The Online Target Date Assignment Problem*

S. Heinz † S. O. Krumke‡ N. Megow§ J. Rambau¶ A. Tuchscherer‖
T. Vredeveld**

November 21, 2005

Abstract

Many online problems encountered in real-life involve a two-stage decision process: upon arrival of a new request, an irrevocable first-stage decision (the assignment of a specific resource to the request) must be made immediately, while in a second stage process, certain "subinstances" (that is, the instances of all requests assigned to a particular resource) can be solved to optimality (offline) later.

We introduce the novel concept of an *Online Target Date Assignment Problem* (ONLINETDAP) as a general framework for online problems with this nature. Requests for the ONLINETDAP become known at certain dates. An online algorithm has to assign a target date to each request, specifying on which date the request should be processed (e. g., an appointment with a customer for a washing machine repair). The cost at a target date is given by the *downstream cost*, the optimal cost of processing all requests at that date w. r. t. some fixed downstream offline optimization problem (e. g., the cost of an optimal dispatch for service technicians). We provide general competitive algorithms for the ONLINETDAP independently of the particular downstream problem, when the overall objective is to minimize either the sum or the maximum of all downstream costs. As the first basic examples, we analyze the competitive ratios of our algorithms for the particular academic downstream problems of bin-packing, nonpreemptive scheduling on identical parallel machines, and routing a traveling salesman.

1 Introduction

Many real-world online problems exhibit a two-stage structure. In a first stage, an immediate online action has to be taken, while in a second stage "certain offline subproblems" (which we will refer to as downstream optimization problems) can be solved to optimality offline. In this paper we provide a general framework for online problems of this type, the *Online Target Date Assignment Problem* (ONLINETDAP).

As an illustration, consider the following scenario arising in the dispatching of service technicians. When a customer calls in, requesting a maintenance service for his washing machine, one of the service technicians has to visit the customer at its location and fix the problem. This

*Supported by the DFG Research Center MATHEON *Mathematics for key technologies* in Berlin.

†Konrad-Zuse-Zentrum für Informationstechnik Berlin, Department Optimization, Takustr. 7, 14195 Berlin, Germany. E-mail: {heinz, tuchscherer}@zib.de

‡University of Kaiserslautern, Department of Mathematics, P.O.Box 3049, Paul-Ehrlich-Str. 14, 67653 Kaiserslautern, Germany. E-mail: krumke@mathematik.uni-kl.de

§Technische Universität Berlin, Institut für Mathematik, Strasse des 17. Juni 136, 10623 Berlin, Germany. E-mail: nmegow@math.tu-berlin.de

¶Universität Bayreuth, Lehrstuhl für Wirtschaftsmathematik, 95440 Bayreuth, Germany. E-mail: Joerg.Rambau@uni-bayreuth.de

‖Konrad-Zuse-Zentrum für Informationstechnik Berlin, Department Optimization, Takustr. 7, 14195 Berlin, Germany. E-mail: {heinz, tuchscherer}@zib.de

**Maastricht University, Department of Quantitative Economics, P.O. Box 616, 6200 MD Maastricht, The Netherlands. E-mail: T.Vredeveld@KE.unimaas.nl

service can be done within a certain time frame, say within a week. The customer must be given the day (and possibly a more narrow time window) when the technician will arrive, while he is on the phone and without knowledge of future service requests, that is, it must be given online. However, until the promised service day arrives, the decision which service technician to send and in which order the customers should be visited can be safely deferred. In other words, the exact scheduling and routing of service technicians for a fixed day can be done optimally offline at the night before.

In this paper, we introduce structures that account for the following dichotomy in many day-to-day resource dispatching problems: First, a resource has to be assigned to a request (e. g., assign a service vehicle to a repair request) and then the processing of all requests assigned to a certain resource can be optimized (find an optimal tour for each service vehicle). The assignment decisions influence the overall cost because they determine the input and thus the optimal costs of the single resource dispatching problems, the *downstream optimization problems*.

Offline, both stages can be integrated to obtain an overall optimal solution, even in many practical applications. However, if for each request the first decision, i.e., the assignment decision, has to be made online, the situation changes: the resulting problem is not offline anymore, but it is neither just the online version of the integrated dispatching problem; it is something in between. In stochastic programming the optimal decisions of a second stage optimization are called a *recourse*. In a way, in this paper we introduce *competitive analysis with recourse*.

Our object of study can be seen as the most extreme distinction between the online requirement of the first decision and the downstream optimization: We present a model where the first decision has to be made immediately and irrevocably before the next request is revealed (no knowledge about the input), while the downstream optimization can be carried out offline (complete knowledge about the input). The resource that has to be assigned to requests in our main actor, the ONLINETDAP, is a target date, a date at which the service should take place.

