Scheduling and Reliable Lead Time Quotation for Orders with Availability Intervals and Lead Time Sensitive Revenues

Pinar Keskinocak R. Ravi Sridhar Tayur

February 1999; Revised February 2000

Abstract

Motivated by applications in the manufacturing and service industries, we consider two models for co-ordinating scheduling with lead time quotation: a basic model with a single customer type, and an enhanced model where an additional second customer type expects immediate service or production. In both models, revenues obtained from the customers are sensitive to the lead-time, there is a threshold of lead-time above which the customer does not place an order and the quoted lead times are 100% reliable. These models are related to well-known scheduling problems, which have been studied in both the off-line and on-line settings.

We introduce the *immediate quotation* case and study it along with the (traditional) on-line version. We provide complexity results for the off-line case, and perform competitive analysis for the on-line cases. A natural question of bridging the gap between the on-line and quotation models leads us to the *delayed* quotation model, which we study briefly. The analysis of these models provides useful qualitative insights as well.

Keywords: Lead Time Quotation, On-line algorithms, Competitive ratio, Complexity, Scheduling.

1 Introduction

In this paper, we study the problem of scheduling and reliable lead time quotation for orders with availability intervals and lead time sensitive revenues (SLTQ). Each order has an arrival time, or release time, and a latest acceptable start time for processing; the difference between the two times is the availability interval or the maximum acceptable lead time. We use the term 'lead time' to denote the time between *starting* the processing of the order and the order's arrival time. Revenues from orders decrease as the (quoted) lead times increase. Our *basic* model has one type of customer, while the *enhanced* model has a second ("urgent") type of customer as well.

For both the basic and enhanced models, we consider four versions of SLTQ based on what information is known and when a decision has to be made.

Off-line (F-SLTQ): In the off-line model, all information about the orders is known in advance. This might be the case if the demand process is very predictable leading to good forecasts, or if most customers place their orders in advance.

On-line (O-SLTQ): Orders arrive over time. The decisions about accepting, rejecting or scheduling an order have to be made only based on the information about the orders that arrived so far, without any knowledge of future orders. This would be the case if the demand is not known in advance, and if forecasting is very difficult. The decisions about an order can be made anywhere between the order's arrival time and latest acceptable start time. This is the traditional on-line version.

Quotation (Q-SLTQ): This is a stringent on-line model, where the decision about accepting/rejecting an order must be made and a lead time must be quoted *immediately* when the order arrives. The quoted lead times are 100% reliable, i.e. the processing of the order has to start *within* the quoted lead time.

Delayed Quotation (D-SLTQ): This is also a stringent on-line model, in which decisions about accepting/rejecting an order and lead time quotation have to be made within q time units after the order arrives, where q is smaller than the maximum acceptable lead time. This model is between the on-line and the quotation models.

To be clear, O-SLTQ, Q-SLTQ and D-SLTQ are all *on-line*, i.e. orders arrive over time and decisions are made without any knowledge of future orders. Any quotation algorithm is a delayed quotation algorithm, which in turn is a traditional on-line algorithm, and this is a one-way inclusion.

The off-line model is studied by methods from mathematical programming. To evaluate the performance of algorithms for the three on-line models, we use *competitive analysis* [29]. In a competitive analysis, an (deterministic) on-line algorithm A is compared to an optimal off-line algorithm. An optimal off-line algorithm knows all the information about the orders in advance and can serve them obtaining the maximum possible total revenue. Given an instance I, let $z_A(I)$ denote the total revenue obtained by using algorithm A, and let $z^*(I)$ denote the revenue obtained by an optimum off-line algorithm for instance I. For maximization problems, we call an on-line algorithm *c-competitive*, if

$$z^*(I) \leq cz_A(I) + a$$

for any instance I (see [7] for a review of competitive on-line algorithms). The factor c is also called the *competitive ratio* of A.

Most of the previous research on the competitive analysis of on-line scheduling algorithms considers models similar to O-SLTQ, where the scheduling decisions about an order can be delayed; see [17], [19] and [24]. In many cases, this delay has no bound as the orders do not have latest acceptable start or completions times. To the best of our knowledge, quotation models for scheduling problems have not received much attention within the context of competitive analysis. The importance of co-ordinating scheduling and quotation cannot be ignored in today's industrial supply chain management. In fact, it is well accepted in many industries that "accurate lead time quotation" is as important as "cost" and "quality" as a performance measure on which customers evaluate suppliers; see [18] [30].

There is additional value in studying the various versions, off-line and on-line, in an unified manner. By comparing the on-line and off-line models, we can evaluate the value of investing in improved information gathering and forecasting methods. Similarly, by comparing the quotation and delayed quotation models, we can analyze the benefits of delaying the decision for a while.

Summarizing, our contributions are the following. We model an important problem within the scheduling framework. We introduce the quotation version in the on-line setting. For the basic and enhanced models, we find competitive ratios for O-SLTQ, Q-SLTQ and D-SLTQ. We briefly present some results on polynomially solvable instances of F-SLTQ. Our analysis also reveals interesting qualitative insights. We now briefly describe two applications that motivated this work.

1.1 Motivation

We are motivated by the following real-world applications. The issues that are described in these examples are typical in the industrial supply chains, where the customer is also a company, rather than an individual consumer.

> Consider a company that produces and supplies customized *rolls* – which are tools in steel mills – to the rapidly growing segment of mini-mills worldwide which produce specialty steel. Since the roll-manufacturing processes are mature, and the production of the different variety of rolls require similar technology, the processing times are nearly deterministic, and within each family of products nearly equal. The key challenge in managing this business is thus not in manufacturing, but rather in the interface between manufacturing and customer service representatives (CSRs), the functional group that accepts orders and guarantees lead times to the customers who demand customized rolls and whose order process is not predictable well. Because of the high variety of rolls differing in finish or diameter, no inventory is kept. Instead, the rolls are built to order. When an order arrives, the CSR quotes a lead time. For longer lead times, it is common practice to give price breaks, so as not to lose customers who have secondary options from where to obtain their rolls. In the past, CSRs have guaranteed lead times without taking into account the shop floor status leading to, among other things, increased expediting, increased use of overtime, missing the promised due dates and losing important customers. To remedy this situation, the firm has now decided to co-ordinate the interface between manufacturing and CSRs. Our model here was motivated to aid in this interface.

A similar problem exists at the customers of these roll-manufacturers as well.

> A specialty alloy division of a large company is a major producer and distributer of steel products and high performance alloys for aerospace, automotive, electronics and other industries worldwide. The division produces a large number of steel products to satisfy its customer's needs. Each product is produced by routing steel through a series of processing steps needed to give it the desired characteristics. One thirds of production is made to stock, and two third is made to order. With the current order inquiry process, the CSR can quote price, lead time etc. immediately, to the customer if that customer has placed the same order earlier. Otherwise, the response time can take anywhere between 24 hours to a week, depending on whether or not a new route must be created. Currently, only about 50% of the orders get quotes immediately at arrival, and this company would like to provide quick response to more orders to improve customer service.

Similar examples exist in the automotive supply chains, in the construction industry, in the paper industry as well as in service companies that provide spare-part kits to customers [30]. Many examples also abound at the individual consumer level: dry-cleaning services, obtaining university transcripts, rush-orders for document services and package delivery. The models studied here have been motivated by projects and discussions at plants of GE, ASKO, Sintermet, TRACO and Blazer Diamond among others.

