to steps such that (ii) is guaranteed. Based on this assignment we construct a partially fractional sequence $\hat{\pi}$ with the desired properties (i) and (ii).

Our DP states are defined by a tuple $[i, A_i, A_{i-1}, \ldots, A_{i-k}, B]$, where $A_j \subseteq M_j \cup L_j$ for $j = i-k, \ldots, i$ and $B \subseteq M_i \cup L_i$ are disjoint sets. A state can either be invalid, or valid; for the latter case we associate a finite value $T[i, A_i, A_{i-1}, \ldots, A_{i-k}, B] \geq t_i$. We set $k = \log_{1+\epsilon} \ell$, so that $(1+\epsilon)^i = \ell(1+\epsilon)^{i-k}$. The index i ranges from $k+1$ to $i_{\max} = \lceil \log_{1+\epsilon} y(J) \rceil$. Notice that with this choice of k we have the property that $S_i \cup M_i = S_{i+k}$ for all steps i.

We say that a proper or fractional sequencing *follows* the DP state $[i, A_i, \ldots, A_{i-k}, B]$ if the items in A_j are assigned to step j for $j = i, \ldots, i-k$ and items in B are assigned to step $i-k-1$ or earlier without splitting any item in $A_i \cup \cdots \cup A_{i-k} \cup B$.

The high level idea behind a DP state $[i, A_i, \ldots, A_{i-k}, B]$ is as follows. If the state is invalid, then we have a certificate that there is no proper solution π that follows the state and satisfies $Y^{\pi}(t) \leq \widetilde{Z}(t)$ for all $t \in [0, t_i]$. If the state is valid, then we will be able to produce a partial fractional solution that spans at least $[0, t_i]$ and follows the state such that the gaps left by the items $A_i \cup \cdots \cup A_{i-k} \cup B$ can be filled using items in S_i while staying under the curve $\widetilde{Z}$ during $[0, t_i]$. Furthermore, $T[i, A_i, \ldots, A_{i-k}, B]$ is the smallest total x-value of such a solution among a certain subset of solutions that includes all proper partial solutions. We spend the rest of this section formalizing this high level description.

To verify if a DP state is valid we will fill the gaps left between the large and medium items by fractionally assigning small items (details follow below). To that end, let $FK(J', x')$ be the minimum possible total y-value that we would observe in a (unrestricted) fractional solution to the knapsack-type problem where we are allowed to use items from $J' \subseteq \{1, \ldots, n\}$ and we want to cover a total x-value of x'. Notice that such a minimum value solution is achieved by a greedy algorithm that packs items in non-decreasing ra-

[1]We describe here the feasibility test for an arbitrary target function. When using it as a subroutine for the approximation scheme in Section 3, then we omit this first simplification step since the input is already a step function.

tio of $\frac{y_j}{x_j}$ for $j \in J'$, with perhaps the last item being packed fractionally.

Formal definition of DP states.

First consider a state $[k+1, A_{k+1}, \ldots, A_1, \emptyset]$. Let $Q = A_{k+1} \cup \cdots \cup A_1$. We will build a partial sequencing of items Q and fill possibly remaining gaps in each step using items in $S_{k+1} \setminus Q$, which are small in step $k+1$, as follows. First we assign the items in A_1. If $x(A_1) < t_1$, we fill the gap until t_1 using items in $S_{k+1} \setminus Q$. This is done by greedily choosing the items with the best $\frac{y_j}{x_j}$ ratio first and possibly splitting a single item at t_1. If there are not sufficiently many items in $S_{k+1} \setminus Q$ to fill the gap until t_1, or if the total y-value assigned to the first step exceeds $1+\epsilon$, then we stop and mark the state as invalid. Otherwise we continue with A_2; we place all items in A_2 in step 2 and again fill any eventual gap until t_2. This continues until either the state is declared invalid or we are done sequencing A_{k+1}. Let τ_{k+1} denote the x-span of the current partial sequencing, i.e., the total x-value of all items assigned so far. If all items A_j start in step j for $j = 1, \ldots, k+1$ and the total accumulated y-value of $A_1 \cup \cdots \cup A_j$ and the small, possibly split items is at most $(1+\epsilon)^j$ for $j = 1, \ldots, k+1$, then we declare the state valid and set $T[k+1, A_{k+1}, \ldots, A_1, \emptyset] = \tau_{k+1}$. Notice that the T-value of the DP state can equal t_i (if there was a gap in the last step) or be strictly larger than t_i (if the last item in A_k sticks beyond t_i). Otherwise, the value of the state is declared in invalid.