There are many variants conceivable of this concept: if the current day is allowed as a target date then the downstream optimization becomes an online problem as well, although a large portion of the data is known before the target date. It is also possible to relax the online requirement of the assignment decision: all requests on a single day might be collected, and the target dates are chosen and communicated at the end of the day. And there are, of course, variants where resources other than dates have to be assigned online (machines, vehicles) before a single resource offline problem has to be solved.

Problems of this type are abundant in reality, and very often the first decision is online. There is, however, almost no theoretical background published on this topic for the case where no stochastic information about future requests is available. And many of the stochastic models, e. g., Markov Decision Processes [5], cannot be solved for practical problem sizes. Therefore we feel that the investigation of the most basic structures in such problems seems adequate. Thus, we get started in this paper by investigating competitive online algorithms for the ONLINETDAP w. r. t. to classical downstream problems.

We think that the introduction of the ONLINETDAP will foster various lines of research, e. g., dealing with competitive analysis for ONLINETDAP w. r. t. various other, maybe more sophisticated downstream problems, with variants of the ONLINETDAP itself, but also with decision support methods for variants of the ONLINETDAP outside competitive analysis.

1.0.1 Problem description.

An instance of the ONLINETDAP consists of a sequence of requests $\sigma = r_1, r_2, \ldots$ and a *downstream problem* Π, an offline optimization problem for which arbitrary subsets of σ are feasible inputs.

Each request r_i has an integral release date $t(r_i)$ and must be assigned immediately and irrevocably to a target date in the time period $t(r_i) + 1, \ldots, t(r_i) + \delta(r_i)$, where $\delta(r_i)$ is the allowed time for deferring the service of request r_i (one week in our service technician scenario), which is also revealed upon arrival of the request. In this paper we consider only the case of uniform

downstream problem	lower bound	upper bound
bin-packing	$3/2$	2
scheduling	$\sqrt{2}$	2
traveling salesman	$\sqrt{2}$	2

Minimizing the total downstream cost (min-total objective).

downstream problem	lower bound	upper bound
bin-packing	2	$\min\{4, \delta\}$
scheduling	$3/2$	$3 - 1/\delta$
traveling salesman	2	$2\delta - 1$

Minimizing the maximum downstream cost (min-max objective).

Table 1: Main bounds on the competitive ratio of best possible deterministic online algorithms for the ONLINETDAP with a certain downstream problem minimizing the total or maximum downstream cost.

deferral times, that is, $\delta(r_i) = \delta$ for all requests r_i, where $1 \leq \delta < +\infty$. For an algorithm ALG we denote the particular date to which request r_i is assigned by $\text{ALG}[r_i] \in \{t(r_i)+1, \ldots, t(r_i)+\delta\}$.

A solution for an ONLINETDAP w. r. t. to downstream problem Π is feasible if

- each request is assigned to a feasible target date, and
- for each single target date, the corresponding instance of Π is feasible, too.

Let σ_d be the subset of requests assigned to date d by an online algorithm ALG. The optimal cost of Π on σ_d is called *downstream cost of* ALG *at date d*, and we denote it by $\text{downcost}(\sigma_d)$.

The overall online cost $\text{ALG}(\sigma)$ of an online algorithm ALG is defined as either the sum of the incurred downstream costs over all dates (min-total problems), or the maximum of the incurred downstream costs over all dates (min-max problems). The goal is to find online algorithms whose *competitive ratios* are as small as possible. An online algorithm ALG is called *c-competitive* if the cost of ALG is never larger than c times the cost of an optimal offline solution. The *competitive ratio* of ALG is the infimum over all $c \geq 1$ such ALG is c-competitive [2].

1.0.2 Our results.

The ONLINETDAP provides a general framework for a large class of online problems and gives a novel view on online optimization. We provide general competitive online algorithms for the ONLINETDAP and analyze them in greater detail w. r. t. classical combinatorial downstream problems such as bin-packing [4, SR1], nonpreemptive parallel machine scheduling [4, SS8] and the traveling salesman problem [4, ND22]. The algorithms we propose do not depend on the downstream problem (although the analysis does). We emphasize that the particular downstream problems discussed in this paper should be seen mainly as illustrating examples for the general framework. Concerning standard online investigations on these problems, [3] gives surveys on online bin-packing and scheduling; the online traveling salesman problem has been considered in [1].

Within the ONLINETDAP framework, our results are online algorithms and lower and upper bounds on their performance guarantees, the competitive ratio, obtained by classical competitive analysis for online algorithms (see, e. g. [2]). In Section 2 we present a 2-competitive algorithm for the min-total objective, i. e., the objective to minimize the total cost summed over all target dates.

In Section 3 we consider min-max problems for which the objective is to minimize the maximum downstream cost that occurs on a target date. Here, we give a general online assignment algorithm that we prove to be 4-competitive for the ONLINETDAP with the bin-packing downstream problem and which is 3-competitive for the scheduling setting. Our main results are summarized in Table 1.0.2. Finally, we observe for both objective functions that special profiles for the downstream problem, as e. g., (un-) bounded number of machines or bins per target date, lead to trivial problems or prevent any deterministic online algorithm from achieving a constant competitive ratio.

