Publication I

Lauri Ahlroth, André Schumacher, Harri Haanpää. On the power of lookahead in online lot-sizing. *Operations Research Letters*, 38(6):522 – 526, November 2010.


Volume 38, Issue 6, November 2010
ELSEVIER
Operations Research Letters
EDITOR-IN-CHIEF
Jan Karel Lenstra
AREA EDITORS
approximation and heuristics
Gerhard Woeginger
book reviews
Marc Uetz
continuous optimization
Patrice Marcotte
financial engineering
Jussi Keppo
game theory
Nimrod Megiddo
graphs and networks
Gianpaolo Oriolo
inventory, reliability and control
Sridhar Seshadri
linear and stochastic optimization
Rüdiger Schultz
logistics and revenue management
Retsef Levi
mixed integer optimization
Christoph Helmberg and
Alexander Martin
sequencing and scheduling
Steef van de Velde
simulation
Jose Blanchet
stochastic networks and queues
Sem Borst and Tetsuya Takine

Operations Research Letters 38 (2010) 522–526

Contents lists available at ScienceDirect

Operations Research Letters

journal homepage: www.elsevier.com/locate/orl

On the power of lookahead in online lot-sizing

Lauri Ahlroth [a,*], André Schumacher [a], Harri Haanpää [b]

[a] Aalto University School of Science and Technology, Department of Information and Computer Science, Finland
[b] Aalto University School of Science and Technology, Department of Communications and Networking, Finland

ARTICLE INFO

Article history:
Received 16 March 2010
Accepted 15 September 2010
Available online 25 September 2010

Keywords:
Competitive analysis
Online algorithm
Lookahead
Lot-sizing

ABSTRACT

We propose an online algorithm for an economic lot-sizing (ELS) problem with *lookahead*, which achieves asymptotically optimal worst-case performance for increasing lookahead. Although intuitive, this result is interesting since deterministic algorithms for previously studied online ELS problems have unbounded competitive ratio. We also prove lookahead-dependent lower bounds for deterministic algorithms.

1. Introduction

The single-commodity economic lot-sizing (ELS) problem has been first studied by Wagner and Whitin [12], who also propose a polynomial time algorithm solving the offline problem to optimality. However, a simple counterexample as in [11] shows that there are no online algorithms with finite competitive ratio even if the amount of future information is arbitrarily large. In this paper we study a continuous time version of the ELS problem, where demands occur at distinct time points, have a minimum quantity and are either satisfied by inventory or by placing an immediate order. We give an asymptotically optimal online algorithm that uses *lookahead*. In our model, an online algorithm with lookahead w is informed of all demands that are to be delivered within the next w time units, where w is a parameter of the algorithm.

We use competitive analysis [7] to study the worst-case performance of online algorithms with lookahead. Competitive analysis is based on comparing the cost of the online algorithm to the optimal offline cost. We say that an algorithm α has performance ratio $a \geq 1$ for a given problem instance if it achieves a cost of a times the optimal cost. The supremum of these ratios over all possible instances is called the *competitive ratio*. Competitive analysis has been used to study the performance of several popular lot-sizing heuristics; see for example [4,5,11].

In this paper we obtain a lookahead-dependent upper bound for the competitive ratio of the proposed algorithm. We also provide lookahead-dependent lower bounds for the competitive ratio of any deterministic online algorithm. We study the asymptotic behavior in both the limits towards a zero lookahead and an infinite lookahead, effectively approaching the offline setting. Our results show that when the lookahead tends to zero, the competitive ratio tends to infinity.

The effect of limited knowledge of the future on the performance of online algorithms has been studied for various problems. For example, [1,2,10] show how limited future knowledge improves the competitive ratio in the paging, list update, and multipass assembly line scheduling problems. Some results, as those in [3,8], are negative in the sense that limited future knowledge does not necessarily improve worst-case performance.