Now consider a state $[i, A_i, \ldots, A_{i-k}, B]$ for $i > k+1$. Then we define

$$T[i, A_i, \ldots, A_{i-k}, B] = \min_{A', B'} \tau, \tag{1}$$

where $[i-1, A_{i-1}, \ldots, A_{i-1-k}, A', B']$ is a valid DP state, and

$$\tau = \max\left\{t_i, T[i-1, A_{i-1}, \ldots, A_{i-k}, A', B'] + x(A_i)\right\},$$

provided that

$$x(S_i \setminus Q) \geq \tau - x(Q)$$

and

$$y(Q) + \mathrm{FK}(S_i \setminus Q, \tau - x(Q)) \leq (1+\epsilon)^i,$$

where $Q = A_i \cup \cdots \cup A_{i-k} \cup B$.

This last condition guarantees that the gaps left after assigning Q can be filled greedily using items in $S_i \setminus Q$ while keeping the accumulated y-value under $(1+\epsilon)^i$.

Feasibility condition.

Now suppose that there is a valid state $[i_{\max}, A_{i_{\max}}, \ldots, A_{i_{\max}-k}, B]$. In this case, we say the DP is *feasible*. The next lemma shows that if there is a proper solution that stays under $\widetilde{Z}$ then the DP is feasible.

LEMMA 1. *If there is a sequence π^* such that $Y^{\pi^*} \leq \widetilde{Z}$, then the DP is feasible.*

PROOF. Let A_i be the set of medium and large items assigned to step i in π^*. Let $B = (A_1 \cup \cdots \cup A_{i-k-1}) \setminus S_i$ and $B' = (A_1 \cup \cdots \cup A_{i-k-2}) \setminus S_{i-1}$. Let τ_i^* be the maximum of t_i and the rightmost point of items in A_i. We show by induction the state $[i, A_i, \ldots, A_{i-k}, B]$ is valid; furthermore, $T[i, A_i, \ldots, A_{i-k}, B] \leq \tau_i^*$.

The base case is $i = k+1$. Let $Q = A_{k+1} \cup \cdots \cup A_1$. Consider the following transformation to π: First, we linearly arrange the items (viewed as rectangles) according to π; second we remove all items not in Q, while keeping Q in place; third, for $j = 1, \ldots, k$, we slide items A_j as much to the left as possible without leaving their assigned step; forth, we greedily and fractionally fill the gaps in $[0, t_{k+1}]$ using items in $S_{i+1} \setminus Q$. Originally, those gaps used to be filled with items in $S_{i+1} \setminus Q$ and were larger before the shifting was performed. Since the original solution stayed under $\widetilde{Z}$ in $[0, t_i]$, so does the new partial fractional solution. Therefore, the state $[k+1, A_{k+1}, \ldots, A_1, \emptyset]$ is valid and $T[k+1, A_{k+1}, \ldots, A_1, \emptyset] \leq \tau_{k+1}^*$.

For the recursive step, $i > k+1$, let $Q = A_i \cup \cdots \cup A_{i-k} \cup B$. First we note that assuming $T[i-1, A_{i-1}, \ldots, A_{i-1-k}, B'] \leq \tau_{i-1}^*$, we can conclude that $T[i, A_i, \ldots, A_{i-k}, B] \leq \tau_i^*$ by (1), provided the state is indeed valid. As we did before, suppose we remove all items not in Q and we fractionally fill the gaps using items in $S_{k+1} \setminus Q$. Clearly the y-value accumulated at τ_i^* cannot be more than before, which was at most $(1+\epsilon)^i$ by virtue of $Y^{\pi^*} \leq \widetilde{Z}$. In other words,

$$y(Q) + \mathrm{FK}(S_i \setminus Q, \tau_i^* - x(Q)) \leq (1+\epsilon)^i,$$

so the test in (1) is successful and thus the state is valid and $T[i, A_i, \ldots, A_{i-k}, B] \leq \tau_i^*$.

