Wave Release Strategies for Order Fulfillment Systems with Deadlines

by

Erdem Çeven

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
August 3, 2013

Keywords: Order release control, wave processing, time guaranteed delivery, cutoff time, fluid approximations

Approved by

Kevin R. Gue, Tim Cook Associate Professor of Industrial and Systems Engineering
Jorge Valenzuela, Professor and Chair of Industrial and Systems Engineering
Chase Murray, Assistant Professor of Industrial and Systems Engineering

Abstract

A distribution center is ultimately in the business of shipping orders. Because every customer's experience is determined by how timely and accurately an order is received, distribution centers should focus not only on how fast they process orders, but also on keeping their operations synchronized and delivering orders on time. In traditional distribution centers, including internet and catalog fulfillment systems, synchronization and economies of scale are achieved with "waves."

A wave is a large group of orders that are picked and packed together, sorted for individual customers, and shipped. Problems arise when those internal processes are not properly coordinated. When operational parameters, such as number and timing of waves, are not planned systematically, missed orders and upset customers are the result.

We present the first systematic investigation of wave planning in order fulfillment systems. We develop continuous fluid models of work content in an order fulfillment system, in order to determine the optimal timing and number of waves with the goal of maximizing service performance against a deadline-oriented metric.

Acknowledgments

I would like to express my deepest gratitude to my advisor, Dr. Kevin R. Gue, for his excellent guidance, patience, brilliant ideas and providing me with an excellent atmosphere for doing research. I am most grateful for his help and support.

I would also like to thank my committee members, Dr. Chase Murray and Dr. Jorge Valenzuela for their thoughtful criticism, time and suggestions. I am grateful to the Naval Postgraduate School, which partially supported this research under Grant NPS-BAA-11-002 at Acquisition Research Program, Auburn University Industrial and Systems Engineering Department and Graduate School, which also funded my graduate work.

I would also like to thank my parents, Hüseyin and Süheyla and my sister Zümrüt. Without their presence, endless support, great understanding and love, my research would not have been possible. They always support me and encourage me with their best wishes.

I would like to thank my all friends who always supported me: Eren Sakınç, Kaan Berberoğlu, Polly Chen. They were always there cheering me up and stood by me through the good times and bad.

This dissertation is dedicated to my parents.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Within a supply chain, products need to be physically moved from the point of origin to the point of consumption. During this process, order fulfillment centers receive product from suppliers, store it for a certain period of time, and fulfill customer orders. In this context, many production and distribution systems can be defined as order fulfillment systems.

Distribution centers (DCs) receive orders from geographically dispersed customers who have increased expectations of getting their products at a desired level of quality and promised time of delivery. With the growing success of e-commerce, distribution centers often receive a large number of small orders which have to be fulfilled within very tight time windows. From 1993 to 2007, smaller size shipments increased by around 107%*. The result of these trends has been more complex distribution centers and more tightly controlled order fulfillment systems.

Today, the world's newest and most successful companies, such as Amazon, Zappos, and WalMart have increased competitiveness by offering better service promises. For example, Amazon customers are familiar with this deadline-driven offer: "Want it delivered Monday, July 11, 2013? Order it in the next 3 hours and 42 minutes, and choose One-Day Shipping at checkout." Amazon is taking its service one step further by investing billions to make next-day delivery standard and same-day delivery an option (Manjoo, 2012). But, how should DCs align their functions properly in order to fulfill orders in these deadline oriented environments?

*2007 Commodity Flow Survey.

In our research, we address the problem of controlling order flow within a DC which operates in a deadline oriented environment. Our main purpose is to develop optimal flow control policies for DCs to increase their service levels.

1.1 Order Flow in Distribution Centers

The basic flow of an order in a distribution center starts with receiving and put away. Receiving is the activity of gathering products from suppliers. The put away process moves products from receiving to designated storage locations. Because the flow is from suppliers to the DC, these activities are usually called inbound operations. Once the products are located in storage locations, they are available for outbound activities. These activities mainly include order picking, accumulation, packing, and shipping.

Order picking, which involves gathering customer orders from storage locations, is the major activity in most DCs. When multiple orders are picked in groups (batches), they must be sorted based on destination. Because multiple customers may be assigned to the same destination, an accumulation process is necessary. Accumulated orders are transferred to packing and then to the shipping docks.

Among all fulfillment activities, order picking constitutes a significant portion of the total warehouse operating expense (de Koster et al., 2006). Although automated picking systems are also common (e.g. automated storage and retrieval systems, robotic picking), the majority of DCs employ picker-to-parts systems (de Koster, 2004).

In picker-to-parts systems, pickers either pick orders individually (discrete picking) or in *batches*. The term *sort-while-pick* is used when pickers pick multiple products simultaneously and immediately sort them. The term *pick-and-sort* is used if sortation is after picking. From an operational point of view, pickers are segregated into *zones* and in each zone, pickers may either pick orders in a *synchronized* (pickers in different zones pick the same batch of orders simultaneously) or *progressive* way (where an order is completed after sequential processing in all zones).

1.2 Wave Processing

Within a DC, picking orders in different ways has long been standard practice. One-size and multi-size batch picking are common in practice, especially when the number of orders (workload) is small. As daily workload increases, DCs require more complex processes. Daily workload can also fluctuate significantly in complex fulfillment environments. In highly complex systems, for example, maximum workload can even be more than twice the average daily workload (van der Berg, 2010). As a consequence, operating characteristics of DCs change dramatically (Table 1.1).

Table 1.1: Operating characteristics of DCs (Source: van der Berg, 2010).

Complexity	Low	Medium	High
Number of operators	≤ 15	15-45	≥ 45
Warehouse area (m^2)	$\leq 5,000$	5,000-15,000	$\geq 15,000$
Orders per day	$\leq 1,000$	1,000-1,000	$\geq 5,000$
Warehouse area	$\leq 5,000$	5,000-15,000	$\geq 15,000$
Separate picking areas	1	2-3	≥ 4
Shipping pattern	1 wave	2-3 waves	Individual
Fluctuating workload	$\leq 50\%$	$\leq 150\%$	$\geq 250\%$

Orders can be picked in a single zone, representing the entire picking region, or in multiple zones. DCs with medium or high complexity require multiple zones. When multiple pickers in different zones pick for the same pool of orders simultaneously, order release is controlled with *waves*. A wave is basically a set of orders grouped by some criteria and which is released to the floor for processing at the same time. Attributes for grouping might be a mode of transportation, a group of stores in retail, high priority orders, orders requiring a specific type of value-added service, or even an individual customer if it orders in large enough quantities.

A basic wave planning and control process involves creation, release, and monitoring. In practice, wave planning is performed manually. Wave planners decide the criteria that determine which orders to include in the wave. Creating waves of orders that are all shipping via UPS, for instance, can ensure that those orders are ready when the trailers are scheduled

for pick-up (Franco, 2006). Wave planners are also responsible for determining the size of waves and releasing them to the floor. If the size (or length) of the wave is too large, it combines tasks and increases pickers productivity; however, it may not be possible to complete by the scheduled truck departure. On the other hand, if the workload is too small, productivity of the pickers will be low, but more orders will be ready for shipment by their deadline.

Most of the world's leading warehouse management systems (WMS) providers, such as SAP and Oracle, offer automatic wave management systems to alleviate inefficiencies of human-based wave management (SAP Business Solutions, 2012; Oracle Warehouse Management User's Guide, 2012). Although these systems are able to monitor the progress of waves dynamically, they lack the ability to prioritize orders when the wave length is longer than two hours. As a consequence, wave planners usually release small waves manually, which creates underutilized resources and does not guarantee completion time of a wave by its deadline. Therefore, there is a need to develop policies that determine the number and size of the waves, their release times, and their contents.

In practice, many DCs release waves sequentially. That is, a new wave does not begin until all the orders in the current wave are completed. Sequential processing of non-overlapping waves is sometimes referred as *fixed wave* systems (Bozer et al., 1988). When orders are grouped in waves based on different attributes, fixed waves create an easily controlled flow of work. For example, when destinations are sorted by distance from the DC and placed into waves (more distant customers are in the first wave, to allow more time for transportation; nearer customers are assigned to later waves), there is no need for additional sortation. Because fixed waves require pickers (and packers) to wait until all others complete their picks, it creates idleness. To prevent the potential idleness of workers, waves can overlap (called *dynamic waves*). Once the orders are dynamically sent downstream, they have to be sorted out. A sortation system usually requires high investment, but more importantly, it adds additional processing time.

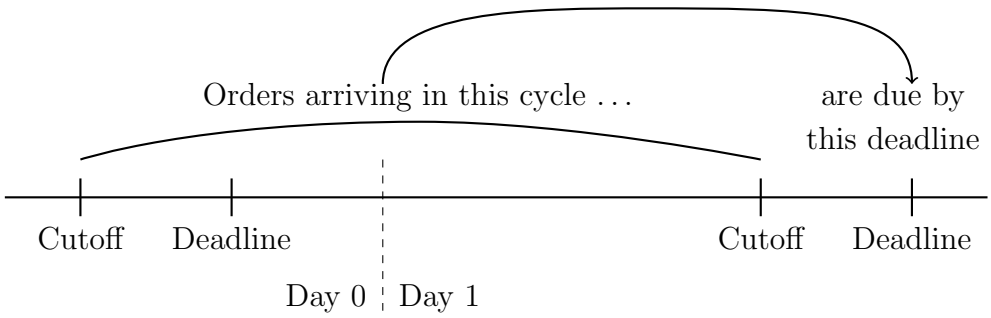


Figure 1.1: Orders arriving between consecutive cutoff times are due on the next deadline.

1.3 Next Scheduled Deadline Metric

Whether a DC employs discrete, batch, or wave picking, fulfillment processes should be designed in such a way that customers receive their orders at the promised time. When balancing internal operations and shipping deadlines, continuous processing versus a cyclical transportation environment should be considered. Doerr and Gue (2011) show that it is more important to make orders ready when the truck is on the shipping dock than it is to minimize flow time. The authors propose a metric called Next Scheduled Deadline (NSD), which measures the fraction of orders arriving during a specific 24-hour period that are processed before a specific truck departure (Figure 1.1).

The 24-hour time window is defined by consecutive "cutoff times." We define *cycle* to specify the time interval between successive cutoff times. To manage customer expectations, DCs publish "cutoff times" and promise to ship orders by a deadline when they are ordered before that time. If a customer places his order after the cutoff time (i.e., after the current cycle), he has no expectation that his order will ship that day (the order will be scheduled to the next day's deadline). If the order is placed before the cutoff time (i.e., before the current cycle), the customer expects his order to be shipped the same day. The cutoff time clarifies expectations and provides a measurable, customer-focused goal for DCs. When the cutoff time is published, the fulfillment system's objective is to ship all orders that arrive between two consecutive cutoff times by the deadline (which is usually before a specific truck departure). While an early cutoff time implies loss in revenues, a late cutoff time

means too many disappointed customers. Consequently, the cutoff times should be set as late as possible, while at the same time, minimizing compensations due to late deliveries. Table 1.2 shows the cutoff times and the fastest delivery promises by the ten largest internet retailers (Internet Retailer, 2013).

Table 1.2: The top ten largest U.S. internet retailers' shipping policies (Source: `http://www.internetretailer.com/`).

Companies	Shipping frequency	Cutoff time	Compensation
1. Amazon.com Inc.	Overnight	4:30 p.m. EST - Prime*	Refund
		6:45 p.m. EST - Non-prime	Refund
2. Staples Inc.	Overnight	5 p.m. EST	N/A
3. Apple Inc.	Overnight	3 p.m. local	N/A
4. Walmart.com	Next business day	N/A	N/A
5. Dell Inc.	Next business day	5 p.m. local	N/A
6. Office Depot Inc.	Same day	10 a.m. local	N/A
7. QVC (Liberty Corp.)	Overnight	12 p.m. EST	N/A
8. Sears Holding Corp.	Overnight	7 p.m. EST	N/A
9. Netflix Inc.	Next business day	N/A	N/A
10. CDW Corp.	Next business morning	5 p.m. EST	N/A

* To increase customer loyalty, Amazon offers two-day free shipping to its prime customers and prioritize their orders.

In addition to top ten internet retailers, many other companies publish their cutoff times to attract customers and gain a competitive advantage. Amazon has taken its service one step further by offering same day delivery with full refunds for late deliveries. In its recently introduced local express delivery policy, Amazon states "When you order using Local Express Delivery before certain cutoff times, your items will be delivered the same day. If the order is placed after the deadline, your order will be delivered the next business day. For same day delivery, you can usually order as late as the times below" (Amazon.com, 2013). Because NSD counts the number of orders that arrive between two consecutive cutoff times and how many of them are actually shipped, a 100% service performance can be achieved with an optimal wave release policy which maximizes the NSD and properly determines the cutoff times.

1.4 Literature Review

The relevant literature can be classified under three main categories: (1) Research which focuses on gaining competitive advantage by increasing customers' experiences, (2) Studies addressing the design and control of order fulfillment systems, and (3) Studies relevant to fluid approximations to queueing networks.

The first group of studies addresses time-based competition of fulfillment systems in which the quality of service (Shang and Liu, 2011) is driven by how timely and accurately their orders are fulfilled. These works, in general, address delivery time quoting, pricing, and capacity decisions with the goal of maximizing expected profit. Most of the studies in this group focus on a uniform delivery time—delivery within a certain time window[†], for all customers. Because demand is driven by the lead time and the price, those studies focus on finding profit maximizing strategies in time competitive environments. The fulfillment system (generally a make-to-order company) selects a uniform delivery guarantee. When the published delivery lead time is short, more customers are attracted which may result in late delivery penalties. To reduce the risk of late completions, companies may publish longer quotes which may cause customers to seek other competitors.

Chatterjee et al. (2002) present a two stage decision model to maximize the profit for a company that quotes uniform delivery dates with incomplete information on real processing time (which makes it hard to estimate future capacity levels). In the first decision step, the capacity is not known and the delivery time is quoted by the company (e.g. the marketing department) which affects the operating characteristics (i.e. the delays in the system) in the second stage. The model considers an environment in which processing times and arrival rates are stochastic and consequently the decisions should reflect the unpredictability of new order arrivals and capacity. So and Song (1998) discuss the relation between delivery time and capacity expansion decisions. The authors developed a mathematical model to determine the optimal delivery time quote under fixed and expandable capacities.

[†]Examples include Pizza Hut's 30 minutes delivery guarantee and FedEx's 10:30 a.m. delivery guarantee

Rao et al. (2005) developed a model for integrating demand and production planning, which determines an optimal planning cycle and a corresponding guaranteed maximum production lead time. Similar to the previous studies, the model optimizes the expected profit by quoting a uniform guaranteed delivery time to all customers, however with a particular focus on updating the production schedule in every fixed point in time. Different than above studies, Duenyas and Hopp (1995) present a non-uniform delivery quote model in which the only controllable variable is accepting or rejecting a customer order. The authors address both finite and infinite capacitated single server queueing systems and demonstrate a control policy with optimal sequencing rules.

The second group of studies focuses on fulfillment operations which mainly include picking, packing, and shipping processes. de Koster et al. (2007) gives a survey of the design and control of order picking systems. In practice, manual picker-to-part systems include both discrete picking, in which orders are assembled one-by-one, and batch picking, in which multiple orders are picked together. They define wave picking as a manual picker-to-part method in which orders for a common destination are released simultaneously for picking. When synchronous picking is performed within small regions of the DC (called zone picking), release for zones is usually controlled with waves. Items in the orders are then consolidated in an automated sortation area (Johnson and Meller, 2002).

Speaker (1975), Huffman (1988) and Frazelle and Apple (1994) define wave picking as a special case of batch-zone picking, where pickers pick very large batches based not on the number of items, but rather on a length of time. A comparison of different order picking policies, including wave picking, is given by Petersen (2000).

Franco (2006) distinguishes between batch and wave picking and discusses drawbacks of wave picking from an industrial perspective. When the DC releases waves, it may be faced with the problem of sorting. To alleviate the sorting problem, DCs can use pick list generation algorithms (Owyong and Yih, 2006).

Some recent studies focus on an alternative policy called waveless picking (Bradley, 2007), in which orders are sent directly to picking upon arrival. Although applications differ in details (Perry, 2007; Morris, 2008), the objective is to decrease the non-productive walking of order pickers. Gilmore (2006) discusses and compares wave and waveless policies. Gallien and Weber (2010) develop a mathematical model to control waveless operations in order to maximize throughput.

As discussed in Chapter 2, our modeling approach is based on fluid approximation of an order fulfillment system in which orders are released into waves. (Fluid models take the advantage of heavy-traffic conditions in queueing models where it is reasonable to replace discontinuities with continuos functions.) Exact queueing models are not appropriate for a number of reasons: (1) DC environment is too complex to analyze with exact models. Because wave policies imply queues and delays during or after the release times (i.e. rush hour), exact queueing analysis of such cases is difficult even for the simplest assumptions. (2) We are interested in systems with stationary and non-stationary arrivals. Arbitrary, non-stationary processes are typically beyond the reach of exact queueing models (Gupta et al., 2006). (3) The system dynamics and the performance measure we focus on cannot be analyzed by pure queueing models.

The use of fluid approximations to queueing models has been investigated by many researchers. The majority of studies have analyzed the behavior of queueing systems in heavy-traffic conditions. We present only studies relevant to our work.

The first model was developed by Newell (1973) to approximate the stochastic behavior of n-server service systems with a large n. Borovkov (1964, 1965) discussed limit theorems for mass service as well as for large values of waiting times. Ridley et al. (2004) implemented a fluid approximation to a priority call center in which the system receives time varying arrivals. The authors investigate the effectiveness of the approximation by comparing it with discrete event simulation.

Liu and Whitt (2010) introduced a deterministic fluid model that serves as an approximation for the $G_t/GI/s_t+GI$ many-server queueing model. The system has a single class of time varying arrivals, general service time distribution (in parallel), time dependent number of servers (s_t) and abandonment time distribution. The fluid model is intended to serve as an approximation for the queueing model when both the number of servers and the arrival rate are large, and then the system experiences occasional periods of significant overloading. Such approximations are usually justified by many-server heavy-traffic (MSHT) limits in which arrival rate and number of servers are increased with the same scaling factor (see Pang and Whitt, 2008 for details).

Fluid models allow us to approximate the dynamics of a queueing system especially when there is a fluctuating (non-stationary) arrival stream. A very detailed study on fluctuating load was given by Gupta et al. (2006), in which system performance was studied under high loads. The authors addressed the performance experienced by customers that arrived to the system when it is in high or low load periods.

Perry and Whitt (2011) analyzed multi-class and pools in a series of papers. The authors followed the same scaling methodology — using the standard many-server heavy-traffic scaling; such that both arrival rate and the number of servers are scaled up by a factor of n.

Ward and Bambos (2003) identified the stability of queueing networks with deadlines. Each arriving job has a deadline, and a single server processes stationary arrivals under first-come-first served discipline. The job abandons the system if it is not completed by its deadline. The authors established stability conditions for this system.

More studies can be found in Dieker (2006) for fluid approximations and extremes in queueing models, Kella and Whitt (1996) for structural properties of storage networks, and Kella and Whitt (1998) and Whitt and Liu (2011) for very general queueing network approximations with fluid models.

This research contributes to the deadline oriented order fulfillment literature in two ways: (1) Our research has an emphasis on organizing internal operations to maximize a *service* objective. With the exception of Doerr and Gue (2011), previous research has focused on traditional measures such as throughput, flow time, or work-in-process inventory. (2) We know of no scientific studies that have investigated systematic wave planning in the presence of deadline-oriented operations. Although wave picking is common in practice, there is still a gap between practice and academic research. Most of the previous academic studies in order picking have focused on efficient picking policies.

1.5 Problem Statement

Our research considers ways in which service performance is maximized by releasing the flow of work optimally to follow on processes. To improve service performance, we propose to explore wave release policies in a DC that operates against a daily deadline. We believe this research is the first comprehensive study of wave planning in a distribution environment, and certainly the first that addresses deadline-oriented operations. We begin with a distribution environment in which the cutoff time is common for all customers and ask,

Problem 1. *What is the optimal timing and number of waves in a DC that operates against a single daily deadline?*

Consider a DC that operates against a daily deadline (e.g. an internet retailer that ships all orders overnight). An important operational decision for this system is when to release orders to pickers. How should managers establish these times? Should they be equally distributed through the day, or does another pattern provide better service? Does the timing really matter from a customer service perspective? Our purpose is to answer these and related questions, all with the goal of maximizing service against a daily deadline.

In our first research question, we address fulfillment systems in which both the arrival process and the server capacity are known. When one of these quantities (or both) is

uncertain, the utilization of the server becomes a random variable. What should be the release times if the server's utilization is unknown? What would be the expected NSD? What is the risk of planning waves against low (or high) utilization? That is,

Problem 2. *How should the timing and number of waves be determined when utilization of the server is uncertain?*

In the first two research questions, we address systems with a single order class. In a more complex fulfillment environment, there are multiple order classes, corresponding to customer groups receiving service at different frequencies. This can be interpreted in our context as orders having different next scheduled deadlines, but the order fulfillment system may choose to work any orders it likes in a wave.

Problem 3. *What should be the timing and the content of waves to maximize service performance across multiple classes of orders?*

This problem is considerably more complex because we must address not only the timing and the number of waves, but also the contents of waves. When a wave is released, it will include orders for the most imminent departure, but may also include orders for future deadlines. The advantage of doing so is obvious, but there is also a disadvantage. An order being worked early consumes capacity that might otherwise be used for an upcoming more urgent order. How to release orders into waves and when to release those waves must be considered for an optimal solution to this problem.

1.6 Organization of the Dissertation

The remainder of this dissertation is organized as follows. In Chapter 2, we introduce optimal wave release strategies for order fulfillment systems in which there is a single class of orders. We present the optimal release times for single and multiple wave release systems, and show how 100% service can be achieved by offering optimal cutoff times. We determine the optimal number of waves when there is an associated fixed time component to waves.

We also present the underlying assumptions behind our models and validation of the fluid model via a discrete event simulation model in this chapter. The models in this chapter constitute a foundation for analyzing more complex systems. In Chapter 3, we discuss how to set release times when the workload is uncertain. We show how to adjust single and multiple waves for a given density function of utilization. We also present a procedure to adjust the wave release times and the cutoff time when the density function is only known empirically. In Chapter 4, we introduce systems of multiple order classes in which there are different deadlines every day. We first address a basic system with multiple order classes and a single wave.We extend our discussion to multiple class, multiple wave systems and present our solution approach. We illustrate the use of the models with numerical examples and discuss how to implement those models in real applications. We offer conclusions and directions for future work in Chapter 5.

Chapter 2

Optimal Release Strategies for Order Fulfillment Systems with Deadlines

2.1 Introduction

Consider a DC that receives orders 24 hours a day, 7 days per week, serving a large enough area to require multiple days of transport to distant customers. Each customer receives service at the same frequency. We assume there is a single deadline such that each arriving order is assigned to a constant next scheduled deadline (NSD) time within a day. We assess the service performance of the DC with NSD.