1.2 The Models

Our basic model has a single customer type and is an appropriate model when the customer orders are similar to each other. Orders arrive over time, and r_j is the *release time* (earliest start time) or *arrival time* of order j. We assume that all arrival times are integers. Each order j of has a processing time $p_j = p$, a maximum acceptable lead time $l_j = l$ and a penalty (or revenue that is lost) $w_j = w$ for each unit of time the order waits before its processing starts. The function $R(d)$ represents the net revenue for (quoted) lead time d, if the order is accepted. We concentrate on the following revenue function:

$$R(d) = \begin{cases} (l-d)w & \text{for } d < l \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

In this revenue function, d denotes the *lead time* in the off-line and on-line models, and the <u>quoted</u> lead time in the quotation and delayed quotation models. If $d \geq l$ for an order, then the customer goes to another vendor. From the supplier's point of view, the supplier has the option of rejecting an order: if it is not possible or desirable to start processing a type i order within $l - 1$ time units of its arrival (due to a busy schedule or in anticipation of future orders), there is no benefit in accepting the order. (Our results on alternative non-increasing revenue functions can be found in [21].)

In F-SLTQ and O-SLTQ, revenues from orders linearly decrease as the lead time increases. In Q-SLTQ and D-SLTQ, revenues linearly decrease as the quoted lead time increases. In Q-SLTQ and D-SLTQ, the quoted lead times are 100% reliable, i.e. once a lead time is quoted, the promised product or service has to be started within the quoted lead time. (Note that in Q-SLTQ and D-SLTQ, although lead times must be quoted immediately and within q time units after an order arrives, respectively, the actual start time for processing can be decided later, any time within the quoted lead time. In some cases, the actual lead time may be shorter than the quoted lead time; however, the revenues are always based on the quoted lead time.) If an order is accepted, its processing must be completed without interruption, i.e. preemption is not allowed. Our objective is to schedule the orders and quote lead times in order to maximize the total revenue. Our models incorporate, through simplifying assumptions, many features present in the real world situations that motivated this work.

The *enhanced* model adds a second type of customer to the basic model. This is motivated by our observations that in some cases there is an urgent customer class in addition to the normal customer type. Urgent orders require very short lead times, whereas normal orders can tolerate relatively longer lead times. We model urgent orders as type 1, which must be processed immediately upon acceptance ($l_1 = 1$). The normal orders are modeled as type 2, which can wait up to a deadline ($l_2 > 1$) before their processing starts. Also, urgent orders have a higher unit revenue compared to normal orders ($w_1 \geq w_2$). Orders within each type have equal processing times, equal maximum acceptable lead times and equal penalties.

In all models, we assume that there is a single machine or server. Note that SLTQ (with multiple types of customers) generalizes the well known scheduling problem of minimizing the sum of weighted completion times subject to release times [21], denoted by $1|r_j|\sum w_j C_j$ (see [15] for taxonomy of scheduling problems).

1.3 Summary of Results

Our main focus is on O-SLTQ and Q-SLTQ and the results are summarized in Table 1. The special case of $p = 1$ (unit length orders) provides a building block for the general p case and so is studied first within each model. LB denotes the lower bound and UB denotes the upper bound on the competitive ratio of on-line and quotation algorithms. We construct instances (with "cruel" adaptive adversaries, where at each time the adversary knows all the actions taken by the online algorithm so far and based on this knowledge constructs the worst possible input of arrivals so as to maximize the competitive ratio) to show the lower bounds. We provide algorithms, with analysis, to show the upper bounds. For a quotation LB entry by '*', we can substitute the on-line lower bound for that problem. Similarly, for an on-line UB entry '*', we can substitute the quotation upper bound for that problem.

Table 1: Summary of results for O-SLTQ and Q-SLTQ.

		On-line		Quotation	
Section	Topic	LB	UB	LB	UB
2.1	Basic Model, $p = 1$	1	1	1.5	1.618
2.2	Basic Model, $p > 1$	$\sqrt{2}$	1.618	1.5	1.618
3.1	Enhanced Model, $p = 1$, $w < l$	1.28	1.618	1.5	$\max\{1 + \frac{w}{l}, 1.755\}$
3.1	Enhanced Model, $p = 1$, $w \geq l$	1	1	2	2.35
3.2	Enhanced Model, $p > 1$	$\max\{\sqrt{2}, \frac{w}{l}\}$	*	*	$\max\{\sqrt{2}, 1.755\frac{w}{l}\}$
4	DLTQ, Basic Model, $p = 1$	1	1		$\min\{1.618, \frac{1}{1-\delta^2}\}$
4	DLTQ, Enhanced Model, R'			w/l	$1/\alpha$ (if $q_i \geq p - 1$)

Recall that D-SLTQ bridges the gap between O-SLTQ and Q-SLTQ. The results on D-SLTQ are discussed briefly in Section 4. Our results on polynomially or pseudo-polynomially solvable instances of F-SLTQ (off-line case, with m types of orders) are summarized in Table 2; $L = \max_i\{l_i\}$ stands for the maximum acceptable lead time and n stands for the number of orders in the problem. A blank box in Table 2 means that this aspect can be arbitrary. We do not discuss F-SLTQ in depth here because of space considerations as well as to retain a focus on the three on-line cases; see [21] for details.

Table 2: Polynomially solvable cases of F-SLTQ.

r_i	p_i	l_i	w_i	Result
equal		equal	equal	$O(n \log n)$
	equal		equal	$O(n^6)$
		$l_i \leq p_i + p_j$		$O(n^2 L^2)$
	1			$O(n^6)$
equal	equal			$O(n^6)$

1.4 Some Qualitative Insights

Some insights we gain by studying and comparing the off-line, on-line, quotation and delayed quotation models are presented below.

1. *SLTQ is similar to, but more difficult than some well known scheduling models.* We show that O-SLTQ is quantifiably harder than the on-line version of $1|r_j, p_j = 1|\sum w_j C_j$.

2. *Comparing O-SLTQ with F-SLTQ.* In some cases, on-line scheduling decisions can be made quite efficiently with good performance guarantees, and sometimes optimally. Thus, in these situations we do not require advance information about future demand.

3. *Comparing Q-SLTQ with O-SLTQ.* Our results also show that the quotation version, where we have to make decisions immediately when an order arrives, can be much harder than the traditional on-line version. So, the difficulty does not only lie in not knowing the demand, but in how soon we have to make a decision when an order arrives.

4. *How to manage quotation.* In order to obtain high revenues, we need to reserve capacity – equivalently, leave space – for future orders, even if there is only a single type of orders.

5. *Enhanced model requires a different strategy than the basic model.* In case of two types of customers, we need to reserve capacity in two different ways: (1) we don't promise capacity beyond a certain number of periods from now, and (2) within the periods we promise capacity, we reserve some capacity for high margin customers. In contrast, in the single type case, it is sufficient not to promise capacity after αl periods, where l is the maximum acceptable lead time of an order (but not reserve space in the first αl periods).

6. *Comparing D-SLTQ with Q-SLTQ.* Partially delaying the quotation decision can improve performance significantly. We are able to quantify the improvement, as well as show results that indicate a continuous improvement as delay increases for our basic model.

7. *Sometimes the delay in D-SLTQ has to be significant.* We show a threshold rule for the enhanced model: after a certain delay, there is significant benefit.

An example of a real world implementation where the insights obtained here have been used is in a laminate plant where quoting accurate lead times and co-ordinating it with scheduling was central to the plant management strategy. In one of the product lines ("the rigid line"), the basic model studied here was considered appropriate as the ten main customers, that accounted for over 80 % of the demand, were nearly homogeneous. The availability interval was 3 weeks, set by industry standards. Based on our models, the laminate manufacturer negotiated with the customers a maximum of one-week delay in quotation. In another product line ("the multi-layer line"), where the enhanced model was considered appropriate, the manufacturer did not negotiate strongly for a delay in quotation as it was not possible to delay the quotation to the point where significant benefits could be realized. Furthermore, the order entry group, that quotes lead time, now uses the schedule information when deciding on the lead time, in a D-SLTQ setting for rigid line and a Q-SLTQ setting for the multi-layer line. For low volume, low margin customers, the O-SLTQ model is appropriate, while for replenishing their own warehouses, since reliable forecasts are available, F-SLTQ is appropriate. The details of the implementation, along with other plant improvement activities such as maintenance, work-force scheduling and re-organization, quality programs and product re-design, are described in [31].

