



ELSEVIER

Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

Computers & Operations Research 33 (2006) 132–157

computers &
operations
research

www.elsevier.com/locate/cor

Comparative approaches to equipment scheduling in high volume factories[☆]

Xinhui Zhang^{a,*}, Jonathan F. Bard^b

^a*Department of Industrial Engineering, Wright State University, 207 Russ Engineering Center, 3640 Colonel Glen Hwy, Dayton, OH 45435, USA*

^b*Graduate Program in Operations Research & Industrial Engineering, The University of Texas, 1 University Station, C2200, Austin, TX 78712, USA*

Available online 13 August 2004

Abstract

Mail processing and distribution centers (P&DCs) are large factories that accept, sort, and sequence mail in preparation for delivery. A central problem in these facilities is how to schedule the equipment over the day to ensure batch production under tight equipment and workforce constraints. The problem falls into the general category of multi-level lot sizing and is notoriously difficult to solve. No exact algorithms exist that are efficient and can consistently provide high-quality solutions.

In the paper, we investigate two specialized approaches for solving this problem. The first is a piece-by-piece LP-based heuristic and the second is a Benders decomposition. The heuristic uses the LP fractional solution as a target and attempts to find an integer solution that is as close to it as possible by minimizing the L_1 -norm. The procedure consists of solving the LP relaxation of the original model and two reduced mixed-integer programs. The process is similar to what has been called ‘piece-by-piece decomposition’ in nonlinear programming. While the goal there is to find better initial starting points for the nonlinear code, our goal is to improve solution quality.

Computational testing using data provided by the Dallas P&DC confirmed the superiority of the piece-by-piece LP-based heuristic with respect to branch and bound, which proved to be computationally expensive, and the Benders algorithm which took several hours to find feasible solutions. In general, the LP-based heuristic was an order of magnitude faster than branch and bound and provided solutions that were able to process more than 99.75% of the daily volume.

© 2004 Elsevier Ltd. All rights reserved.

[☆] This work was supported in part by the National Science Foundation under Grant #DMI-0218701.

* Corresponding author.

E-mail addresses: xinhui.zhang@wright.edu (X. Zhang), jbard@mail.utexas.edu (J.F. Bard).

Keywords: Multi-level lot sizing; Piece-by-piece decomposition; LP-based heuristic; Equipment scheduling; Postal operations; Integer programming; Benders decomposition

1. Introduction

The United States Postal Service (USPS) operates a network of approximately 275 major mail processing and distribution centers (P&DCs) around the country. These facilities contain a large variety of automated equipment for collecting, processing and distributing the mail, and serve an interface between the local service area and the nation. It has long been a challenge to efficiently manage the use of this equipment. The problem is complicated by over 40 different mail categories for letters and flats, the interactions of the corresponding mail streams, tight resource constraints, and the requirement to ensure batch production (e.g., see Berman et al. [1], Jarrah et al. [2]).

Mail arrives at a facility over a 24-h period in highly varying patterns. The average volume at a medium-size center is more than 5,000,000 pieces per day, and depending on its characteristics, each piece undergoes several operations including stamp cancellation, address identification, and barcoding. It is then sorted and perhaps delivery route sequenced before leaving the facility. Although each type of equipment is designed to perform a basic task, such as reading an address, machines must be reset periodically as the category of mail changes. To reduce unproductive time, it is common to process the mail in batches as is typically done in a job shop.

There are two major factors that limit the capacity of a facility. The first is the amount of equipment available and the second is the size of the workforce. For our purposes, the size of the workforce is measured by the total number of full-time shifts available per day. A full-time shift is defined as 8 consecutive hours with predetermined start and end times, and must satisfy certain legal and contractual restrictions.

Zhang and Bard [3] were the first to develop a comprehensive framework for solving the equipment scheduling problem at P&DCs. Their approach called for the sequential solution of three large-scale mixed-integer programs (MIPs). The first is aimed at finding the maximum amount of mail that can be processed in a day; the second determines the minimum number of shifts required to operate the equipment subject to a user-specified percentage of the maximum volume being processed; and the third minimizes a combination of machine startups and weighted working periods to ensure batch production. The third model was the most difficult of the three to solve, primarily due to the presence of the workforce capacity constraints. Computation times on the order of hours were required to achieve a 2% or 3% optimality gap when a commercial branch and bound code was used.

The purpose of this paper is to investigate two specialized approaches to solving the MIP associated with the batch production problem—model 3. The first approach is an LP-based heuristic that uses the fractional solution as a target and attempts to find an integer solution that is as close to it as possible. This is achieved by minimizing the sum of the absolute deviations from the LP solution; i.e., minimizing the L_1 -norm. The solution process is divided into three parts. First, the LP relaxation is solved and then two additional MIPs are solved in sequence, each with additional constraints and variables. The aim is to refine the solution obtained in the previous stage. The methodology is similar to piece-by-piece decomposition designed to find good initial solutions to nonlinear optimization problems [4]. Our computations show that the heuristic is able to find near-optimal solutions an order of magnitude faster than branch and bound.