2 Minimizing total downstream cost

In this section, we consider the ONLINETDAP with the objective to minimize the total downstream cost summed up over all target dates (min-total objective). Particular downstream problems we deal with are bin-packing, scheduling on parallel machines, and the traveling salesman problem.

We first present our main competitiveness result which is an online algorithm formulated independently of the downstream problem. Let us say that a target date is *used*, if a request has been assigned to it.

Algorithm PackTogetherOrDelay (PTD) Assign a request r to the earliest date in the feasible range $t(r)+1,\ldots,t(r)+\delta$ which is already used. If no used target date is feasible for request r, then assign it to the latest feasible target date, that is, to $t(r)+\delta$.

The above algorithm always finds a feasible solution under the assumption that the amount of requests that can be assigned to the same target date is not restricted (we call this the case of *unlimited resources*). Under this assumption at any moment in time at most one feasible target date is used by PTD.

Theorem 2.1. *Consider the* ONLINETDAP *w. r. t. downstream problem* Π *with the min-total objective. Assume that there are unlimited resources in* Π *and suppose that the following properties hold for any subinstance* $\bar{\sigma}$ *of* σ*:*

i. *The optimal offline cost for the downstream problem* Π *is a monotonously increasing function, that is,* $\mathsf{OPT}(\bar{\sigma}) \le \mathsf{OPT}(\sigma)$ *(i. e.,* Π *is monotone).*

ii. *For each disjoint partition* $\sigma^{(1)},\ldots,\sigma^{(k)}$ *of the subsequence* $\bar{\sigma}$ *holds the inequality* $\mathrm{downcost}(\bar{\sigma}) \le \sum_{i=1}^{k} \mathrm{downcost}(\sigma^{(i)})$ *(i. e.,* Π *allows for synergy).*

Then, algorithm PTD *is* 2*-competitive.*

Proof. For a given sequence of requests σ consider the target dates $d_1 < d_2 < \ldots < d_k$ that PTD chooses. Denote by σ_{odd} (and σ_{even}) the subsequence of requests that the algorithm assigns to target dates d_i with odd (respective even) index i.

Observe that, if the input to PTD were solely σ_{odd} or σ_{even}, then each request would still be assigned to the same target date as when operating on σ. Therefore,

$$\mathsf{PTD}(\sigma) = \mathsf{PTD}(\sigma_{\text{odd}}) + \mathsf{PTD}(\sigma_{\text{even}}). \tag{1}$$

Moreover, we know by definition of the algorithm that the difference between any two used target dates is at least δ. Thus, the distance between any two different target dates designated for two requests of the subsequence σ_{odd} (or σ_{even}, respectively) is at least 2δ. This implies that no two requests of the same subsequence σ_{odd} (or σ_{even}, respectively) that have not been assigned to the same target date share a single feasible target date. Therefore, no algorithm can assign such two requests to the same target date. With property (ii) we conclude that

$$\mathsf{PTD}(\sigma_{\text{odd}}) = \mathsf{OPT}(\sigma_{\text{odd}}) \quad \text{and} \quad \mathsf{PTD}(\sigma_{\text{even}}) = \mathsf{OPT}(\sigma_{\text{even}}).$$

It follows with (1) and the monotonicity condition (i) that we have online cost

$$\mathsf{PTD}(\sigma) \;=\; \mathsf{OPT}(\sigma_{\text{odd}}) + \mathsf{OPT}(\sigma_{\text{even}}) \;\le\; 2\,\mathsf{OPT}(\sigma).$$

□

Note, that in the case that property (ii) only holds in a relaxed version with a factor α, i. e., $\mathrm{downcost}(\bar{\sigma}) \le \alpha \sum_i^k \mathrm{downcost}(\sigma^{(i)})$, PTD is 2α-competitive.

We will now demonstrate the power of Theorem 2.1 by applying it to various instantiations of the ONLINETDAP.

2.1 Downstream bin-packing

In bin-packing n items with sizes $s_1, \ldots, s_n$ need to be packed in unit sized bins. The objective is to find a packing such that the total size of the items packed in one bin does not exceed the bin's capacity and the total number of bins needed to pack the items is minimized. In ONLINETDAP w. r. t. bin-packing, a request $r = (t(r), s(r))$ is given by its release date $t(r)$ and its size $0 < s(r) \leq 1$. We assume that the number of available bins per day is not bounded because this would disable any deterministic online algorithm to guarantee a feasible solution. The objective is to find an assignment of requests to feasible target dates that minimizes the total sum of used bins of all target dates.

The following theorem gives a lower bound on the competitive ratio of any deterministic online algorithm.