The lot-sizing problem with limited lookahead has been studied previously but usually only by experimentally comparing the performance of different heuristics. The experiments reported in [6], for example, indicate the counterintuitive result that generally suboptimal heuristics may outperform a locally optimal solver when run in a rolling-horizon fashion. The results in [11] also show that algorithms operating in a rolling-horizon fashion can produce arbitrarily bad solutions. The worst-case instances in [11] are exploiting the fact that demands can be of arbitrary small quantities. Our results imply that if demands have a minimum quantity, then lookahead can improve algorithm performance, up to asymptotic optimality.

The version of lookahead considered here represents only one way of incorporating limited future information into online algorithms. Dooly et al. [8], for example, provide instances of the TCP acknowledgment problem that are difficult for algorithms that have knowledge of the arrival time of the next l packets in the future. Due to the similarity of the problems (make-to-stock versus make-to-order), we expect this type of *demand lookahead* also to be ineffective for the lot-sizing problem. Our results hold for the *time*

* Corresponding address: Aalto University School of Science and Technology, Department of Information and Computer Science, P.O. Box 15400, FI-00076 Aalto, Finland. Tel.: +358 9 47025141; fax: +358 9 47023277.

E-mail addresses: lauri.ahlroth@tkk.fi (L. Ahlroth), andre.schumacher@tkk.fi (A. Schumacher), harri.haanpaa@tkk.fi (H. Haanpää).

0167-6377/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.orl.2010.09.010

lookahead model where one foresees the demands placed in the next w time units.

The paper is organized as follows. In Section 2 we introduce the problem and the online setting with lookahead which we analyze in this paper. In Section 3 we show lower bounds for the competitive ratio of any deterministic online algorithm with lookahead, distinguishing between small and large lookahead. Section 4 presents an online algorithm whose worst-case performance can be bounded by a function of lookahead. Finally, Section 5 presents our conclusions.

2. The economic lot-sizing problem

In this section we first consider the ELS problem in the offline setting, discuss properties satisfied by an optimal solution and then present an integer programming formulation of the problem. The resulting IP is similar to the one proposed by Levi et al. [9] for the make-to-order version of the ELS problem with discrete time. We then describe the online setting.

2.1. Offline problem

Consider a sequence of customer demands arriving at a factory warehouse over a continuous time horizon. A demand is a pair (t_i, d_i), where $t_i \in [0, T]$ is the *delivery time*, and d_i is the size of the demand, i.e., the *quantity* of inventory requested. We require $d_i \geq d_{\min} > 0$, where $d_{\min}$ is the minimum demand quantity. We frequently refer to a demand by just its index $i \in I$, where $I = \{1, 2, \dots, |I|\}$ and $|I| < \infty$.

Any demand has to be satisfied either from the inventory or by placing an order at the delivery time. Each order incurs a *setup cost* of K units, independent of how many units of commodity are ordered. Ordered inventory becomes available immediately. Storing d units of inventory over a time period of length t incurs a *holding cost* of $h \cdot d \cdot t$ units. The objective is to minimize the sum of setup and holding costs while satisfying all demands in time.

By Lemma 2.1, we may combine demands with the same delivery time to a single demand, or alternatively assume distinct delivery times, without affecting the optimal order schedule. Thus, w.l.o.g. we assume $0 = t_1 < \cdots < t_{|I|} = T$. Denote the sequence of demands in the order of their delivery time by S, where $S = ((t_i, d_i))_i$ for $i \in I$.

The crucial difference to the widely used model of [12] is that we fix a lower bound $d_{\min} > 0$ on the demand quantities. In the traditional model, it is usually assumed that time is discrete and the input consists of a non-negative demand quantity for each time period. We can model such an instance by choosing S to contain the set of non-zero demands of the original instance and $d_{\min}$ to be equal to their minimum quantity. Additionally, we do not use the discrete-time assumption but allow arbitrary delivery times. Before formulating the problem as an IP, we describe properties of an optimal solution.

