Dynamic Multi-Period Routing With Two Classes

Sven O. Krumke* Christiane Zeck*

Abstract

In the Dynamic Multi-Period Routing Problem, one is given a new set of requests at the beginning of each time period. The aim is to assign requests to dates such that all requests are fulfilled by their deadline and such that the total cost for fulling the requests is minimized. We consider a generalization of the problem which allows two classes of requests: The 1^{st} class requests can only be fulfilled by the 1^{st} class server, whereas the 2^{nd} class requests can be fulfilled by either the 1^{st} or 2^{nd} class server. For each tour, the 1^{st} class server incurs a cost that is α times the cost of the 2^{nd} class server, and in each period, only one server can be used. At the beginning of each period, the new requests need to be assigned to service dates. The aim is to make these assignments such that the sum of the costs for all tours over the planning horizon is minimized.

We study the problem with requests located on the nonnegative real line and prove that there cannot be a deterministic online algorithm with a competitive ratio better than α. However, if we require the difference between release and deadline date to be equal for all requests, we can show that there is a $\min\{2\alpha, 2 + 2/\alpha\}$-competitive algorithm.

Keywords: vehicle routing, multi-period optimization, online algorithms, competitive analysis

1 Introduction

We consider a vehicle routing problem similar to the one studied by Angelelli et al. [2]. In the *Dynamic Multi-Period Routing Problem (DMPRP)* there is a finite number of time periods $t = 1, \dots, T$. At the beginning of each time period, a new set of requests is released. Each request r_i has a *release date* $d(r_i)$ and a *deadline date* $D(r_i)$ and

*Department of Mathematics, University of Kaiserslautern, Paul-Ehrlich-Str. 14, 67663 Kaiserslautern, Germany. {krumke,zeck}@mathematik.uni-kl.de

must be fulfilled in one of the periods $d(r_i), \dots, D(r_i)$. The allowed time for deferring the requests, i.e., the difference between deadline date and release date, is called *deferral time* $\delta(r_i) = D(r_i) - d(r_i)$. Fulfilling a request means that a vehicle has to go to the request's location in the plane. For this purpose, there is a single vehicle (also called server) available which has to return to the depot at the end of each time period.

At the beginning of each period, when new requests become known, they have to be assigned irrevocably to a feasible *target date*, i.e., the date when they will be fulfilled. On a certain target date, the vehicle fulfills all requests assigned to that date in one Traveling Salesman Tour.

The goal is to make the assignment of requests to target dates such that the total distance traveled over the planning horizon is minimal.

Note that the assignment of requests to target dates has to be made without the knowledge of future requests, i.e., in an online manner, whereas the planning of the Traveling Salesman Tour for a specific day is performed offline.

There has been done a fair amount of research about Multi-Period Vehicle Routing Problems. Ausiello et al. [3] consider the Vehicle Routing Problem with release times. Angelelli et al. [2] introduce the DMPRP and assume deferral times of 1 while allowing requests in the first and last period that have a deferral time of 0. They show that the algorithms IMMEDIATE and DELAY that serve all requests as soon as they arrive or as late as possible, respectively, have a competitive ratio of 2. For the case that requests are located on the nonnegative real line, they present the algorithm SMART which they prove to be $\sqrt{2}$-competitive, and hence optimal if the number of time periods is 2. In their paper [1], Angelelli et al. show that for an arbitrary horizon length T, the competitive ratio of SMART is 1.5 for requests located on the nonnegative real line. Heinz et al. [6] consider a more general framework called the Online Target Date Assignment Problem which also comprises the DMPRP with customers located in the Euclidean plane. They present the algorithm PackTogetherOrDelay which is 2-competitive for the DMPRP with uniform deferral times. Gassner et al. [5] prove that under certain conditions, the algorithm SMART by Angelelli et al. [2] is 1.8284-competitive for the Online Target Date Assignment Problem and they present the algorithm CLEVER with a competitive ratio of 1.5.