Theorem 2.2. *No deterministic online algorithm for* ONLINETDAP *w. r. t. bin-packing minimizing the min-total objective has a competitive ratio less than* $3/2$.

Proof. The adversarial sequence starts with a request r_1 released at time 0 with size $s(r_1) < 1/2$. Consider an online algorithm, ALG, that does not assign this request to its deadline δ. Then at time ALG$[r_1]$ a second request is released with size $s(r_2) = 1 - s(r_1)$. ALG cannot assign this request to the same date as the first request and therefore it needs two bins, whereas the optimum needs only one.

Now consider an online algorithm ALG that assigns the first request to its deadline δ. Then at time $t(r_2) = 1$ a second request with size $s(r_2) = s(r_1)$ is released. If the algorithm does not pack this item with the first request, then it needs two bins and the optimum needs only one. Otherwise, at time $t(r_3) = \delta - 1$ and $t(r_4) = \delta$ two requests are released both with size $s(r_3) = s(r_4) = 1 - s(r_1)$. To pack these items, ALG needs two extra bins, thus in total three bins, whereas the optimum would pack request r_1 and r_3 to date δ and item r_2 and r_4 to $\delta + 1$, needing only two bins. ☐

Since the properties of Theorem 2.1 are met, we immediately have the following result.

Theorem 2.3. *The competitive ratio of* PTD *minimizing the total number of used bins for* ONLINETDAP *w. r. t. bin-packing is* 2.

That PTD cannot achieve a better competitive ratio than 2, can be shown by the following instance. For given $k \in \mathbb{N}, k \geq 3$, let $\varepsilon < 1/(2k-4)$ and $\sigma = \sigma^{(1)} \cup \ldots \cup \sigma^{(k)}$. $\sigma^{(1)}$ consists of the following three requests:

$$r_1 = (0, 1), \quad r_2 = (1, 1/2 - \varepsilon), \quad r_3 = (\delta, 1/2 + \varepsilon).$$

For $i = 2, \ldots, k$, the subsequence $\sigma^{(i)}$ is defined by

$$\sigma^{(i)} = ((i\delta - 1, 1/2 + (i-2)\varepsilon), (i\delta, 1/2 - (i-2)\varepsilon)).$$

The cost of PTD on this sequence is PTD$(\sigma) = 2k + 1$. On the other hand, the number of required bins of the optimal offline algorithm is OPT$(\sigma) = k + 1$. By letting $k \to \infty$, the lower bound follows.

We conjecture that the following online algorithm, PACKFIRSTORDELAY, has a better performance guarantee than PTD although the analysis for the general problem seems more difficult.

Algorithm PackFirstOrDelay (PFD) If there is a used target date to which the current request r can be assigned without increasing the number of necessary bins, then the earliest of these dates is chosen. Otherwise, assign the latest possible date, $t(r) + \delta$.

This algorithm achieves a better solution on the lower bound instance for PTD from above. However, there exist instances for which PFD performs worse than PTD, as for example: $r_1 = (0, 2/5)$, $r_2 = (0, 1/5)$, $r_3 = (0, 1/5)$, $r_4 = (\delta - 1, 2/5)$, $r_5 = (\delta - 1, 2/5)$, and $r_6 = (\delta - 1, 2/5)$.

If all items have identical size the problem becomes much easier.

Theorem 2.4. *Consider the* ONLINETDAP *w. r. t. bin-packing with the min-total objective. Then,* PFD *is optimal if all item sizes are equal.*

Proof. Assume that the bin-packing instance at each date is solved in such a way that at most one bin is partially filled. Given a sequence σ, let $\text{PFD}(\sigma) = f + p$, where f is the number of full bins and p is the number of partially filled bins. Let $d_0 < d_1 < \ldots < d_p$ be the dates on which PFD has partially filled bins. Let σ' be the subsequence consisting of all requests that are packed in a full bin and for each partially filled bin the request that opened this bin. Note that $\text{PFD}(\sigma') = \text{PFD}(\sigma)$.

We partition σ' into subsequences σ_ℓ consisting of all requests $r \in \sigma'$ with $d_{\ell-1} \le t(r) < d_\ell$. As the last request in σ_ℓ and the first request in $\sigma_{\ell+1}$ are both assigned using the delay tactic of PFD, we know that there is no overlap in the feasible target dates of requests of different subsequences. Hence, $\text{OPT}(\sigma') = \sum_\ell \text{OPT}(\sigma_\ell)$. Moreover, PFD packs the items of a subsequence in all but one fully filled bins and thus $\text{PFD}(\sigma_\ell) = \text{OPT}(\sigma_\ell)$. Combining these equalities, we get

$$\text{PFD}(\sigma) = \text{PFD}(\sigma') = \sum_\ell \text{PFD}(\sigma_\ell) = \sum_\ell \text{OPT}(\sigma_\ell) = \text{OPT}(\sigma') \le \text{OPT}(\sigma).$$