The second approach investigated is based on Benders decomposition. Because the results were not as good, we give fewer details on the implementation.

Batch production in P&DCs differs in several ways from batch production in the processing industries. In chemical manufacturing, for example, activity times are generally assumed to be constant regardless of the batch size, although minimum and maximal production quantities are often established to ensure safety. In mail processing, the time of an operation is determined by several factors including the volume, the characteristics of the mail piece, and the machine parameters. Moreover, the primary objective in chemical manufacturing is to minimize the total makespan while in mail processing the objective is to minimize the number of transitions and working periods. For more discussion of batch production in the chemical, food, and refinery industries, see Blömer and Günther [5] and Reklaitis et al. [6].

The remainder of the paper is organized as follows. In the next section, we give an overview of mail flow and equipment scheduling at P&DCs. In Section 3, we present the mixed-integer programming model, review the literature on related problems, and discuss our previous computational experience. The LP-based heuristic is discussed in Section 4 and the Benders decomposition algorithm in Section 5. Computation results are presented in Section 6 and several concluding remarks are offered in Section 7.

2. Equipment scheduling at mail processing centers

Fig. 1 displays the major operations and mail flows associated with letter processing at the Dallas facility. Here, arcs (arrows) represent mail flows and nodes (blocks) represent the processing operations or jobs (the operation numbers and the equipment used to run these jobs are also listed). All mail first arrives at the receiving area and is broken down into one of several pre-defined groups based upon origination, physical size, quality of address, and degree of previous processing. Each group has a different composition and follows different routes (series of operations) through the facility. The major groups are (1) local collection mail gathered from the street boxes, and (2) incoming mail partially processed at another P&DC.

The local collection mail can be classified as stamped, metered, pre-barcoded, and manual. The manual qualifier refers to letters that, due to peculiar physical characteristics (e.g., irregular shape), get culled as they travel through the belt feeding mail into the automation system. The stamped letters are first sent to the AFCS (advanced facer-canceller system), operation 015, where their stamps are located and canceled. Pre-barcoded letters, machine-printed envelopes and handwritten pieces are also separated at this stage for faster processing with the automation equipment.

The machine-printed envelopes as well as the metered arrivals are fed into MLOCs (multiline optical character readers), operation 881 in the figure, where the machines attempt to read the full address, and if successful, print a 3-, 5-, or 11-digit barcode on the envelope. The MLOCs are also equipped with a limited number of bins to perform primary sorting. Output to certain other P&DCs is dispatched promptly while the remaining mail pieces require a secondary sorting, operation 892, on a DBCS (delivery barcode sorter) to channel them to more specific destinations.

The handwritten letters separated at the AFCS and machine-printed envelopes unreadable by an MLOC are sprayed with a florescent ID on the back and sent to a remote barcode sorter (RBCS), operation 271, to obtain a barcode. While the letters are being transferred, video images of the address fields are transmitted to a remote encoding center (the dashed REC block in Fig. 1) to be identified. The barcodes found are then retrieved and sprayed when the letters enter the RBCS to be sorted and dispatched.