The DC has a fixed capacity μ which depends on the workforce and which is sufficient to process all arriving orders. Arriving orders accumulate in a buffer until the next wave is released, at which time the quantity of orders in that wave decreases at rate μ until the wave is complete. Upon completion of a wave, the server may go idle, or another wave can be released. The release of a wave before completion of the current wave (overlapping) is disallowed. While the server is working a wave, orders in the next wave accumulate, and the cycle continues. Figure 2.1 illustrates the inventory graphs for this system when there are three waves.

Without loss of generality, we assume a day starts (and ends) at a deadline. That is, the start (and end) time of a day is equal to 0 (and 1). For now, we assume the cutoff times are equal to deadlines.

The workload in a wave is determined by the time since the most recent wave was released. For example, if orders arrive at constant rate of λ and there are three waves at times 0.3, 0.6, and 0.8 in a day of length 1, the workloads for each wave are $(1-0.8)\lambda+(0.3-0)\lambda = 0.5\lambda$, $(0.6-0.3)\lambda = 0.3\lambda$, and $(0.8-0.6)\lambda = 0.2\lambda$. The first wave consists of orders arriving between the last wave from yesterday until midnight and orders arriving from midnight until

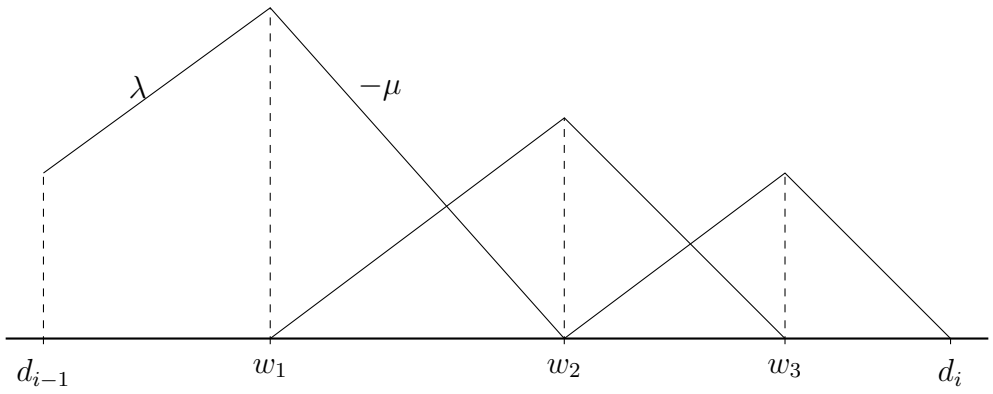


Figure 2.1: A single class, three wave system. d_i indicates the deadline on day i; w_j is the release time for wave j.

the release of the first wave today. The objective is to determine wave release times that maximize the fraction of orders completed by the deadline.

2.2 Single Wave Systems

Consider a single wave system with wave release time w_1. Day 0 is effectively a "warm up day." Denote the release time of the wave by w_1 and constant arrival rate by λ orders per day. Then, $w_1\lambda$ orders arrive before the release on Day 0. The server finishes processing in $w_1\lambda/\mu$ time. Waves on all following days contain λ orders, so the server works $\rho = \lambda/\mu$ of the time.

NSD in a single wave system is a function of release time w_1, arrival rate λ, and server capacity μ. By definition,

$$\text{NSD} = \frac{\text{\# orders worked that arrived in the current cycle and completed before } d_i}{\text{\# orders that arrived in the current cycle}}.$$

Because cycle defines unit length of day between successive cutoff times, the denominator is λ. The numerator is more complicated. Each wave consists of orders that arrived the previous day (but after the wave release, and therefore did not make the deadline) and orders that arrived during the current day. (Recall that we assume cutoff times equal to deadlines). Only the latter count toward NSD, but we assume the former must be worked

before the current day's work (i.e., we assume first-come, first-served discipline). Therefore, the number of orders released today has two components: $(1 - w_1)\lambda$ orders from yesterday and $w_1\lambda$ orders from today.

If the server finishes the wave before the deadline, then

$$\text{NSD} = \frac{w_1\lambda}{\lambda} = w_1. \tag{2.1}$$

If the server finishes the wave after the deadline, only $(1 - w_1)\mu$ orders are processed before the deadline, of which $(1 - w_1)\lambda$ do not count toward NSD because they arrived the previous day. In this case,

$$\text{NSD} = \frac{(1 - w_1)\mu - (1 - w_1)\lambda}{\lambda} = \frac{(1 - w_1)(\mu - \lambda)}{\lambda} = (1 - w_1)\left(\frac{1}{\rho} - 1\right), \tag{2.2}$$

where $\rho = \lambda/\mu$.

The server finishes exactly at the deadline when $w_1 = 1 - \rho$; therefore,

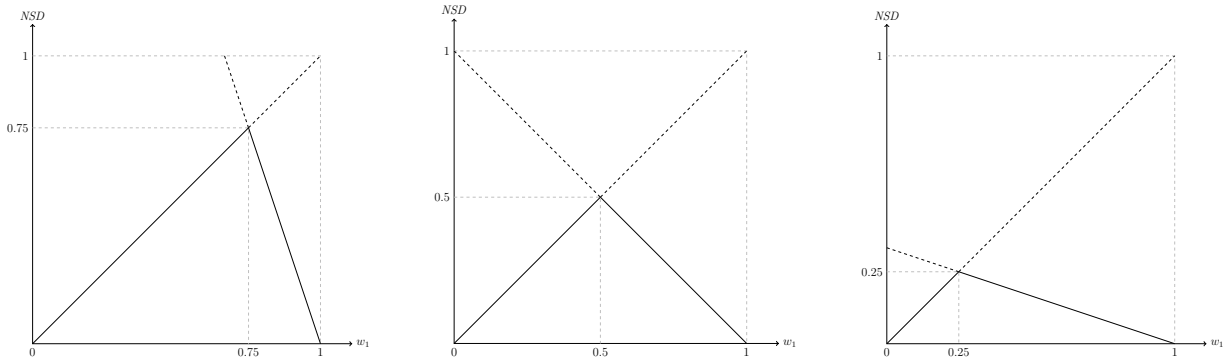
$$\text{NSD} = \begin{cases} w_1, & \rho \leq 1 - w_1 \\ (1 - w_1)(1/\rho - 1), & \rho \geq 1 - w_1. \end{cases} \tag{2.3}$$

and the maximum possible NSD occurs when $w_1 = 1 - \rho$, for any value of ρ.

Figure 2.2 illustrates how NSD changes with the release time w_1 for three values of utilization. We have just shown that

Proposition 2.1. *For a system with a single wave, the optimal release time* $w_1^* = NSD^* = 1 - \rho$.

The proposition confirms two points of intuition. First, the server should begin work as late as possible in order to allow as many orders as possible to make it into the wave. Second, the wave should finish exactly at the deadline.



(a) $\rho = 0.75$. (b) $\rho = 0.5$. (c) $\rho = 0.25$.

Figure 2.2: Different realizations of optimal w_1.

Recall that NSD is defined as the fraction of orders arriving between two consecutive cutoff times that finish by the next deadline. As Doerr and Gue (2011) note, NSD is merely an accounting measure that can be manipulated by shifting the cutoff time; that is, it can be increased by making the cutoff time earlier, or decreased by making it later. In neither case do customers receive their orders any sooner or later. Nevertheless, the cutoff time can serve at least two purposes; if it is published, the cutoff time establishes expectations for customers (e.g., Amazon's overnight delivery guarantee); if not, it can be used as an internal metric to motivate workers (Doerr and Gue, 2011). For our purposes, an *optimal cutoff time* is the latest possible time for which NSD is 100 percent. Proposition 2.1, then, suggests

Proposition 2.2. *In a deterministic order fulfillment system with a single wave, the optimal cutoff time equals the optimal wave release time* $w_1 = 1 - \rho$.

2.3 Multiple Wave Systems

In practice, distribution centers typically use multiple waves per day—usually 2–6, depending on the workload and number of destinations that must be accommodated. Our model of a multi-wave system is built on the simple insight that the number of orders, or *load*, in a wave is the product of the arrival rate λ and the time since the previous wave

was released. (We require that waves consist of all orders available to be released.) Before developing equations for a multi-wave system, we require two results.

Proposition 2.3. *In an optimal solution, work on the final wave ends at the deadline.*

Proof. Let w_N be the last wave release time in an N-wave system. When the last wave ends at the deadline, $(1 - w_N)\lambda$ orders do not make the deadline and $w_N\lambda$ do. Therefore,

$$\text{NSD} = \frac{w_N\lambda}{\lambda} = w_N.$$

Now assume to the contrary of the proposition that there exists a feasible set of release times such that w_N does not finish at the deadline and the corresponding NSD $\geq w_N$. There are two cases:

Case 1: Assume there is an optimal solution in which the final wave w_N finishes Δt before the deadline ($0 < \Delta t < 1 - w_N$), with NSD $= x$. Now, shift every wave release Δt to a later time. The system is still feasible, and now NSD $= x + \Delta t$, a contradiction.

Case 2: Assume there is an optimal solution in which the final wave w_N finishes after the deadline, such that NSD $\geq w_N$. Because the final wave finishes after the deadline, the number of orders arriving today that make today's deadline is the number worked before the deadline $(1 - w_1)\mu$ minus the number that arrived after the last wave in the previous day $(1 - w_N)\lambda$. Therefore,

$$\text{NSD} = \frac{(1 - w_1)\mu - (1 - w_N)\lambda}{\lambda},$$

which can be rewritten as

$$\text{NSD} = w_N + \left(\frac{1 - w_1}{\rho}\right) - 1. \tag{2.4}$$

Because the server begins working at w_1, the number of orders completed before the deadline is no greater than $(1 - w_1)\mu$, which must be less than total arrivals in a day λ. That

is,

$$\begin{aligned} (1-w_1)\mu &< \lambda \\ \frac{(1-w_1)\mu}{\lambda} &< 1 \\ \frac{1-w_1}{\rho} &< 1. \end{aligned}$$

Substituting into Equation 2.4 implies NSD$< w_N$, a contradiction. ■

Proposition 2.3 suggests that the level of unworked inventory at the end of each day is the same, thus the optimal solution provides a stable workload among days. Should an optimal solution provide a stable work-in-process inventory at the end of each day regardless of the initial conditions? No: when the initial work content is greater than $\mu - w_1(\lambda + \mu)$, the stability condition does not hold (see Appendix A).

In a system with multiple waves, it is possible that idleness exists between waves. However,

Proposition 2.4. *In an optimal solution, the server is idle only between the deadline and the first wave release on the following day.*

Proof. Let w=$\{w_1, w_2, ..., w_N\}$ be a set of feasible wave release times. Assume there exists idle time Δt_i between consecutive waves w_i and w_{i+1}. It suffices to show that all $\Delta t_i = 0$.

During wave i, the server completes $[(w_{i+1} - w_i) - \Delta t_i]\mu$ orders. From Proposition 2.3, we know there is no idle time between the end of the final wave and the deadline, so $(1-w_N)\mu$ orders are worked and contribute to NSD in wave N. NSD is the number of orders worked today that arrived today, divided by the number that arrived today,

$$\begin{aligned} \text{NSD} &= \frac{[(w_2 - w_1) - \Delta t_1 + (w_3 - w_2) - \Delta t_2 + \ldots + (w_N - w_{N-1}) - \Delta t_{N-1} + (1 - w_N)]\mu}{\lambda} \\ &= \frac{(1 - w_1 - \sum_{i=1}^{N-1} \Delta t_i)\mu}{\lambda}, \end{aligned}$$

which is highest when $\Delta t_i = 0$ for all i, and there is no idle time after the first wave release. ■

We are now ready to develop equations for optimal wave releases. Let $0 \leq w_i^m \leq 1$ be the time of the i-th wave on day m. Quantity $\lambda(1-w_N^{m-1})$ orders arrive between the deadline and midnight on day $m-1$, and λw_1^m orders arrive on day m before the first wave releases. Therefore, the first wave on day m contains $\lambda(1 - w_N^{m-1} + w_1^m)$ orders. Because there is no idle time between waves (Proposition 2.4), release times must satisfy the following system of recursive equations.

$$\begin{aligned} \lambda(1 - w_N^{m-1} + w_1^m) &= \mu(w_2^m - w_1^m), \\ &\vdots \\ \lambda(w_n^m - w_{n-1}^m) &= \mu(w_{n+1}^m - w_n^m), \\ &\vdots \\ \lambda(w_N^m - w_{N-1}^m) &= \mu(1 - w_N^m). \end{aligned} \tag{2.5}$$

We use the `RSolve` function in MATHEMATICA to solve for the optimal release times for day m. In a steady state ($\lim_{m\to\infty} w_j^m$) for an N wave system,

$$w_j = \begin{cases} \dfrac{j-1}{N}, & \text{for } \lambda = \mu \\ 1 - \dfrac{\rho^j - \rho^{N+1}}{1-\rho^N}, & \text{for } \lambda < \mu. \end{cases} \tag{2.6}$$

Figures 2.3–2.5 show the optimal wave release times for different levels of utilization. Each horizontal line corresponds to a different system, with the indicated number of waves. Dots on the line correspond to optimal release times. Because there is no idle time between waves (Proposition 2.4), the time of the first wave w_1 does not change, but later wave times adjust as the number of waves increases. In fact, Equations 2.5 suggest a relationship between

Figure 2.3: Optimal wave release times: $\rho = 0.5$.

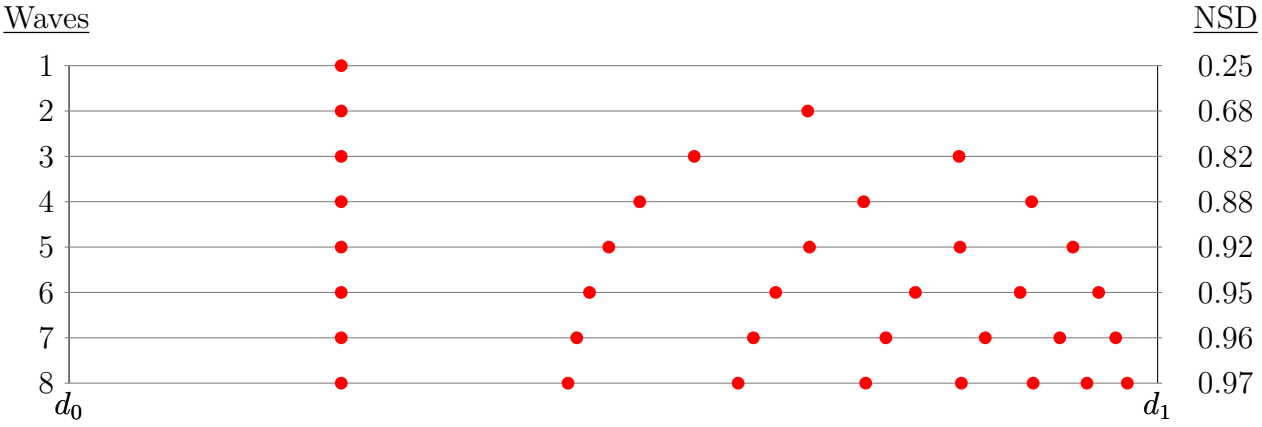


Figure 2.4: Optimal wave release times: $\rho = 0.75$.

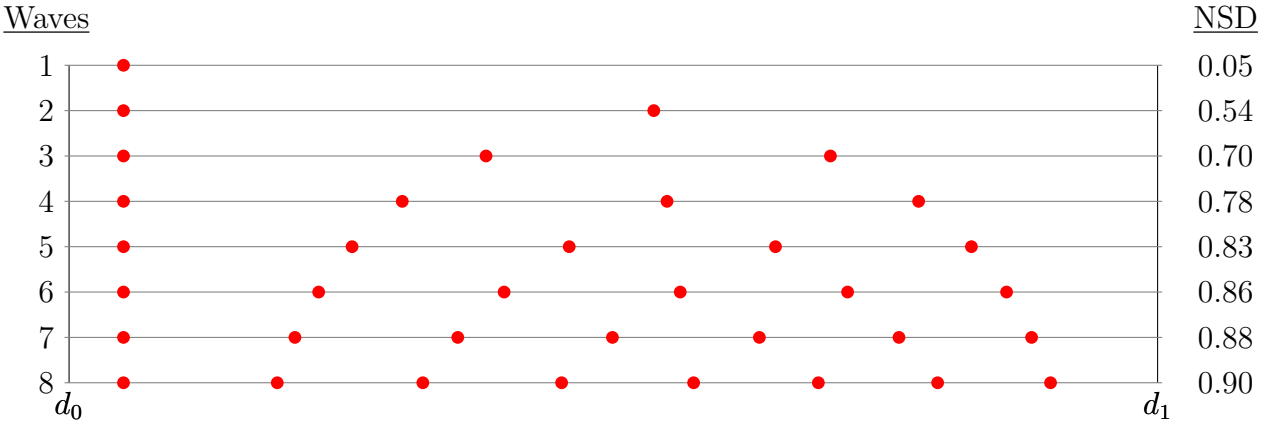


Figure 2.5: Optimal wave release times: $\rho = 0.95$.

consecutive release time intervals:

$$\frac{w_{n+1} - w_n}{w_n - w_{n-1}} = \rho,$$

which means that successively earlier wave lengths are $1/\rho$ larger. This can be seen in Figures 2.3 and 2.4 especially.

Recall that only orders arriving after the final wave will not make the deadline, and therefore NSD $= w_N$. In the expression above, substituting N for j and simplifying gives the optimal NSD for a system with N waves and utilization ρ,

$$\text{NSD}^* = w_N = 1 - \frac{\rho^N(1-\rho)}{1-\rho^N}. \tag{2.7}$$

As expected, $w_N = 1-\rho$ when $N = 1$, and w_N (and NSD) converges to 1 as the number of waves $N \to \infty$. Figure 2.6 illustrates how NSD varies for systems with one to five waves. As utilization increases, the maximum possible NSD decreases, converging eventually to $(N-1)/N$ (Equation 2.6). The plot on the right shows how many hours before the deadline the cutoff time should be set in order to achieve 100 percent NSD.

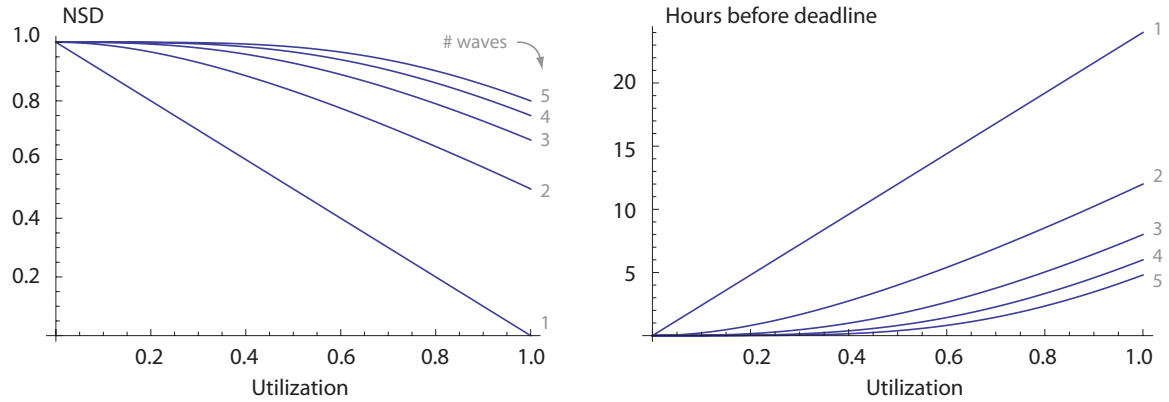


Figure 2.6: Two perspectives on maximum possible NSD for different numbers of waves.

The plot illustrates that an aggressive service promise via a late cutoff time can be kept only by increasing the number of waves, or by adding capacity (decreasing utilization), or

both. The plot also shows that there is little marginal benefit to increasing the number of waves beyond four or five.

2.4 Optimal Number of Waves

We have shown that the last wave release time w_N and therefore NSD converges to 1 as the number of waves N $\to \infty$. Figures 2.3–2.5 show that there is little marginal benefit to increasing the number of waves beyond four or five. From Equation 2.5, for $j > 2$, we have

$$\lambda(w_j - w_{j-1}) = \mu(w_{j+1} - w_j).$$

Rewriting this expression gives us the wave-size (or workload) ratio of consecutive waves:

$$\frac{L_{j+1}}{L_j} = \frac{\lambda}{\mu} = \rho,$$

where L_j denotes the workload of the j-th wave. More generally, the workload ratio of any two waves in a system with multiple waves can be written as:

$$\frac{L_{j+n}}{L_j} = \rho^n.$$

Wave size ratios depend on the assumption that processing rate is constant without regard to the size of the wave. However, for many order picking systems, there is a fixed time component to a wave. For example, in a manual picking system, workers must walk a tour to gather items in the wave. Thus, the processing time is comprised of a fixed time T to walk the tour and a variable processing time Lp that depends on the number of picks in the wave L and the time p to make a single pick. The processing rate can be calculated by dividing the total number of picks with the total time required to traverse the picking area

and pick orders. When the number of orders in a wave is equal to L, the process rate is

$$\mu(L) = \frac{L}{T + Lp}.$$

Modifying Equation 2.5 with $\mu(L)$,

$$\begin{aligned} T + p\lambda(1 - w_N^{m-1} + w_1^m) &= w_2^m - w_1^m, \\ &\vdots \\ T + p\lambda(w_n^m - w_{n-1}^m) &= w_{n+1}^m - w_n^m, \\ &\vdots \\ T + p\lambda(w_N^m - w_{N-1}^m) &= 1 - w_N^m. \end{aligned} \tag{2.8}$$

We use the `RSolve` function in MATHEMATICA to solve for the optimal release times for day m and determine the steady state release times for an N-wave system. The right-hand side of the above equations refers to the allowed time of a wave in order to complete the work-content $p\lambda(w_n^m - w_{n-1}^m)$ including the fixed time T (left-hand side). This observation leads to:

Proposition 2.5. *When processing rate is a function of walk time, pick time and wave size, the optimal number of waves*

$$N^* = \left\lfloor \frac{1 - \lambda p}{T} \right\rfloor.$$

Proof. For a system with N-waves, the first wave is released at $w_1 = 1 - NT - \lambda p$. Because $w_1 \geq 0$, $1 - NT - \lambda p \geq 0$, and therefore the maximum number of waves is

$$N^* = \left\lfloor \frac{1 - \lambda p}{T} \right\rfloor.$$

■

Proposition 2.5 leads us to the plots in Figure 2.7, where we illustrate different levels of picking volumes when $T = 0.05$ (i.e., 1.2 hours of total fixed time including operations in picking, packing, and shipping).