□

2.2 Downstream parallel-machine scheduling

In this section, we consider the ONLINETDAP w. r. t. nonpreemptive machine scheduling of jobs on identical parallel machines to minimize the makespan, i. e., the latest completion time of all jobs on all machines of one date. The overall objective is now to minimize the sum of makespans over all target dates. For convenience we will use standard scheduling terminology, i. e., a request r is a job that has a processing time denoted by $p(r)$. We denote a request by an ordered pair of release date and processing time, $r = (t(r), p(r))$. The number of machines available per date is denoted by m. Note, that in case $m = 1$, the problem is trivial since any target date assignment yields a total downstream cost of $\sum_{r\in\sigma} p(r)$, for any sequence σ. Therefore, we assume for the remainder of this section that more than one machine are available each date.

Consider the general online algorithm PTD. Also for this setting with the scheduling downstream problem, Theorem 2.1 applies and PTD is 2-competitive. The analysis is tight as the following sequence shows:

$$r_1 = (0, \varepsilon), \quad r_2 = (\text{PTD}[r_1] - 1, 1), \quad r_3 = (\text{PTD}[r_1], 1),$$

where $\varepsilon < 1$. The costs incurred by the algorithm are $\text{PTD}(\sigma) = 2$, whereas optimal offline costs are $\text{OPT}(\sigma) = 1 + \varepsilon$. Thus, we have shown:

Theorem 2.5. *The deterministic online algorithm* PTD *has a competitive ratio of* 2 *for the* ONLINETDAP *for downstream scheduling on identical parallel machines ($m > 1$) subject to minimize the sum of makespans induced on all target dates.*

Moreover, we obtain the following general lower bound result for this problem setting.

Theorem 2.6. *No deterministic online algorithm can achieve a competitive ratio less than $\sqrt{2}$ for the* ONLINETDAP *minimizing the total downstream cost caused by nonpreemptive scheduling on more than one machine.*

Proof. In order to obtain this bound consider for a given online algorithm ALG the following sequence:

$$r_1 = (0, 1), \quad r_2 = (\text{ALG}[r_1] - 1, 1 + \sqrt{2}).$$

If ALG assigns a target date different from $\text{ALG}[r_1]$ to request r_2, then no further requests are given. Thus, ALG's cost is $\text{ALG}(r_1, r_2) = 2 + \sqrt{2}$, whereas an offline optimum yields a solution with cost $\text{OPT}(r_1, r_2) = 1 + \sqrt{2}$, which gives a ratio of $\sqrt{2}$.

Assume that ALG assigns request r_2 to the same date as r_1, and a third request $r_3 = (\text{ALG}[r_1], 1+\sqrt{2})$ is given. Then the cost of the online algorithm is $2 + 2\sqrt{2}$, whereas the optimal offline costs are $2 + \sqrt{2}$. Again, the ratio of the incurred costs of ALG and OPT is $\sqrt{2}$. □

Note that the lower bound construction heavily depends on different processing times of jobs. Let us briefly consider the restricted setting where we assume that all requests have equal processing time. In this case, we can easily transform the ONLINETDAP w. r. t. parallel machine scheduling into an ONLINETDAP w. r. t. bin-packing: Each request $(t(r_j), p(r_j))$ is transformed into a request $(t(r_j), s(r_j) = 1/m)$, i. e., to each job corresponds an item of size $1/m$, where m is the number of machines in the scheduling problem and we assume unit bin capacity in the bin-packing problem. Both problems are equivalent; therefore the results from the previous section carry over, and thus, we have with PFD an optimal online algorithm.

Corollary 2.7. PFD *is an optimal algorithm for the* ONLINETDAP *with downstream problem scheduling of jobs with equal processing times for the min-total objective.*

2.3 Traveling salesman problem

In this section, we consider the ONLINETDAP with the downstream problem of finding a minimal tour of a traveling salesman problem, i. e., for a given set of points in a metric space (request set) a tour has to be found, from the origin through all points ending in the origin. The overall objective is now to minimize the sum of the optimal tour lengths on all target dates.

For this problem setting we provide the following general lower bound.

Theorem 2.8. *No deterministic online algorithm has a competitive ratio less than $\sqrt{2}$ for the* ONLINETDAP *w. r. t. a traveling salesman problem on $\mathbb{R}_+$ as the downstream problem minimizing the total downstream cost.*

Proof. Consider the following simple instance: At time 0, request r_1 with distance 1 from the origin is given. In order to be better than 2-competitive an algorithm has to assign the request to target date δ, because otherwise an identical request would be given at the chosen target date. Now, a second request r_2 appears at time 1 with distance $1 + \sqrt{2}$ to the origin. If the algorithm assigns it to some target date different from δ, then no more requests are released and the ratio of costs of an online algorithm to those of the optimum is $\sqrt{2}$. Otherwise, a third request at the same location of request r_2 is released at time δ. In this case the ratio of costs is $\sqrt{2}$. □

As before, the conditions in Theorem 2.1 are also met for the traveling salesman problem as the downstream problem.