This holds for all i, and in particular for $i_{\max}$; so it follows that the DP is feasible. □

Extracting a fractional solution.

If the DP is feasible then there must be a chain of states

$$\begin{aligned}
\tau_{k+1} &= T[k+1, A_{k+1}, \ldots, A_1, \emptyset] \\
\tau_{k+2} &= T[k+2, A_{k+2}, \ldots, A_2, B_{k+2}] \\
&\vdots \\
\tau_{i_{\max}} &= T[i_{\max}, A_{i_{\max}}, \ldots, A_{i_{\max}-k}, B_{i_{\max}}]
\end{aligned}$$

such that each one attains the minimum in (1) for the next state, where $B_i = (A_{i-k-1} \cup \cdots \cup A_1) \setminus S_i$.

Notice that such a chain of states specifies a sequence of disjoint sets $A_1, \ldots, A_{i_{\max}}$. Given this sequence, we build a fractional solution as follows. First we assign items in A_i to $[\tau_{i-1}, \tau_{i-1} + x(A)]$ for $i = 1, \ldots, i_{\max}$. Then, we fill the gaps using the remaining non-large items in a greedy fashion: When trying to fill a gap in step i we choose a yet-unassigned item j minimizing $\frac{y_j}{x_j}$ with $y_j \leq (1+\epsilon)^i/\ell$, that is, we restrict ourselves to items that are not large in the current step. We call this algorithm RESTRICTED FRACTIONAL KNAPSACK (RFK) because we restrict the set of items we can used in each step and we fix some items (the large and medium items in each step chosen by the chain of DP states) in advance. Notice that by this construction of the solution, an item can only be split at the end of a step and that such an item cannot be large in that step. We denote this solution as $\hat{\pi}$.

LEMMA 2. *If the DP is feasible then* RFK *produces a fractional solution $\hat{\pi}$ such that $Y^{\hat{\pi}} \leq (1+\epsilon) \cdot \widetilde{Z}$. Furthermore, in $\hat{\pi}$ at most one new item is split per step and such an item is not large in that step.*

PROOF. First we argue that the algorithm indeed produces a solution when the DP is feasible. Notice that this is

not immediately obvious since the algorithm may not have enough non-large items to fill the gaps left after assigning the items in $A_1, \ldots, A_{i_{\max}}$. Suppose, for the sake of contradiction that the algorithm cannot fill the gap at some step i, and this is the first time this happens. Let $Q = A_i \cup \cdots \cup A_{i-k} \cup B_i$. By the definition of $T[i, A_i, \ldots, A_{i-k}, B_i]$ we are guaranteed that $x(S_i \setminus Q) \geq \tau_i - x(Q)$. Since RFK successfully filled the gaps up to step $i-1$ and it is allowed to use in step i any item in $(S_i \cup M_i) \setminus Q$, it can fill also the gap in step i. We have reached a contradiction, so RFK can fill all gaps.

The fact that at most one non-large new item can be split at each step follows directly from the definition of RFK. It only remains to show that the fractional solution stays below $(1+\epsilon)\cdot\widetilde{Z}$. We show this by induction on the number of steps.

Our base case is the first state in the chain of feasible states, $T[k+1, A_{k+1}, \ldots, A_1, \emptyset]$. Let $Q = A_{k+1} \cup \cdots \cup A_1$. Notice that the procedure used to establish the validity of the state places items $A_1, \ldots, A_{k+1}$ in the same place as RFK; however, it uses items in $S_{k+1} \setminus Q$ to fill the gaps, while RFK is allowed to use items $(S_1 \cup M_1) \setminus Q = S_{k+1} \setminus Q$ in step 1, and items $S_{k+2} \setminus Q$ in step 2, etc. In other words, the greedy procedure used to establish the validity of $[k+1, A_{k+1}, \ldots, A_1, \emptyset]$ is more restricted than RFK in its choice of items to fill the gaps. Therefore, the accumulated y-value by RFK is less than the accumulated y-value by the procedure defining the state, which in turn is guaranteed to be stay below $\widetilde{Z}$ in $[0, \tau_{k+1}]$.