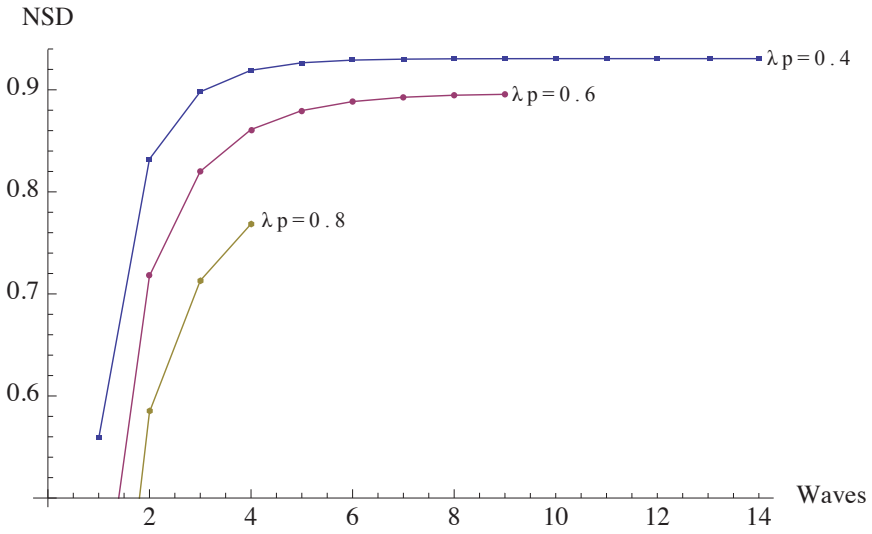


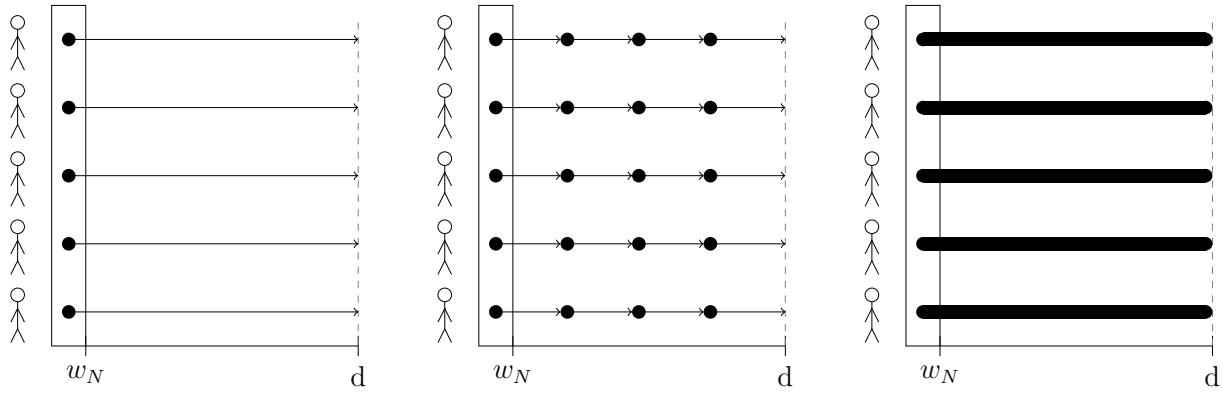
Figure 2.7: Change in NSD with respect to the number of waves when there is a fixed time component associated with each release.

As λp increases, the optimal number of waves decreases. For example, when $\lambda p = 0.4$, the optimal number of waves is $N^* = 14$ and maximum NSD=0.93. (However, there is little marginal return beyond 5 waves). Suppose $\lambda p = 0.8$, then the optimal number of waves is only 4 and the maximum NSD is 0.77. In this case, workers either require more time to pick items or must to pick more items, both resulting in longer processing times and therefore fewer waves.

2.5 Validation of Fluid Approximation

In the previous sections, we assume a fluid model, which assumes a continuous stream of work that flows into the system. In a typical order fulfillment setting, however, orders arrive at discrete times. Therefore, we should determine when the fluid approximation is valid. Fluid models as approximations to queueing systems have an extensive literature (see especially Bernd S. et al., 2011; Dai, 1995; Dai and Jennings, 2003).

Recall that, the accuracy of the fluid approximation improves as the arrival and service rate or the number of servers becomes large (Section 1.4). The typical method of validating a fluid approximation is to increase the number of servers and the corresponding arrival rate while keeping the utilization constant (Pang and Whitt, 2008). The arrival rate of a system with k servers is $\lambda_k = k\mu$. We use a similar but different approach. Because our interest is not the number of servers, but the length of the job (i.e., the service rate), we scale up the service rate parameter μ and keep the number of servers constant, thereby maintaining a constant utilization.



(a) 5 orders each of which takes two hours of processing.

(b) 20 orders each of which takes half hour of processing.

(c) Infinitely many number of orders completed instantaneously.

Figure 2.8: Accuracy of the fluid approximation improves as orders become more indivisible.

Figure 2.8 illustrates the approach. In the leftmost example, the final wave of the day begins two hours before the deadline, in a system with five workers. The final wave contains five orders, each taking two hours to complete. The work content, then, is 10 order-hours. The middle figure shows the same amount of work content spread among 20 orders, each taking 30 minutes to complete. When the wave is released, 15 orders wait in the queue until a server is free. The fluid model is a limiting case in which the work content is divided into infinitely small portions, which are then completed in a continuous stream of output.

If the time to process an order is stochastic, we should expect that some orders will finish after the deadline, and that the expected number of orders missing the deadline will vary based on how the work content is modeled—or said another way, based on how long

the expected processing time is with respect to the length of the final wave. By contrast, the fluid model assumes work is deterministic and that all orders in the final wave make the deadline. Is this a problem?

To answer this question, we simulated a three-stage order fulfillment system, corresponding to the picking, packing, and shipping functions in a DC. To illustrate the point, consider a fulfillment system in which each stage has twenty servers, representing workers in those functions. We consider three levels of utilization: 0.5, 0.75, and 0.95. For a four wave system, the lengths of the final waves are equal to $24 \times (1 - w_4) = 56$ minutes, 2.78 and 5.27 hours for $\rho = 0.5,\ 0.75$, and 0.95, respectively.

The goal is to determine at what ratio of expected processing time to length of final wave the fluid model breaks down, and whether common practice is typically less than that ratio. We begin with a very long expected processing time of 1,024 minutes (17 hours), then half the processing time in successive experiments until expected processing time is just 1 minute. Processing time is divided equally among the three stages, and processing times in each stage are exponentially distributed. We adjust the (exponential) arrival rate to maintain the appropriate utilization. Runs last 30 simulated days, with 3 days of warm-up and 25 replications. For each run, we compare the average simulated NSD with the fluid model approximations, which were 96.7%, 88.6% and 78.1% for $\rho = 0.5, 0.75, 0.95$. We show how to find wave release times and resulting NSD levels in Section 2.3.

Table 2.1 shows the results. As expected, the model performs very poorly when expected processing time is longer than the length of the final wave (ratio greater than 1). When the ratio is less than about 0.3, the fluid model approximation is within one percent of the simulated value. The fluid model always overestimates NSD because it assumes full utilization of the (single) server and deterministic service time. In the simulated (and a real) system, the final wave sometimes finishes early, but NSD is no higher because no more orders get processed before the deadline. However, sometimes the final wave finishes late and orders fail to meet the deadline. NSD in this case is lower than predicted by the fluid

Table 2.1: Validation of fluid model via simulation.

	$\rho = 0.50$			$\rho = 0.75$			$\rho = 0.95$		
Processing time	E[NSD]	error(%)	ratio	E[NSD]	error(%)	ratio	E[NSD]	error(%)	ratio
3,072	1.8	94.8	53.9	2.6	85.9	18.4	3.2	74.9	9.7
1,536	8.4	88.2	26.9	13.2	75.3	9.2	12.8	65.9	4.9
768	29.5	67.1	13.5	37.4	51.2	4.6	37.2	40.9	2.4
384	60.9	35.8	6.7	63.0	25.6	2.3	62.7	15.4	1.2
192	79.4	17.2	3.4	79.4	9.2	1.2	74.1	3.9	0.6
96	88.8	7.9	1.7	86.6	1.9	0.6	78.0	0.1	0.3
48	93.8	2.9	0.8	88.3	0.2	0.3	78.1	0	0.2
24	96.2	0.5	0.4	88.4	0.2	0.1	78.1	0	0.1
12	96.6	0.1	0.2	88.5	0.1	0.1	78.1	0	0
6	96.7	0	0.1	88.5	0.1	0	78.1	0	0
3	96.7	0	0.1	88.5	0	0	78.1	0	0

model. In our experience, a ratio of average processing time to length of final wave of about 1/3 is realistic for most distribution environments.

Chapter 3

Setting Wave Release Times in the Presence of Uncertainty

3.1 Introduction

In the previous chapter, we discussed how to set release times when the workload is known. Most order fulfillment systems, however, experience workload fluctuations from day to day. Some fluctuations are cyclical. Think of weekly patterns, such as peak workload on Sunday as experienced by online retailers (Bates, 2012). Because of workload uncertainty, planning operations based on an average daily workload results in poor performance. Workload can also fluctuate during the day as, for example, customers place orders in the morning or after work.

Consider a DC that has been designed to accommodate a certain workload. There are two potential consequences: (1) If the DC has to process significantly higher volumes, then it will not finish all orders in time, resulting in lower NSD. (2) If the workload is lower, then the system completes orders before the deadline, and labor resources are underutilized. In addition to workload uncertainty, worker absenteeism and other causes of variable capacity lead to uncertainty in utilization.

We have shown that wave release times (and the number of waves) can have a significant impact on NSD for a given utilization level. But how should the release times be set when utilization is uncertain? What is the risk of releasing waves too early or too late?

3.2 The Case of a Single Wave

We start by describing how to adjust a single wave. For a fixed release time w_1, we showed that (Equation 2.3),

$$\text{NSD} = \begin{cases} w_1, & \rho \leq 1 - w_1 \\ (1 - w_1)(1/\rho - 1), & \rho \geq 1 - w_1. \end{cases}$$

Figure 3.1 shows NSD as a function of utilization for three specific values of w_1.

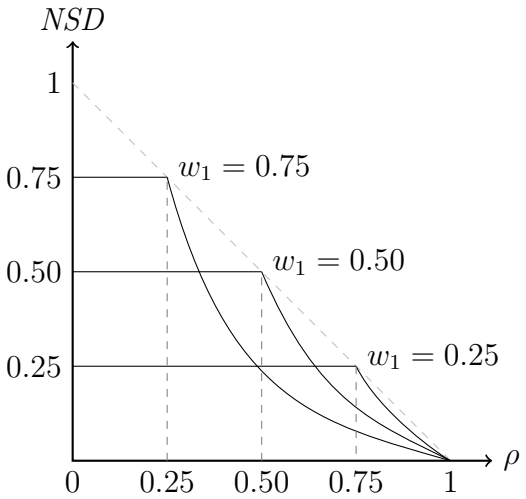


Figure 3.1: Service performance versus utilization ρ, for three wave release times.

When $w_1 = 0.5$, for example, NSD is at its maximum 50% for all values of $\rho \leq 0.5$. In such cases, the server would finish at or before the deadline, but there is no increase in performance because all orders arriving after the release still miss the deadline. For $\rho > 0.5$, performance on NSD drops off sharply because many orders in the wave finish after the deadline. The curves illustrate that, in the presence of uncertainty in workload or capacity (i.e., uncertain ρ), setting the release time too late (high w_1) risks a significantly lower performance on NSD if utilization is actually high, but setting it too early (low w_1) limits the highest possible NSD if utilization is actually low.

Consider two systems—one with a fixed utilization 0.5 and one with utilization distributed Uniform[0,1]. The optimal release time for the first system can be calculated from

Proposition 2.1: $w_1 = 1 - \rho = 0.5$. To compute the maximum expected NSD in the second case, we use Equation 2.3 with density function $\{f(\rho) = 1; 0 \leq \rho \leq 1\}$.

$$\begin{aligned} E[\text{NSD}] &= \int_0^1 \text{NSD}(w_1, \rho) f(\rho)\, d\rho \\ &= \int_{\rho=0}^{1-w_1} w_1 f(\rho)\, d\rho + \int_{\rho=1-w_1}^{1} (1 - w_1) \left(\frac{1}{\rho} - 1\right) f(\rho)\, d\rho \\ &= (w_1 - 1) \log_e(1 - w_1). \end{aligned} \tag{3.1}$$

Taking the derivative and setting equal to zero,

$$\frac{d}{dw_1} E[\text{NSD}] = \log_e(1 - w_1) + 1 = 0,$$

which has solution

$$w_1 = 1 - \frac{1}{e} \approx 0.6321,$$

which is later than in the deterministic case. From Equation 2.3, maximum NSD for this release time is approximately 0.6321, but *expected* NSD is $(1 - 1/e - 1) \log_e(1 - 1 - 1/e) = 1/e \approx 0.3679$. Expected NSD for $w_1 = 0.5$ would have been 0.3466. This simple example shows that releasing a single wave later than it otherwise would have been released improves expected NSD.

3.3 Adjusting Multiple Wave Release Times

That uncertainty should lead to a *later* release time seems to violate the intuition that a later release time is "riskier." As we now show, it is *not* true that uncertainty of utilization leads to a later optimal release time for the more practical case of multiple wave releases per day.

To see why, consider the optimal wave release times for many values of ρ (call them $\hat{\rho}$). We can then compute expected NSD for each set of times, given a density of utilization $f(\rho)$. Because the observed utilization ρ may not equal the planned utilization $\hat{\rho}$, we must

develop expressions for NSD for two possible cases. If $\rho < \hat{\rho}$, the last wave finishes early and there is idleness between consecutive waves. Only orders arriving after the last wave miss the deadline, so NSD $= w_N$.

If $\rho > \hat{\rho}$, the first wave does not finish before the release of the second; the second does not finish before release of the third; and so on, and the last wave does not finish before the deadline. From Equation 2.4,

$$\text{NSD} = w_N + \left(\frac{1 - w_1}{\rho}\right) - 1.$$

From Proposition 2.3, we know $w_1 = 1 - \hat{\rho}$. Therefore, $\hat{\rho} = 1 - w_1$ and

$$\text{NSD} = \begin{cases} w_N, & \rho \leq 1 - w_1 = \hat{\rho} \\ w_N + \left(\frac{1 - w_1}{\rho}\right) - 1, & \rho \geq 1 - w_1 = \hat{\rho}. \end{cases} \tag{3.2}$$

Figure 3.2 illustrates the relationship between planned utilization $\hat{\rho}$ and observed utilization ρ. Planning for a high utilization means an earlier first release time w_1, and therefore a lower maximum NSD; planning for low utilization allows a higher NSD if observed utilization is less than planned, but increases the chance that observed utilization will exceed planned. Which curve provides the highest expected NSD depends on the density of utilization $f(\rho)$. The curves in Figure 3.2 are easy to generate, and for a given $f(\rho)$ we can easily integrate them to compute expected NSD.

To illustrate the procedure, we assume utilization is distributed Uniform[0,1] and use Equation 3.2 to compute NSD for many values of planned utilization $\hat{\rho}$. Each curve in Figure 3.3 is a piecewise-linear representation of expected NSD for 100 values of $\hat{\rho}$. The maximum point for a single wave system is E[NSD] = 0.3679, which equals the optimal value of the single-wave model as expected. Release times for the best four-wave solution are $w_1 = 0.32,\ w_2 = 0.597,\ w_3 = 0.785$, and $w_4 = 0.913$, which produce E[NSD] = 0.855. Had we used the deterministic solution for $\rho = 0.5$, release times would have been $w_1 = 0.5,\ w_2 = 0.767,\ w_3 = 0.9,\ w_4 = 0.967$, with E[NSD] = 0.813. In other words, adjusting

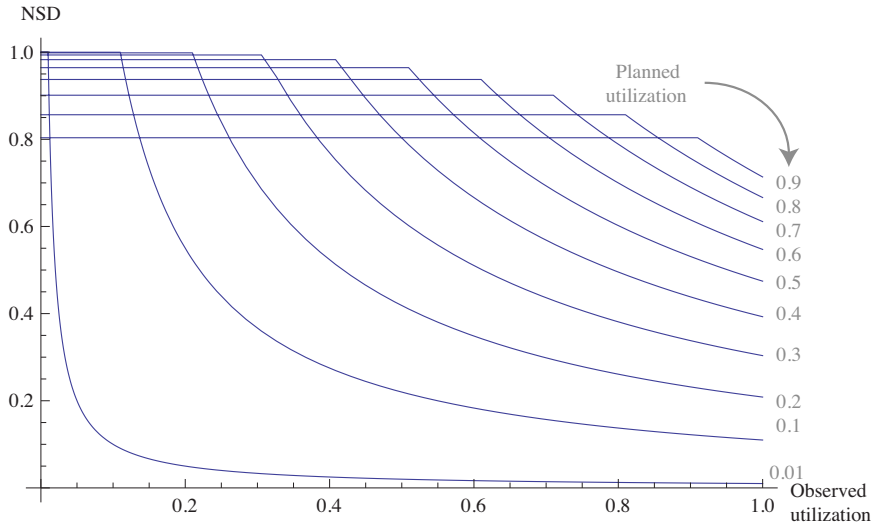


Figure 3.2: Each curve shows how NSD changes for different levels of observed utilization with respect to planned utilization.

release times for the uncertain utilization increases expected NSD by 4.2 percent. Table 3.1

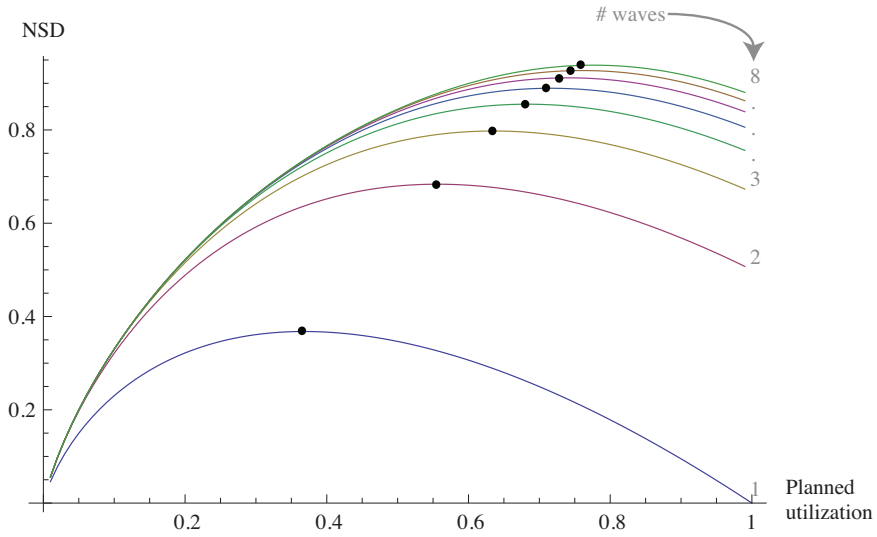


Figure 3.3: Maximum expected NSD achieved when release times are adjusted to earlier times.

shows the percent improvement in expected NSD by adjusting the release times for other numbers of waves.

Table 3.1: Adjusting release times in the presence of workload uncertainty improves expected NSD.

Number of waves	1	2	3	4	5	6	7	8
E[NSD|ρ=0.5] (%)	34.7	67.9	77.5	81.3	83.0	83.9	84.3	84.5
E[NSD] (%)	36.8	68.4	79.8	85.5	88.9	91.2	92.7	93.9
Improvement (%)	2.1	0.4	2.3	4.3	5.9	7.3	8.4	9.4

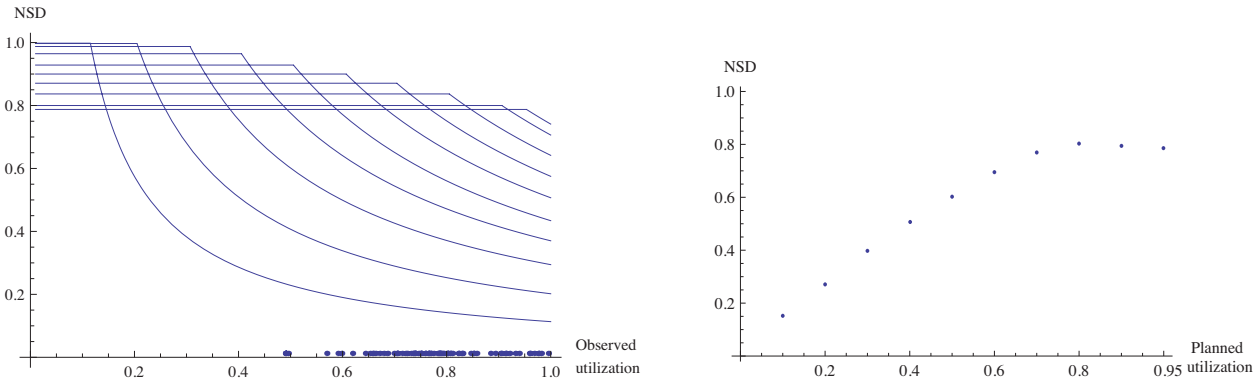


Figure 3.4: Ten different levels of planned utilization for the case study (left) and corresponding average NSD (right).

3.4 An Empirical Procedure to Adjust Release Times

In previous section, we showed how to adjust release times for a known density function of utilization. In practice, of course, the density is unknown, so variability of utilization must be handled in another way. Here we describe an empirical procedure to increase expected performance on NSD, given historical utilization data. (We used 80 day workload and shipment data of an industrial DC to analyze the variability of utilization).

First, we generate performance curves for many levels of planned utilization, as in Figure 3.2. Each curve corresponds to a different set of release times and tells what NSD would be for any value of observed utilization. Next, for each value of planned utilization (and its associated curve) we simply record what NSD could have been for each day in the data set. The planned utilization giving the highest average NSD gives us the best set of release times.

The left plot in Figure 3.4 shows curves corresponding to ten different levels of planned utilization. Data points on the bottom correspond to the observed utilization for each day

in the data set. The right plot shows average NSD for the ten levels of planned utilization. The highest value of NSD is 80.4%, which corresponds to a planned utilization of 80% and release times $w_1 = 0.200,\ w_2 = 0.424,\ w_3 = 0.690$, and $w_4 = 0.841$.

As expected, the best value of planned utilization when accounting for variability of utilization is greater than the average utilization in the historical data. Higher planned utilization means earlier wave releases and a reduced chance that the final wave will not finish before the deadline.

3.5 Setting the Cutoff Time

To this point we have assumed that the cutoff time equals the deadline, which is hardly realistic if the cutoff time is to be published to customers. How should a cutoff time be established such that orders make the deadline "almost always?"

Note that all orders arriving after the final wave will miss the deadline, so the cutoff time should be no later than w_N, but setting it earlier than w_N means the final wave contains orders not due at the upcoming deadline. We conclude, then, that the cutoff time should equal w_N. Why not release the final wave earlier to get a headstart on the final orders arriving for the next deadline?