Theorem 2.9. PTD *has a competitive ratio of* 2 *for the* ONLINETDAP *w. r. t. minimizing the tour length in a traveling salesman problem as a downstream problem for the min-total objective.*

In order to show that this result is tight, consider two requests released at time 0 and 1, with distances ε and 1 from the origin, respectively. Let the distance between r_1 and r_2 be equal to the sum of their distances to the origin, $1+\varepsilon$. If a third request is released at time δ in exactly the same position as r_2, then the ratio of total sum of route length for PTD to OPT tends to 2 for $\varepsilon \to 0$.

3 Minimizing maximum downstream cost

In this section, we consider ONLINETDAP subject to minimize the maximum downstream cost over all target dates for the downstream problems bin-packing, scheduling on parallel machines, and the traveling salesman problem.

As in the previous section, we firstly present a general online algorithm that is independent of the specific downstream problem.

Algorithm Balance (BAL) Assign a given request to the earliest feasible target date such that the increase in the objective value, i. e., the maximum downstream cost over all dates, is minimal.

Notice that processing each request requires BAL to solve several instances of the downstream problem optimally. However, computing optimal solutions may not be feasible under real-time aspects because of the complexity of the downstream problem. But in the analysis of our algorithm we only use such upper bounds on the offline optimum that are also satisfied by simple approximation algorithms. Therefore, all results presented in this section still hold true if the optimization is done approximately and all algorithmic computations can be accomplished in polynomial time.

3.1 Downstream bin-packing

We analyze the ONLINETDAP with bin-packing as downstream problem subject to minimizing the maximum number of used bins over all target dates. The notation and downstream problem definition is similar as in Section 2.1.

Our first result is a general lower bound on the competitive ratio of any online algorithm.

Theorem 3.1. *For the* ONLINETDAP *with min-max objective for downstream bin-packing no deterministic online algorithm has a competitive ratio of less than* 2*.*

Proof. In order to obtain this bound we consider a sequence σ with the following two first requests: $r_1 = (0, \varepsilon)$ and $r_2 = (0, \varepsilon)$ for some $\varepsilon < 1/2$.

If the considered online algorithm ALG assigns the same target date to both requests, then sequence σ is completed by the requests:

$$r_3 = (0, 1 - \varepsilon), \quad r_4 = (0, 1 - \varepsilon), \quad r_j = (0, 1) \quad 5 \leq j \leq \delta + 2.$$

Obviously, we have $\text{ALG}(\sigma) \geq 2$ and $\text{OPT}(\sigma) = 1$.

Suppose now that the online algorithm assigns different target dates to the requests r_1 and r_2, then the following additional requests are given:

$$r_3 = (0, 1 - 2\varepsilon), \quad r_j = (0, 1) \quad 4 \leq j \leq \delta + 2.$$

Again, any deterministic online algorithm is forced to open at least two bins on some date, i. e., $\text{ALG}(\sigma) \geq 2$, whereas the optimum has only cost $\text{OPT}(\sigma) = 1$. ☐

Next we analyze the algorithm BAL for the ONLINETDAP with downstream bin-packing.

Theorem 3.2. *The algorithm* BAL *is* 4*-competitive for the* ONLINETDAP *with downstream bin-packing subject to minimizing the maximum number of used bins over all target dates.*

Proof. The crucial observation is the following: Given a request r, the total size of all items assigned by BAL within the time frame $t(r) + 1, \ldots, t(r) + \delta$ is bounded from below by half the number of bins required, whenever more than one bin is used in this period of dates.

This claim can be shown by induction on the number of requests assigned to any of the considered dates. Obviously, the claim holds when none of the considered dates has yet been used. Assume that the claim is true after k requests have been assigned to the dates $t(r) + 1, \ldots, t(r) + \delta$ and let r_{k+1} be another request. If $s(r_{k+1}) \geq 1/2$, the claim obviously also holds after assigning r_{k+1}. So assume that $s(r_{k+1}) < 1/2$. If BAL can assign r_{k+1} to some date without increasing the number of used bins at that date, we are also done. But if BAL needs to use a new bin at the assigned date, we know that previously the load of each bin at the dates $t(r) + 1, \ldots, t(r) + \delta$ was at least $1 - s(r_{k+1}) > 1/2$, which proves the claim.