For the inductive case (step $i > k+1$) we show that $Y^{\hat{\pi}}(\tau_i) \leq (1+\epsilon)^{i+1}$, assuming that the property holds for earlier steps. Let $Q = A_i \cup \cdots \cup A_{i-k} \cup B_i$. By inductive hypothesis, it follows that $Y^{\hat{\pi}}(\tau_{i-k-1}) \leq (1+\epsilon)^{i-k}$. Notice that $\hat{\pi}$ places $A_{i-k} \cup \cdots \cup A_i$ in the interval $[\tau_{i-k-1}, \tau_i]$ and fills any eventual gap with items that are not large at each step. As we argued in the base case, the procedure used to define $T[i, A_i, \ldots, A_{i-k}, B_i]$ uses a greedy algorithm to fill these gaps with items in S_i and that such an algorithm is more restricted than RFK in its choice of items during the interval $[\tau_{i-k-1}, \tau_i]$. It follows that the total gain in y-value by RFK to fill the gaps in $[\tau_{i-k-1}, \tau_i]$ is at most the gain of the greedy procedure to fill the gaps in $[0, \tau_i]$, which is at most $FK(S_i \setminus Q, \tau_i - x(Q))$. Therefore,

$$\begin{aligned} Y^{\hat{\pi}}(\tau_i) &\leq (1+\epsilon)^{i-k} + y(Q \setminus B_i) + FK(S_i \setminus Q, \tau_i - x(Q)) \\ &\leq (1+\epsilon)^i/\ell + y(Q) + FK(S_i \setminus Q, \tau_i - x(Q)) \\ &\leq \epsilon^2(1+\epsilon)^i + (1+\epsilon)^i \\ &= (1+\epsilon)^{i+1} \end{aligned}$$

where the second to last inequality follows from the facts that $T[i, A_i, \ldots, A_{i-k}, B_i]$ is a valid state and $\ell = \epsilon^{-2}$. □

4.3 Turning a fractional solution into a feasible sequence

Suppose we obtained a fractional solution $\hat{\pi}$ as described in Lemma 2. Now we show how to turn it into a proper sequence by increasing the accumulate y-value by at most a factor $1+\epsilon$.

LEMMA 3. *For a given a fractional solution $\hat{\pi}$ such that $Y^{\hat{\pi}} \leq (1+\epsilon)\cdot\widetilde{Z}$, having at most one new split item per step and only for items that are not large, let π be the proper solution that orders the items by their first occurrence in $\hat{\pi}$. Then $Y^{\pi} \leq (1+\epsilon)^2 \cdot \widetilde{Z}$.*

PROOF. Consider some $t \in [t_{i-1}, t_i)$ in step i. We show that when going from $\hat{\pi}$ to π, the increase in the accumulated y-value at t is bounded by $\epsilon \cdot (1+\epsilon)^i = \epsilon \cdot \widetilde{Z}(t)$. Only items that were split in $\hat{\pi}$ and that where partially assigned in steps $1, 2, \ldots, i-1$ may contribute additional y-volume to this increase. We know that there is at most one new split item per step and that such an item is not large in the respective steps, i.e., the new split item in step a has y-value at most $(1+\epsilon)^a$. Therefore, the total increase in the accumulated y-value in step i is at most $\sum_{a=1}^{i-1}(1+\epsilon)^a/\ell \leq (1+\epsilon)^i/(\epsilon\ell)$. Since $\ell = \epsilon^{-2}$, we have that π satisfies $Y^{\pi}(t) \leq (1+\epsilon)^i + (1+\epsilon)^i/(\epsilon\ell) = (1+\epsilon)^2 \cdot \widetilde{Z}(t)$ for any t in step i, and thus for all t. □

4.4 Summarizing the feasibility test and proof of Theorem 2

In summary, our approximate feasibility test proceeds as follows. First, we simplify the target curve and compute $\widetilde{Z} \leq (1+\epsilon)Z$, for a given $\epsilon > 0$. Then we run the DP. If the DP is not feasible, then by Lemma 1 it follows that the instance is not feasible, so we can report so. Otherwise, if the DP is feasible, then by Lemma 2 we can find a fractional solution whose accumulated y-value stays below $(1+\epsilon)\cdot\widetilde{Z}$. Finally, this fractional solution can be turned into a proper sequencing π at the cost of an additional factor $(1+\epsilon)$ using Lemma 3. Thus, π satisfies $Y^{\pi} \leq (1+\epsilon)^3 \cdot Z$. Adjusting the parameter ϵ according to the desired accuracy of the test, proves one part of the claim of Theorem 3.