Define the cutoff time *setback* to be the time between the cutoff time and the deadline. The left plot in Figure 3.5 shows daily performance on NSD for the data set when release times are set according to $\hat{\rho} = 0.8$ and the cutoff time is equal to w_N, for a setback of 3.8 hours.

All customer requests were met on 53 of the 80 days, or 66.3% of the time, which is analogous to Type 1 service in inventory theory. The same policy yields a fill rate (Type 2 service) of 97.24%. The plot on the right shows Type 1 and Type 2 service levels for different levels of $\hat{\rho}$. In each case, the cutoff time is set to the new final release time w_N.

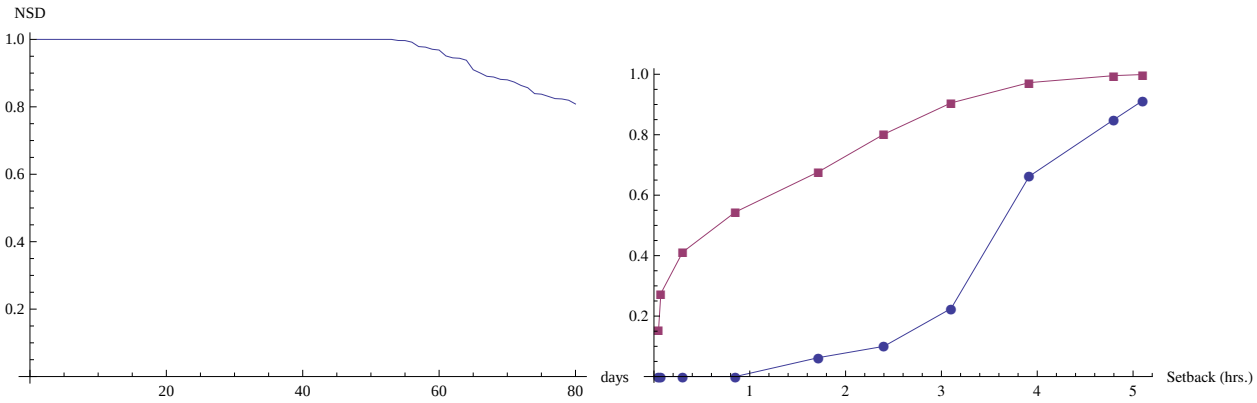


Figure 3.5: Performance results when the cutoff time is adjusted. On the left, NSD for each of the 80 days in the data, sorted from high to low. On the right, Type 1 and Type 2 performance for different levels of cutoff time setback.

3.6 Adjusting The Number of Waves

Proposition 2.5 shows how to determine the optimal number of waves for a given utilization ρ. When ρ is uncertain, N becomes uncertain and determining an optimal policy is more difficult.

Consider a fulfillment system in which planned utilization is less than observed utilization. High utilization implies releasing fewer waves; however, Equation 2.6 suggests more waves to achieve the same level of NSD. As a consequence, there is a risk of releasing more waves and missing some portion of the orders. Our objective is to determine the number of waves for a fulfillment system in which the utilization is uncertain.

We first consider a system with a single wave release. As in Section 2.4, the fixed time component is T. If the server completes the wave before the deadline, then the wave will be completed at $w_1 + \lambda/\mu + T$. Because the wave is completed before the deadline, $w_1\lambda$ orders count toward NSD, and NSD $= w_1$. If the server finishes the wave after the deadline, $(1 - w_1 - T)\mu$ orders are processed, of which $(1 - w_1)\lambda$ do not count toward NSD. In this

case,

$$\begin{aligned} \text{NSD} &= \frac{(1-w_1-T)\mu-(1-w_1)\lambda}{\lambda} \\ &= \frac{(1-w_1)(\mu-\lambda)}{\lambda}-\frac{T}{\rho} \\ &= (1-w_1)\left(\frac{1}{\rho}-1\right)-\frac{T}{\rho}. \end{aligned}$$

Consequently,

$$\text{NSD} = \begin{cases} w_1, & \rho \le 1-w_1-T \\ (1-w_1)\left(\frac{1}{\rho}-1\right)-\frac{T}{\rho}, & \rho \ge 1-w_1-T \end{cases}$$

which is maximized when $w_1^* = 1-\rho-T$. As expected, the optimal release time is now T earlier and as it increases, the wave should be released earlier, resulting in a lower NSD.

When the workload is uncertain, the *expected* NSD can be calculated from

$$E[\text{NSD}] = \int_0^1 \text{NSD}(w_1,\rho,T)f(\rho)\,d\rho. \tag{3.3}$$

To find the optimal release time of a single wave, we take the derivative of the above expression with respect to w_1 and set it equal to zero.

To illustrate the procedure, consider the following example. Assume that the server's utilization is distributed Uniform[0,1]. What is the improvement in expected NSD when the release time of a single wave is adjusted based on the fixed time wave component? Using the above equation,

$$\begin{aligned} E[\text{NSD}] &= \int_0^1 \text{NSD}(w_1,\rho,T)f(\rho)\,d\rho \\ &= \int_{\rho=0}^{1-w_1-T} w_1\,d\rho + \int_{\rho=1-w_1-T}^{1}\left[(1-w_1)\left(\frac{1}{\rho}-1\right)-\frac{T}{\rho}\right]d\rho \\ &= (w_1+T-1)\log\left[\frac{w_1+T-1}{T-1}\right] \end{aligned}$$

and $E[\text{NSD}]$ is maximized when

$$w_1^* = \left(1 - \frac{1}{e}\right)(1 - T).$$

When ρ is uncertain, the adjusted release time is always later than what it would otherwise be: $w_1 \geq 1 - \hat{\rho} - T$. We find the expected NSD by inserting w_1^* into Equation 3.3:

$$E[\text{NSD}|w_1^*] = \frac{1 - T}{e}.$$

The expected NSD is a linear function of T for the optimal value of w_1^*. Figure 3.6 shows that releasing a single wave later than it otherwise would have been released improves expected NSD.

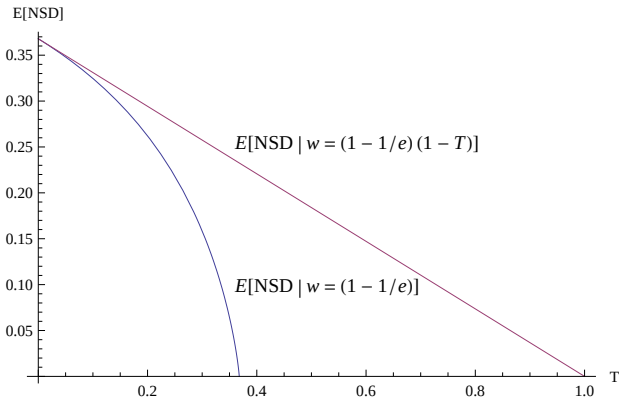


Figure 3.6: Expected NSD considering only uncertainty (bottom curve) and expected NSD considering both uncertainty and the fixed time component (top curve).

For the above example, although considering the fixed time component does not produce significant improvement (1.21%) when we account for multiple stages (e.g. packing and shipping), it produces more than 3.5%.

Now, consider a multiple wave system. A lower server utilization suggests releasing more waves ($\lim_{\rho \to 0} N^* = 1/T$). A higher utilization, on the other hand, implies fewer waves. What should be the optimal number of waves when the utilization is uncertain?

As before, when waves are completed before the deadline NSD $= w_N$, otherwise the pickers process $\mu(1 - w_1 - NT)$ many orders, of which $\lambda(1 - w_N)$ of them arrive after the last release and thus do not count toward NSD. The resulting NSD is $[\mu(1 - w_1 - NT) - \lambda(1 - w_N)] = w_N + (1 - w_1 - NT)/\rho - 1$. Consequently,

$$\text{NSD} = \begin{cases} w_N, & \rho \leq 1 - w_1 - NT = \hat{\rho} \\ w_N + \dfrac{1 - w_1 - NT}{\rho} - 1, & \rho \geq 1 - w_1 - NT = \hat{\rho}. \end{cases} \tag{3.4}$$

For given values of w_1, w_N, we can calculate NSD; however, for different values of ρ, the number of waves N changes. To determine the optimal N that maximizes expected NSD, we follow a similar but slightly different procedure than the one in Section 3.3. The steps of the procedure are

(1) For a given T and $f(\rho)$, determine $\hat{\rho} = E[\rho]$. Determine $N = \lfloor (1 - \hat{\rho})/T \rfloor$.

(2) Use the recursive system given in Equation 2.8 to determine w_1, w_N and NSD_N.

(3) Insert w_1, w_{N^*} into Equation 3.4 and use it in the following steps.

(4) Calculate the loss $\ell(\rho)$ associated with releasing N waves.

(4.1) Calculate down-side loss $\ell_d(\rho)$:

For $\rho \leq \hat{\rho}$ evaluate Equation 2.8 and determine resulting NSD_d,

$\ell_d(\rho) = \text{NSD}_d - \text{NSD}_N$.

(4.2) Calculate up-side loss $\ell_u(\rho)$:

For $\rho \geq \hat{\rho}$, repeat (3) and determine resulting NSD_u,

$\ell_u(\rho) = \text{NSD}_N - \text{NSD}_u$.

(4.3) When the actual utilization is ρ, total loss $\ell(\rho) = \ell_d(\rho) + \ell_u(\rho)$.

(5) Determine expected NSD and expected loss of using N waves.

(5.1) Expected NSD:

$$E[\text{NSD}|N] = \int_{\rho=0}^{1} \text{NSD}(w_1, w_N, \rho) f(\rho) d\rho.$$

(5.2) Expected loss of using N waves:

$$E[\ell|N^*] = \int_{\rho=0}^{1} \ell(\rho)d\rho.$$

(6) Calculate expected NSD after expected loss $E[\text{NSD}|N] - E[\ell|N]$.

(7) While $N > 0$, decrement N by one and go to step 2.

(8) Select the N^* that produces the maximum $E[\text{NSD}]$.

We illustrate the procedure with the following example. Assume that the utilization is distributed Uniform[0,1] and $T = 0.1$. Proposition 2.5 suggests $N = \lfloor(1 - 0.5)/0.1\rfloor = 5$. When $\rho = 0.5$, the first and the last release times are $w_1 = 0,\ w_5 = 0.8$ resulting in NSD=0.8.

Suppose that observed utilization is lower than planned, to say $\rho = 0.25$. Although the maximum number of waves should be $N^* = \lfloor(1-0.25)/0.1\rfloor = 7$, because the number of waves were decided based on the planned utilization, the number of waves would still be five. A lower observed utilization, however, suggests different release times: $w_1 = 0.25,\ w_5 = 0.8664$ (NSD=86.6%). Consequently, there will be a 6.64% loss in NSD due to releasing five waves, but not adjusting the release times (downside loss).

Now suppose that observed utilization $\rho = 0.75$. The maximum number of waves should be $N^* = \lfloor(1 - 0.75)/0.1\rfloor = 2$. The resulting NSD would be 0.536. Because the maximum number of waves is two, releasing five waves must produce a lower NSD. When there are five waves, total fixed time is 50% of the day, resulting in late completion of the waves (because remaining capacity will not be enough to complete the waves before the deadline). Releasing five waves will produce an NSD equal to 0.397. The loss in NSD due to planning for low utilization is 0.536-0.397=0.139 (upside loss).

If the planned utilization is equal to observed utilization ($\hat{\rho} = \rho = 0.5$), then there would be no loss in NSD. For all other values of ρ, there is either downside or upside loss which needs to be considered. We calculate the loss for each value of ρ with increments of 0.1 (Figure 3.7).

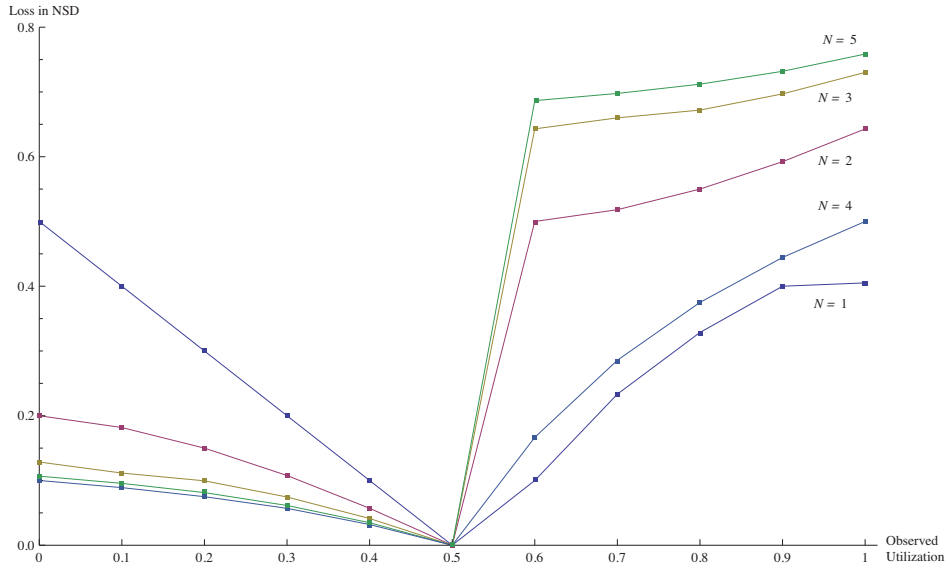


Figure 3.7: Loss associated with using N waves.

In Figure 3.7, for data points where $\rho \leq \hat{\rho}$, there would be a potential increase if release times were adjusted. In this case, releasing more waves decreases the loss (more orders will be completed). When observed utilization is higher than planned, there are too many waves and therefore the picking density is low (due to excessive fixed time in operations). As a result, the last wave (in this example, the fifth wave) will be completed after the deadline and NSD will be lower than a system with fewer waves. Taking the expectation of $\ell(\rho)$, we calculate the expected loss of releasing a certain number of waves. Subtracting each loss from its expected NSD value, we determine the optimal number of waves that maximizes NSD. For our example $\max\{E[\text{NSD}|N = j] - E[\ell|N = j]\} = \max\{0, 0.228, 0.267, 0.279, 0.454\} = 0.454$ when $N^* = 5$. Recall that we have assumed $T = 0.1$ which means 2.4 hours per wave. In a more realistic fulfillment system, the total fixed time in all processes is no longer than two hours. If we assume $T = 2$ hours, then Proposition 2.5 suggests releasing $(1-0.50)/0.083 = 6$ waves. The solution for a six wave system suggests $w_1 = 0$ and $w_6 = 0.837$. In this case, the expected NSD $\text{E}[\text{NSD}|N = 6, T = 2 \text{ hours}] = 0.433$. Following the procedure given above, the expected NSD is maximized when $N^* = 2$. Release times for two wave system are $w_1 = 0$ and $w_2 = 0.502$ which produce $\text{E}[\text{NSD}] = 0.621$ (18.8% higher than a six wave system).

In the above example, we assume a known density function for the server's utilization and demonstrate the use of our procedure to adjust the number of waves. In practice, the density function is not known, therefore, historical utilization distribution should be used to determine the optimal number of waves. The procedure, in this case, will be slightly different than the one we present above (summation is required instead of integration). Nevertheless, we can use the approach given in Section 3.4 to determine the expected NSD and insert it into the Step 5 in the above procedure.

Chapter 4

Wave Release Strategies for Systems with Multiple Order Classes

4.1 Introduction

Our first and second research questions address systems with a single class of orders. That is, the DC ships all customer orders at the same time. In many order fulfillment environments, there are multiple deadlines per day. Different deadlines may be the result of multiple delivery modes. For example, a retailer may have a deadline for FedEx, UPS, and even one or more LTL carriers. Some retailers, on the other hand, may control their own shipment operations. Customers farther from the DC may have earlier deadlines (to allow for timely deliveries); customers located closer to the DC may be assigned to later deadlines. Consider a DC that processes three different order classes with three waves (Figure 4.1).

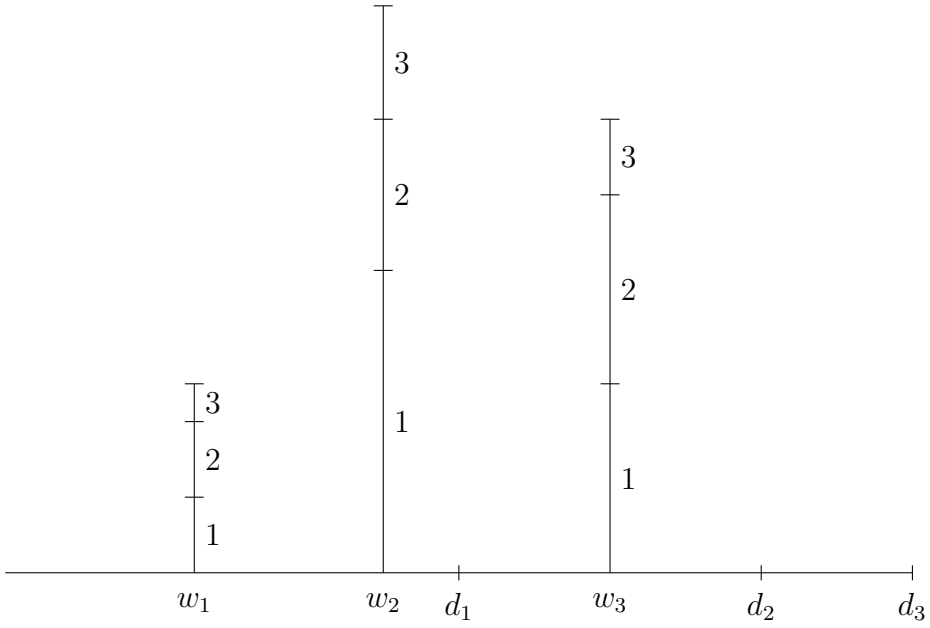


Figure 4.1: A three-class, three wave system.

Each class has a different deadline $d_i, i \in \{1, 2, 3\}$. When a wave is released, it includes the most imminent orders, but may also include orders for future deadlines. (For waves 1 and 2, the most imminent orders are class 1 orders, having a deadline d_1.) Should waves release all classes of waiting orders, or release only the most imminent ones? When should the waves be released, and what should be the content of the waves? In this chapter, our objective is to determine the optimal timing and mixture of waves in multiple order class, multiple wave systems.

Before specifying an initial model, we need to define a "system NSD," which accounts for all order classes. Because there are multiple classes of orders (deadlines), each class has its particular NSD:

$$\text{NSD}_i = \frac{\#\text{ class } i \text{ orders that arrived in the current cycle and completed before } d_i}{\#\text{ class } i \text{ orders that arrived in the current cycle}},$$

$i \in \{1, \ldots, K\}$. Because the time between successive deadlines is one day, the denominator is λ_i. Therefore, the total workload in a K class system is $L = \sum_{i=1}^{K} \lambda_i$. The numerator depends on multiple factors: the number of order classes K; arrival rates $\lambda_i, i \in \{1, \ldots, K\}$, deadlines $d_i, i \in \{1, \ldots, K\}$ and the number of waves N. When those parameters and variables are defined, the system NSD can be written as a weighted combination of each NSD_i:

$$\text{NSD} = \frac{1}{L}\left(\sum_{i=1}^{K} \lambda_i \text{NSD}_i\right). \tag{4.1}$$

We use Equation 4.1 throughout this chapter to calculate system NSD. (Hereafter, NSD denotes the system NSD.) The server's utilization is equal to the total workload divided by the total processing capacity μ:

$$\rho = \frac{\sum_{i=1}^{K} \lambda_i}{\mu} = \frac{L}{\mu}.$$

For now, we assume that the cutoff time of each order class is equal to its deadline and that completed orders are instantaneously ready for shipment. In a system with K order classes, waves complete some classes before their deadlines and some after their deadlines. Releasing multiple order classes in waves reflects realistic operations. In practice, pickers pick orders from all classes in their picking tours. Consequently, waves include multiple order classes that are processed together.

4.2 Single Wave Systems With Multiple Order Classes

Our approach is similar to the one given in Section 2.2. Because there is a single wave, all orders accumulated between the release times of two consecutive days constitute the total workload. That is, the server processes orders of all classes in a single wave.

The simplest problem is to determine a single wave release time when there are two order classes with deadlines d_1 and d_2. The first deadline d_1 represents the earliest deadline in a day, followed by the second deadline d_2. Without loss of generality, we assume $d_2 = 1$. When should the single wave be released to maximize system NSD?

We denote the completion time of a single wave by f_1. When the wave is released, there are λ_i many class i orders of which $\lambda_i(1 - w_1)$ arrived in the previous day and $\lambda_i w_1$ arrived in the current day. The total number of orders at time w_1 is equal to the sum of orders from all classes: $\sum_{i=1}^{K} \lambda_i = L$. The server starts working at a rate of μ and completes the wave at

$$\begin{aligned} f_1 &= w_1 + \frac{L}{\mu} \\ &= w_1 + \rho. \end{aligned} \tag{4.2}$$

What is the system NSD if the wave is completed after d_1, or d_2? The answer depends on its release time. For a two class, single wave system, there are five possible cases.

Case 1. The wave is released before the first deadline and completes all work before the first deadline.

Figure 4.2 illustrates this case. All orders from both classes are released at w_1. The total number of orders released is equal to $\lambda_1 + \lambda_2$. With a rate μ, the server starts processing

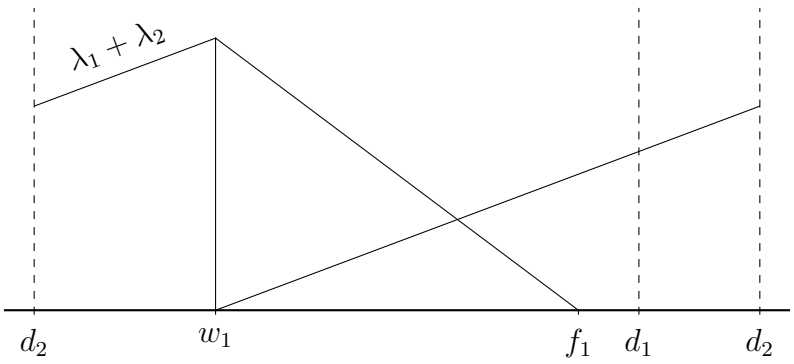


Figure 4.2: Case 1: $w_1 < d_1$ and $f_1 < d_1$.

orders of both classes and completes the wave at $f_1 = w_1 + (\lambda_1 + \lambda_2)/\mu$. Because $f_1 < d_1$, all orders within the wave are completed and only orders that arrived after the release do not count toward NSD_i. The number of orders that arrive after the release is equal to $\lambda_i(d_i - w_1)$. Then,

$$\begin{aligned} \text{NSD}_1 &= 1 - \frac{\lambda_1(d_1 - w_1)}{\lambda_1} \\ &= 1 - (d_1 - w_1). \end{aligned} \tag{4.3}$$

$$\begin{aligned} \text{NSD}_2 &= 1 - \frac{\lambda_2(d_2 - w_1)}{\lambda_2} \\ &= 1 - (d_2 - w_1). \end{aligned} \tag{4.4}$$

From Equation 4.1, the system NSD is

$$\begin{aligned} \text{NSD} &= \frac{\lambda_1}{\lambda_1 + \lambda_2}\text{NSD}_1 + \frac{\lambda_2}{\lambda_1 + \lambda_2}\text{NSD}_2 \\ &= (1 + w_1) - \frac{\lambda_1 d_1 + \lambda_2 d_2}{\lambda_1 + \lambda_2}. \end{aligned} \tag{4.5}$$

Case 2. The wave is released before the first deadline but completed at $d_1 < f_1 < d_2$ (Figure 4.3).