Lemma 2.1. *In an optimal solution, all simultaneously occurring demands are satisfied by only one order.*

Proof. Suppose that in a solution the demands with delivery time t are satisfied by two or more different orders, where A is the first and B is the last order. Consider the demands or partial demands with delivery time t satisfied by A. Ordering these might as well be postponed until the order B, causing savings in holding cost but not causing any new setup costs. The original schedule was not optimal, and the claim follows. □

Corollary 2.2. *In an optimal solution, every demand is satisfied by one order.*

Lemma 2.3. *An optimal schedule satisfies the zero inventory ordering property: orders occur only at times when there is a demand and the inventory is empty.*

Proof. If the solution contains an order at a time when there is no demand, that order can be postponed until the next demand. If there is an order at a time when the inventory is not empty, a part of the previous order can be moved to the current one. In each case one saves holding cost without affecting setup costs, so the original schedule was not optimal. □

We now present an IP formulation of the problem that excludes suboptimal solutions not satisfying the two preceding lemmas. We introduce binary variables x_{ij} for all $i, j \in I$ with $i \leq j$ such that when set to 1 they indicate that demand j is satisfied by ordering at time t_i. We further introduce binary variables y_i for all $i \in I$ that correspond to ordering inventory at time t_i. The actual quantity of inventory ordered results from the demands satisfied by that particular order. The problem can then be formulated as follows.

ELS
$$\min \sum_{i \in I} \Big[K y_i + \sum_{j > i} (t_j - t_i) h d_j x_{ij} \Big]$$
$$\text{s.t.} \sum_{i \leq j} x_{ij} = 1 \qquad \forall j \in I \qquad (1)$$
$$x_{ij} \leq y_i \qquad \forall i, j \in I,\ i \leq j \qquad (2)$$
$$x_{ij}, y_i \in \{0, 1\} \qquad \forall i, j \in I, i \leq j.$$

Constraint (1) requires each demand to be satisfied prior to or at its delivery time. Eq. (2) ensures that a demand can only be satisfied by an order. The objective function accounts for the setup and holding cost of each order.

2.2. Online problem

In the online setting, one has only limited knowledge of the future when fixing the ordering decisions. We use a model of *time lookahead* [3] w, which means that at time t_i one sees all the demands $j \in I$ for which the delivery time satisfies $t_j \leq t_i + w$. Seeing a demand means that the corresponding delivery time t_j and demand quantity d_j are known. However, no information about the length T of the instance or the demands j for which $t_j > t_i + w$ is available.

The decision made at time t_i includes permanently fixing values for y_i and x_{ij} for all seen $j \geq i$, i.e., choosing which of the next seen future demands to order at time t_i and which to be ordered later. Note that this setting is different from [11], where the amount of ordered commodity may change after the order was placed. The values x_{ij} for unseen $j \geq i$ are implicitly set to zero, which means that no algorithm is allowed to anticipate unseen future demands by ordering them before seeing them. We do not consider this as a real restriction since anticipation leads to an infinite competitive ratio. Any pre-ordered but unseen demand may occur after an arbitrarily long period (if at all), leading to arbitrarily high holding cost while the optimal total cost would be bounded.

In order to simplify the exposition of our analysis, we eliminate the constants K, h and $d_{\min}$ by scaling the problem without affecting the nature of the problem. Namely, we scale the objective function by $1/K$, measure demands as relative to the minimum demand $\delta_i := d_i/d_{\min}$, and rescale times as $\tau_i := t_i h d_{\min}/K$, $\Gamma := T h d_{\min}/K$. It turns out that the properties of the problem are largely defined by the scaled lookahead $\omega := w h d_{\min}/K$. After scaling, the objective function takes the form

$$\min \sum_{i \in I} \Big[y_i + \sum_{j > i} (\tau_j - \tau_i) \delta_j x_{ij} \Big].$$

From now on, we use the scaled variables instead of the original ones. Since we view ω as a parameter of the algorithm, any instance of the scaled problem is completely determined by the sequence of demands S. Thus, we use the terms demand sequence and problem instance interchangeably.