We consider the DMPRP with requests located on the nonnegative real line $\mathbb{R}^+$, the depot being the origin. In this setting, the length of the tour for one day is determined by the maximum distance of a request to the origin: The vehicle goes to this most distant location and back to the origin. We study the following generalization of the DMPRP: In the DMPRP with two classes, each request is either a 1^{st} class request or a 2^{nd} class request. We will denote this by $cl(r_i) = 1^{st}$ or $cl(r_i) = 2^{nd}$, if r_i is a 1^{st} or 2^{nd} class request, respectively. According to that, we have two servers: The 1^{st} class server can fulfill both 1^{st} and 2^{nd} class requests, whereas the 2^{nd} class server can only serve 2^{nd} class requests. We assume that the 1^{st} class server is more expensive, i.e., the cost of a tour by the 1^{st} class server incurs a cost that is α times the cost incurred by the 2^{nd} class server and $\alpha > 1$. Furthermore, we will assume that on each day t at most one server can be used. The aim is to make assignments of requests to target dates such that the cost occuring over the planning horizon is minimized. In this setting, a request r_i can be characterized by a tuple $(cl(r_i)|t(r_i), T(r_i) \,|\, dist(r_i))$ consisting of its class, earliest and latest feasible target date and its distance from the origin.

An application of the DMPRP can be found in the service delivery management of a company. Customers call in and ask for a service which the company, by contract, has to fulfill within a certain period of time. Such a procedure is common in roadside assistance for example, see [7]. As for the two classes, the requests may be different in what it takes to fulfill them. There might be normal requests that can be fulfilled using the usual tools and by every staff member, and there might be special requests that can only be fulfilled by an expert using special equipment, which certainly leads to a price difference.

In a more general sense, one could also perceive a similarity of the DMPRP with two classes to metrical task systems. In the DMPRP with two classes, the type of server used to fulfill requests in one period could be interpreted as the "state" of the service in the sense of a metrical task system. Hence, in both models, the cost for fulfilling requests directly depends on the current state and it is possible to change states.

In this paper, we provide competitive analysis for the DMPRP with two classes. For the basics on competitive analysis, we refer the reader to the book of Borodin et al. [4]. In Chapter 2, we show that, in general, there cannot be a an algorithm which beats the triviality barrier if deferral times are arbitrary. In Chapter 3, we give a $\min\{2\alpha, 2+2/\alpha\}$-competitive algorithm for the case of uniform deferral times.

2 A Lower Bound on the Competitive Ratio

One possible approach to the DMPRP with two classes is to simply ignore the classes and to use algorithms solving the corresponding DMPRP with one class to make the assignments of requests to target dates. Then, on dates to which only 2^{nd} class requests have been assigned, the cheaper 2^{nd}nd class server can be used, otherwise, the 1^{st} class server has to be used.

Theorem 1. If an online algorithm for the DMPRP with one class is c-competitive, then it is αc-competitive for the DMPRP with two classes.

Proof. Given an instance σ of the DMPRP with two classes, let σ^1 be the corresponding instance such that all requests are 1^{st} class requests and σ^2 the corresponding instance with only 2^{nd} class requests. Then, for any online algorithm ALG, we have

$$ALG(\sigma) \leq ALG(\sigma^1) \leq \alpha ALG(\sigma^2) \leq \alpha c OPT(\sigma^2) \leq \alpha c OPT(\sigma),$$

which proves the claim. □

Thus, we can easily obtain a 2α-competitive algorithm for the DMPRP as considered by Angelelli et al. [2] extended to two classes by applying one of the algorithms IMMEDIATE or DELAY which serve all requests as soon or as late as possible, respectively. In fact, for the scenario considered by Angelelli et al. [2], which, besides requests with deferral time 1, allows requests that have to be fulfilled in the same period in which they appear, there cannot be a deterministic online algorithm with a competitive ratio smaller than α as the following theorem shows.