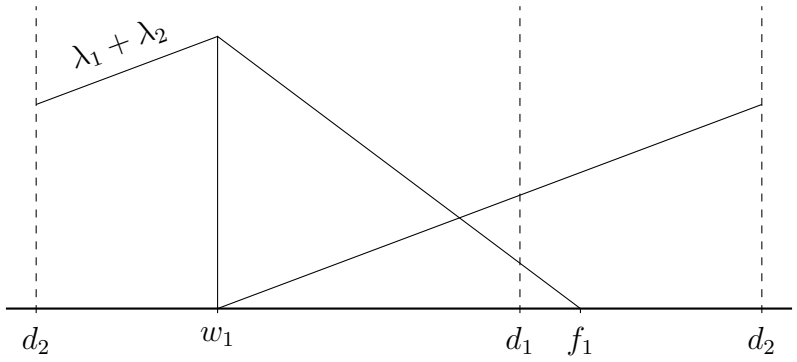


Figure 4.3: Case 2: $w_1 < d_1$ and $d_1 < f_1 < d_2$.

Because $d_1 < f_1$, $\lambda_1/(\lambda_1+\lambda_2)\mu(d_1 - w_1)$ many class 1 orders will be completed by d_1. Again $\lambda_1(d_1 - w_1)$ many of those do not count toward NSD. NSD for class 1 is

$$\begin{aligned} \text{NSD}_1 &= \frac{\frac{\lambda_1}{\lambda_1+\lambda_2}\mu(d_1 - w_1) - \lambda_1(d_1 - w_1)}{\lambda_1} \\ &= \left(\frac{\mu}{\lambda_1+\lambda_2} - 1\right)(d_1 - w_1). \end{aligned} \tag{4.6}$$

Because the server completes the work before the second deadline ($f_1 < d_2$), NSD for class 2 is same as in Equation 4.4. Therefore,

$$\text{NSD} = \frac{\lambda_1}{\lambda_1+\lambda_2}\left(\frac{\mu}{\lambda_1+\lambda_2} - 1\right)(d_1 - w_1) + \frac{\lambda_2}{\lambda_1+\lambda_2}\big[1 - (d_2 - w_1)\big]. \tag{4.7}$$

Case 3. The wave is released before the first deadline and completed after the second deadline (Figure 4.4).

Because class 2 orders are completed after d_2, we simply replace d_1 in Equation 4.6 with d_2 to calculate

$$\text{NSD}_2 = \left(\frac{\mu}{\lambda_1+\lambda_2} - 1\right)(d_2 - w_1). \tag{4.8}$$

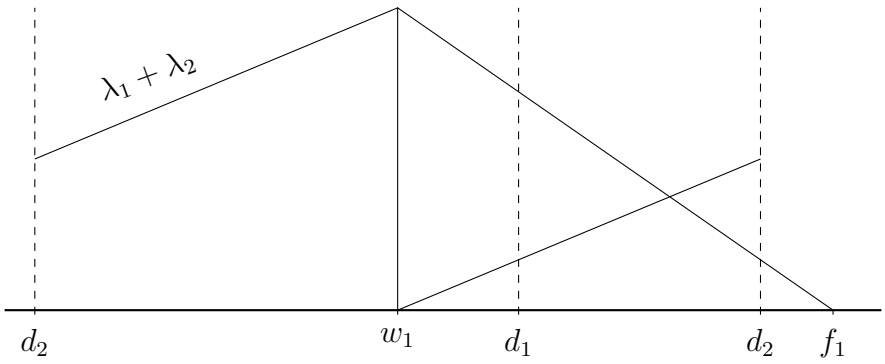


Figure 4.4: Case 3: $w_1 < d_1$ and $f_1 > d_2$.

The system NSD is:

$$\begin{aligned} \text{NSD} &= \frac{\lambda_1}{\lambda_1+\lambda_2}\left(\frac{\mu}{\lambda_1+\lambda_2}-1\right)(d_1-w_1)+\frac{\lambda_2}{\lambda_1+\lambda_2}\left(\frac{\mu}{\lambda_1+\lambda_2}-1\right)(d_2-w_1) \\ &= \frac{\mu\big[\lambda_1(d_1-w_1)+\lambda_2(d_2-w_1)\big]}{(\lambda_1+\lambda_2)^2}. \end{aligned} \tag{4.9}$$

Case 4. The wave is released after the first deadline and the server completes the work before the second deadline (Figure 4.5).

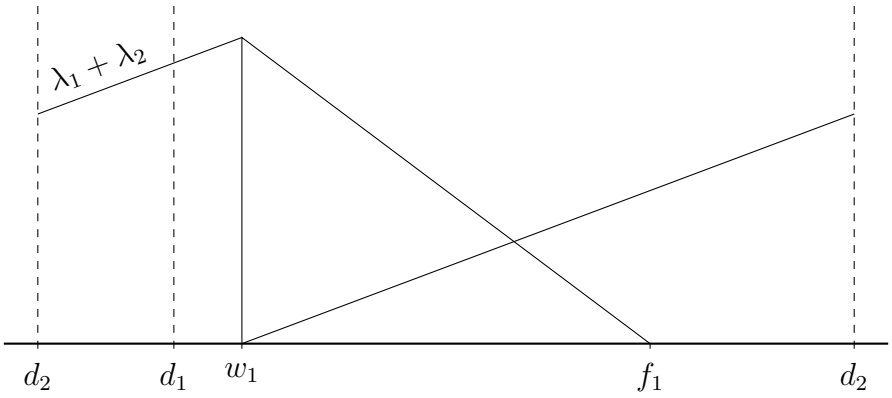


Figure 4.5: Case 4: $w_1 > d_1$ and $f_1 < d_2$.

The server processes classes simultaneously, and we assume that the instantaneous completion rate is proportional to the arrival rates of classes; however, now all class 1 orders miss

their deadline. Because $f_1 \leq d_2$, the system NSD can be written from Equation 4.4:

$$\begin{aligned} \text{NSD} &= \frac{\lambda_2}{\lambda_1 + \lambda_2}\text{NSD}_2 \\ &= \frac{\lambda_2}{\lambda_1 + \lambda_2}\big[1 - (d_2 - w_1)\big]. \end{aligned} \tag{4.10}$$

Case 5. The wave is released after the first deadline and the server completes the work after the second deadline (Figure 4.6).

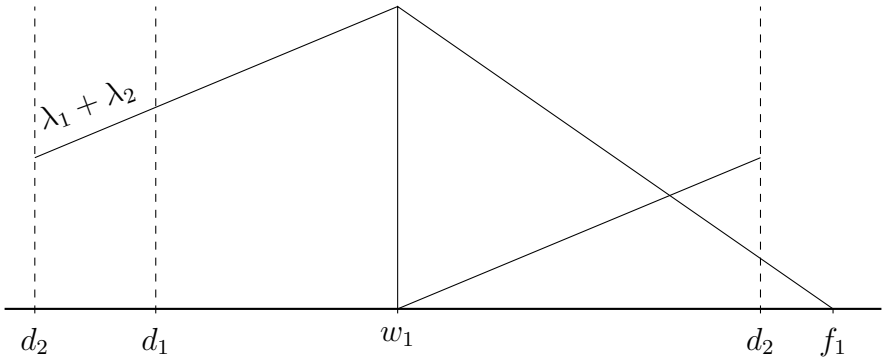


Figure 4.6: Case 5: $w_1 > d_1$ and $f_1 > d_2$.

Using Equation 4.8, the system NSD is:

$$\begin{aligned} \text{NSD} &= \frac{\lambda_2}{\lambda_1 + \lambda_2}\text{NSD}_2 \\ &= \frac{\lambda_2}{\lambda_1 + \lambda_2}\left(\frac{\mu}{\lambda_1 + \lambda_2} - 1\right)(d_2 - w_1). \end{aligned} \tag{4.11}$$

To demonstrate how each class NSD changes with respect to the single wave release time, consider the following example. Suppose there are two order classes having arrival rates $\lambda_1 = \lambda_2$. The first and the second class of orders have deadlines $d_1 = 0.5$, $d_2 = 1$. The processing capacity of the server $\mu = (\lambda_1 + \lambda_2)/\rho$, where $\rho = 0.5$. Using NSD functions for each class, we obtain the piecewise functions of NSD_1 and NSD_2. Figure 4.7 shows each class NSD (and the resulting system NSD) with respect to wave release time w_1.

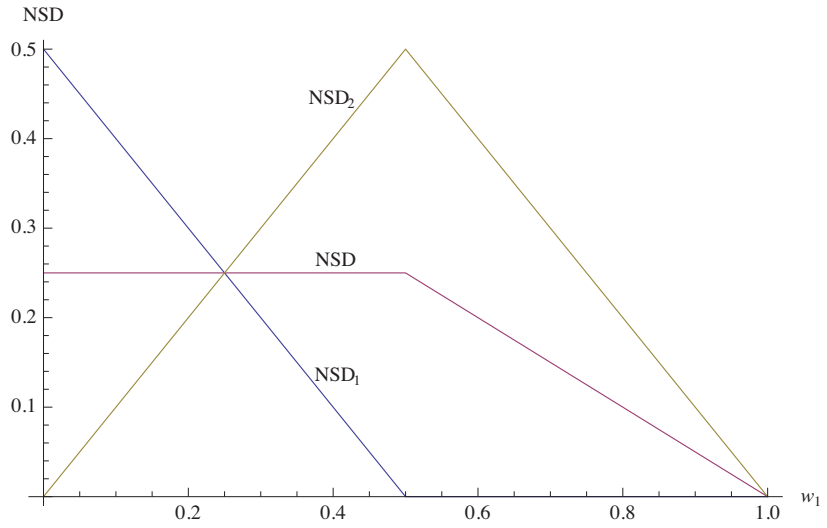


Figure 4.7: Change in the system NSD in a two class, single wave system.

Figure 4.7 shows the change in system NSD when $\rho = 0.5$. As expected, the function of NSD_2 is the same as in single class, single wave systems. When $w_1 < 1 - \rho$, the server completes fewer number of class 2 orders by d_2. On the other hand, when $w_1 > 1 - \rho$, the server completes the wave after d_2, which produces lower NSD_2. Consequently, NSD_2 is maximized when $w_1 = 1 - \rho$. Although releasing the wave at $w_1 = 0.5$ maximizes NSD_2, all class 1 orders would be missed by d_1, which results in $NSD_1 = 0$. What should be the single wave release time to maximize system NSD?

Figure 4.7 shows that the system NSD is maximized when $w_1 \leq 1 - \rho$. Although the system NSD does not change when $w_1 \leq 0.5$, each class NSD is affected by the release time. For example, when $w_1 = 0.3$, $NSD_1 < NSD_2$ and when $w_1 = 0.2$, $NSD_1 > NSD_2$. Consequently, the optimal release time depends on the importance of high NSD for each class. If the classes are equally important, then the optimal single wave release time $w_1 = 0.25$.

Combining equations 4.5–4.11, we define the system NSD as a piecewise function of release time w_1:

$$\text{NSD} = \begin{cases} (1+w_1) - \dfrac{\lambda_1 d_1 + \lambda_2 d_2}{\lambda_1+\lambda_2}, & \text{if } f_1 \le d_1 \\ \dfrac{\lambda_1}{\lambda_1+\lambda_2}\left(\dfrac{\mu}{\lambda_1+\lambda_2}-1\right)(d_1-w_1) + \dfrac{\lambda_2}{\lambda_1+\lambda_2}\big[1-(d_2-w_1)\big], & \text{if } w_1 \le d_1 < f_1 \le d_2 \\ \dfrac{\mu\big[\lambda_1(d_1-w_1)+\lambda_2(d_2-w_1)\big]}{(\lambda_1+\lambda_2)^2}, & \text{if } d_2 < f_1,\ w_1 \le d_1 \\ \dfrac{\lambda_2}{\lambda_1+\lambda_2}\big[1-(d_2-w_1)\big], & \text{if } f_1 < d_2,\ w_1 > d_1 \\ \dfrac{\lambda_2}{\lambda_1+\lambda_2}\left(\dfrac{\mu}{\lambda_1+\lambda_2}-1\right)(d_2-w_1), & \text{if } f_1 > d_2,\ w_1 > d_1 \end{cases} \tag{4.12}$$

Equation 4.12 is composed of continuous linear pieces forming a *polygonal curve*. Each piece forms a linear segment, and the system NSD is maximized at one of the extreme points of the segments. As a consequence, we can decompose the problem into distinct linear programming (LP) sub-problems. The maximum system NSD can be found by solving each sub-problem and then choosing the w_1 that maximizes the system NSD.

When there are more than two classes of orders, the number of subproblems increases. Fortunately, the structure of the problem allows us to generalize results of a single class system (Section 2.2) and develop the system NSD function for single wave systems with more classes. In a system with K order classes, for example, when the server completes the wave before the deadline d_i, only orders that arrive after the release do not count toward its NSD, and

$$\text{NSD}_i = 1 - (d_i - w_1).$$

When the server completes the single wave after the deadline d_i, we know that

$$\begin{aligned} \text{NSD}_i &= \frac{\mu(d_i-w_1)(\lambda_i/L) - \lambda_i(d_i-w_1)}{\lambda_i} \\ &= \frac{\lambda_i\big[(\mu/L)(d_i-w_1) - (d_i-w_1)\big]}{\lambda_i} \\ &= (d_i-w_1)\left(\frac{1}{\rho}-1\right). \end{aligned}$$

If the server starts processing the wave after d_i, non of the class i orders count toward its NSD, thus $\text{NSD}_i = 0$. As a consequence,

$$\text{NSD}_i = \begin{cases} 1 - (d_i - w_1), & \text{if the wave completes before } d_i \\ (d_i - w_1)\left(\dfrac{1}{\rho} - 1\right), & \text{if the wave starts before } d_i \text{ and completes after } d_i \\ 0, & \text{otherwise} \end{cases}$$

and the system NSD for a single wave system is equal to $\sum_{i=1}^{K} \lambda_i \text{NSD}_i / L$.

4.3 Multiple-Class, Multiple-Wave Systems

When there is a single wave and multiple classes, the wave releases orders from all classes. Therefore a single wave is a mixture of all classes and each class comprises some proportion of the wave based on its arrival rate. In practice, distribution centers typically have multiple customer classes whose orders must be processed together (to achieve economies of scale in operations) within multiple waves. The server may process single or many classes in a wave. That is, an individual wave may be dedicated to a single class or to a composition of multiple classes. There are two limiting situations: (1) Each wave is dedicated to a single class —a *class exclusive wave policy*, in which orders of a single class comprise the total workload of a wave. (2) Compositions of waves are based purely on the arrival rates of classes — a *pure mix wave policy*. The number of orders from class i in a wave is equal to the workload of the wave times the proportion of total arrivals of class i to the total daily load. These two policies can be classified under a more general policy — a *mixture policy*, in which compositions of classes in waves are decision variables. We are interested in optimal mixture policies. This problem involves multiple order classes, each having a different deadline, and multiple waves per day. Because wave releases are common across order classes, they must be established to maximize multiple objectives. Further, each wave must have an optimal composition of classes which maximizes the system NSD. What should be the timing and the content of

multiple waves to maximize the system NSD? This problem is considerably more complex because we must address not only the timing of waves, but also which available orders to release in each wave.

4.3.1 A Two-Class, Two-Wave Order Release System

We start by defining a simple version of the multiple-class multiple-wave system in which there are two classes and two waves ($K = 2, N = 2$). Arrival rates and deadlines of the classes are λ_1, λ_2 and d_1, d_2 such that $d_1 \leq d_2$. Without loss of generality, we assume $d_2 = 1$.

Two quantities are of interest: (1) the unworked inventory from the previous cycle, and (2) the orders that arrive within the current cycle. Recall that the cycle of a class is determined by the length of a unit time between two consecutive deadlines. Class 1 orders that arrive after the last release and before d_1 in the previous cycle form the unworked inventory for this class. (Similarly, class 2 orders that arrive after the last release and before d_2 form class 2's unworked inventory.) Figure 4.8 illustrates the unworked inventory from previous cycles and the total number of orders that arrive in the current cycle before the first release of the day.

In Figure 4.8, the amount of unworked class 1 inventory depends on the release time of the last wave. Because w_2 is the last wave before d_1, the unworked inventory for class 1 is equal to $\lambda_1(d_1 - w_2)$. For class 2 orders, the unworked inventory by d_2 is equal to $\lambda_2(d_2 - w_2)$. The total number of orders that arrive within the current cycle before the first release is a function of the first release time w_1. Together with the unworked inventory, orders that arrive within the current cycle define the *work content* of a wave.

Although releasing all unworked inventory first is a guarantee for satisfying absolute business requirements (e.g., if an order misses a deadline, the DC should ship the order by the second deadline), because workers are indifferent to which order they process, there may be orders which will not be completed even after a couple or more deadlines. (We discuss

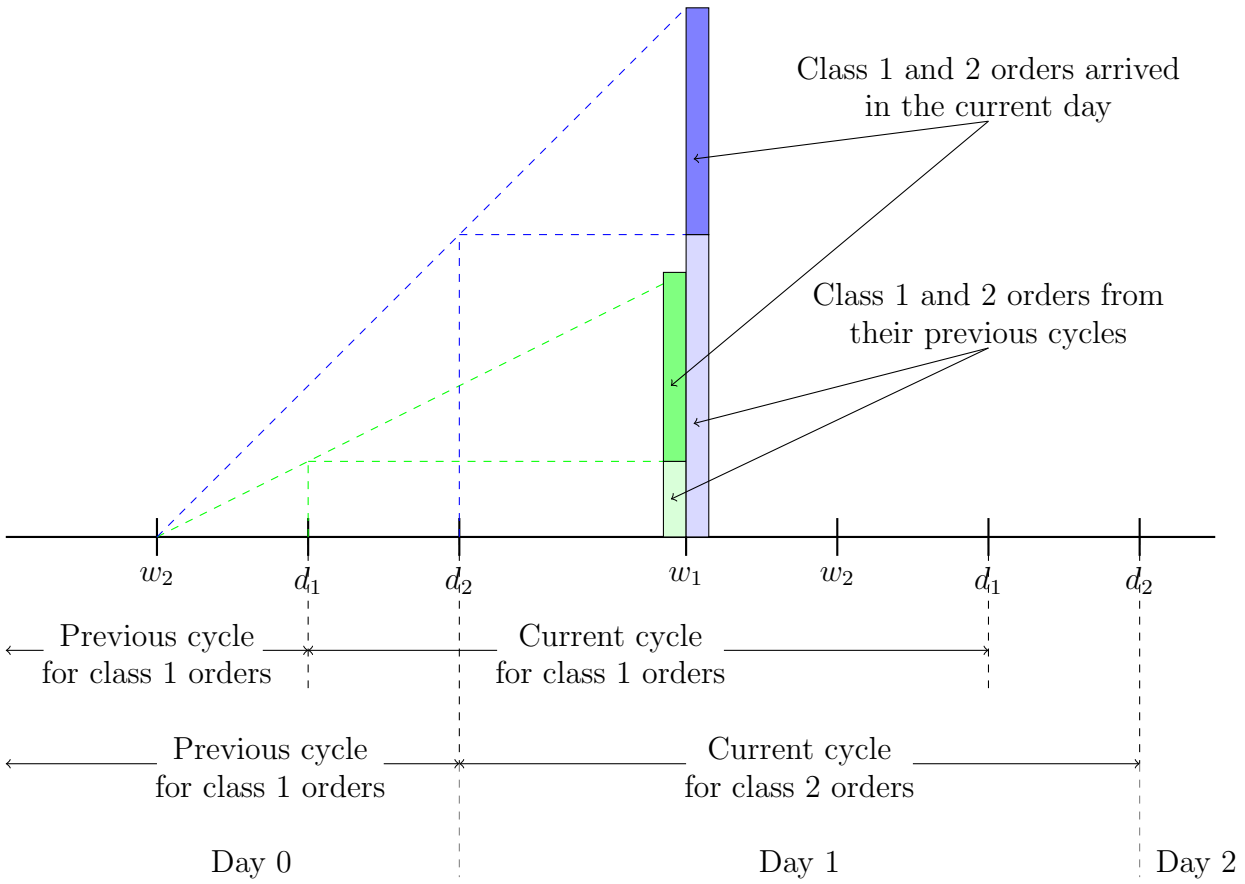


Figure 4.8: A wave is comprised of orders from their previous cycles (unworked inventory) and orders that arrive in the current day.

the implications of different sequencing rules in Section 4.3.2). Consequently, the unworked inventory can be distributed to the waves to ship more orders that arrive in their current cycle.

Define y_{ij} as the fraction of unworked inventory of class i orders released in wave w_j. If $w_2 \leq d_1$, then the amount of unworked class 1 inventory released in wave j is $\lambda_1(d_1 - w_2)y_{1j}$; otherwise, $\lambda_1(d_1 - w_1)y_{1j}$. We write similar expressions for class 2 unworked inventory: unworked inventory for class 2 in the second wave is equal to $\lambda_2(d_2 - w_2)y_{21}$ and $\lambda_2(d_2 - w_2)y_{22}$. The number of class 1 orders that arrive in their current cycle is equal to $\lambda_1(1 - d_1 + w_1)$. The number of class 2 orders that arrive in their current cycle is analogous: $\lambda_2(1 - d_2 + w_1) = \lambda_2 w_1$.

We are now ready to write the necessary conditions for a two-class, two-wave system. Let x_{ij} be the number of class i orders that arrive in their current cycle and are released in

wave j ($0 \leq x_{ij}$). Because the total number of class i orders is $\lambda_i(1 - d_i + w_1)$, we have the following condition for the first wave: $x_{i1} \leq \lambda_i(1 - d_i + w_1),\ i = 1, 2$. For the second wave, we have $x_{i2} \leq \lambda_i(w_2 - w_1),\ i = 1, 2$. The second wave must start after the server completes the total work in the first wave, so $w_2 \geq f_1$. Because we do not know which wave is the last one before a specific deadline, we denote the release time of a wave before d_1 by w_ι. Then, completion time of the first wave

$$f_1 = \begin{cases} w_1 + \dfrac{x_{11} + \lambda_1(d_1 - w_2)y_{11} + x_{21} + \lambda_2(d_2 - w_2)y_{21}}{\mu}, & \text{if } w_\iota = w_2 \\ w_1 + \dfrac{x_{11} + \lambda_1(d_1 - w_1)y_{11} + x_{21} + \lambda_2(d_2 - w_2)y_{21}}{\mu}, & \text{otherwise.} \end{cases}$$

We allow the second wave to be completed after d_2. We also require the condition to release all unworked inventory in waves, or $\sum_{j=1}^{2} y_{ij} = 1, i = 1, 2$. Finally, wave release times should satisfy $0 \leq w_j \leq 1,\ j = 1, 2$.