1.5 Literature Review

Most of the literature in machine scheduling problems focuses on sequencing decisions only (see [25] for a recent review on machine scheduling problems), assuming that the due dates are preset and/or there is no availability interval (there may be a release time, but usually there is no latest start or completion time). Minimizing (weighted) tardiness, number of tardy jobs, lateness, flow times and completion times are among common objectives. Once the due dates are set, different rules are used for sequencing such as earliest due date, minimum slack and critical ratio [2]. In contrast, we consider the combined problem of due date setting and scheduling, where we need to quote a due date and then schedule an order to ensure that it is completed before the quoted due date.

Combined due date setting and sequencing problems are considered in [3], [4], [5], [6], [11], [13], [23], [26], [32] where the performance of different rules are compared via simulation. Analytical procedures are discussed in [8], [9], [12] [27], [28], [20], [33]. In all of these papers it is assumed that the customer will place an order no matter how late the quoted due date is. In our model, we assume that each order is available for processing within a certain time interval and an order will be lost if it is not processed within its interval of availability. See [10] for a review of scheduling research involving due date determination decisions.

A variant of SLTQ is *Scheduling of Intervals (SOI)*, where $l = 1$ for all the orders. The on-line case is studied in [14] and [34] while the off-line version is considered in [1]. Another variant of F-SLTQ is studied in [16], where $R_i(d)$ is a positive constant, say K_i, if $d \leq l_i$, and zero otherwise.

1.6 Organization of the Paper

The paper is organized as follows. In Section 2, we study the basic model with single customer type, first with unit-length orders followed by the case of non-unit length orders. In Section 3, we study the enhanced model with two customer types, first with unit length orders followed by non-unit length orders. In Section 4, we study the delayed quotation problem. We conclude in Section 5. Some of the proofs are presented in the appendix for improved readability.

1.7 Preliminaries

We define some terms that are frequently used in the paper.

Let σ denote the schedule generated by an on-line (quotation) algorithm. To do the competitive analysis, it is sometimes convenient to divide σ into "phase"s, where each phase consists of a sequence of consecutively scheduled orders. Phase i starts at time t_i, if the following conditions hold:
1. An order is scheduled to time t_i and the arrival time of that order is also t_i.
2. All the accepted orders which arrived before t_i are processed before t_i (i.e. no more accepted, but not yet processed orders).
Let B_i denote the ordered set of orders scheduled in phase i. Note that the orders in B_i are served consecutively, and there may be some idle time after the last order in B_i, before the next phase starts. Let t'_i be the completion time of the last order in B_i. Let z^*_i be the maximum revenue one could make from the orders which arrived in phase i and z_i be the revenue made by the algorithm.

2 Basic Model

2.1 On-line and Quotation Results for the Basic Model with Unit Processing Times

Consider the following algorithm[1].

Algorithm O-HRR: (On-line Highest Remaining Revenue)
Whenever the machine is idle and there are orders available for scheduling, pick an order j with the largest remaining revenue (denoted by $rem_j(t)$ if we are in time t) and schedule it next. In case of ties, choose the order with the largest w_j.

We first show that Algorithm O-HRR is an optimum algorithm for O-SLTQ when we have unit length orders. (Thus, LB and UB are both equal to 1.) We then show a lower bound of 1.5 for the competitive ratio of any algorithm for Q-SLTQ with unit length orders. Thus, we show that even in this special case, on-line quotation algorithms are quantifiably harder to design than traditional on-line algorithms.

We present an on-line quotation algorithm (called Q-FRAC) with competitive ratio at most 1.618 for Q-SLTQ with unit length orders. We provide an example to show that our analysis about the performance of this algorithm is tight.

Our first proposition follows from the standard pairwise interchange argument.

Proposition 1 *Algorithm O-HRR is optimal for the basic model with unit length orders.*

The result of Proposition 1 implies that for the basic model with unit length orders, an on-line algorithm which does not have any information about future orders is as good as an optimum off-line algorithm which knows all the data in advance. This is because when we commit capacity, we do so only for one unit of time and we have all the information we need. Next, we see that quotation (the Q-SLTQ version), in contrast, can benefit from more information.

To see the basic trade-off in quotation decisions, let us consider the following scenario. Suppose that in some period, part of our future capacity is already reserved for the orders

[1]We use the first letter of the algorithm name as O and Q to denote on-line and quotation algorithms respectively.

which arrived earlier and a new order arrives. Even if we quote the shortest possible lead time for the newly arrived order, the revenue we will get from this order is not going to be too high. If we accept this order, we will further utilize our capacity (for a low revenue). If new orders arrive in the future (which could give us higher revenue provided that we can quote short lead times) we may not be able to quote short lead times for those orders. On the other hand, if we do not accept this order now, we lose the revenue from this order. If we do not receive enough orders in the future, we may end up with un-utilized capacity.

Based on the above, we create an instance to show a lower bound on the competitive ratio of quotation algorithms for Q-SLTQ.

Proposition 2 *The competitive ratio of any quotation algorithm for Q-SLTQ with unit length orders is at least 1.5.*

Proof Consider an instance where $l = 2$ and $w = 1$. At time zero, the machine is available and two orders arrive. Any optimal algorithm should schedule one of these orders to time zero. If the algorithm rejects the second order, no more orders arrive, so we have $z^*/z = 3/2 = 1.5$. If the algorithm decides to accept the second order, then this order must be scheduled to time 1. In this case, two orders arrive at time 1. One of them is lost, since the machine is busy at time 1. In general, if the algorithm accepts the second order which arrived at time t, it must be scheduled to time $t + 1$ and two more orders arrive at time $t + 1$. If the algorithm rejects the second order at time t, no more orders arrive. Suppose that the algorithm rejected the second order at $t \geq 1$. We have $z = 2 + t$ and $z^* = 2(t + 1) + 1 = 2t + 3$. $z^*/z \geq 5/3$ for $t \geq 1$. Hence, $z^*/z \geq 1.5$ for $t \geq 0$. □

Thus, the fact that we have to commit capacity at the time an order arrives makes the problem harder. We now complement this lower bound of 1.5 with an upper bound, by analyzing the following algorithm.

Algorithm Q-FRAC: (Quotation-FRACtional revenue)
Choose $0 < \alpha < 1$. At time t, schedule each order to the earliest available position, *only if* a revenue of at least αl can be obtained. Reject all the other orders which arrived at time t.

The main idea of Algorithm Q-FRAC is to accept orders only if they yield a certain fraction (α) of the maximum possible revenue (l in this case), so as to reserve capacity for orders that may arrive at a later time and bring more revenue. If we think of the next l time periods as our planning window, this means that we can promise our capacity for the first $(1-\alpha)l$ periods of the planning window, but we should keep the capacity of the last αl periods free for future orders. Our analysis chooses the value of this fraction to optimize the worst case performance.

Theorem 1 *If $\alpha = 0.618$, then Algorithm Q-FRAC has competitive ratio $z^*/z \leq 1/\alpha = 1.618$ for the case of single type unit length orders.*

Proof Consider an arbitrary phase i. (Refer to Section 1.7 for definition and associated notation.) Note that for any order accepted by the algorithm, we get a revenue at least αl. We consider two cases:

CASE 1: $t'_i - t_i \leq (1-\alpha)l$

In this case, we have exactly $k = t'_i - t_i$ arrivals during the time interval $[t_i, t_{i+1})$, and the maximum possible revenue one can get is kl. By the choice of the algorithm, we get revenue at least $k\alpha l$ for this time interval.