Theorem 2. No deterministic online algorithm for the DMPRP with two classes can have a competitive ratio smaller than α.

Proof. Consider the following instance σ. In period 1, two requests $r_1 = (1^{st}|1, 2|1)$ and $r_2 = (2^{nd}|1, 2|b)$ with $b > 1$ appear. Let $2\alpha b > 2\alpha + 2b$, i.e., it is cheaper to serve the two requests separately. Any deterministic online algorithm ALG has to choose one of the following strategies.

- If ALG serves both requests on the same day, let no new request appear in period 2. Then, the optimal strategy would have been to serve them separately, and hence,

$$\frac{\text{ALG}(\sigma)}{\text{OPT}(\sigma)} = \frac{2\alpha b}{2b + 2\alpha} \overset{b\to\infty}{\to} \alpha.$$

- If r_1 is served immediately and r_2 is postponed, let request $r_3 = (1^{st}|2, 2|1)$ appear. Then, delaying r_1 and serving r_2 immediately would have yielded the best outcome.

$$\frac{\text{ALG}(\sigma)}{\text{OPT}(\sigma)} = \frac{2\alpha + 2\alpha b}{2b + 2\alpha} \overset{b\to\infty}{\to} \alpha.$$

- If r_1 is postponed and r_2 is served immediately, let $r_3 = (2^{nd}|2, 2|b)$. In this case, serving r_1 immediately and postponing r_2 would have been the best decision and therefore,

$$\frac{\text{ALG}(\sigma)}{\text{OPT}(\sigma)} = \frac{2b + 2\alpha b}{2\alpha + 2b} \overset{b\to\infty}{\to} 1 + \alpha.$$

This shows that ALG cannot be better than α-competitive. □

3 Algorithms for the Case of Uniform Deferral Times

From now on, we assume uniform deferral times, i.e., $D(r_i) - d(r_i) = \delta$ for all i. Thus, the time between learning about a request and its deadline date is equal for all requests.

The following algorithm by Heinz et al. [6] has a competitive ratio of 2 for the DMPRP with one class, customers on the real line and uniform deferral times.

ALG 1: PackTogetherOrDelay (PTD). Assign a request r_i to the earliest date in the feasible range $d(r_i), \dots, D(r_i)$ to which a request has already been assigned. If no such date is feasible for request r_i, assign it to its deadline date $D(r_i)$.
On a specific target date, use the cheapest possible server, i.e., use the 2^{nd} class server if only 2^{nd} class requests have to be served, and the 1^{st} class server else.

Now that we have two classes of requests and servers, the following holds.

Theorem 3. The competitive ratio of PTD for the DMPRP with two classes is exactly 2α.

Proof. By Theorem 1, since PTD is 2-competitive for the case of one server, it is 2α-competitive for the problem with two servers. The following instance σ shows that 2 is actually the competitive ratio. Let

- $r_1 = (1^{st}|1, \delta|\epsilon)$,
- $r_2 = (2^{nd}|2, \delta + 1|1)$,
- $r_3 = (2^{nd}|\delta + 1, 2\delta|1)$,
- $r_4 = (1^{st}|\delta + 1, 2\delta|\epsilon)$,

where $0 < \epsilon \leq 1$.

PTD assigns r_1 and r_2 to period $t = \delta$, r_3 and r_4 are assigned to period $t = 2\delta$. Both times, the 1^{st} class server has to be used. Hence, a cost of $2\alpha + 2\alpha$ is incurred. A better strategy, however, is to serve r_1 in period $t = 1$ with the 1^{st} class server, r_2 and r_3 in period $t = \delta + 1$ with the 2^{nd} class server and r_4 in $t = \delta + 2$ with the 2^{nd} class server at a total cost of $2\alpha\epsilon + 2 + 2\alpha\epsilon$. So we get the ratio

$$\frac{\text{PTD}(\sigma)}{\text{OPT}(\sigma)} = \frac{2\alpha + 2\alpha}{2\alpha\epsilon + 2 + 2\alpha\epsilon}$$

which approaches 2α as ϵ approaches 0. □

PTD is based on the following properties of the DMPRP with one class: It is efficient to fulfill requests only on deadline dates. Any request that is scheduled before a deadline date can easily be postponed to the next deadline date without increasing the cost. The reason for this is closely related to the second property: It is advantageous to aggregate requests since the cost for a tour is only determined by the request furthest away from the origin, whereas the other requests are fulfilled at no additional cost.