Now we can prove that BAL is 4-competitive. Let r_k be the first request in a given sequence σ such that the maximum downstream cost is attained, i. e., $\text{BAL}(r_1, \ldots, r_k) = \text{BAL}(\sigma)$. Notice that the assigned target date for r_k is $\text{BAL}[r_k] = t(r_k) + 1$. Let $\bar{\sigma}$ be the subsequence of all requests from σ up to r_k that have been assigned a target date $d \geq t(r_k) + 1$. On the one hand, we have:

$$\text{OPT}(\sigma) \geq \frac{1}{2\delta - 1} \sum_{r \in \bar{\sigma}} s(r) > \frac{1}{2\delta} \sum_{r \in \bar{\sigma}} s(r). \tag{2}$$

On the other hand, BAL uses in total $\delta(\mathsf{BAL}(\sigma)-1)+1$ bins in the time period from $t(r_k)+1$ to $t(r_k)+\delta$. Since we may assume $\mathsf{BAL}(\sigma)>1$ (otherwise there is nothing to show), the sum of all item sizes assigned to theses dates is at least half the number of bins required by BAL. This implies,

$$\frac{1}{2}\left(\delta(\mathsf{BAL}(\sigma)-1)+1\right) \leq \sum_{r\in\bar{\sigma}} s(r).$$

Together with (2), we can bound the cost of BAL by

$$\mathsf{BAL}(\sigma) \leq \frac{2}{\delta}\sum_{r\in\bar{\sigma}} s(r) + 1 - \frac{1}{\delta} < 4\,\mathsf{OPT}(\sigma) + 1 - \frac{1}{\delta}.$$

Finally, the integrality of $\mathsf{BAL}(\sigma)$ and $\mathsf{OPT}(\sigma)$ gives $\mathsf{BAL}(\sigma) \leq 4\,\mathsf{OPT}(\sigma)$. □

For small values δ the FIRSTFIT Algorithm that assigns a given request r to its earliest feasible target date $t(r)+1$, improves the competitiveness result of Theorem 3.2. It is easy to see that FIRSTFIT has a competitive ratio of δ.
As in Section 2.1, the situation improves significantly for equal item sizes.

Theorem 3.3. *The algorithm* BAL *is* 2*-competitive for the* ONLINETDAP *with downstream bin-packing subject to minimizing the maximum number of used bins over all target dates if all requests have equal sizes.*

Proof. Let r_k be the first request in a given sequence σ such that the maximum downstream cost is attained, i. e., $\mathsf{BAL}(r_1,\ldots,r_k)=\mathsf{BAL}(\sigma)$. Moreover consider on date $t(r_k)+1$ an optimal packing which only uses one bin partially. With respect to such an optimal packing all bins at the dates $d>t(r_k)$ except one on the date $t(r_k)+1$ are filled with a maximum number of items, because of equal item sizes. Since OPT requires the same number of bins distributed onto at most $2\delta-1$ dates, we have

$$\mathsf{OPT}(\sigma) \geq \frac{1}{2\delta-1}\delta(\mathsf{BAL}(\sigma)-1) > \frac{1}{2}(\mathsf{BAL}(\sigma)-1).$$

This implies $\mathsf{BAL}(\sigma) < 2\,\mathsf{OPT}(\sigma)+1$, which gives the theorem by the integrality of $\mathsf{BAL}(\sigma)$ and $\mathsf{OPT}(\sigma)$. □

Theorem 3.4. *For the* ONLINETDAP *with min-max objective for downstream bin-packing where all requests have equal sizes, no deterministic online algorithm has a competitive ratio of less than* $3/2$.

Proof. Let s denote the size of all requests, and consider an arbitrary online algorithm ALG and the following sequence σ of requests. $\delta\lfloor 1/s\rfloor$ requests are given at date 0. In order to achieve a competitive ratio better than 2, ALG must not use more than one bin each date. Next, at date 1 additionally $(\delta+2)\lfloor 1/s\rfloor$ requests are given, which gives $\mathsf{ALG}(\sigma)\geq 3$ and $\mathsf{OPT}(\sigma)=2$.

□

3.2 Downstream parallel-machine scheduling

In this section we consider the ONLINETDAP w. r. t. nonpreemptive machine scheduling on parallel machines subject to minimize the maximum makespan over all target dates. Notations and the exact downstream problem definition is used as in Section 2.2. Note, that if an infinite number of machines is available at each date, i. e., $m=\infty$, then the problem becomes trivial since any feasible solution yields a downstream cost of $\max_{r\in\sigma} p(r)$, for any sequence σ. In the following we assume a bounded number of machines.

In this problem setting where the number of available machines per date is bounded ($m<\infty$) the following instance shows a lower bound of $3/2$ on the competitive ratio of any deterministic online algorithm. Given $m\delta$ requests with release date 0 and processing time 1, only an algorithm ALG that assigns m jobs to each date can be better than 2-competitive. However, at date 1 are given $m(\delta+2)$ more requests with processing time 1, then ALG has a makespan of at least 3

whereas the optimum makespan over all dates equals 2. Note that this request sequence contains only requests with equal processing time. Thus, we have shown the following:

Theorem 3.5. *No deterministic online algorithm can achieve a competitive ratio less than $3/2$ for the* ONLINETDAP *w. r. t. scheduling to minimize the maximum makespan over all target dates, where the number of available machines per date is bounded and the processing times for all requests are equal.*

We next prove that the general algorithm BAL for the ONLINETDAP w. r. t. scheduling on parallel machines is $(3 - 1/\delta)$-competitive.