The necessary conditions for a feasible solution are relatively easier to define, but the objective function is more challenging. By definition, if the first wave is completed before d_1, then all orders of class 1 that arrive in the current cycle count toward its NSD. This quantity is, then, equal to x_{11}. If $f_1 > d_1$, then only some proportion of class 1 orders will be completed by the deadline. The first wave includes class 1 orders both from the previous cycle and the current cycle: x_{11} and $\lambda_1 y_{11}(d_1 - w_1)$, respectively. When $f_1 > d_1$, the number of class 1 orders completed by d_1 is equal to $\mu(d_1 - w_1)\lambda_1/(\lambda_1 + \lambda_2)$. Assuming that the server randomly selects which order to process (i.e. the server is indifferent to orders of unworked inventory and orders that arrive in the current cycle), the number of class 1 orders that arrive in the current cycle and are completed by d_1 is equal to $x_{11}\big[\mu(d_1 - w_1)\lambda_1/(\lambda_1 + \lambda_2)\big]/\big[x_{11} + \lambda_1 y_{11}(d_1 - w_1)\big]$.

For convenience, we denote the number of class i orders that account for their NSD in wave j by n_{ij}. Simplifying terms, the number of class 1 orders completed by d_1 when $f_1 > d_1$ is

$$n_{11} = \frac{x_{11}(d_1 - w_1)}{\rho[x_{11} + y_{11}(d_1 - w_1)]}.$$

Similarly, if $f_2 \leq d_1$, the number of class 1 orders completed in wave 2 is x_{12}; otherwise $(w_\iota = w_2)$

$$n_{12} = \frac{x_{12}(d_1 - w_2)}{\rho[x_{12} + y_{12}(d_1 - w_2)]}.$$

Let us now consider class 2 orders. If the first wave completes before the second deadline ($f_1 \leq d_2$, but not necessarily $w_1 \leq d_1$), then all class 2 orders count toward its NSD which is equal to x_{21}; otherwise

$$n_{21} = \frac{x_{21}(d_2 - w_1)}{\rho[x_{21} + y_{21}(d_2 - w_1)]}$$

many class 2 orders count toward its NSD. Finally suppose that the server completes the second wave by d_2. Because all class 2 orders that arrive in the current day and are ready by their deadline, x_{22} many of them count toward its NSD. On the other hand, if $f_2 > d_2$, then

$$n_{22} = \frac{x_{22}(d_2 - w_2)}{\rho[x_{22} + y_{22}(d_2 - w_2)]}$$

many class 2 orders count toward its NSD. The completion time of wave 2 is analogous to f_1:

$$f_2 = \begin{cases} w_2 + \dfrac{x_{12} + \lambda_1(d_1 - w_2)y_{12} + x_{22} + \lambda_2(d_2 - w_1)y_{22}}{\mu}, & \text{if } w_\iota = w_2 \\ w_2 + \dfrac{x_{12} + \lambda_1(d_1 - w_1)y_{12} + x_{22} + \lambda_2(d_2 - w_2)y_{22}}{\mu}, & \text{otherwise.} \end{cases}$$

Then, we have the following expression for the number of class i orders that are processed in wave j which count toward the system NSD:

$$n_{ij} = \begin{cases} x_{ij}, & \text{if } f_j \leq d_i \\ \dfrac{x_{ij}(d_i - w_\iota)}{\rho[x_{ij} + y_{ij}(d_i - w_\iota)]}, & \text{otherwise} \end{cases} \tag{4.13}$$

for $i = 1, 2$ and $j = 1, 2$. Note that variables n_{ij} are conditional expressions and we need to introduce new binary variables to the objective function in order to handle these expressions. Finally, we require the last wave to be completed before the release of the next day's first

wave. That is $f_2 \leq 1 + w_1$. Consequently, we formulate the problem as follows:

$$\begin{aligned}
\max \text{ NSD} = \quad & \frac{\sum_{i=1}^{2}\sum_{j=1}^{2} n_{ij}}{\sum_{i=1}^{2} \lambda_i} \\
\text{s.t.} \quad & x_{11} \leq \lambda_1(1 - d_1 + w_1) \\
& x_{21} \leq \lambda_2(1 - d_2 + w_1) \\
& x_{12} \leq \lambda_1(w_2 - w_1) \\
& x_{22} \leq \lambda_2(w_2 - w_1) \\
& y_{11} + y_{12} = 1 \\
& y_{21} + y_{22} = 1 \\
& f_1 \leq w_2 \\
& f_2 \leq 1 + w_1 \\
& w_2 \leq 1 \\
& 0 \leq w_j \qquad j = 1, 2 \\
& 0 \leq x_{ij} \qquad i = 1, 2; j = 1, 2 \\
& 0 \leq y_{ij} \qquad i = 1, 2; j = 1, 2
\end{aligned}$$

Example. Suppose that there are two classes of orders with stationary arrival rates $\lambda_1 = \lambda_2$. Assume the last deadline is at $d_2 = 1$. For different deadlines of d_1 and utilization levels of ρ, what is the optimal system NSD for these systems?

The variables of interest are $\{w_1, w_2, x_{11}, x_{12}, x_{21}, x_{22}, y_{11}, y_{12}, y_{21}, y_{22}\}$ which can be determined by solving the above problem. Using MATHEMATICA's `NMaximize` function, we solve for the variables that maximize the system NSD for a two-class, two-wave system. A detailed discussion on the optimization methodology is given in Section 4.3.2. Each point in Figure 4.9 illustrates a solution for the system NSD with given d_1 and ρ values.

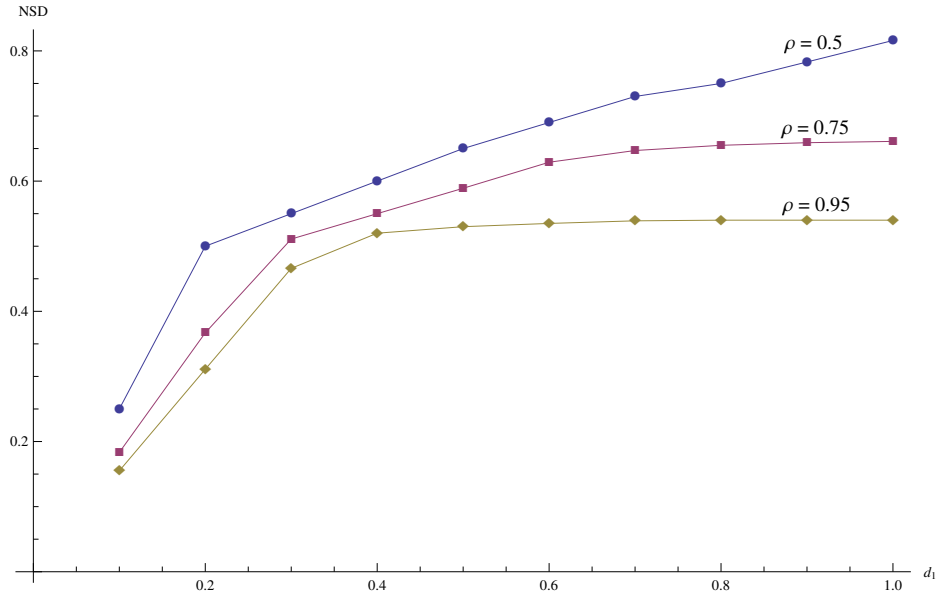


Figure 4.9: Solutions for a two class, two wave system ($\lambda_1 = \lambda_2$).

As expected, system NSD improves as the utilization decreases. When deadline d_1 is early, the server cannot process many class 1 orders on time, and system NSD is low. This observation suggests that

Observation 1. *Setting a common deadline for all customer classes improves system NSD.*

What would be the potential improvement in system NSD if the DC were able to change the deadlines? For example, if class 1 orders have a deadline at noon ($d_1 = 0.5$) and the DC is able to set the deadlines, the system NSD will be improved by 16.6% and 7.2% when there is a common deadline (for $\rho = 0.5$ and $\rho = 0.75$). Although improvements are significant,

Observation 2. *There is a little marginal benefit to having a common deadline when utilization is high.*

This can be seen especially when $\rho = 0.95$. Because the system is heavily loaded, the server is able to complete fewer class 1 and class 2 orders by d_1 and d_2. When d_1 is early, releasing class 2 orders in the first wave results in fewer completed class 1 orders by d_1, which produces a lower system NSD. Consequently, the first wave must be class exclusive. What

should be the content of the second wave? Should it be class exclusive, or should it include some class 1 orders?

Denote the percentage of class i orders in wave j by c_{ij}. Then,

$$c_{ij} = \frac{x_{ij} + \lambda_i(d_i - w_\iota)y_{ij}}{\sum_{i=1}^{2}[x_{ij} + \lambda_i(d_i - w_\iota)y_{ij}]}. \tag{4.14}$$

Figures 4.10–4.12 illustrate the optimal timing of waves and mixtures of classes for different values of ρ. The horizontal axis indicates different values of d_1. Green and blue bars show the proportions of class 1 and class 2 orders, respectively; and dark (and light) bars indicate the proportions in wave 1 (and wave 2).

The first pairs of bars in the figures show the solutions when $d_1 = 0.1$. When $\rho = 0.5$, the solution allocates only class 1 orders to the first wave and only class 2 orders to the second wave (a class exclusive wave policy). However, as utilization increases, the solution also allocates class 1 orders to the second wave (the first pairs of bars in Figures 4.11 and 4.12). Why does the solution allocate more class 1 orders to the second wave? An increased utilization implies a later wave completion, consequently the solution allocates more class 1 orders to the second wave in order to complete more class 1 orders in the first wave by d_1.

For $\rho = 0.5$, the solution produces the same release times and the contents of waves for all $d_1 \leq 0.3$; however, the resulting system NSD improves as the first deadline is set to a later time. When $d_1 > 0.3$, the number of class 2 orders in the first wave and the number of class 1 orders in the second wave start to increase—suggesting more homogeneous waves. We also observe that the solution produces more homogeneous waves as the deadlines get closer for higher values of utilization. When $d_1 = d_2$, the solution proposes a pure mix wave policy for $\rho = 0.5$. Consequently,

Observation 3. *When d_1 is early, the solution suggests class exclusive waves and as d_1 approaches d_2, the solution suggests more homogeneous waves.*

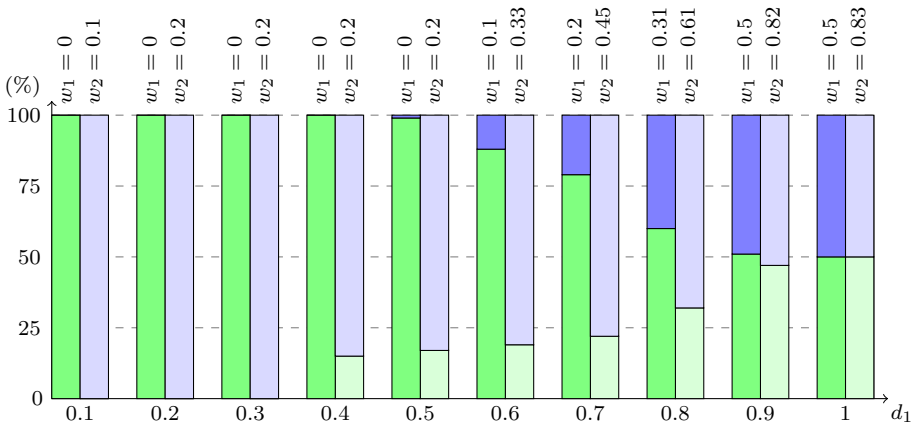


Figure 4.10: Timing of waves and class mixtures when $\rho = 0.50$.

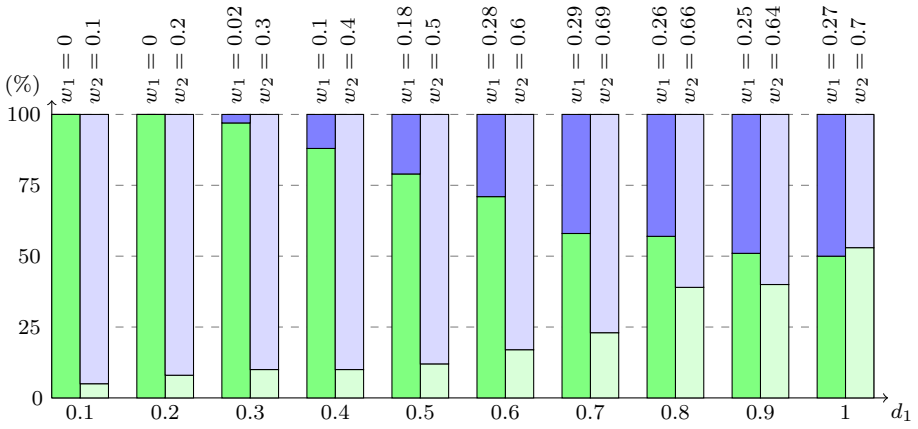


Figure 4.11: Timing of waves and class mixtures when $\rho = 0.75$.

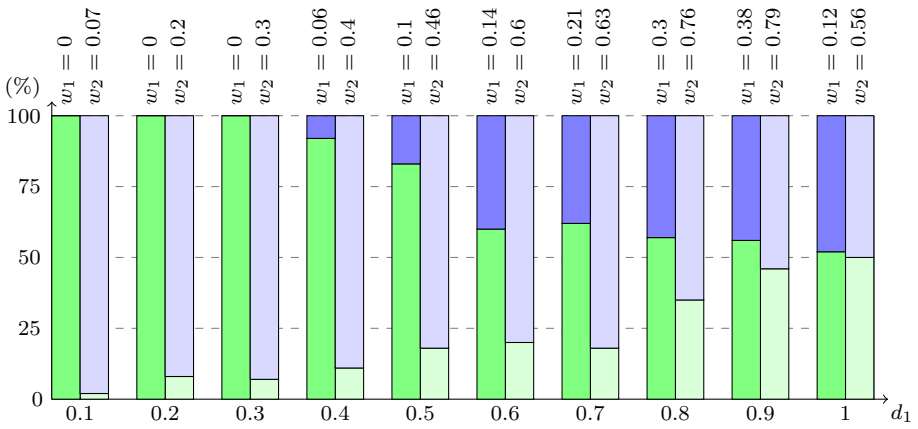


Figure 4.12: Timing of waves and class mixtures when $\rho = 0.95$.

For other values of ρ, the waves have nearly the same number of class 1 and class 2 orders when $d_1 = d_2$. This is because the first wave contains only orders that arrive in the current cycle and all unworked inventory from the previous cycles is released in the second wave (i.e., the solution produced $y_{11} = y_{21} = 0$). As a result, the first wave release time is later than the one in single class systems (e.g. the release time of the first wave in a single class system is equal to 0.25 when $\rho = 0.75$, but the solution found the first release time as $w_1 = 0.27$).

Setting Cutoff Times

We have one remaining question corresponding to the service level: How many hours before the deadline(s) should the cutoff time(s) be set in order to guarantee all customer orders are shipped by their deadlines?

In single class systems with deterministic arrival and processing rates, we first assumed the cutoff time is equal to the deadline. Analytical solutions produce maximum NSD, and by setting cutoff time equal to the last wave release time, we show that all customers who place their orders before the cutoff time receive their orders by the next deadline. That is, service level (which is analogous Type 1 service in inventory theory) is 100% when cutoff time is equal to the last release time. Because NSD counts the number of orders that arrived in the current cycle and shipped by its deadline, NSD is also 100% when cutoff time is set optimally. Therefore, maximizing NSD is the same as achieving the latest possible cutoff time for single class systems.

Consider multiple-class systems. Denote cutoff time of class i by c_i. Recall that NSD_i is defined as the fraction of class i orders that arrived between successive c_is and completed before d_i . To this point, we have assumed that $c_i = d_i$. Numerical solution of the analytical model determines wave release times and their contents which maximize NSD_i. Does setting cutoff times equal to the last wave release times produce 100% service level?

Suppose there are two-class, two-wave release system in which $\lambda_1 = \lambda_2, d_1 = 0.8$, $d_2 = 1$. When $\rho = 0.75\%$, release times for this system are $w_1 = 0.26$, $w_2 = 0.66$, which are completed at $f_1 = 0.66$, and $f_2 = 1$. The resulting system NSD is equal to 66%. If cutoff times are equal to deadlines ($c_1 = d_1; c_2 = d_2$), then service level is also equal to 66%.

If there is a common cutoff time for both classes which is set to w_2, then all class 2 customers receive their orders by d_2, thus service level for class 2 customers is equal to 100%. If class 1 orders would have the same cutoff time, then all class 1 orders in the first wave would be completed by d_1; however, too few class 2 orders in the second wave would be completed by their deadline. That is, class 1 customers would have a service level which is less than 100%. If the cutoff time were equal to w_1, then all class 1 and class 2 customers who place their orders before $c = w_1$ would receive their orders on time; however, because the cutoff time for class 2 customers is early, they must wait longer for their orders.

Now, suppose there is a unique cutoff time for each class. When cutoff times are equal to the last wave release times completed before the deadlines, all class 1 and class 2 customers receive their orders by d_1 and d_2, therefore service level is equal to 100%. Then, for our example, $c_1 = 0.26$, and $c_2 = 0.66$. Table 4.1 shows the sets of cutoff times that guarantee 100% service level for different values of d_1.

Table 4.1: Cutoff times for different deadlines d_1 in a two class, two wave system ($\lambda_1 = \lambda_2$).

	$\rho = 0.50$		$\rho = 0.75$		$\rho = 0.95$	
d_1	c_1	c_2	c_1	c_2	c_1	c_2
0.1	0	0.10	0	0.10	0	0.07
0.2	0	0.20	0	0.20	0	0.20
0.3	0	0.20	0.02	0.30	0	0.30
0.4	0	0.20	0.10	0.40	0.06	0.40
0.5	0	0.20	0.18	0.50	0.14	0.14
0.6	0.10	0.33	0.26	0.60	0.14	0.14
0.7	0.20	0.45	0.29	0.29	0.21	0.21
0.8	0.31	0.61	0.25	0.66	0.3	0.3
0.9	0.50	0.825	0.25	0.25	0.38	0.38
1	0.83	0.83	0.70	0.70	0.56	0.56

Because solutions of the analytical model determine wave release times (and their contents) without regard to the cutoff times (i.e. cutoff times are not decision variables in the model), they do not produce optimal cutoff times. That is, maximizing system NSD is not same as achieving the latest possible cutoff times when there are multiple order classes. Cutoff times in Table 4.1 can be used to establish lower bounds for optimal cutoff times; however, a model that produces optimal cutoff times is significantly more complicated and will be the subject of future research.

4.3.2 Systems with More Classes and Waves

In this section, we extend two-class, two-wave systems to more general systems in which there are K different types of customers whose orders are released in N waves. Because the number of waves in a typical application is usually less than six and we are interested in systems with a reasonable number of deadlines (i.e. transportation modes), we focus on systems with up to six classes and six waves.

Our objective is to find a vector of variables $\{w_1, w_2, \ldots, w_N, c_{11}, \ldots, c_{ij}, \ldots, c_{KN}\}$ that maximizes system NSD. Recall that the number of class i orders in wave j that count toward the system NSD is defined by n_{ij} in Equation 4.13. Because variables n_{ij} are conditional expressions, we introduce a new binary variable z_{ij}:

$$z_{ij} = \begin{cases} 1, & \text{if } f_j \leq d_i \\ 0, & \text{otherwise} \end{cases}$$

for all $i \in \{1, \ldots, K\}$ and $j \in \{1, \ldots, N\}$. That is, variables z_{ij} indicate if wave j is completed before or after deadline i. Incorporating z_{ij} into Equation 4.13, the system NSD function

may be written as follows:

$$\text{NSD} = \frac{\sum_{i=1}^{K}\sum_{j=1}^{N}\left[x_{ij}\left(z_{ij} + \frac{(1-z_{ij})(d_i - w_\iota)}{\rho(x_{ij} + y_{ij}(d_i - w_\iota))}\right)\right]}{\sum_{i=1}^{K}\lambda_i}. \tag{4.15}$$

Generalizing our findings in a two class, two wave system, we determine the following necessary conditions. First, the number of class i orders that arrive in its current cycle and released in wave j should not be greater than the number of orders accumulated until the first wave release time w_1. That is, the solution decides how many class i orders to release and therefore the wave is *selective*. This condition should also hold for following waves. Consequently, we have

$$x_{i1} \leq \lambda_i(d_K - d_i + w_1) \quad i \in \{1, \ldots, K\}, \tag{4.16}$$

$$x_{ij} \leq \lambda_2(w_j - w_{j-1}) \quad \forall i \in \{1, \ldots, K\}, \quad \forall j \in \{2, \ldots, N\}. \tag{4.17}$$

A second set of constraints is required to allocate unworked inventories (from the previous day) to the waves. Defining continuous variables y_{ij}, we have

$$\sum_{j=1}^{N} y_{ij} = 1 \qquad \forall i \in \{1, \ldots, K\}. \tag{4.18}$$

In practice, Equation 4.18 implies absolute business requirements such as shipment guarantee by the "second scheduled deadline;" however, because pickers pick orders in a mixed sequence, all orders may still not be ready by their deadline. (We discuss the implications of order sequencing below.)