For the DMPRP with two classes, however, the situation is different. The instance presented in the previous proof makes it clear that it can be reasonable to serve 1^{st} and 2^{nd} class requests separately with different servers. In case they have the same deadline date, this requires serving one of them before a deadline date. The following Theorem shows that we still have properties similar to those in the one class case.

Theorem 4. Let σ be a given instance of the DMPRP with two classes of requests and servers. Let $\mathcal{D}$ be the set of all occurring deadline dates. Then there exists an optimal offline solution which only fulfills requests on days t such that $t \in \mathcal{D}$ or $t + 1 \in \mathcal{D}$, i.e., on deadline dates or one day before a deadline date.

Proof. First, consider the following situation: let t be a date with $t \notin \mathcal{D}$ but $t+1 \in \mathcal{D}$ a day before a deadline but not a deadline itself, and consider an optimal assignment that uses both, t and $t+1$. Then, the assignment must be such that servers of different classes are used in period t and $t+1$. Otherwise it would be cheaper to serve all of them together on day $t+1$.

Now, consider an optimal assignment that assigns a subset of requests $\bar{\sigma} \subseteq \sigma$ to a day t with $t, t+1 \notin \mathcal{D}$. Then the requests in $\bar{\sigma}$ could as well be postponed to the next deadline date or the day before, depending on which server is needed for them, without affecting feasibility. Serving the requests $\bar{\sigma}$ together with other requests requiring the same server, in consequence of moving them to a new target date, does not incur any extra cost. This shows how an optimal assignment can be transformed into one that only uses target dates t with $t \in \mathcal{D}$ or $t+1 \in \mathcal{D}$. □

Based on the above structural results, the next algorithm follows the strategy of assigning requests of different classes only to different target dates.

ALG 2. For $k \geq 0$, define the interval

$$I_k = [k\delta + 1,\, (k+1)\delta].$$

- If k is even: Serve all 1^{st} class requests released in I_k in period $t = (k+1)\delta$ with the 1^{st} class server, and all 2^{nd} class requests in period $t = (k+1)\delta + 1$ with the 2^{nd} class server.
- If k is odd: Serve all 2^{nd} class requests released in I_k in period $t = (k+1)\delta$ with the 2^{nd} class server, and all 1^{st} class requests in period $t = (k+1)\delta + 1$ with the 1^{st} class server.

ALG 2 divides the planning horizon in intervals comprising δ days each. Since we have uniform deferral times of δ, the earliest possible deadline date of a request released in I_k is $k\delta + 1 + \delta = (k+1)\delta + 1$. ALG 2 serves all requests released in I_k before or in that period and thus yields feasible assignments for each k.

For requests released in I_k two target dates are used. Note that the second target date used for requests released in I_k is $(k+1)\delta + 1$, whereas the first target date used for requests released in the next interval I_{k+1} is $(k+2)\delta$. So, they coincide for a deferral time of $\delta = 1$. The distinction between odd and even values of k in the algorithm makes sure that even if those target dates coincide, the algorithm does not assign requests of different classes to the same date.

Theorem 5. ALG 2 is $2 + \frac{2}{\alpha}$-competitive for the DMPRP with two classes, uniform deferral times and customers located on the real line. For $\delta > 1$, the competitive ratio is exactly $2 + \frac{2}{\alpha}$.