Let us further define some notation used in bounding the performance of algorithms. By $C_\alpha(S)$ we refer to the cost of algorithm α on the instance S, i.e., the objective value of the solution provided by α for S. Denote by OPT an optimal offline solver, so $C_{\mathrm{OPT}}(S)$ is the optimal cost for the instance S when all demands are revealed immediately to the solver. We may also use the notation C_α for the cost when the instance S is clear from the context. The *competitive ratio* of an online ELS algorithm α with lookahead ω is defined by

$$r_\alpha(\omega) = \sup_S \frac{C_\alpha(S)}{C_{\mathrm{OPT}}(S)},$$

where the supremum is taken over instances S. For given lookahead ω, we define the competitive ratio of the ELS problem by taking the infimum over algorithms with lookahead ω,

$$r(\omega) = \inf_\alpha \sup_S \frac{C_\alpha(S)}{C_{\mathrm{OPT}}(S)}.$$

Lemma 2.4. *The competitive ratio $r(\omega)$ is nonincreasing in ω.*

Proof. From an algorithm α with lookahead ω we can easily construct an algorithm α' with lookahead $\omega' > \omega$ by simply ignoring the additional part of lookahead. The algorithms α and α' make the same decisions, so $r_{\alpha'}(\omega') = r_\alpha(\omega)$. Thus, $\inf_\beta r_\beta(\omega') \le \inf_\alpha r_\alpha(\omega)$, and we are done. □

3. Lower bounds for $r(\omega)$

In this section we prove lower bounds on the competitive ratio of any deterministic online ELS algorithm as a function of the lookahead ω. This is done by considering some simple input sequences where the online algorithms make bad decisions. We summarize the results first before presenting the proof. Theorem 3.1 is stronger for $\omega \le 1/2$ while Theorem 3.3 is stronger for $\omega > 1/2$.

Theorem 3.1. *In the interval $2/(L+1)^2 < \omega \le 2/L^2$, where $L \in \mathbb{Z}_+$, the competitive ratio of any deterministic online ELS algorithm is bounded by*

$$r(\omega) \ge \frac{1}{2/L - 1/L^2}.$$

Proof. The result follows from combining Lemma 3.4 with Lemma 2.4. □

Corollary 3.2. *By letting $\omega = 2/L^2$ and substituting $L = \sqrt{2/\omega}$ into the bound in Theorem 3.1, we find that $r_\alpha(\omega) \in \Omega(\omega^{-1/2})$ as $\omega \to 0$.*

Theorem 3.3. *The competitive ratio of any deterministic online ELS algorithm is bounded by $r_\alpha(\omega) \ge 1 + 1/(6\lceil\omega\rceil + 4)$, for all $\omega > 0$.*

Proof. The result follows from combining Lemma 3.7 with Lemma 2.4. □

Lemma 3.4. *When ω is of the form $\omega = 2/L^2$, where $L \in \mathbb{Z}_+$, the competitive ratio is bounded by*

$$r(\omega) \ge \frac{1}{\sqrt{2\omega} - \omega/2}.$$

Proof. Consider a demand sequence S where M demands of unit size have consecutive separation of $\omega_\varepsilon = \omega + \varepsilon$. An online algorithm α sees at most one demand at any time and thus pays in total $C_\alpha(S) = M$ for the setups. However, an offline solver might order everything in the first stage and pay $\sum_{i=1}^{M-1} i\omega_\varepsilon = \omega_\varepsilon M(M-1)/2$ for holding the inventory. We do not claim this strategy to be optimal but use it to obtain a lower bound on the competitive ratio as

$$r(\omega) \ge \inf_\alpha \frac{C_\alpha(S)}{C_{\mathrm{OPT}}(S)} \ge \frac{M}{1 + \omega_\varepsilon M(M-1)/2}. \tag{3}$$

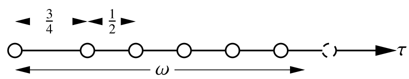

Fig. 1. Illustration of construction for proof of Lemma 3.7 with $\omega = 3$.

In (3), letting $\varepsilon \to 0+$ allows replacing ω_ε by simply ω. By elementary differentiation, the lower bound of (3) is maximized by setting

$$M = \sqrt{2/\omega}. \tag{4}$$

The claim follows by substituting $\omega = 2/L^2$ and (4) into (3). □

In the forthcoming analysis we need to make some assumptions on the decisions made by the online algorithms. Thus, we define a subset $\mathcal{A}$ of online ELS algorithms and show that it actually contains the best algorithms, allowing us to restrict ourselves to $\mathcal{A}$.