Recall that the wave contents are determined by the unworked inventory from the previous cycles and the number of orders that arrive in the current cycle. For all $j \in \{2, \ldots, N\}$,

the completion times of wave j

$$f_j = w_j + \frac{\sum_{i=1}^{K} \left[x_{ij} + \lambda_i (w_j - w_{j-1}) y_{ij}\right]}{\mu}.$$

The expression for f_1 is more complicated. As we showed in a two-class, two-wave system, the first wave contains y_{i1} of class i orders from its previous cycle. We do not know which is the last wave before d_i, so we denote the last wave release time before a specific deadline by w_ι. The total number of orders released in the first wave is then equal to $\sum_{i=1}^{K} \left[x_{i1} + \lambda_i (d_i - w_\iota) y_{i1}\right]$ and the first wave's completion time

$$f_1 = w_1 + \frac{\sum_{i=1}^{K} \left[x_{i1} + \lambda_i (d_i - w_\iota) y_{i1}\right]}{\mu}.$$

The optimal solution requires that each wave must be released after the completion of the previous one (non-overlapping waves). Therefore,

$$f_j \leq w_{j+1} \quad \forall j \in \{1, \ldots, N-1\}, \tag{4.19}$$

$$f_N \leq 1 + w_1 \quad j = N, \tag{4.20}$$

We allow the server to complete the last wave after the last deadline. (Recall that we have assumed that the last deadline $d_K = 1$, which defines the current day.) The last wave of the current day should then be released before 1. Then,

$$w_N \leq 1. \tag{4.21}$$

Together with the non-negativity of variables, necessary conditions 4.16–4.21 lead to the following optimization problem for a K class, N wave system:

$$\text{maximize NSD} = \frac{\sum_{i=1}^{K}\sum_{j=1}^{N}\left[x_{ij}\left(z_{ij}+\frac{(1-z_{ij})(d_i-w_\iota)}{\rho(x_{ij}+y_{ij}(d_i-w_\iota))}\right)\right]}{\sum_{i=1}^{K}\lambda_i}$$

$$\begin{aligned}
\text{subject to} \quad & x_{i1} \le \lambda_i(d_K - d_i + w_1) && \forall i \in \{1,\ldots,K\},\ j = 1 \\
& x_{ij} \le \lambda_i(w_j - w_{j-1}) && \forall i \in \{1,\ldots,K\},\ \forall j \in \{2,\ldots,N\} \\
& \sum_{j=1}^{N} y_{ij} = 1 && \forall i \in \{1,\ldots,K\} \\
& f_j \le w_{j+1} && \forall j \in \{1,\ldots,N-1\} \\
& f_N \le 1 + w_1 && j = N \\
& w_N \le 1 \\
& z_{ij} \in \{0,1\} && \forall i \in \{1,\ldots,K\},\ \forall j \in \{1,\ldots,N\} \\
& 0 \le w_j && j \in \{1,\ldots,N\} \\
& 0 \le x_{ij} && i \in \{1,\ldots,K\}; j \in \{1,\ldots,N\} \\
& 0 \le y_{ij} && i \in \{1,\ldots,K\}; j \in \{1,\ldots,N\}
\end{aligned}$$

The model has linear and nonlinear constraints and a nonlinear objective function including binary variables and non-convexities in the continuous variables (we discuss this issue below); therefore it falls into a class of nonlinear optimization problems with binary variables.

Although writing expressions for the constraints is relatively easy, because we do not know which wave is the last wave before a certain deadline, it is more challenging to write the objective function. We develop an algorithm in `Visual Basic for Applications` that evaluates all possible cases and generates the whole problem for a given number of classes and waves. Because nonlinearities imply many local maxima, solving problems optimally is difficult (if not impossible). Consequently, we choose a meta-heuristic solution algorithm to search for the optimal values of variables that maximize the system NSD. We present the solution approach and validation of the model below.

Optimization Methodology

Before presenting the proposed optimization method, it is useful to explore the structure of K class, N wave problems. These problems have multiple variables which form a polytope in $3(NK+1)$ dimensions. In addition to nonlinearities of variables, due to dependency between variables, we expect the objective function to be nonlinear and non-convex. Note that the objective function surface is not known in advance; however we can characterize it by generating many random solutions. For example, Figure 4.13 illustrates the surface of the objective function in a two class, two wave system in which $\lambda_1 = \lambda_2$ and $d_1 = 0.5, d_2 = 1$.

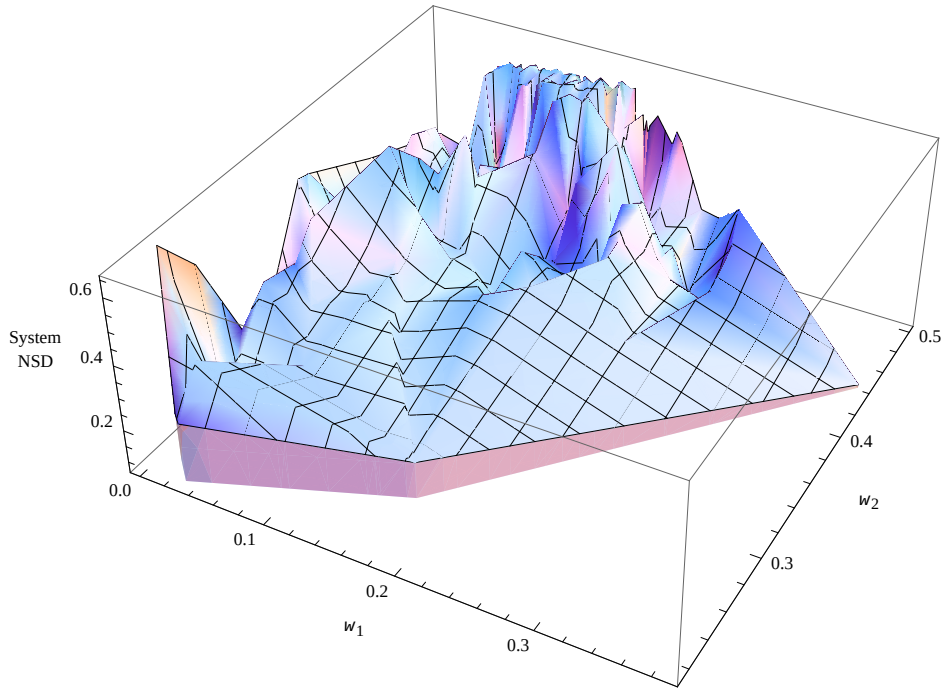


Figure 4.13: Admissible pairs of (w_1, w_2) indicate that the objective function surface is non-convex.

To expose the structure of the objective function, we generate more than 2,300 admissible pairs of release times (w_1, w_2) and plot them with the corresponding objective value. For this problem, we observe that the objective function is non-convex. Although this observation does not imply that any other problem has the same non-convexity, due to an added complexity of the mixture of classes in waves, we expect the objective functions of more complicated problems to be both nonlinear and non-convex.

We use a meta-heuristic called differential evolution (DE), which was first introduced by Storn and Price (1995) to optimize problems with a real-valued, non-differentiable, multi-modal objective functions with nonlinear constraints. Because the method searches on a search space, it does not guarantee optimality; however, our test results showed that differential evolution is an appropriate method for multiple-class, multiple-wave problems. (We discuss the details of the method in Appendix B.)

Model Validation

The analytical models assume a fluid model in which discrete arrivals are represented as a continuous stream of work. Consequently, we must validate the solutions before addressing systems with more classes and waves.

We represent the fulfillment system as a three-stage queueing network, corresponding to the picking, packing, and shipping processes. We assume each stage has twenty servers. For the validation experiment, we simulate systems with two classes and two waves at a utilization level of 0.5. Orders arrive to the system according to a Poisson process with rate $\lambda_1 = \lambda_2 = 1$ order per minute and processing times of workers are distributed exponentially with a rate of five minutes. We choose the relatively simple case of two waves and two classes because it is much easier to interpret the results than it would be for more complex cases.

An arrival to the system triggers a number of logical decisions to determine if it will be released in the next wave or not. Suppose that an order arrives in the current day before the first release time. It is assigned to the queue of the first wave. If the arrival occurs after the first and before the second wave release time, it is assigned to the second wave queue. Otherwise, the order waits in the release queue until the first release in the following day. Because each class has a different cycle regulated by its deadline, the amount of unworked inventory must be handled carefully. For class 2 orders that arrive after the last wave, y_{21} percent of them must wait until the first release and y_{22} percent of them must wait until the second release in the following day. For convenience, denote the arrival time of an order in

the current day by t_A. Delay time of a class 2 order is equal to $w_1 + d_2 - t_A$, if it arrived after w_2 and assigned to the first wave in the following day. (The delay time is $w_2 + d_2 - t_A$ if the order is assigned to the second wave in the next day.) When a class 1 order arrives after w_2, another decision is required. If $t_A \leq d_1$, then it is assigned to the first wave on the next day; otherwise it is assigned to the next day's first or second wave release queues. Once orders are released in waves, they are sequentially processed in picking, packing and shipping.

The analytical models assume the server randomly selects which order to process. That is, the server is indifferent to the arrival times of orders. Figure 4.14 demonstrates the realization of this assumption.

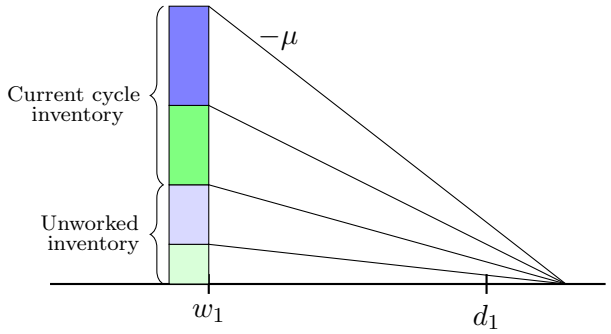
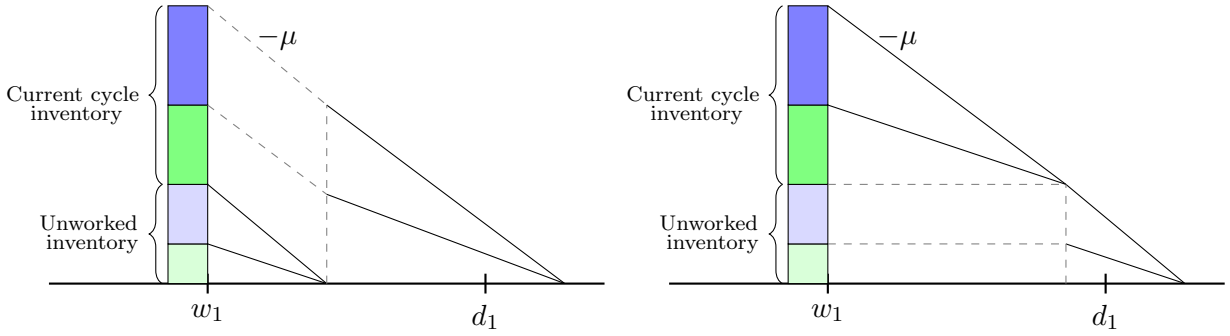


Figure 4.14: Analytical models assume the server randomly selects orders.

In Figure 4.14, the server completes the first wave after d_1 and some unworked class 1 inventory will not be processed by its (second) deadline. In practice, because pickers pick orders both from the previous and the current cycles in the same picking tour, there may be remaining unworked inventory at d_1. Further, the amount of unworked and current cycle inventories are only known at the time of release. In simulation, each order arrival changes the number of orders in the release queue and because we do not know which cycle (and wave) the order should be assigned to preserve the class weights, it is not possible to maintain the class weights in waves. Consequently, we use first-come, first served (FCFS) and last-come, first served (LCFS) sequencing policies to set bounds on the expected system NSD.

In the FCFS policy, the server processes the orders as they appear in the queue. On the other hand, the orders that arrive in the current cycle are worked first in the LCFS policy. In

both policies, the server has the same processing capacity; however, each rule may result in different system NSD. (Because the wave content is divided into two parts, the starting time of the second part can be thought as a *sub-wave*.) Figure 4.15 illustrates the implications of these two policies.



(a) FCFS policy results in lower system NSD. (b) LCFS policy results in higher system NSD.

Figure 4.15: Different sequencing rules result in different simulated average system NSD.

When the FCFS policy is used, the server can only start the orders that arrive in the current cycle after completing all unworked inventory. All unworked inventory is completed by d_1 (100% of orders are guaranteed for delivery by the second deadline); however, because the server completes the current cycle inventory after d_1, there are too many class 1 orders that will not be ready by the deadline. Consequently, the average system NSD will be lower than the expectation. When orders are processed based on the LCFS policy, the server completes more current cycle orders, therefore the average system NSD will be higher than the expectation. (In this case, the DC should not promise a guarantee for the second day delivery.) In Figures 4.14–4.15, we ignored the presence of cutoff times. Because cutoff times correspond to the wave release times which are completed by the deadlines, when they are set optimally, FCFS and LCFS policies should produce the same service level (i.e., the plots above will be irrelevant).

In each simulation run, we gradually increased the first deadline d_1 and kept the second deadline $d_2 = 1$. We insert the output of the analytical model (release times and y_{ij} values) as an input to the simulation model. Runs last seven simulated days, with 30 replications.

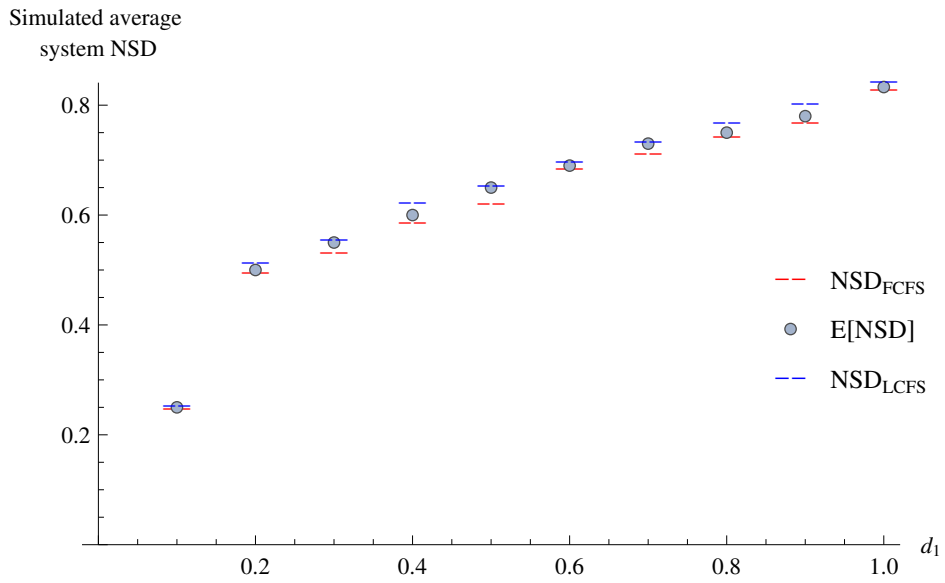


Figure 4.16: Model validation results with simulation when $\rho = 0.5$.

We verified the simulation model by first comparing the average utilization level of each stage with our expectation, and then inserting arbitrary release times to observe how system NSD changes. The average utilization of the workers was observed close to our expectation. As expected, different release times resulted in lower average daily system NSD. To validate the model, we compare the simulated average daily system NSD with the results of numerical solutions in each run. In Figure 4.16, the circular points correspond to the system NSD found by the solution of the analytical model for each value of d_1. Lower and upper bounds (found by applying FCFS and LCFS rules) are shown with horizontal notches. We observe that the expected system NSD is within its bounds, suggesting that the model valid.

Numerical Examples

To justify the solution method and explore the underlying characteristics of more complex systems, we conduct a numerical analysis. Because the number of order classes and wave releases in most applications is at most six, we limit the experiment to the systems with fewer than six classes and six waves.

We first test the performance of the optimization method by transforming multiple class systems into single class systems. That is, we set all deadlines equal to one while all other parameters remain unchanged. For each problem, we investigated three different utilization levels. There are 108 problems total.

Closed form solutions for single class, multiple wave systems are available (Section 2.3), and we use Equation 2.6 to assess the performance of the search method for each set of problems. Table 4.2 summarizes the performance of the solution method.

Table 4.2: Summary of performance metrics.

	Utilization level								
	$\rho = 0.5$			$\rho = 0.75$			$\rho = 0.95$		
Num of Waves	% of Solns Optimal	Max Gap	Avg Gap	% of Solns Optimal	Max Gap	Avg Gap	% of Solns Optimal	Max Gap	Avg Gap
2	100%	0%	0%	40%	5.76%	3.74%	40%	2.98%	2.74%
3	100%	0%	0%	40%	0.70%	0.62%	40%	1.01%	0.71%
4	80%	1.43%	1.43%	40%	2.09%	0.75%	40%	1.13%	0.67%
5	80%	2.35%	1.52%	60%	1.74%	1.20%	40%	0.35%	0.31%
6	60%	2.15%	1.24%	60%	2.02%	1.67%	60%	0.49%	0.41%

In Table 4.2, we solve six problems for a given utilization level and number of classes. Columns labeled with "% of Solns Optimal" represent the percentage of the solutions in which the method found the optimal solution. The method performs better when utilization is equal to 0.5; however, the percentage of problems for which the method produced optimal solutions appears to be only 13.3%. Although the method produced an optimal solution only for a small percentage of the test problems, the average gap between the numerical and the analytical methods is about 1.14%.

Example. Suppose each class has the same arrival rate $\lambda_i = \lambda_{i+1}, i \in \{1, \ldots, 5\}$, and the total processing capacity is equal to $\mu = \sum_{i=1}^{6} \lambda_i/\rho$ where $\rho = 0.5, 0.75, 0.95$. We assume that deadlines are equally distributed through the unit length of time. That is, $d_i = i/6, i \in \{1, \ldots, 6\}$. Figure 4.17 shows the numerical results of the analytical models.

In Figure 4.17, each data point illustrates a system NSD found by solving the corresponding problem with K classes and N waves. The system NSD for each value of ρ is shown

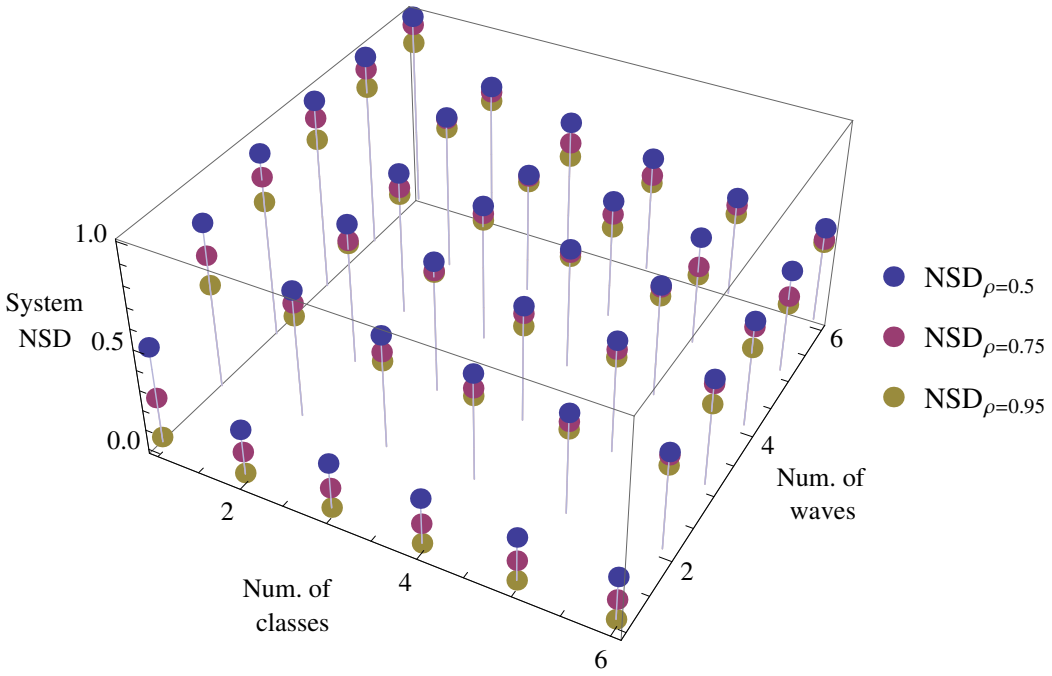


Figure 4.17: Numerical solutions of analytical models.

with a different color. We measure the computational time in seconds. The method solves a problem in less than a minute on average (41.8 seconds); however, the computational time increases as the number of classes and waves increases. (We observe that the computational time does not significantly change for different values of utilization.) When $K = 6, N = 6$, the method obtained the solution in more than four minutes. Table 4.3 shows the details of computational times for each problem.

Figure 4.17 led us to a number of observations: (1) When utilization increases, it impacts the system NSD. The impact of high utilization on system NSD is most significant when there is a single class of orders or when there is a single wave. (2) System NSD improves as the number of waves increases. As in single class systems, we observe that system NSD is improved as the number of waves increases, without regard to the number of classes. The maximum system NSD is observed when there is a single class of orders and the number of waves is six. On the other hand, the system NSD is minimum when there are six classes and

Table 4.3: Computational times in seconds.

	Number of waves					
Num. of classes	2	3	4	5	6	ρ
2	7.9	15.7	19.9	21.2	25.9	0.5
	8.0	14.3	18.5	19.8	28.2	0.75
	9.9	16.1	17.7	17.3	23.8	0.95
3	11.0	20.0	30.9	57.5	74.1	0.5
	11.0	22.9	32.1	64.6	81.1	0.75
	12.6	17.1	27.8	56.1	83.7	0.95
4	16.5	35.3	69.2	96.8	124.3	0.5
	20.6	26.8	67.4	132.2	123.6	0.75
	17.1	30.3	61.9	120.9	93.2	0.95
5	55.6	48.6	76.8	140.2	221.8	0.5
	47.4	48.9	74.6	134.5	242.5	0.75
	42.1	47.2	79.4	111.2	247.5	0.95
6	48.5	72.8	106.3	143.4	209.8	0.5
	45.5	93.1	113.4	184.7	223.5	0.75
	43.6	78.4	95.6	169.9	274.5	0.95

a single wave, which suggests that, (3) Systems with multiple deadline should compensate with more wave releases.

4.4 Implications for Practice

The results of multiple-class, multiple-wave problems suggest implications for practice. When there are multiple classes, many factors affect service performance, including the timing of waves and their contents. Increasing the number of waves seem to improve system NSD.

The solutions of single class systems suggest that the first wave should be released at $w_1 = 1 - \rho$. Although the server is busy ρ of the time, the first wave release time should be set to a different time in multiple-class systems. This observation implies possible idleness of workers between completion and release times of waves. That is, the workers may go idle in order to improve the system NSD. Managers would have to think carefully how to use this time productively, perhaps with replenishment or other necessary operations.

In an order fulfillment system that must fulfill orders of multiple classes, deadlines may significantly impact service performance. We have assumed deadlines that are evenly distributed throughout the day. What would the system NSD be if the deadlines were close to midnight? What should be the wave release times and contents? Should we expect later cutoff times in this case?

Earlier deadlines imply an earlier first wave release, and when deadlines are closer to midnight, waves should be released later. We observe that system NSD improves when deadlines are close together and set as late as possible. Late wave release times and deadlines close to midnight also imply later cutoff times. Consequently, DCs may offer late cutoff times to their customers while maintaining an acceptable level of system NSD.

The numerical results of the analytical models show that a wave should contain more orders of a particular class if that wave is the closest wave to the deadline of that class. When deadlines are closer, the waves should contain the same amount of orders from classes.

The sequence of picks in waves also affects the system NSD. Releasing late orders first guarantees second day delivery, but it results in lower system performance. When pickers pick the most imminent orders first, it will produce a higher next day fulfillment rate; however, if an order is not delivered by its first promised deadline, then it may also not be ready for the shipment by the second deadline. Managers should consider the tradeoff between absolute business requirements and marketing promises.