Proof. For a given sequence of requests σ let σ_{even} (σ_{odd}) be the subsequence of all requests released in an interval I_k with even (odd) index k. We have

$$\text{ALG } 2(\sigma) \leq \text{ALG } 2(\sigma_{even}) + \text{ALG } 2(\sigma_{odd}).$$

For $\delta > 1$ equality holds, whereas for $\delta = 1$ we can have a strict inequality because target dates for requests released in σ_{even} and σ_{odd} coincide.

For a fixed $k \geq 0$ let $\sigma_k \subseteq \sigma$ be the subset of requests released in I_k, and let $r_{k,1}$ ($r_{k,2}$) be the 1^{st} (2^{nd}) class request among them with maximum distance to the origin. Then, by definition of ALG 2 we have

$$\text{ALG } 2(\sigma_k) = 2\,\alpha\,\text{dist}(r_{k,1}) + 2\,\text{dist}(r_{k,2}).$$

An optimal strategy either serves all requests in σ_k in one tour on the same day or it serves requests of different classes on two different days and thus,

$$\begin{aligned}\text{OPT}(\sigma_k) = \min\{&2\alpha \max\{\text{dist}(r_{k,1}), \text{dist}(r_{k,2})\},\\ &2\alpha\,\text{dist}(r_{k,1}) + 2\,\text{dist}(r_{k,2})\}.\end{aligned}$$

If the minimum is attained for $2\alpha\,\text{dist}(r_{k,1}) + 2\,\text{dist}(r_{k,2})$, ALG 2 yields the optimal solution. Otherwise we get a ratio of

$$\frac{\text{ALG } 2(\sigma_k)}{\text{OPT}(\sigma_k)} \leq \frac{2\alpha\text{dist}(r_{k,1}) + 2\text{dist}(r_{k,2})}{2\alpha \max\{\text{dist}(r_{k,1}), \text{dist}(r_{k,2})\}} \leq 1 + \frac{1}{\alpha}.$$

Let r_k be a request belonging to σ_k and let r_{k+2} be a request in σ_{k+2}. Then the deadline date $D(r_k)$ of r_k and the release date $d(r_{k+2})$ of r_{k+2} fulfill

$$D(r_k) \leq (k+1)\delta + \delta = (k+2)\delta < (k+2)\delta + 1 \leq d(r_{k+2}).$$

Hence, no algorithm can assign r_k and r_{k+2} to the same target date. Thus, we have

$$\text{ALG } 2(\sigma_{even}) \leq (1 + \frac{1}{\alpha})\,\text{OPT}(\sigma_{even}) \leq (1 + \frac{1}{\alpha})\,\text{OPT}(\sigma)$$

and the same holds for σ_{odd}. Putting everything together we obtain

$$\text{ALG } 2(\sigma) \leq \text{ALG } 2(\sigma_{even}) + \text{ALG } 2(\sigma_{odd}) \leq (2 + \frac{2}{\alpha})\,\text{OPT}(\sigma).$$

So, ALG 2 is $2 + \frac{2}{\alpha}$-competitive. On the following instance this bound is tight if $\delta > 1$. Let σ be a request sequence with

- $r_1 = (2^{nd}|1, \delta + 1|\epsilon)$,
- $r_2 = (1^{st}|1, \delta + 1|\epsilon)$,
- $r_3 = (1^{st}|\delta, 2\delta|1)$,
- $r_4 = (2^{nd}|\delta, 2\delta|1 + \epsilon)$,
- $r_5 = (1^{st}|\delta + 1, 2\delta + 1|1)$,
- $r_6 = (2^{nd}|\delta + 1, 2\delta + 1|1 + \epsilon)$,

where $0 < \epsilon \leq 1$. ALG 2 serves r_1 and r_4 in period $t = \delta$ with the 1^{st} class server, r_2 and r_3 in period $t = \delta + 1$ with the 2^{nd} class server. Request r_6 is served in $t = 2\delta$ with the 2nd class server and r_5 in $t = 2\delta + 1$ with the 1st class server. For $\delta > 1$, all these target dates are distinct. So the cost incurred by ALG 2 is

$$\text{ALG } 2(\sigma) = 2(1 + \epsilon) + 2\alpha + 2(1 + \epsilon) + 2\alpha.$$