Theorem 3.6. BAL *is $(3 - 1/\delta)$-competitive for the* ONLINETDAP *with downstream scheduling to minimize the maximum makespan over all target dates for a bounded number of available machines per date.*

Proof. Consider a request sequence σ served by BAL and let r denote the first request that causes the maximum makespan. Consider the schedule obtained by BAL before r is released with respect to the offline optimum and let w denote the load of a least loaded machine over all feasible target dates.

Then, the BAL's makespan is at most $w + p(r)$. Since all feasible target dates for r have load of at least wm, the total load in that time period is at least $wm\delta + p(r)$.

Any of the corresponding requests in that time period could not be issued earlier than δ dates before the release date of request r. Hence, even an optimal offline algorithm OPT obeying feasibility conditions has at least the following cost on sequence σ:

$$\mathsf{OPT}(\sigma) \geq \frac{wm\delta + p(r)}{(2\delta - 1)m} > \frac{w\delta}{2\delta - 1}.$$

Hence, we have:

$$w < \left(2 - \frac{1}{\delta}\right) \mathsf{OPT}(\sigma).$$

Since $\mathsf{OPT}(\sigma)$ is bounded from below by $p(r)$, we conclude

$$\mathsf{BAL}(\sigma) \leq w + p(r) < \left(2 - \frac{1}{\delta}\right) \mathsf{OPT}(\sigma) + \mathsf{OPT}(\sigma) = \left(3 - \frac{1}{\delta}\right) \mathsf{OPT}(\sigma).$$

□

The following sequence σ shows for $\varepsilon \to 0$ that BAL is not better than 2-competitive:

$$r_i = \begin{cases} (0, 1/2 + \varepsilon) & \text{if } i \in \{1, \dots, m(\delta - 1)\}, \\ (0, 1) & \text{if } i \in \{m(\delta - 1) + 1, \dots, m\delta\}, \\ (\delta - 1, 1) & \text{if } i \in \{m\delta + 1, \dots, m(2\delta - 1) + 1\}. \end{cases}$$

Note, that this lower bound construction is based on jobs with different processing times. Now, let us briefly consider the restricted setting where we assume that all requests have equal processing time. Then, the downstream problem scheduling is equivalent to the bin-packing problem of uniform items as we described in Section 2.2. Hence, the results from the previous section carry over.

Corollary 3.7. *The algorithm* BAL *is 2-competitive for the* ONLINETDAP *with min-max objective for downstream scheduling if all jobs have identical processing times. Furthermore, no deterministic online algorithm can achieve a competitive ratio of less than $3/2$ in this setting.*

3.3 Traveling salesman problem

In this section we analyze the traveling salesman problem as downstream problem for the ONLINETDAP with objective to minimize the maximum downstream cost. Similar to the downstream problems considered before, the algorithm BAL is trivially $(2\delta - 1)$-competitive since the requests assigned to the date at which the maximum tour length is attained can at most be spread over $2\delta - 1$ dates. On the other hand, we have the following lower bound on the competitive ratio of any online algorithm.

Theorem 3.8. *No deterministic online algorithm for the* ONLINETDAP *w. r. t. the traveling salesman problem as downstream problem minimizing the maximum tour length achieves a competitive ratio less than* 2*.*

Proof. Consider a metric space induced by the unweighted star graph with at least $\delta + 1$ leaves. First, δ requests in δ different leaves are given at date 0. In case an algorithm ALG assigns more than one request to one date, it cannot be better than 2-competitive. Otherwise, let r be the request with $\mathrm{ALG}[r] = 1$. At date 1 another request associated with the point not yet used is released as well as a request for the point of request r, yielding $\mathrm{ALG}(\sigma) \geq 2$. In contrast, $\mathrm{OPT}(\sigma) = 1$ since OPT is able to assign both requests for the same point to the same target date. ☐

References

[1] Giorgio Ausiello, Esteban Feuerstein, Stefano Leonardi, Leen Stougie, and Maurizio Talamo, *Algorithms for the on-line travelling salesman.*, Algorithmica **29** (2001), no. 4, 560–581.

[2] Allan Borodin and Ran El-Yaniv, *Online computation and competitive analysis*, Cambridge University Press, 1998.

[3] Amos Fiat and Gerhard J. Woeginger (eds.), *Online algorithms, the state of the art*, Lecture Notes in Computer Science, vol. 1442, Springer, 1998.

[4] Michael R. Garey and David S. Johnson, *Computers and intractability (a guide to the theory of NP-completeness)*, W.H. Freeman and Company, New York, 1979.

[5] Martin L. Putermann, *Markov decision processes*, Wiley Interscience, 2005.