Chapter 5

Conclusions & Future Research

To attract more customers and increase the market share, order fulfillment systems offer aggressive service promises. That kind of offer can put pressure on DCs in their fulfillment operations. The purpose of this dissertation is to increase customer service by improving order releases with optimal timing of waves. In this dissertation, we present the first systematic investigation of wave planning in deadline-oriented order fulfillment systems.

Contribution 1. *We propose the first fluid model for a single-class, multiple-wave system that operates against a daily deadline.*

Systems with a single order class have correspondence in industry, especially in internet retailing where DCs promise overnight deliveries. We showed that properly timed waves should be smaller as the deadline approaches, rather than of uniform size, as intuition might suggest. The service performance can be increased by adding more waves, but there is a marginal benefit to increasing the number of waves beyond four or five. The optimal number of waves is based on both the pick density and the fixed time component of waves, and there should be fewer waves when the fixed time component is long. The optimal policy provides a more stable workload as well as more tractable flow of work. Maximum service performance can be achieved by setting the cutoff time equal to the latest release time.

Contribution 2. *We determined the optimal number and timing of waves in the presence of workload uncertainty.*

We addressed daily workload fluctuations which may be the result of natural demand variability or the result of internal workforce variability. We discussed how to adjust wave

release times and the number of waves to minimize the risk of workload exceeding capacity. In the presence of workload uncertainty, we showed that sacrificing a little from the maximum achievable service level hedges against a drastic drop in performance if utilization is high. While the release time of a single wave is later than what it would otherwise be, multiple waves should be released earlier to maximize expected NSD. We discussed the implications of wave release times on the cutoff times. We showed that the cutoff time should be no later than the last release time. Setting the cutoff time as late as possible is important for improving the service promise.

Contribution 3. *We showed how to establish the timing and contents of waves in order fulfillment systems with multiple order classes.*

We extended our single class systems to multiple class systems in which customers are grouped into ordering clusters, and each cluster receives service at a different frequency. We modeled the problem as a nonlinear optimization problem and used a heuristic method to search for optimal solutions. We justified the mathematical models with discrete event system simulation. We presented numerical examples of multiple-class, multiple-waves systems and discussed the implications of the results.

The methods in this dissertation rely on certain assumptions. To determine maximum system NSD, we have assumed that cutoff times are equal to deadlines. Although this assumption maximizes system NSD, resulting cutoff times are not optimal. Future research should consider cutoff times as decision variables and determine optimal cutoff times. We disallowed release of a wave if the server is busy. Although this assumption alleviates the need for additional sortation systems and makes the control of flow easier, it requires pickers to wait until all others complete their processing. To increase the productivity of workers, DCs might release waves dynamically, in which case waves are allowed to overlap and there is a requirement for sortation. Future research should address dynamic wave releases so that

worker productivity will be as high as possible and fulfillment operations are designed to maximize service performance.

The validation experiments of multiple-class, multiple-wave systems showed that system NSD could be improved by implementing a more general policy. Future research should develop more powerful models which consider the tradeoff between absolute business requirements and marketing promises. Future research should also consider fluctuating work-force levels in order fulfillment systems. An important issue is to determine an optimal work-force level. New models are needed to determine the minimum required (hourly) work-force levels that guarantee an acceptable level of service for systems that release orders in waves.

Our research should be of interest to practitioners in at least two ways. First, it helps the workers to clearly articulate the service to be delivered. This is likely to help fulfillment systems control the gap between customer expectations and actual performance. Second, it allows DCs to utilize guaranteed delivery as a powerful marketing strategy. Although wave planning is offered by some commercial WMS providers, they do not determine the optimal number of waves nor the optimal release times. Our research could be used by WMS vendors or consultants for better operational planning.

Bibliography

Amazon.com (2013). Local Express Delivery Ordering Deadlines. http://www.amazon.com/gp/help/customer/display.html/?nodeId=201117750.

Bates, D. (2012). The Secret to Online Shopping Revealed: Buy on a Tuesday, And Don't Wait For the January Sales. http://goo.gl/aTo6o.

Bernd S., Fabian, W., Dashkovskiy, S., Schonlein, M., Makuschewitz, T., and Kosmykov, M. (2011). Some Remarks on Stability and Robustness of Production Networks Based on Fluid Models. *Dynamics in Logistics: 2. International Conference, LDIC, Bremen, Germany*, 1:27–35.

Borovkov, A. A. (1964). Some Limit Theorems in the Theory of Mass Service. *SIAM*, 9(4):550–565.

Borovkov, A. A. (1965). Some Limit Theorems in the Theory of Mass Service, II Multiple Channels Systems. *SIAM*, 10(3):375–400.

Bozer, Y., Quiroz, M. A., and Sharp, G. P. (1988). An Evaluation of Alternative Control Strategies and Design Issue for Automated Order Accumulation and Sortation Systems. *Material Flow*, 4:265–282.

Bradley, P. (2007). Smoothing The Waves. DC Velocity.

Chatterjee, S., Slotnick, S. A., and Sobel, M. J. (2002). Delivery Guarantee and the Interdependence of Marketing and Operations. *Production And Operations Management*, 11:393–410.

Dai, J. (1995). On Positive Harris Recurrence of Multiclass Queueing Networks: A Unified Approach via Fluid Limit Models. *Ann. Appl. Probab.*, 5:49–77.

Dai, J. (1999). Stability of Fluid and Stochastic Processing Networks.

Dai, J. and Jennings, O. (2003). *Stability of General Processing Networks, in: Stochastic Modeling and Optimization*. Springer.

de Koster, R. (2004). How to Assess a Warehouse Operations in a Single Tour. Technical report, RSM Erasmus University.

de Koster, R., Le-Duc, T., and Roodbergen, K. J. (2006). Design and Control of Warehouse Order Picking: a literature review. Technical report, Erasmus Research Institute of Management.

de Koster, R., Le-Duc, T., and Roodbergen, K. J. (2007). Design and Control of Warehouse Order Picking: A Literature Review. *European Journal of Operational Research*, 182:481–501.

Dieker, A. (2006). Extremes and fluid queues. Ph.d. Thesis. Universteit van Amsterdam - Netherlands.

Doerr, K. and Gue, K. R. (2011). A Performance Metric and Goal Setting Procedure for Order Fulfillment Operations. Forthcoming in *Production and Operations Management*.

Duenyas, I. and Hopp, W. J. (1995). Quoting Customer Lead Times. *Management Science*, 41(1):43–57.

Franco, M. D. (2006). Batch vs. Wave Picking. *Operations + Fulfillment*, pages 57–58.

Frazelle, E. H. and Apple, J. M. (1994). *Warehouse Operations*. McGraw-Hill, NY.

Gallien, J. and Weber, T. (2010). To Wave or Not to Wave? Order Release Policies for Warehouses with an Automated Sorter. *Manufacturing Service Oper. Management*, 12(4):642–662.

Gilmore, D. (2006). Warehouse Management: To Wave or not to Wave - Part 2. *Supply Chain Digest*.

Gupta, V., Harchol-Balter, M., Wolf, A. S., and Yechiali, U. (2006). Fundamental Characteristics of Queues with Fluctuating Load. *SIGMETRICS Perform. Eval. Rev.*, 34:203–215.

Huffman, J. R. (1988). *Order Picking Systems*. McGraw-Hill, NY.

Internet Retailer (2013). Internet Retailer Top 500 Guide. `http://www.internetretailer.com/`.

Johnson, M. E. and Meller, R. D. (2002). Performance Analysis of Split-Case Sorting Systems. *Manufacturing Service Oper. Management*, 4:258–274.

Kella, O. and Whitt, W. (1996). Stability and Structural Properties of Stochastic Storage Networks. *J. Appl. Prob*, 33:1169–1180.

Kella, O. and Whitt, W. (1998). Linear Stochastic Fluid Networks. *J. Appl. Prob*, 36:244–260.

Liu, J. and Lampinen, J. (2005). A fuzzy adaptive differential evolution algorithm. *Soft Computing*, 9(6):448–462.

Liu, Y. and Whitt, W. (2010). A Fluid Approximation for Large-scale Service Systems. *SIGMETRICS Perform. Eval. Rev.*, 38:27–29.

Manjoo, F. (2012). How Amazons Ambitious New Push for Same-day Delivery Will Destroy Local Retail. `http://goo.gl/pfXUS`.

Morris, J. (2008). American Eagle Outfitters on Cutting Labor Expenses. *Logistics World*.

Newell, G. (1973). *Approximate Stochastic Behavior of n-Server Service Systems with Large n*. Springer.

Oracle Warehouse Management User's Guide (2012). Wave Planning. `http://goo.gl/S3hgO`.

Owyong, M. and Yih, Y. (2006). Picklist Generation Algorithm with Order-Consolidation for Split-Case Module Based Fulfillment Centres. *International Journal of Production Research*, 44-21:4529–4550.

Pang, G. and Whitt, W. (2008). Heavy-traffic Limits for Many-Server Queues with Service Interruptions.

Perry, D. (2007). Continuous Processing Using a Sorter. Technical report, Vargo Adaptive Software LLC.

Perry, O. and Whitt, W. (2011). A Fluid Limit for an Overloaded X Model via An Averaging principle. Technical report, e.

Petersen, C. (2000). An Evaluation of Order Picking Policies For Mail Order Companies. *Production and Operations Management*, 9:319–335.

Rao, U. S., Swaminathan, M. J., and J., Z. (2005). Demand and Production Management with Uniform Guaranteed Lead Time. *Production And Operations Management*, 14:400–412.

Ridley, A. D., Massey, W., and Fu, M. (2004). Fluid Approximation of a Priority Call Center with Time-varying Arrivals. *Simulation Conference, 2003. Proceedings of the 2003 Winter*, pages 69–77.

SAP Business Solutions (2012). Wave Picks. `http://goo.gl/dlkm9`.

Shang, W. and Liu, L. (2011). Promised Delivery Time and Capacity Games in Time-Based Competition. *Management Science*, 57(3):599–610.

So, K. C. and Song, J.-S. (1998). Price, Delivery Time Guarantees and Capacity Selection. *European Journal of Operational Research*, 111(1):28 – 49.

Speaker, R. (1975). Bulk Order Picking. *Industrial Engineering*, 7-12:14–18.

Storn, R. (1996). On The Usage of Differential Evolution for Function Optimization. Technical report, North American Fuzzy Information Processing Society.

Storn, R. and Price, K. (1995). Differential Evolution - A Simple and Efficient Adaptive Scheme For Global Optimization Over Continuous Spaces. Technical report, International Computer Science Institute.

Storn, R. and Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359.

Mathematica Tutorial (2013). Mathematica Tutorial: Numerical Nonlinear Global Optimization. `http://goo.gl/D8cLJ`.

Ursem, R. K. and Vadstrup, P. (2003). Parameter Identification of Induction Motors using Differential Evolution. CEC-2003.

van der Berg, J. P. (2010). *Integral Warehouse Management: The Next Generation in Transparency, Collaboration and Warehouse Management Systems*. Management Outlook Publications.

Ward, A. R. and Bambos, N. (2003). On the Stability of Queueing Networks with Job Deadlines. *Journal of Applied Probability*, 40:293–304.

Whitt, W. and Liu, Y. (2011). Nearly Periodic Behavior in the Overloaded G/D/s+GI Queue. *Stochastic systems*, 1:1–71.

Appendices

Appendix A
Feasibility and Stability Conditions

A set of wave releases is *feasible* if each wave can be completed before the next one begins. Let N be the number of waves in a day, and let w_i denote the time of the i-th wave release time everyday. The first wave on day 1 is at w_1^1. The wave is composed of all orders accumulated until that time, which is equal to $\lambda(w_1^1 - 0)$. The first wave is completed at $w_1^1 + \lambda(w_1^1 - 0)/\mu$. The completion time of the first wave should be less than w_2^1 to be feasible. The second wave which releases all orders accumulated within $[w_1^1, w_2^1]$, is feasible if the first one completes before the second wave. The feasibility condition for wave two is then $w_1^1 + \lambda(w_2^1 - w_1^1)/\mu \leq w_2^1$. Similar expressions can be written for the rest of waves on day 1. On day two, the first wave does not only release orders that arrive that day, but also releases the unworked orders from the first day. The workload of the first wave on day 2 should satisfy $w_1^2 + \lambda(1 - w_N^1 + w_1^2)/\mu \leq w_1^2$ (the length of a day is represented as the interval of [0,1]). On day m, we have the following expressions for feasibility:

$$\begin{aligned} w_1^m + \frac{\lambda(1 - w_N^{m-1} + w_1^m)}{\mu} &\leq w_2^m, \qquad i = 1, \\ w_i^m + \frac{\lambda(w_i^m - w_{i-1}^m)}{\mu} &\leq w_{i+1}^m, \ 1 < i < \mathrm{N}, \\ w_N^m + \frac{\lambda(w_N^m - w_{N-1}^m)}{\mu} &\leq 1, \qquad i = \mathrm{N}. \end{aligned} \tag{A.1}$$

Any system that satisfies Equations A.1 is feasible.

Stability conditions for queueing models can be found in Dai (1999). In our context, the system is *stable* if work-in-process inventory remains at a constant level for a given initial fluid level. We start with no beginning inventory, I(0)=0. The server does not work on the orders that arrived after w_N. Total number of orders that arrive between the last release and the end of the first day is $\lambda(1 - w_N)$. For an N wave system, the last wave has the following workload: $\lambda(w_N - w_{N-1})$. The last wave may or may not finish at one, but works on $\mu(1 - w_N)$ orders. If the last wave completes the work on the second day, $\{\lambda(w_N - w_{N-1}) - \mu(1 - w_N) > 0$. Otherwise there will be zero work left from the interval $[w_{N-1}, w_N]$. Then work-in-process inventory at end of day 1 is equal to

$$I_1 = \lambda(1 - w_N) + \max\{\lambda(w_N - w_{N-1}) - \mu(1 - w_N), 0\}.$$

For the second day, work-in-process inventory is equal to

$$\begin{aligned} I(2) =& I(1) + \max\{\lambda(w_N - w_{N-1}) - \mu(1 - w_N), 0\} \\ =& \lambda(1 - w_N) + 2\big(\max\{\lambda(w_N - w_{N-1}) - \mu(1 - w_N), 0\}\big). \end{aligned}$$

We define WIP for day m as:

$$\begin{aligned} I(m) =& I_{m+1} + \max\{\lambda(w_N - w_{N-1}) - \mu(1 - w_N), 0\} \\ =& \lambda(1 - w_N) + m\big(\max\{\lambda(w_N - w_{N-1}) - \mu(1 - w_N), 0\}\big). \end{aligned}$$

Note that, in a steady state $\lim_{m\to\infty} I(m) = \lambda(1 - w_N)$ only if $\lambda(w_N - w_{N-1}) = \mu(1 - w_N)$, otherwise the system is feasible but not stable.

Now, we establish the stability condition when $I(0) \neq 0$. Because the first wave has λw_1+I(0) orders to process before the first release of the next day, this quantity should not exceed the available capacity:

$$\begin{aligned} w_1 + \frac{\lambda w_1 + I(0)}{\mu} \leq& 1, \\ I(0) \leq& \mu - w_1(\lambda + \mu). \end{aligned}$$

Appendix B
Comparison of Search Methods and Details of Differential Evolution

We solve multiple class, multiple wave system problems with MATHEMATICA's built in function `NMaximize`. The function provides a number of search algorithms including direct search methods such as Nelder-Mead, differential evolution, simulated annealing, and random search (Mathematica Tutorial, 2013).

Although Storn (1996) showed that differential evolution is an appropriate method for solving real-valued, non-differentiable, multi-modal objective functions with non-linear constraints, we first run preliminary tests to assess the performance of the method. The purpose of the preliminary tests is twofold: selecting the best method for our purpose, and assess the quality of the selected method. We test nine problems to compare the performances of different numerical methods: differential evolution, simulated annealing, Nelder-Mead, and random search. Note that MATHEMATICA's built-in function `NMaximize` supports all these methods, therefore we test each one by specifying the method within the function.

Based on the problem structure, `NMaximize` chooses which method to use automatically; however, it also lets users to select a specific method (and its parameters) for their purposes. We start by running a number of preliminary tests to assess the performance of candidate optimization methods given in `NMaximize`. (Note that there are other possible heuristic methods such as particle swarm optimization which is not included in the function.) We compare the candidate methods as follows. We first transformed multiple class problem into a single class system by setting a common deadline ($d_1 = d_2 = \ldots = d_K$) and then comparing the results with analytical solutions. During the preliminary tests, we assume 2–4 waves and test the performance of the methods at three levels of utilization: $\rho = 0.5,\ 0.75,\ 0.95$. We limit computational time with 600 seconds.

Because our first task is to select an appropriate method, we use the default parameter sets of the methods during the preliminary tests. (Method specific parameters can be found in Mathematica Tutorial, 2013.) The results of the preliminary test are given in Table B.1 in which best solutions are shown with asterisks.

Table B.1 suggests that differential evolution is an appropriate method for the problems we address. Simulated annealing (SA) performed very poorly when compared to the other methods (especially when $N = 3$ and as utilization increases). Although random search (RS) finds the optimal solutions for all values of utilization when $N = 2$, this method did not find any solution within the specified time limit for $N > 2$.

Differential evolution attempts to find global solution by executing three main routines: mutation, recombination, and selection. The first step is to generate an initial solution which requires $\mathcal{D}$ dimensions to define the variables. (For a K class, N wave system, it requires $N(3K + 1)$ dimensions.) In this step, randomly generated points between their bounds constitute an initial feasible solution S. The value of the objective function in the initialization step is denoted by f_1.

Table B.1: Comparison of objective function values in the preliminary run.

ρ	Waves	Analytical (%)	DE (%)	SA(%)	NM(%)	RS(%)
0.5	2	83.3	83.3*	83.3*	83.3*	83.3*
	3	92.9	92.9*	32.2	91.4	-
	4	96.7	96.7*	96.7*	96.7*	-
0.75	2	67.9	67.9*	55.2	67.9*	67.9*
	3	81.8	81.3*	31.6	79.7	-
	4	88.4	88.4*	87.6	88.4*	-
0.95	2	53.7	53.6	40.0	52.6	53.7*
	3	69.9	69.5*	35.3	69.5*	-
	4	78.1	78.0*	77.8	77.8	-

After specifying the control parameters (explained below) and generating the initial solution, the algorithm executes its routines to search the solution space. The algorithm iteratively repeats cycles of routines until it is terminated by stopping criteria. By default, these criteria are based on both the convergence of variables and objective function value in the successive iterations. We force the algorithm to terminate if the objective value difference between two consecutive iterations is less than 10^{-6} or the computational time is greater than 600 seconds.

Because differential evolution uses sets of solutions in each generation, it requires a population size parameter P_s. In each generation, a population size of P_s is maintained. We denote the i^{th} solution in generation G by S_{iG}. A mutation routine is executed in the next step. Note that the set of solutions $S_{iG}, i \in P_s$ contains vectors of variables x_{iG}. For each target vector i, mutation operator first selects three random points: x_{uG}, x_{wG}, and x_{zG}. Then the operator generates trial vector x_{TG} based on the selected points. This calculation is done by adding the scaled down (by a factor α) difference of the two vectors to the first one: $x_{TG} = x_{uG} \oplus \alpha \times [x_{wG} - x_{zG}]$ (operators "$\oplus$" and "$\times$" denote vectorial summation and scalar multiplication operators). Because mutation operation replaces all points (as a vector) in the solution by new points in each generation, it expands the solution space. Mutation routine generates the trial vector and with the crossover routine, the algorithm replaces some part of the target vector with this new vector. To crossover, a uniform generated random variable u is compared with the crossover constant p_c. If $u < p_c$, then recently generated trial vector replaces the target vector after the crossover point uP_s. Note that DE uses operators similar to the genetic algorithm, however its computational power comes from the ability to replace all points in the solution with new points in each generation. The last routine is the selection which actually is a greedy approach to replace the solution. In the routine, the generated vector is replaced with the existing one only if it produces a feasible and a better solution, otherwise it preserves the existing vector (we denote the existing vector by $x_{jG_{-1}}$). Pseudo-code for the differential evolution is given in Figure B.1.

The structure of the algorithm in `NMaximize` is basically same as general DE algorithm; however default control parameters in the function are different. Storn and Price (1995) and Liu and Lampinen (2005) recommended the following control parameters for the algorithm: $P_s = 10\mathcal{D}, p_c \in [0.8, 1], \alpha \in [0.5, 1]$. (Recall that, $\mathcal{D} = N(4K + 1)$ for the problems we

Require: $\mathcal{D}, P_s, p_c, \alpha$
Initialize x_{j1} randomly between its bounds (x_{j1}^L, x_{j1}^U)
Set the initial feasible solution: $S \leftarrow S_{01}$
Evaluate the objective function f_1
repeat
  for $i = 1, i \leq P_s, i++$ **do**
    Generate a population S_{iG}
    Randomly select three distinct points in S_{iG}: x_{uG}, x_{wG}, x_{zG}
    Determine a trial point with mutation: $x_{TG} = x_{uG} \oplus \alpha \times (x_{wG} - x_{zG})$
    Find a cross over point:
    Generate a uniform random variable $u \sim U[0, 1]$
    if $u \leq p_c$ **then**
      Cross over point index $i = u$
    end if
    Determine the candidate solution S_C and evaluate its objective value f_C
    if $x_{jG}, j \in N$ is not feasible **then**
      Penalize f_C by setting $x_{jG} = x_{jG_{-1}}$
    else
      if f_C is greater than f_i **then**
        Update the solution: $S \leftarrow S_C$
      end if
    end if
  end for
until Termination criterion is met.
return $S = \{w_1, \ldots, w_N, x_{11}, \ldots, x_{KN}, y_{11}, y_{KN}\}$

Figure B.1: Differential Evolution Pseudo-code

address.) Ursem and Vadstrup (2003) claimed that using $p_c = 0.2, F = 0.35$ results in faster convergence.

The preliminary results in Table B.1 indicate that DE performed worst for systems in which there are three waves at utilization levels $\rho = 0.75, 0.95$. Consequently, we decide to tune the control parameters of the method and hopefully improve the results. `NMaximize` chooses the same population size as in Storn and Price (1995) and Liu and Lampinen (2005) with different crossover and scale factors: $p_c = 0.5, \alpha = 0.6$. Although these two parameters are chosen differently, because the performance of the differential evolution algorithm is more sensitive to the choice of scaling constant than to the choice of crossover constant (Storn and Price, 1995; Storn and Price, 1997), we particularly focus on tuning the parameter α.

We adjust the parameter $\alpha \in [0.1, 1]$ with increments of 0.1 in the algorithm. Other parameters remained unchanged during the tuning process. We measured how much the method deviates from the optimal objective value (in percentage) at different levels of α. We observed that the mean squared percentage error is minimized when $\alpha = 0.6$. Consequently, we decide to preserve the control parameter α equal to 0.6 for larger problems.