It remains to show that the running time of the algorithm is for any fixed $\epsilon > 0$ is polynomial in the input encoding. Rounding the target function, which we assume to be encoded polynomially in the dual-value sequencing instance J, requires polynomial time since we have to solve at most $\log_{1+\epsilon} y(J)$ knapsack problems each with running time $\mathcal{O}(n^3/\epsilon)$ [12]. The computational effort for our DP is $\mathcal{O}(\log_{1+\epsilon} y(J) n^{(k+2)\ell^2})$. To see that, observe that by definition there can fit at most ℓ^2 large and medium items in each step. In each state we store the large-and-medium-item assignment for $k+1$ steps, and we have in total $\log_{1+\epsilon} y(J)$ steps. To compute the value of a DP state of the form $[i, *]$ we need to inspect all compatible states of the form $[i-1, *]$, which involved trying all possible choices of medium and large items for step $i-k-1$. With our choices of $k = \log_{1+\epsilon} \ell$ and $\ell = 1/\epsilon^2$, the total running time is polynomial in the input encoding of J for any fixed $\epsilon > 0$. This concludes the proof of the covering part of Theorem 3. □

5. HARDNESS OF TESTING EXACT FEASIBILITY FOR A GIVEN TARGET FUNCTION

In this section we show that the $(1+\epsilon)$-approximate feasibility test in Theorem 3 is best possible. We prove that the problem of deciding exact feasibility for a given target curve is strongly $\mathcal{NP}$-hard.

THEOREM 4. *Given an instance of the dual-value sequencing problem with a given target curve T it is strongly $\mathcal{NP}$-hard to decide if there is a sequence π with $Y(t)^{\pi} \leq T(t)$ for all t. This is true even in the case of proportional values and when the encoding length of the target function is linear in the remaining input.*

PROOF. The proof is by reduction from 3-PARTITION: given a set of integers $\{a_1, \ldots, a_{3n}\}$, does there exist a partition into triples such each triplet add up to the same value A. It is well known that this problem is strongly $\mathcal{NP}$-hard even when all integers are in the range $(A/4, A/2)$ [8]. We consider only 3-partition instances having this property.

Given an instance of 3-partition, we build a dual-value sequencing instance where each number a_j induces an item with values $x_j = y_j = a_j$. The target function is set to $T(t) = A\left\lceil \frac{t}{A} \right\rceil$. It is worth pointing out that in this case the target curve T is given to us explicitly and its complexity is linear on the number of items.

It is easy to see that there is a sequence π of all items such that $Y(t)^\pi \leq T(t)$, for any $t \geq 0$, if and only if the 3-partition instance admits a partition into triples adding up to A. □

6. DUAL-VALUE SEQUENCING WITH AN UNKNOWN PACKING CONSTRAINT

In this section we argue that the methods and results in Sections 3, 4, and 5 can be obtained similarly for dual-value sequencing with unknown packing constraint.

To obtain a PTAS, we again first approximate the *lower bound* function $\bar{Y}^*$ by a step function. Let t_i^* be the *largest* value such that $Y^*(t_i) \leq (1+\epsilon)^i$. Using an FPTAS for knapsack, we can find in polynomial time values $t_i \in [t_{i-1}^*, t_i^*]$. We define $Z^*(t) = (1+\epsilon)^{i-1}$ for $t \in [t_i, t_{i+1})$. This step function lies *below* $\bar{Y}^*$; more precisely, $(1-\epsilon)^2\bar{Y}^*(t) \leq Z(t) \leq \bar{Y}^*(t)$. Again, we combine a binary search for $\alpha^*(J)$ with an approximate feasibility test for a given target function. Such a test returns for a given function Z either a sequence π with $\bar{Y}^\pi(t) \geq (1-\epsilon)\cdot Z(t)$, for all t, or it states that there is no π^* that satisfies $\bar{Y}^\pi(t) \geq Z(t)$ for all t.