An optimal strategy would be to serve r_1 and r_2 in period $t = 1$ with the 1^{st} class server, and r_3, r_4, r_5, r_6 in period $t = \delta + 1$ with the 1st class server. The total cost is

$$\text{OPT}(\sigma) = 2\alpha\epsilon + 2\alpha(1 + \epsilon).$$

This yields a ratio of

$$\frac{\text{ALG } 2(\sigma)}{\text{OPT}(\sigma)} = \frac{2\alpha + 2(1 + \epsilon) + 2\alpha + 2(1 + \epsilon)}{2\alpha\epsilon + 2\alpha(1 + \epsilon)} = \frac{4\alpha + 4 + 4\epsilon}{2\alpha + 4\alpha\epsilon}$$

which approaches $2 + \frac{2}{\alpha}$ as ϵ approaches 0. □

We know that ALG 2 is $2 + \frac{2}{\alpha}$-competitive, whereas PTD is 2α-competitive. It suggests itself to choose between ALG 2 and PTD depending on which competitive ratio is smaller for a given α. We have $2 + \frac{2}{\alpha} \leq 2\alpha$ if and only if $\alpha \geq \frac{1}{2} + \frac{1}{2}\sqrt{5}$. So ALG 2 is better for large values of α, whereas PTD is better for small values of α.

Intuitively, if the 1^{st} class server is very expensive compared to the 2^{nd} class server, it seems reasonable to only use it if absolutely necessary, that is, solely for 1st class requests. This corresponds to the strategy of ALG 2. Otherwise, if the 1st class server is not much more expensive, it makes sense to serve all requests with the 1st class server. This is the strategy of PTD.

ALG 3. If $\alpha \leq \frac{1}{2} + \frac{1}{2}\sqrt{5}$ apply PTD, else apply ALG 2.

Theorem 6. ALG 3 is $\min\{2\alpha, 2 + \frac{2}{\alpha}\}$-competitive for the DMPRP with two classes, uniform deferral times and customers located on the real line.

In the worst case, if $\alpha = \frac{1}{2} + \frac{1}{2}\sqrt{5}$, ALG 3 is $1 + \sqrt{5} \approx 3.2361$-competitive.

References

[1] E. Angelelli, M.W.P. Savelsbergh, and M.G. Speranza. Competitive analysis of a dispatch policy for a dynamic multi-period routing problem. *Operations Research Letters*, 35(6):713 – 721, 2007.

[2] E. Angelelli, M.G. Speranza, and M.W.P. Savelsbergh. Competitive analysis for dynamic multiperiod uncapacitated routing problems. *Networks*, 49(4):308–317, 2007.

[3] G. Ausiello, E. Feuerstein, S. Leonardi, L. Stougie, and M. Talamo. Algorithms for the on-line travelling salesman. *Algorithmica*, 29(4):560–581, 2001.

[4] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, New York, 1998.

[5] E. Gassner, J. Hatzl, S.O. Krumke, and S. Saliba. Clever or smart: Strategies for the online target date assignment problem. *Discrete Applied Mathematics*, 158(1):71–79, 2010.

[6] S. Heinz, S.O. Krumke, N. Megow, J. Rambau, A. Tuchscherer, and T. Vredeveld. The online target date assignment problem. In T. Erlebach and G. Persiano, editors, *WAOA*, volume 3879 of *Lecture Notes in Computer Science*, pages 230–243. Springer, 2005.

[7] S. Saliba, S.O. Krumke, and S. Westphal. Online-optimization of large-scale vehicle dispatching problems. *Electronic Notes in Discrete Mathematics*, 25:145–146, 2006.