Definition 3.5. Let $\mathcal{A}$ be the set of online algorithms α for the ELS problem for which the storing time for any individual demand is strictly less than 1.

Theorem 3.6. *Any lower bound for $\inf_{\alpha\in\mathcal{A}} r_\alpha(\omega)$ is a lower bound for the competitive ratio of online algorithms for the ELS problem.*

Proof. For any algorithm β, there exists an $\alpha \in \mathcal{A}$ that always obtains a solution with cost at most the cost obtained by β. Let α be the modification of β that places a separate order at τ_i for any demand i with holding time at least 1 in algorithm β. Each such change costs 1 in setup costs but, as $\delta_i \ge 1$, saves at least 1 in holding costs for this order without affecting the costs of other orders. □

Lemma 3.7. *For integers ω, $r_\alpha(\omega) \ge 1 + 1/(6\omega + 4)$.*

Proof. By Theorem 3.6, we may assume that $\alpha \in \mathcal{A}$. Assume we have the situation shown in Fig. 1 of initially 2ω demands of unit size, where the first two are separated by 3/4 and all later consecutive demands by 1/2 time unit. The last demand is hence at time $\omega - 1/4$. Since $\alpha \in \mathcal{A}$, at most two demands can be satisfied by a single order due to Definition 3.5, so an algorithm $\alpha \in \mathcal{A}$ has two choices: either it orders the first demand alone or it orders the two first demands together. In the former case, the adversary does not add any further demands. Denote this instance by S. In the latter case, the adversary adds one more demand at time $\omega + 1/4$, which is just slightly after the current seen time window. Denote this instance with one additional demand at time $\omega + 1/4$ by S'.

After the initial decision, the online algorithm sees $2\omega - 1$ demands with separations 1/2 for both S and S'. The optimal cost is achieved by ordering these as $\omega - 1$ pairs and one single demand. Thus, for instance S algorithm α can achieve at best the cost $C_\alpha(S) = 2 + (\omega - 1)(1 + 1/2) = 3\omega/2 + 1/2$, while a complete pairing would save one order and increase holding costs by 3/4, leading to a total cost of $(6\omega + 1)/4$. The performance ratio here has a lower bound of

$$C_\alpha(S)/C_{\mathrm{OPT}}(S) \ge 2(3\omega + 1)/(6\omega + 1) = 1 + 1/(6\omega + 1).$$

In the case S' with an additional demand, we similarly obtain at best $C_\alpha(S') = 1 + 3/4 + 1 + (\omega - 1)(1 + 1/2) = 3\omega/2 + 5/4$, while ordering the first demand alone and pairing the others would save 1/4 in holding costs. Thus, the performance ratio here has a lower bound of

$$C_\alpha(S')/C_{\mathrm{OPT}}(S') \ge (6\omega + 5)/(6\omega + 4) = 1 + 1/(6\omega + 4).$$

Whichever decision algorithm α makes, the adversary can ensure that the lower bound of $1 + 1/(6\omega + 4)$ holds for the performance ratio. Thus, the competitive ratio $r_\alpha(\omega)$ has the same lower bound for integer ω. □

4. An asymptotically optimal algorithm

In this section we introduce a simple online ELS algorithm with lookahead and show upper bound for its competitive ratio. The algorithm divides time into intervals that are optimized separately. We also show that a simpler variant of the first algorithm is suboptimal, motivating some details in the chosen algorithm.

4.1. Preliminaries

Before presenting the algorithm analysis, we characterize a set of worst-case instances for a class of online algorithms. By definition any $\alpha \in \mathcal{A}$ necessarily has zero inventory on an interval formed by two consecutive demands separated by at least one time unit. It seems natural that a good online algorithm should indeed recognize these long intervals, and not let the demands before and after the break affect the schedules on the other side of the interval.

Definition 4.1. Let $\mathcal{S}$ denote the set of all ELS input sequences and let $\mathcal{D} \subseteq \mathcal{S}$ be the set of *dense* sequences where each two consecutive demands are separated by less than 1 time units.