To adopt the method in Section 4 for obtaining the approximate test for target function Z, we basically replace the minimization criterion by a maximization criterion: Let $FK(J', x')$ denote now the *maximum* possible total y-value in a fractional solution when packing items from $J' \subseteq \{1, \ldots, n\}$ into an interval of length x'. This maximum value is achieved by a greedy algorithm that packs items in *non-increasing* order of ratios y_j/x_j for $j \in J'$, with perhaps the last item being packed fractionally. With a DP we assign large and medium items to steps without splitting them, and we fill the gaps by greedily packing small items according to the maximized ratio y_j/x_j. This allows to argue similarly to Lemma 1 that the DP must be infeasible if there is no sequence π^* such that $Y^{\pi^*}(t) \geq Z(t)$ for all t. On the other hand, if the DP is feasible, then there is a fractional solution $\hat{\pi}$ with $\bar{Y}^{\hat{\pi}} \geq (1-\epsilon)\cdot Z$, with at most one new split item per step and such item is not large in this step. Now, we derive from $\hat{\pi}$ a proper solution π by sequencing items in order of their *last* occurrence in $\hat{\pi}$. Similarly to the proof of Lemma 3, we can show that this transformation from $\hat{\pi}$ to π decreases the total y-value accumulated by time t by at most $\epsilon\cdot(1+\epsilon)^i \leq \epsilon\cdot Z(t)$ for t in step i, i.e., $t \in [t_i, t_{i+1})$.

This gives the following result.

THEOREM 5. *There is an approximate feasibility test for the dual-value sequencing problem with a given target curve Z that either reports that there is no π^* such that $\bar{Y}^{\pi^*}(t) \geq Z(t)$, for all t, or it outputs a permutation π such that $\bar{Y}^\pi(t) \geq (1-\epsilon)\cdot Z(t)$, for all t, for any given $\epsilon > 0$ in polynomial time.*

In terms of approximation Theorem 5 is best possible since testing exact feasibility is $\mathcal{NP}$-hard.

THEOREM 6. *Given an instance of the dual-value sequencing problem with a given target curve T it is strongly $\mathcal{NP}$-hard to decide if there is a sequence π with $Y(t)^\pi \geq T(t)$ for all t. This is true even in the case of proportional values and when the encoding length of the target function is linear in the remaining input.*

PROOF. The proof is nearly identical to the proof of Theorem 4. Given an instance of 3-partition in the same notation, we construct the same dual-value sequencing instance and define the slightly modified target function $T(t) = A\left\lfloor \frac{t}{A} \right\rfloor$. Again, it is easy to see that there is a sequence π of all items such that $Y(t)^\pi \geq T(t)$, for any $t \geq 0$, if and only if the 3-partition instance admits a partition into triples adding up to A. □

We embed the test in the binary search framework. Observe that in the packing variant of dual-value sequencing, there is no upper bound on the robustness factor for all instances (as it was 4 in the covering variant). However, for each instance J holds the upper bound $\alpha^*(J) \leq y(J)$. And this robustness factor is achieved by an arbitrary sequence. Thus, we can run a standard binary search for $\alpha^*(J) \in [1, y(J)]$ such that in iteration $i \in \{1, 2, \ldots\}$ with α_i we run the approximate feasibility test with given accuracy parameter ϵ and target function $Z_i(t) = \alpha_i \in Z^*(t)$. After $k = \lceil \log y(J)/\epsilon \rceil$ iterations the length of the search interval for $\alpha^*(J)$ has reduced to ϵ. Notice that k is polynomial in the encoding length of instance J, and we can conclude with the same argumentation as in the proof of Theorem 2.

THEOREM 7. *There is a PTAS that computes a $(1-\epsilon)\cdot\alpha^*(J)$-robust solution for dual-value sequencing with an unknown packing constraint problem. Furthermore, the algorithm computes a value α with $(1-\epsilon)\alpha^*(J) \leq \alpha \leq \alpha^*(J)$.*

Combined with Theorem 2 this concludes the proof of Theorem 1.