CASE 2: $t'_i - t_i > (1-\alpha)l$

By the choice of the algorithm, we cannot have any arrivals between $t'_i - (1-\alpha)l$ and t_{i+1}. The revenue we get during the interval $[t_i, t'_i)$ is at least

$$z_i \;\geq\; \frac{l(l+1)}{2} - \frac{\alpha l(\alpha l - 1)}{2} + \alpha l(t'_i - t_i - (1-\alpha)l).$$

The first two terms of the right hand side above give us a lower bound for the revenue of the first $(1-\alpha)l$ orders scheduled in B_i (if they all arrived at time t_i). The last term gives a lower bound for the revenue of the remaining orders. By rearranging terms, we have

$$z_i \;\geq\; \frac{(1-\alpha)^2}{2}l^2 + \frac{(1+\alpha)}{2}l + \alpha l(t'_i - t_i).$$

The maximum revenue z^*_i one can get from the arrivals in $[t_i, t'_i)$ is

$$z^*_i \;\leq l(t'_i - t_i - (1-\alpha)l) + \tfrac{l(l+1)}{2}.$$

During $[t_i, t_i' - (1-\alpha)l]$, the maximum revenue one can get is $l(t_i' - t_i - (1-\alpha)l)$, hence the first term of the right hand side above. Since there are no arrivals between $t_i' - (1-\alpha)l$ and t_{i+1}, the maximum revenue one can get between $t_i' - (1-\alpha)l$ and t_{i+1} is $\frac{l(l+1)}{2}$, which is the second term of the right hand side above. By rearranging terms, we have

$$z^* \quad \leq \quad l(t_i' - t_i) + (\alpha - \frac{1}{2})l^2 + \frac{l}{2}.$$

To balance the requirements of the two cases, we set

$$\frac{\alpha - \frac{1}{2}}{\frac{(1-\alpha)^2}{2}} = \frac{1}{\alpha}$$

and obtain $\alpha = 0.618$. Thus, for $\alpha = 0.618$ the ratio z_i^*/z_i is at most $1/\alpha = 1.618$ for any phase i, implying that the competitive ratio of this algorithm is at most $1/\alpha$. □

Our first example below shows that our analysis of Algorithm Q-FRAC is tight.

Example 1. Consider the case where all the orders are type 2. At time zero, l orders arrive. At time t, $t = 1, \ldots, n$, only one order arrives. For very large $n >> l$, the optimum solution is to schedule only one order at each time, to obtain a total revenue $z^* = nl$. The algorithm above will schedule at time zero $(1-\alpha)l$ orders to the first $(1-\alpha)$ periods, and then the order which arrives at time t will be scheduled to time $t + (1-\alpha)l$, $t = 1, \ldots, n$. The revenue of this schedule is $z = \frac{(1-\alpha)^2}{2}l^2 + \frac{(1+\alpha)}{2}l + \alpha ln$. The ratio z^*/z approaches $1/\alpha$ for large n.

2.2 On-line and Quotation Results for the Basic Model with Non-unit Processing Times

In contrast to the optimal performance of O-HRR for unit processing times, we show a lower bound of $\sqrt{2}$ for the competitive ratio of *any* on-line algorithm for O-SLTQ in case of non-unit processing times. (Thus, the non-unit processing time case is harder than the unit length case for O-SLTQ.) In this setting, there is an incentive to invest in better forecasting or obtaining advance information about orders. On the other hand, for Q-SLTQ, the upper bound carries over to this case from the $p = 1$ case. This indicates that Q-SLTQ does not get harder because of non-unit processing times. (The lower bounds always carry over from O-SLTQ to Q-SLTQ.)

Proposition 3 *If $p > 1$, then the competitive ratio of any on-line algorithm for O-SLTQ is at least $\sqrt{2}$.*

Proof Consider an instance with $l = p$. The machine is free at time zero, and an order arrives. Since all the orders are of the same type and since the machine is free at time zero, any good online algorithm should start processing this order immediately. If the processing of this order starts later, this will not only decrease the revenue one can obtain from this order but may also delay the processing of other orders that might arrive later. At time $t = (\sqrt{2} - 1)l$ another order arrives. If the algorithm decides to reject this order, no more orders arrive and we have $z^*/z = \frac{l+l(\sqrt{2}-1)}{l} = \sqrt{2}$. If the algorithm accepts this order, it will start processing it at some time $t \geq p$. Then, at time $p + 1$ another order will arrive (and can make revenue at most 1, since it can be scheduled only after the current order's processing is completed). In this case $z \leq l + 1 + l(\sqrt{2} - 1)$, $z^* = 2l$ and $z^*/z \geq \sqrt{2}$ for large l. $\square$

Proposition 4 *If $p > 1$, then the competitive ratio of any quotation algorithm for for Q-SLTQ is at least 1.5.*

Proof The proof is similar to the proof of proposition 2 and can be constructed by using an instance where $p = 2$ and $l = 4$. We skip the details.

Theorem 2 *If $\alpha = 0.618$, then Algorithm Q-FRAC has competitive ratio $z^*/z \leq 1/\alpha = 1.618$ for single type with $p > 1$.*

The proof of Theorem 2 is similar to the proof of Theorem 1. We skip the details.

In summary, O-SLTQ is more difficult than F-SLTQ only when $p > 1$ (so can benefit from information), while $Q - SLTQ$ is hard primarily because of *when* a decision has to be made.

3 Enhanced Model

Recall that in the enhanced model there are two types of customers; an urgent type who would like the product immediately, and a normal type whose availability window is longer.

First we study the case where all the orders have unit processing times. Without loss of generality we assume that there is at most one type 1 arrival (urgent type) in each time period and that $w_1 = w \geq 1$ and $w_2 = 1$. Let $l_1 = 1$ and $l_2 = l$. Note that the maximum revenue one can make from a type 1 order is w and the maximum revenue one can make from a type 2 order is l. We distinguish between the cases $w < l$ and $w \geq l$.

- $\underline{w < l.}$ The lower bound for O-SLTQ is 1.2808. The lower bound on Q-SLTQ is 1.5. This indicates that Q-SLTQ may be harder than O-SLTQ. We present an on-line algorithm for O-SLTQ with competitive ratio 1.618. We also present a quotation algorithm with competitive ratio at most $\max\{1 + \frac{w}{l}, 1.755\} \leq 2$. We use techniques similar to those applied in the basic model; the algorithms are a hybrid of the earlier idea of scheduling an order only if it yields at least a certain fraction of the maximum revenue (like algorithm Q-FRAC), along with the idea of scheduling the more profitable type of order first.

- $\underline{w \geq l.}$ We show that Algorithm O-HRR is an optimum on-line algorithm for O-SLTQ. We show that any quotation algorithm must have competitive ratio at least 2. We present a quotation algorithm for Q-SLTQ with competitive ratio at most 2.3524. Here, we use an interesting new idea of scheduling the less profitable orders by leaving evenly spaced gaps to allow for scheduling of the more profitable orders with shorter deadline, should they arrive later.

On-line version, O-SLTQ. The following propositions show that when $p = 1$, the online version of the enhanced model is harder than the corresponding basic model *only if* $w < l$. This is because when $w \geq l$, and an urgent order arrives, it is optimal to schedule it, and, if it does not arrive, then scheduling the normal orders based on remaining revenue is optimal (since the commitment of capacity is only one time unit). In the $w < l$ case, however, the static priority of urgent orders over normal orders is neither immediate, nor optimal.

Proposition 5 *If $w \geq l$, then Algorithm O-HRR is optimal for the enhanced model with unit length orders.*

Proposition 6 *Any on-line algorithm for O-SLTQ has competitive ratio at least 1.2808 if $w < l$.*

[**Remark.** Consider an algorithm that gives priority to the jobs with largest unit revenue, instead of largest remaining revenue.

Algorithm O-HUR: (On-line Highest Unit Revenue)
Whenever the machine is idle and there are orders available for scheduling, pick an order j with the largest w_j and schedule next. In case of ties, choose the order with the largest $rem_j(t)$.

One can see that O-SLTQ is harder than the online version of $1|r_j, p_j = 1|\sum w_j C_j$ by noting that Algorithm O-HUR finds the optimum solution for the latter problem while any on-line algorithm for O-SLTQ has competitive ratio at least 1.2808, even if there are only two types of orders.]