Definition 4.2. We say that an algorithm has the *splitting property*, if the following holds: for any instance with a gap of at least 1 between two consecutive demands, the schedule returned by the algorithm is the same as would be obtained by running the algorithm separately on the demands before the gap and the demands after it. Let $\mathcal{B} \subset \mathcal{A}$ be the set of algorithms with the splitting property.

Theorem 4.3. *Let $\alpha \in \mathcal{B}$. Then*

$$\sup_{S \in \mathcal{S}} \frac{C_\alpha(S)}{C_{\mathrm{OPT}}(S)} = \sup_{S' \in \mathcal{D}} \frac{C_\alpha(S')}{C_{\mathrm{OPT}}(S')}.$$

Proof. By a trivial subset argument

$$\sup_{S \in \mathcal{S}} \frac{C_\alpha(S)}{C_{\mathrm{OPT}}(S)} \geq \sup_{S' \in \mathcal{D}} \frac{C_\alpha(S')}{C_{\mathrm{OPT}}(S')},$$

so we only need the reverse inequality.

Fix a non-dense demand sequence S. There is a pair of consecutive demands with delivery times at least 1 time unit apart. This gap divides the input sequence into two nonempty subsequences L and R that consist of orders before and after the gap, respectively. We show that the performance ratio of α on the demands on either L or R alone is at least as large as on the total sequence $S = L \cup R$.

Recall that by the definition of $\mathcal{B}$ the ordering decisions to satisfy demands in L and R are independent of demands in the other subset. We thus have the equality $C_\alpha(S) = C_\alpha(L) + C_\alpha(R)$ and similarly for the optimal costs. Using the algebraic identity $(a + b)/(c + d) \leq \max\{a/c, b/d\}$ for positive numbers we obtain

$$\frac{C_\alpha(S)}{C_{\mathrm{OPT}}(S)} = \frac{C_\alpha(L) + C_\alpha(R)}{C_{\mathrm{OPT}}(L) + C_{\mathrm{OPT}}(R)} \leq \max\left\{\frac{C_\alpha(L)}{C_{\mathrm{OPT}}(L)}, \frac{C_\alpha(R)}{C_{\mathrm{OPT}}(R)}\right\}.$$

Thus, we may use either the part L or R to lower bound the performance ratio and eliminate at least one separation of length at least 1. Working recursively, we eliminate all large separations without lowering the performance ratio, that is, we find a dense $S' \in \mathcal{D}$ satisfying $C_\alpha(S)/C_{\mathrm{OPT}}(S) \leq C_\alpha(S')/C_{\mathrm{OPT}}(S')$. The proof is finished. □

Hence, when looking for worst-case bounds for an algorithm in $\mathcal{B}$, we may assume that the demand sequence S is dense. We can also obtain a lower bound on the cost of an optimal solution for dense instances.

Theorem 4.4. *Let S be dense, and Γ the time of last demand. The optimal cost is at least $C_{\mathrm{OPT}}(S) \geq \Gamma + 1$.*

Proof. Let the demands occur at times $0 = \tau_1 < \tau_2 < \cdots < \tau_n = \Gamma$, where by denseness $\tau_i - \tau_{i-1} \leq 1$. We claim that for $2 \leq i \leq n$, we can charge a cost of at least $\tau_i - \tau_{i-1}$ to demand i. Indeed, if an optimal solution did not produce at time τ_i, demand i incurs a delay cost of at least $\tau_i - \tau_{i-1}$, since it has been stored for at least this long. If instead there was an order at time τ_i, we charge demand i the ordering cost of $1 \geq \tau_i - \tau_{i-1}$. We also charge the first setup cost of 1 to the first demand. Now the total cost from the demands is at least

$$C_{\mathrm{OPT}}(S) \geq 1 + \sum_{i=2}^{n} (\tau_i - \tau_{i-1}) = 1 + \tau_n - \tau_1 = 1 + \Gamma. \quad \square$$

4.2. GREEDYSPLIT

For large lookahead, it seems natural to exploit the optimal schedule for the current lookahead window, as if assuming there are no more demands arriving after the seen window. An algorithm resulting from this assumption may use an offline solver in a rolling-horizon fashion and update the schedule whenever new information arrives. A simpler variant, which we call GREEDYFIX, divides the problem instance into intervals of length ω and optimizes these separately. When one interval of length ω is completed, the algorithm finds the next unsatisfied demand and starts over. This algorithm is much easier to analyze but turns out to be suboptimal. We give the suboptimality proof and then present a refinement that has asymptotically optimal performance.