7. CONCLUDING REMARKS

We leave as an open problem to determine the complexity of finding an optimal-robust solution for an arbitrary instance of the dual-value sequencing problem. We conjecture that the problem is at least $\mathcal{NP}$-hard—notice that it is not even clear that the problem belongs to $\mathcal{NP}$ or co-$\mathcal{NP}$.

Admittedly, our techniques are strongly tailored to the sequencing problem at hand. However, we feel that our idea of approximating an (unknown) instance-sensitive guarantee does have greater applicability. We hope that our work inspires others to revisit optimization problems where an absolute approximation or competitive ratio has been found to be tight for a certain subclass of instances. For such problems, it may still be possible to improve upon these absolute guarantees by focusing on instances-sensitive guarantees.

8. REFERENCES

[1] S. Angelopoulos and A. López-Ortiz. Interruptible algorithms for multi-problem solving. In *Proceedings of IJCAI*, pages 380–386, 2009.

[2] L. Becchetti, S. Leonardi, A. Marchetti-Spaccamela, and K. Pruhs. Online weighted flow time and deadline scheduling. *Journal of Discrete Algorithms*, 4(3):339–352, 2006.

[3] D. Bertsimas and M. Sim. The price of robustness. *Operation Research*, 52(1):35–53, 2004.

[4] P. C. Bouman, J. M. van den Akker, and J. A. Hoogeveen. Recoverable robustness by column generation. In *Proceedings of ESA*, pages 215–226, 2011.

[5] C. Büsing, A. M. C. A. Koster, and M. Kutschka. Recoverable robust knapsacks: the discrete scenario case. *Optimization Letters*, 5(3):379–392, 2011.

[6] M. Bădoiu, K. Dhamdhere, A. Gupta, Y. Rabinovich, H. Räcke, R. Ravi, and A. Sidiropoulos. Approximation algorithms for low-distortion embeddings into low-dimensional spaces. In *Proceedings of SODA*, pages 119–128, 2005.

[7] L. Epstein, A. Levin, A. Marchetti-Spaccamela, N. Megow, J. Mestre, M. Skutella, and L. Stougie. Universal sequencing on a single unreliable machine. *SIAM Journal on Computing*, 41(3):565–586, 2012.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979.

[9] M. Groß, J.-P. W. Kappmeier, D. R. Schmidt, and M. Schmidt. Approximating earliest arrival flows in arbitrary networks. In *Proceedings of ESA*, pages 551–562, 2012.

[10] J. Hartline and A. Sharp. An incremental model for combinatorial maximization problems. In *Proceedings of WEA*, pages 36–48, 2006.

[11] R. Hassin and S. Rubinstein. Robust matchings. *SIAM Journal on Discrete Mathematics*, 15(4):530–537, 2002.

[12] O. H. Ibarra and C. E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of the ACM*, 22(4):463–468, 1975.

[13] L. Jia, G. Lin, G. Noubir, R. Rajaraman, and R. Sundaram. Universal approximations for tsp, steiner tree, and set cover. In *Proceedings of STOC*, pages 386–395, 2005.

[14] N. Kakimura, K. Makino, and K. Seimi. Computing knapsack solutions with cardinality robustness. In *Proceedings of ISAAC*, pages 693–702, 2011.

[15] G. Lin, C. Nagarajan, R. Rajaraman, and D. P. Williamson. A general approach for incremental approximation and hierarchical clustering. *SIAM Journal on Computing*, 39(8):3633–3669, 2010.

[16] R. Mettu and G. Plaxton. The online median problem. *SIAM Journal on Computing*, 32(3):816–832, 2003.

[17] C. H. Papadimitriou and M. Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *Proceedings of FOCS*, pages 86–92, 2000.

[18] S. J. Russell and S. Zilberstein. Composing real-time systems. In *Proceedings of IJCAI*, pages 212–217, 1991.

[19] G. Yu. On the max-min 0-1 knapsack problem with robust optimization applications. *Operations Research*, 44(2):407–415, 1996.

[20] S. Zilberstein, F. Charpillet, and P. Chassaing. Optimal sequencing of contract algorithms. *Annals of Mathematics and Artificial Intelligence*, 39:1–18, 2003.