We now find an upper-bound for the case $w < l$. Consider the following algorithms.

Algorithm O-1HRR(On-line 1-first then O-HRR):
If a type 1 order is available at time t, schedule that order. Otherwise, schedule an order with the largest remaining revenue.

Algorithm O-HYBRID:
Let $\alpha = \frac{\sqrt{5}-1}{2}$. If $w \geq \alpha l$, use algorithm O-1HRR; otherwise, use algorithm O-HRR.

Theorem 3 *The competitive ratio of algorithm O-HYBRID is at most 1.618.*

The theorem follows from propositions 7 and 8. Proposition 7 shows that if $w \geq \frac{\sqrt{5}-1}{2} l$, then Algorithm O-1HRR has competitive ratio at most 1.618. Proposition 8 shows that if $w \leq \frac{\sqrt{5}-1}{2} l$, then Algorithm O-HRR has competitive ratio at most 1.618.

Proposition 7 *The competitive ratio of Algorithm O-1HRR is at most l/w.*

Proof Consider an arbitrary instance I, where the algorithm scheduled k type 1 orders (i.e. there were type 1 arrivals in k periods) and made revenue $z = C + wk$. We claim that the optimum solution is $z^* \leq C + lk$. To see why this is the case, consider another instance I', where there is a type 2 arrival instead of every type 1 arrival. Let z' be the optimum solution for I'. z' is clearly better than z^*, and can be obtained by scheduling the order with the largest remaining revenue at any time. But then $z' = C + lk \geq z^*$ and $z^*/z \leq l/w$.

Observation: Suppose that in an optimum solution an order j is scheduled at time t, with remaining revenue $rem_j(t)$, although there was another order k available for scheduling at time t, with $rem_k(t) > rem_j(t)$. Then, order k must also be accepted and scheduled later in the optimum solution.

Proposition 8 *The competitive ratio of Algorithm O-HRR is* $1 + \frac{w}{l}$.

Proof Consider an arbitrary phase i. Suppose that x of the type 1 orders which are rejected during phase i by Algorithm O-HRR are accepted in the optimum solution. The algorithm rejected these type 1 orders, because some type 2 orders with remaining revenue $> w$ were available when they arrived. (If there were no type 1 arrivals, then z_i would be the optimum solution.) By scheduling these x type 1 orders, the optimum algorithm made wx more revenue during this phase. In the best case, the type 1 orders are scheduled to the interval $[t'_i - x, t'_i]$, and the x type 2 orders are scheduled immediately after t'_i. Therefore, the optimum algorithm will lose at least $1 + 3 + \ldots + 2x - 1 = x^2$ due to those x type 2 orders (cf. previous observation).

Let z_i and z_i^* denote the revenue made by the algorithm and by the optimum solution in phase i, respectively. We have $z_i^* \le z_i + xw - x^2$, and $z_i^*/z_i \le 1 + \frac{wx - x^2}{z_i}$. During the first x periods of phase i, the revenue made by the algorithm is at least $l + (l-1) + \ldots + (l - x + 1) \ge \frac{2l-x}{2}x$ (This happens when all these orders arrive in period t_i). Hence, $z_i \ge \frac{2l-x}{2}x$ and

$$\frac{z_i^*}{z_i} \le 1 + \frac{wx - x^2}{\frac{2l-x}{2}x} = 1 + \frac{2w - 2x}{2l - x} \le 1 + \frac{w}{l}.$$

Since this is true for any phase, the competitive ratio of this algorithm is at most $1 + \frac{w}{l}$. □

Quotation version, Q-SLTQ. Next we turn to quotation algorithms.

Algorithm Q-FRAC-HYBRID:

Choose $0 < \alpha < 1$. If there is a type 1 arrival at time t:

Schedule the type 1 order, if there are no type 2 orders available for scheduling; otherwise, schedule the type 1 order only if $w > \alpha l$.

If there are type 2 arrivals at time t, schedule the type 2 orders to the earliest available positions, as long as you will make at least αl revenue for each order scheduled. Reject all the remaining type 2 orders.

The proof of the next theorem is in the appendix.

Theorem 4 *If $w < l$ and $\alpha = 0.56984$, then Algorithm Q-FRAC-HYBRID has competitive ratio $z^*/z \leq \max\{1 + \frac{w}{l}, \frac{1}{\alpha}\}$.*

Note that for $w = l$, we have an on-line algorithm which gives the optimum solution, but any quotation algorithm has competitive ratio at least 2 as shown by Proposition 9 below. This indicates that the quotation version continues to be harder than the on-line version in the enhanced model.

Proposition 9 *If $w \geq l$, then any on-line quotation algorithm for Q-SLTQ has competitive ratio $z^*/z \geq 2$.*

We now turn to construct a good algorithm for Q-SLTQ. Here is the intuition. First, recall that w is the largest revenue we can make from a type 1 order and l is the largest revenue we can make from a type 2 order. If $w \geq l$, the revenue we can make from a type 1 order can be arbitrarily larger than the revenue we can make from a type 2 order. Also note that a type 1 order is lost, if it is not scheduled immediately at its arrival. Therefore, in the quotation version, it is crucial to leave some capacity free for possible future type 1 arrivals, while we accept and quote lead times for type 1 orders. As in the single type case, we will accept type 2 orders only if they yield a certain fraction (α) of the maximum possible revenue (l in this case), in order to reserve capacity for other type 1 or type 2 orders which may arrive at a later time (and bring more revenue). If we think of the next l time periods as our planning window, this means that we should keep the capacity of the last αl periods free for future orders. However, due to the possible revenue difference between type 1 and type 2 orders, reserving capacity only in this fashion is not enough. Therefore, we also need to leave "gap"s (i.e. reserve capacity) between type 2 orders, while we quote lead times. These "gap"s may be filled later with type 1 orders or high revenue type 2 orders. So, in our proposed algorithm Q-GAP, in addition to leaving the last αl periods, we also leave β fraction of the first $(1 - \alpha)l$ periods in the planning window free while quoting lead times for type 2 orders.

Algorithm Q-GAP:

Choose $0 \leq \alpha \leq \beta \leq 0.5$. If the machine is available, schedule a type 1 order as soon as it arrives. Quote lead times for type 2 orders leaving the machine free for at least β fraction of the time (as evenly as possible), if you will make at least αl revenue for each order. If the machine is available and there are no new arrivals, "pull" the order with the earliest quoted due date (which must be type 2) and process it.

Our analysis chooses the values of α and β to optimize the worst case performance of the algorithm.

Theorem 5 *If $\alpha = 0.4251$, algorithm Q-GAP has competitive ratio at most $\frac{1}{\alpha} = 2.3524$ for unit length orders.*

Proof We first prove the theorem for $l > 3$. Consider an arbitrary phase i. By the definition of the phases and the choice of the algorithm, there are no arrivals in $[t'_i, t_{i+1}]$. If no type 2 order is scheduled in that phase, then it consists of a single type 1 order, and we have $z^*_i = z_i$.

If type 2 orders are scheduled in phase i, let n_i be the number of type 1 orders scheduled during the interval $[t_i, t'_i]$. Since we leave the machine free for at least β fraction of the time, and since a type 2 order is "pulled" only if there is no type 1 arrival (hence, increasing the free space), at least β fraction of all type 1 arrivals are scheduled in a phase.

Consider the following two cases:

CASE 1: $t'_i - t_i - n_i < (1-\alpha)(1-\beta)l$
This means that the number of type 2 arrivals during this phase was $t'_i - t_i - n_i$ (and all of them are scheduled), because otherwise the algorithm would schedule more type 2 orders. Furthermore, the maximum number of type 1 arrivals in this phase is $\frac{1}{\beta}n_i$ which is at most $(t'_i - t_i)$, and hence $n_i \leq (t'_i - t_i)\beta$. So, we have

$$z^*_i \leq (t'_i - t_i - n_i)l + w\frac{1}{\beta}n_i \quad \text{and} \quad z_i \geq (t'_i - t_i - n_i)\alpha l + wn_i.$$

Therefore, $z^*_i / z_i \leq \max\left\{\frac{1}{\alpha}, \frac{1}{\beta}\right\}$.