Theorem 4.5. *For $\omega > 1$, $r_{\mathrm{GREEDYFIX}}(\omega) \geq 3/2$.*

Proof. For any given $\omega > 1$ consider the instance with unit demands at times $\tau_1 = 0, \tau_2 = \omega, \tau_3 = \omega + \varepsilon$. At time $\tau = 0$ the algorithm sees two demands separated by a distance ω and decides to order both of them separately, paying a cost of 2. Since the algorithm commits to this decision, it needs to satisfy the third demand by an additional order even if it is optimal to satisfy the second and the third demand with the same order. The total cost is 3 while $2 + \varepsilon$ would suffice, leading to performance ratio $3/(2 + \varepsilon) \to 3/2$ as $\varepsilon \to 0$. □

Obviously, the rolling-horizon version performs better here, but the flaws are also fixed by a small modification that does not complicate the analysis too much. The essential mistake of GREEDYFIX was to commit to orders for demands separated by long periods without demands. We give a refined algorithm that handles these long gaps separately, and call this algorithm GREEDYSPLIT or just GS.

```
repeat
  let i be the next unsatisfied demand
  if ∃ consecutive demands separated by ≥ 1 in [τ_i, τ_i + ω]
    let τ_l be the time of the earlier demand of the pair
    find and apply an optimal schedule for [τ_i, τ_l]
  else
    find and apply an optimal schedule for [τ_i, τ_i + ω]
```

Algorithm 1. GREEDYSPLIT algorithm

In other words, the order planning in GS is split into disjoint intervals of length at most ω. Shorter intervals than ω are used when there is a long gap between consecutive demands. Each interval starts with a demand, and between consecutive intervals there is at least a small gap during which no demands arrive.

Note that some intervals may have several optimal solutions. However, it is easy to see that there always exists an optimal solution that stores inventory for each order in the interval for strictly less than 1 time unit. By letting GS choose this solution for the current interval and by noting that decisions on different intervals are independent, it follows that $GS \in \mathcal{B}$.

To bound the competitive ratio from above, consider a dense demand sequence S. This really represents an arbitrary worst-case instance because of Theorem 4.3. An execution of GS induces a collection of disjoint subintervals of $[0, \Gamma]$ that cover all the

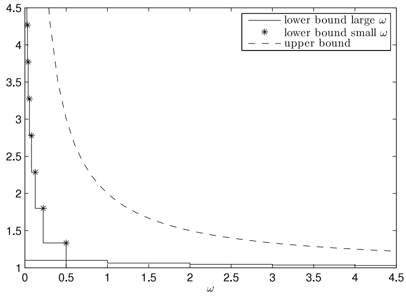

Fig. 2. Lower and upper bounds for $r(\omega)$ as a function of the lookahead ω; the figure shows the lower bounds from Section 3 and the upper bound for GS of Section 4.

demands. Each such interval corresponds to a call of the offline solver for the instance restricted to the interval. Denote the interval starting times by $0 = b_1 < b_2 < \cdots < b_N$. From the denseness of S and the definition of GS we conclude that all the intervals, possibly excluding the last which is cut off at Γ, are of length ω. Because all the demands lie in $[0, \Gamma]$, we must have $(N-1)\omega < \Gamma$.