CASE 2: $t'_i - t_i - n_i \geq (1-\alpha)(1-\beta)l$

$$z_i \geq \left(\frac{l(l+1)}{2} - \frac{\alpha l(\alpha l - 1)}{2}\right)(1-\beta) + (t'_i - t_i - n_i - (1-\alpha)(1-\beta)l)\alpha l + wn_i$$

We claim that the first term of the right hand side above denotes the minimum revenue made from the first $(1-\alpha)(1-\beta)l$ type 2 orders scheduled in phase i. The second term denotes the minimum revenue made from the remaining type 2 orders. The last term denotes the revenue made from the type 1 orders. After rearranging terms, we get

$$z_i \geq (1-\beta)(1-\alpha)^2\frac{l^2}{2} + (1-\beta)(1+\alpha)\frac{l}{2} + \alpha(t_i' - t_i - n_i)l + wn_i.$$

Before computing an upper bound on z_i^*, again note that there are no arrivals in $[t_i', t_{i+1}]$. Let $\delta = \alpha + \beta - \alpha\beta$.

$$z_i^* \leq (t_i' - t_i - n_i)\, l + \frac{1}{\beta}wn_i + \frac{\delta l(\delta l + 1)}{2}$$

The first two terms of the right hand give an upper bound on the maximum revenue one could make during the interval $[t_i, t_i']$. The third term of the right hand side is an upper bound on the maximum revenue one could make from type 2 orders that were rejected by the algorithm but could be scheduled after t_i'. The algorithm rejected such orders which arrived in $[t_i, t_i']$, if they had remaining revenue less than αl. If we did not leave β fraction of the time free, and if we scheduled such an order after t_i', we could make at most $\alpha l + \beta(1-\alpha)l$, which is equal to δl giving the third term. After rearranging terms, we get

$$z_i^* \leq \delta^2\frac{l^2}{2} + \delta\frac{l}{2} + (t_i' - t_i - n_i)\, l + \frac{1}{\beta}wn_i$$

and

$$\frac{z_i^*}{z_i} \leq \max\left\{\frac{\delta^2}{(1-\beta)(1-\alpha)^2}, \frac{\delta}{(1-\beta)(1+\alpha)}, \frac{1}{\alpha}, \frac{1}{\beta}\right\}.$$

If we choose $\alpha = \beta$ and solve for $\frac{\delta^2}{(1-\beta)(1-\alpha)^2} = \frac{1}{\alpha}$, we get $\alpha = 0.4251$.

For $\alpha = 0.4251$, $z_i^*/z_i \leq 1/\alpha$ in any phase i, and the competitive ratio of this algorithm is at most $1/\alpha = 2.3524$.

Now we do the analysis for $l \leq 3$ and $\alpha = 0.4251$. If there is no type 1 arrival, but there are type 2 arrivals in a given period, the revenue made by the algorithm in that time period is l. The maximum revenue one could make from the the orders which arrived in that time period is $l(l+1)/2$. If there is a type 1 arrival in a given period, the revenue made by the algorithm is w, whereas the maximum revenue one could make from the orders that arrived in that time period is $w + l(l-1)/2$. Since $w \geq l$, we have $z_i^*/z_i \leq \max\{(l+1)/2, 1 + (l-1)/2\} \leq 2$ for $l \leq 3$. $\square$

We created an example where the ratio z^*/z goes to $\frac{1}{(1-\alpha^2)(1-\alpha)}$ (which is equal to 2.1231 for $\alpha = 0.4251$), which shows that the analysis of this algorithm is almost tight.

Next we turn to $p > 1$. First, we show a lower bound for the competitive ratio of any on-line algorithm. This same lower bound can be used for any quotation algorithm as well.

Proposition 10 *Any on-line algorithm with* $p > 1$ *has competitive ratio at least* $\max\{\sqrt{2}, \frac{w}{l}\}$.

Next, we show that the performance of Q-FRAC is within a constant of this lower bound. The proof of Theorem 6 is similar to the proof of Theorem 4.

Theorem 6 *Algorithm Q-FRAC has competitive ratio at most* $\max\{2, 1.75488\frac{w}{l}\}$ *with* $p > 1$.

4 Delayed Quotation Problems

In terms of decision making, the on-line and quotation versions consider two extremes. In this section we consider the delayed quotation problem (D-SLTQ), which generalizes Q-SLTQ by allowing a waiting time q_i for decision making, such that $0 \le q_i < l_i$ (for a type i order). (Interestingly, an emerging topic of interest within the computer science community is the study of on-line models with a "look ahead" feature, which parallels the delayed quotation feature of our model [22].)

We have at least two options on how we think a customer behaves in this setting.

- *No change in the revenue function.* The first option is that the revenue function remains the same, i.e. revenue is lost for every unit of time an order waits before its processing starts.

- *The revenue function is more lenient.* In this case, if the quoted lead time is less than q_i, then the full revenue is obtained. Decisions have to be made within q_i time periods, but revenues start to decrease only if the quoted lead time is longer than q_i. Furthermore the availability interval is now $q_i + l_i$, i.e. longer by q_i units. One such revenue function is the following.

$$R'_i(d) = \begin{cases} w_i(l_i - \max\{0, d - q_i\}) & \text{if } d < l_i + q_i \; i = 1, 2 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

In both of these delayed quotation models, without loss of generality we can assume that $q = (1-\delta)(l-1)$ for some $0 \leq \delta \leq 1$. If $q = l-1$ ($\delta = 0$), D-SLTQ with revenue function R reduces to the standard on-line scheduling problem O-SLTQ. If $q = 0$ ($\delta = 1$), then D-SLTQ is equivalent to the immediate quotation model Q-SLTQ.

First, let us consider D-SLTQ for our basic model with a single type of customer. To quantify the impact of delaying the quotation decision on performance, we first design a new algorithm, Q-HRR, which is a modified version of O-HRR. Theorem 7 shows that by using algorithms Q-FRAC and Q-HRR together, we can obtain an increase in the revenues as q increases.

Algorithm Q-HRR (Quotation version of O-HRR):
At time t, from the set of orders available for scheduling, choose the one with the largest remaining revenue and process. Reject all the orders with remaining revenue $\leq \delta l$. (These are the orders whose quotation time is over.)

Theorem 7 *Consider D-SLTQ in the basic model with unit length orders and let* $q = (1-\delta)(l-1)$. *There is an on-line quotation algorithm with competitive ratio at most* $\min\{1.618, \frac{1}{1-\delta^2}\}$.

Proof Let σ denote the schedule generated by the above algorithm. Again, we do the analysis by dividing σ into phases. Phase i starts at time t_i, if the following conditions hold:
1. An order is scheduled to time t_i and the arrival time of that order is also t_i.
2. There are no orders waiting for quotation at time t_i.
Let t'_i be the start time of the last order scheduled in phase i.
Note that the machine will always be busy during the interval $[t_i, t'_i]$. Let z^*_i be the maximum revenue one could make from the orders which arrived during phase i and z_i be the revenue made by the algorithm. Now, consider the following two cases:

CASE 1: $t'_i - t_i < (1-\delta)l$
This means that the number of arrivals in this phase was exactly $t'_i - t_i$ (and all of them are scheduled). Hence, $z^*_i = z_i$ in this phase.