Consider the following modification to the schedule obtained by an optimal offline solver OPT: whenever the schedule holds d units of inventory from an interval $[b_i, b_i + \omega]$ until the following interval, move these d units from the last order in the previous interval to an additional order at time b_{i+1}, so that no inventory is carried over. This modification increases the cost due to additional orders by at most $N - 1$. Clearly, the algorithm GS could have made the same decisions leading to this schedule. Thus, the actual schedule of GS has at most this modified cost for each interval, and the total cost for GS is bounded by

$$C_{GS}(S) \le C_{OPT}(S) + (N-1) < C_{OPT}(S) + \Gamma/\omega. \tag{5}$$

Theorem 4.4 implies that

$$C_{OPT}(S) \ge \Gamma + 1. \tag{6}$$

Combining Eqs. (5) and (6), we arrive at the bound

$$\frac{C_{GS}(S)}{C_{OPT}(S)} \le \frac{C_{OPT}(S) + \Gamma/\omega}{C_{OPT}(S)} = 1 + \frac{\Gamma/\omega}{C_{OPT}(S)} \le 1 + \frac{1}{\omega}.$$

Thus, we obtain our main result that GREEDYSPLIT is asymptotically optimal for sufficiently large lookahead. Fig. 2 summarizes our results.

Theorem 4.6. *The competitive ratio of GREEDYSPLIT is upper bounded by $r_{GS}(\omega) \le 1 + 1/\omega$, which tends to 1 for sufficiently large ω.*

5. Conclusions

The lower and upper bounds obtained can be solely expressed in terms of the scaled lookahead $\omega = whd_{\min}/K$. In particular, allowing arbitrarily small positive demands to be placed has the same blow-up effect on the competitive ratio as using only an arbitrarily small lookahead. Thus, our result of the competitive ratio tending to infinity when $\omega \to 0$ generalizes previous results for rolling-horizon algorithms without minimum demand in [11]. Additionally, the parameter ω has a natural interpretation in the sense that it is equal to the relative cost of storing a minimum demand quantity over the length of the lookahead window compared to the setup cost. One may ask whether other online problems with lookahead that have practical relevance exhibit a similar relation between future knowledge and cost factors.

Acknowledgements

The authors thank Pekka Orponen for fruitful discussions on the topic, and an anonymous reviewer of an earlier version of the paper for suggesting an improved bound in Theorem 4.4. The first author has been supported by the Academy of Finland under grant 128823, and the second author jointly by the Helsinki Graduate School of Computer Science and Engineering and by the Nokia Foundation.

References

[1] S. Albers, On the influence of lookahead in competitive paging algorithms, Algorithmica 18 (1997) 283–305.
[2] S. Albers, A competitive analysis of the list update problem with lookahead, Theoretical Computer Science 197 (1998) 95–109.
[3] L. Allulli, G. Ausiello, L. Laura, On the power of lookahead in on-line vehicle routing problems, in: Lecture Notes in Computer Science, vol. 3595, Springer, Berlin, Heidelberg, 2005, pp. 728–736.
[4] S. Axsäter, Worst case performance for lot sizing heuristics, European Journal of Operational Research 9 (1982) 339–343.
[5] S. Axsäter, Performance bounds for lot sizing heuristics, Management Science 31 (1985) 634–640.
[6] J.D. Blackburn, R.A. Millen, Heuristic lot-sizing performance in a rolling-schedule environment, Decision Sciences 11 (1980) 691–701.
[7] A. Borodin, R. El-Yaniv, Online Computation and Competitive Analysis, Cambridge University Press, 1998.
[8] D.R. Dooly, S.A. Goldman, S.D. Scott, TCP dynamic acknowledgment delay (extended abstract): theory and practice, in: STOC'98: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, ACM, New York, NY, USA, 1998, pp. 389–398.
[9] R. Levi, R.O. Roundy, D.B. Shmoys, Primal–dual algorithms for deterministic inventory problems, Mathematics of Operations Research 31 (2006) 267–284.
[10] R. Motwani, V. Saraswat, E. Torng, Online scheduling with lookahead: multipass assembly lines, INFORMS Journal on Computing 10 (1998) 331–340.
[11] W. van den Heuvel, A.P.M. Wagelmans, Worst-case analysis for a general class of online lot-sizing heuristics, Operations Research 58 (2010) 59–67.
[12] H.M. Wagner, T.M. Whitin, Dynamic version of the economic lot size model, Management Science 5 (1958) 89–96.