CASE 2: $t'_i - t_i \geq (1-\delta)l$
In this case we made the maximum possible revenue, except possibly losing δl orders at

the end of the phase. The minimum revenue we made during this phase is

$$z_i \geq \frac{l(l+1)}{2} - \frac{\delta l(\delta l - 1)}{2} \text{ and we lost at most } \frac{\delta l(\delta l - 1)}{2}.$$

$$\text{Therefore, } z_i^* \leq z_i + \frac{\delta l(\delta l - 1)}{2} \text{ and } \frac{z^*}{z} \leq \frac{l^2 + l}{(1-\delta^2)l^2 + (1+\delta)l} \leq \frac{1}{1-\delta^2}.$$

The ratio decreases, as δ decreases, i.e. as q increases.
If $\delta \leq 0.618$, then algorithm Q-HRR gives $z^*/z \leq 1.618$. So, one can use algorithm Q-FRAC for $\delta > 0.618$, and algorithm Q-HRR for $\delta \leq 0.618$. □

The result of Theorem 7 quantifies the increase in revenues, as the waiting time $q = (1-\delta)(l-1)$ increases. For $\delta \leq 0.618$, we are able to show that revenues increase quadratically as the waiting time increases linearly. One can compare the revenue increase with the "cost" of asking a customer to wait for a quote, and decide whether it is worth delaying the decision.

The following two results show how delayed quotation impacts the revenues, for the second revenue function R' in the enhanced model. In this case, we observe that there is a sharp decrease in the worst case performance guarantees, once the waiting time exceeds a threshold of $q = p - 1$.

First, we show that if $q_i \leq p-1$ for one of the types, then delaying the quotation decision does not improve the worst case performance.

Proposition 11 *If $q_i < p - 1$ for some i, then w/l is a lower bound on the competitive ratio of any quotation algorithm for enhanced model with revenue function R'.*

Proof Consider an instance where $l_1 = l_2 = 1$, $q_1 < p - 1$ and $q_2 = p$. At time zero, the machine is idle and a type 2 order arrives. Any on-line algorithm has two options, either it will reject the order, or it will accept it and start processing it at some time $t < p + 1$. (The decisions can be made within p time units.) If the algorithm rejects the order, another type 2 order will arrive at time p, and the type 2 orders will continue to arrive every p time periods, as long as the algorithm rejects them. If the algorithm does not accept any of these type 2 orders, we have $z = 0$ and $z^* = t/p$ at time t. If the algorithm decides to accept a type 2 order which arrived at time t, it will start processing this order at time $t' < t + p$ and no more type 2 orders will arrive. Then, at time $t' + 1$

a type 1 order will arrive. In this case, since the machine is busy for the next $p-1$ time periods, this type 1 order is lost and we have $z = 1$ and $z^* = t/p + w$. $\square$

A similar lower bound can be shown for revenue function R.

Next, we consider a modified version of Algorithm Q-GAP.

Algorithm Q-LONG-GAP:
Choose $0 \le \alpha, \beta \le 0.5$. Schedule a type 1 order to the earliest available position. Quote lead times for type 2 orders based on the following conditions:
1. Leave the machine free for p period long intervals, for at least β fraction of the time (as evenly as possible).
2. If the revenue you will make from a type 2 order is less than αl, reject that order.
If the machine is available and there are no new arrivals, "pull" the order with the earliest quoted due date (which must be type 2) and process it.

The following result shows that once the decisions can be delayed longer than a threshold of $p-1$, revenues increase significantly as the competitive ratios decrease sharply to a constant.

Theorem 8 *If $q_1 = q_2 = p - 1$, $l_1 = 1$ and $\alpha = 0.4251$, then Algorithm Q-LONG-GAP has competitive ratio at most $1/\alpha$ for revenue function R'.*

The proof of Theorem 8 is similar to the proof of Theorem 5.

In summary, we see that delaying the decision can be beneficial, although the benefit may be monotone in some cases, and a threshold type for others.

5 Conclusion

Motivated by real applications, we considered the problem of scheduling and lead time quotation when revenues are decreasing with lead times and the orders have an availability interval. We studied, for a basic model and an enhanced model, four versions – F-SLTQ, O-SLTQ, Q-SLTQ and D-SLTQ – that differ in what information is known and when decisions have to be taken. We have provided complexity results for the off-line case, and competitive analyses for the on-line cases. Several useful qualitative insights about the relative difficulty of the versions leads to improved managerial decision making about when to collect more information and when to delay a quotation decision. These insights have been used as part of real world implementations of accurate lead time quotation; see [31] for an example.

Our approach for measuring the performance of on-line algorithms was to use competitive analysis, in which the performance of an on-line algorithm is compared to the performance of an optimum off-line algorithm, which knows the input sequence in advance. Although competitive analysis allows us to obtain theoretical bounds on the “worst case” performance of on-line algorithms, this approach so far has not had much impact on the development of real systems. One reason is that (in practice) the orders in the near future are at least partially predictable, but competitive analysis assumes that an on-line algorithm has no information about the future. Another reason is that in practice the probability that a problem instance will cause an online algorithm to realize its worst case performance may be very small. For example, in the SLTQ context it is very unlikely that a group of clients will select the release times and the lead times in order to frustrate the decision maker. However, to have a good competitive ratio an online algorithm still has to be designed to perform well in such worst possible instances. Despite its limitations, we believe that this approach is a good alternative and complements other existing methods (such as queuing) widely used within our community.

Future research considers computational testing to find average case performance of the algorithms as well as study randomized algorithms. The study of a general case that includes non-equal processing times with $m > 2$ types is also underway.

Appendix

Proof of Theorem 4 The proof is similar to the proof of Theorem 1. In every phase, at most one type 1 order is scheduled. If a phase consists of type 1 orders only, then it has exactly one order in it. In that case, we have $z_i^* = z_i$.

If at least one type 2 order is scheduled in a phase i, we consider two cases:

CASE 1: $t_i' - t_i \leq (1-\alpha)l$
In this case, we can have at most $k = t_i' - t_i$ type 2 arrivals during the interval $[t_i, t_{i+1})$.

All the type 2 orders which arrived in this time interval are accepted, and scheduled in the order of nondecreasing arrival times. If there were no type 1 arrivals during this period, this would be the optimum solution (this can be shown by a simple interchange argument). But there may be some type 1 orders, which are rejected due to the type 2 orders scheduled consecutively by the algorithm. Consider an instance I' with the same type 2 arrivals as in this phase, but also with a type 1 arrival in each period. Let z' be the optimum solution for that instance. Clearly, $z' \geq z_i^*$. If the optimum algorithm accepts x of those type 1 orders in I', then it will make an extra wx revenue. As before, the optimum algorithm will lose at least $1 + 3 + \ldots + 2x - 1 = x^2$ due to those x type 2 orders. (This lower bound is attained if all of the x type 1 orders are consecutively scheduled in the interval $[t_i' - x, t_i']$, which minimizes the revenue loss due to the delayed type 2 orders.) $x \leq k$, since there are k periods. We have $z_i^* \leq z' \leq z_i + wx - x^2$ and $z_i \geq l + (l-1) + \ldots + (l-k+1) \geq lk - \dfrac{k^2}{2} \geq lx - \dfrac{x^2}{2}$. Hence, $z_i^*/z_i \leq 1 + \dfrac{w}{l}$.

CASE 2: $t_i' - t_i > (1-\alpha)l$

By the choice of the algorithm, we cannot have any type 2 arrivals between $t_i' - (1-\alpha)l$ and t_{i+1}. Furthermore, we cannot have any type 1 arrivals between t_i' and t_{i+1}. By the definition of the phases, only the first order in B_i may be a type 1 order, and all the other orders must be type 2 orders.

If the first order is a type 1 order, the revenue (denoted by z_i) we will get for this time interval will be at least

$$z_i \;\; \geq \;\; w + \frac{l(l-1)}{2} - \frac{\alpha l(\alpha l - 1)}{2} + \alpha l(t_i' - t_i - (1-\alpha)l)$$

For the first order, which is of type 1, we get revenue w; for the following $(1-\alpha)l - 1$ orders, which must be of type 2, we get revenue at least $\frac{l(l-1)}{2} - \frac{\alpha l(\alpha l-1)}{2}$; finally, for each of the remaining orders, we get revenue at least αl, which gives us the last term of the above right hand side. Since $w \geq \alpha l$ by the choice of the algorithm, we have

$$z_i \;\; \geq \;\; \frac{(1-\alpha)^2}{2} l^2 + \frac{(3\alpha - 1)}{2} l + \alpha l(t_i' - t_i).$$

If all the orders in B_i are type 2 orders, then the revenue we will get for this time interval will be at least

$$z_i \;\; \geq \;\; \frac{l(l+1)}{2} - \frac{\alpha l(\alpha l - 1)}{2} + \alpha l(t_i' - t_i - (1-\alpha)l)$$

The first two terms of the right hand side above give us a lower bound for the revenue of the first $(1-\alpha)l$ orders scheduled in B_i, and the last term gives a lower bound for the revenue of the remaining orders. By rearranging terms, we have

$$z_i \;\geq\; \frac{(1-\alpha)^2}{2}l^2 + \frac{(1+\alpha)}{2}l + \alpha l(t'_i - t_i).$$

$$\text{The maximum revenue is } \; z^*_i \leq l(t'_i - t_i) + \frac{\alpha l(\alpha l - 1)}{2}.$$

During $[t_i, t'_i]$, the maximum revenue one can get is $l(t'_i - t_i)$, hence the first term of the right hand side above. Since there are no type 2 arrivals between $t'_i - (1-\alpha)l$ and t_{i+1}, and no type 1 arrivals between t'_i and t_{i+1}, the maximum revenue one can get between t'_i and t_{i+1} is $\frac{\alpha l(\alpha l-1)}{2}$, which is the second term of the right hand side above. By rearranging terms, we have

$$z^* \leq l(t'_i - t_i) + \frac{\alpha^2}{2}l^2 - \frac{\alpha}{2}l.$$

If we choose $\alpha = 0.56984$, then for the interval $[t_i, t_{i+1})$, the ratio z^*/z is at most $1/\alpha$.

References

[1] E.M. Arkin, E.B. Silverberg (1987), "Scheduling jobs with fixed start and end times", *Discrete Applied Mathematics* 18, 1-8.

[2] K.R. Baker (1984), "Sequencing rules and due-date assignments in a job shop", *Management Science* Vol. 30, No. 4, 1093-1104.

[3] K.R. Baker, J.W.M. Bertrand (1981), "A comparison of due-date selection rules", *AIIE Transactions*, Vol. 13, No. 2, 123-131.

[4] K.R. Baker, J.W.M. Bertrand (1982), "A Dynamic Priority Rule for Scheduling Against Due-Dates", *Journal of Operations Management*, Vol. 3, No. 1, 37-42.

[5] J.W.M. Bertrand (1983), "The effect of workload dependent due-dates on job shop performance", *Management Science*, Vol. 29, No. 7, 799-816.

[6] J.H. Bookbinder, A.I. Noor (1985), "Setting job-shop due-dates with service-level constraints", *J. Oper. Res. Soc.*, Vol. 36, No. 11, 1017-1026.

[7] A. Borodin, R. El-Yaniv (1998), Online Computation and Competitive Analysis, Cambridge University Press, United Kingdom.

[8] S. Chand, D. Chhajed (1992), "A single machine model for determination of optimal due dates and sequence", *Operations Research*, Vol. 40, No. 3, 596-602.

[9] T.C.E. Cheng (1984), "Optimal due-date determination and sequencing of n jobs on a single machine", *J. Oper. Res. Soc.*, Vol. 35, No. 5, 433-437.

[10] T.C.E. Cheng, M.C. Gupta (1989), "Survey of scheduling research involving due date determination decisions", *European Journal of Operational Research* 38, 156-166.

[11] R.W. Conway (1965), "Priority dispatching and job lateness in a job shop", *The Journal of Industrial Engineering*, Vol. 16, No. 4, 228-237.

[12] I. Duenyas (1995), "Single facility due date setting with multiple customer classes", *Management Science*, Vol. 41, No. 4, 608-619.

[13] S.Elion, I.G. Chowdhury (1976), "Due Dates in Job Shop Scheduling", *International Journal of Production Research*, Vol. 14, No. 2, 223-237.

[14] U. Faigle and W.M. Nawijn (1994), "Note on scheduling intervals on-line", *Discrete Applied Mathematics* 58, 13-17.

[15] R.L.Graham, E.L.Lawler, J.K.Lenstra and A.H.G.Rinnooy Kan (1979), "Optimization and approximation in deterministic sequencing and scheduling: A survey", *Annals of Discrete Mathematics*, No. 5, 287-326.

[16] N.G. Hall and M.J. Magazine (1994), "Maximizing the value of a space mission", *European Journal of Operational research* 78, 224-241.

[17] L. Hall, D.B. Shmoys and J. Wein (1996), "Scheduling to minimize average completion time: Off-line and online algorithms", in *Proceedings of the 7th ACM-SIAM Symposium on Discrete Algorithms*, 142-151.

[18] Robert B. Handfield and Ernest L. Nichols, Jr., *Introduction to Supply Chain Management*, Prentice Hall, NJ, 1999.

[19] J.A.Hoogeveen and A.P.A.Vestjens (1996), "Optimal On-Line Algorithms for Single Machine Scheduling", in *Proceedings of the Fifth Conference on Integer Programming and Combinatorial Optimization*, 404-414.

[20] R. Kapuscinski and S. Tayur (1997), "100% reliable quoted lead times", *GSIA working paper*, Carnegie Mellon University, Pittsburgh, PA.

[21] P. Keskinocak (1997), Satisfying Customer Due Dates Effectively, Ph.D. Thesis, GSIA, Carnegie Mellon University.

[22] P. Keskinocak (1998), "On-line Algorithms: How Much is it Worth to Know the Future?", IBM Research Report RC21340(96133).

[23] S. Miyazaki (1981), "Combined scheduling system for reducing job tardiness", *Int. J. Prod. Res.*, Vol. 19, No. 2, 201-211.

[24] C.Phillips, C.Stein and J.Wein (1995), "Minimizing Average Completion Time in the Presence of Release Dates", to appear in *Mathematical Programming*, earlier version appeared in *Algorithms and Data Structures: 4th International Workshop, WADS'95, Kingston, Canada, August 1995, Proceedings*, 86-97.

[25] M. Pinedo (1995), *Scheduling: Theory, Algorithms and Systems*, Prentice Hall, New Jersey.

[26] G.L. Ragatz and V.A. Mabert (1984), "A simulation analysis of due date assignment rules", *Journal of Operations Management*, Vol. 5, No. 1, 27-39.

[27] A. Seidman, S.S. Panwalker and M.L. Smith (1981), "Optimal assignment of due-dates for a single processor scheduling problem", *Int. J. Prod. Res*, Vol. 19, No. 4, 393-399.

[28] A. Seidman and M.L. Smith (1981), "Due date assignment for production systems", *Management Science*, Vol. 27, No. 5, 571-581.

[29] D.D. Sleator and R.E. Tarjan (1985), "Amortized efficiency of list update and paging rules", *Communication of the ACM* 28, 202-208.

[30] G. Stalk Jr., and T. H. Hout (1990), *Competing Against Time*, The Free Press, New York.

[31] S. Tayur (1998). " Improving Operations and Quoting Accurate Lead Times in a Laminate Plant", to appear in *Interfaces.*

[32] J.K. Weeks (1979), "A simulation study of predictable due-dates", *Management Science*, Vol. 25, No. 4, 363-373.

[33] L.M. Wein (1991), "Due-date setting and priority sequencing in a multi-class M/G/1 queue", *Management Science*, Vol. 37, No. 7, 834-851.

[34] G.J. Woeginger (1995), "On-line scheduling of jobs with fixed start and end times", *Theoretical Computer Science* 130, 5-16.

This research was sponsored in part by National Science Foundation (NSF) grant no. CCR-0122581.