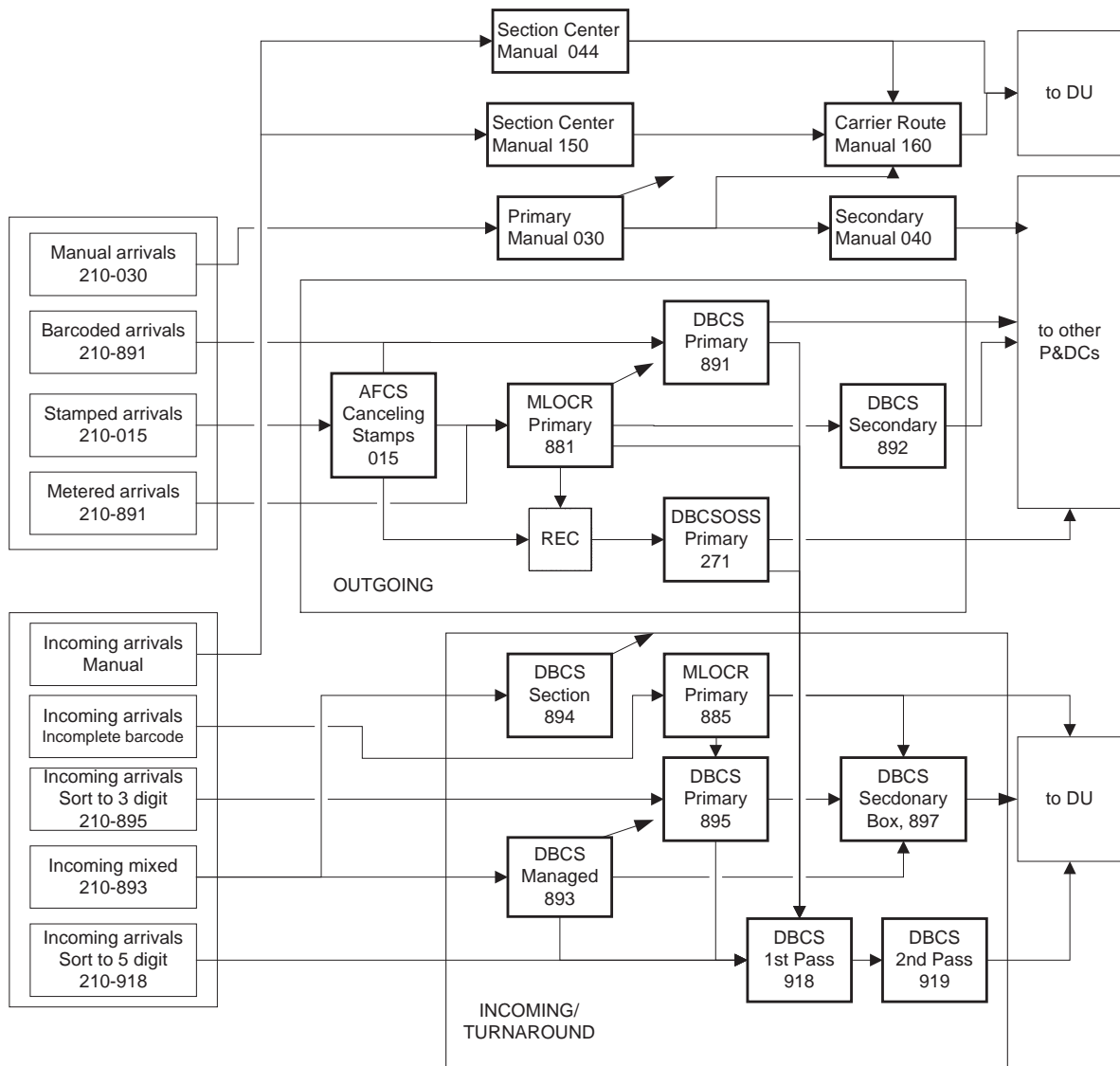


Fig. 1. Major mail flows in a P&DC.

The pre-barcoded arrivals with no stamps are sent directly to a DBCS, operation 891, to be sorted and dispatched. Finally, rejects that failed in the above operations, as well as the letters that are not fed into the automation system, are sent to the manual section, which consists of a primary (operation 030) and secondary (operation 040), to be separated and dispatched.

Mail sent to the local service area can arrive already sorted to either 3 or 5 digits. The 3-digit sorted letters are first sent to a DBCS, operation 895, to receive two additional digits. Then, together with the 5-digit sorted letters, are dispatched to firms through operation 897 or to delivery units (DUs) through delivery point sequencing operations. These operations are designed to reduce the workload of the carriers

in the DUs by sorting the letters to each carrier route, operation 918, and then to walk order for each route, operations 919. The full letter automation process contains more than 20 different arrival streams, 30 operations, and more than 200 flows within the facility.

As mentioned, Zhang and Bard [3] modeled the equipment scheduling problem as a MIP with multi-criteria and used a three-stage approach to find solutions. The need for service organizations like the USPS to balance several objectives at once is one factor that distinguishes them from manufacturing. While it is common in manufacturing to have a single objective, such as maximizing throughput or minimizing production costs, many service organizations are driven by the conflicting objectives of minimizing personnel costs and maximizing customer satisfaction. The fact that a flexible workforce can lead to sharply reduced personnel costs (e.g., see Bard et al. [7]) introduces an additional complication that manufacturers do not have to face because shifts and weekly schedules are, for the most part, standardized. Unfortunately, it is not possible to develop a tractable P&DC equipment scheduling model with the objective of minimizing personnel costs. Therefore, it is necessary to separate the two issues, giving rise to the three-stage approach.

In the first stage, we seek a schedule that processes as much mail as possible while satisfying all operational, technological, and resource (machine only) constraints. This is done by minimizing the inventory remaining at the end of the day and is equivalent to maximizing the amount of mail that is processed. In the second stage and in light of the solution obtained in the first stage, we seek a schedule that minimizes the number of 8-h shifts needed in a day. This is approximately equivalent to minimizing the daily personnel costs. Finally, we solve a third optimization problem aimed at reducing the number of startups to ensure batch production subject to remaining mail volume and both equipment and workforce capacity constraints. This problem is the focus of the current paper and is shown to be NP-hard in Appendix A.

3. Mathematical model

An equipment schedule is the specification of the start times for each operation to be performed in a facility, the machine type to be used for that operation for a given start time, and the corresponding end times. Looking at the mail flows in Fig. 1, such a schedule would indicate, for example, how many MLOCs to set up for outgoing primary operation 881 and how many to set up for incoming primary operation 885, as well as the time intervals during which each is to be run. The objective in developing such a schedule is to minimize some measure of total machine setups and overall machine usage. At this point, we are not concerned with assigning batches of mail to specific machines. A post-processor is used to handle this aspect of the problem (see Zhang and Bard [8]).

In the development of the model we make use of the notation given below. Because the problem decomposes by day, it is necessary to specify initial inventory conditions for each day in the planning horizon.

Indices

i, o	input and output mail streams
n, p, s	nodes (a unique operation is associated with each node)
g	machine groups
t	time periods; $t \in T = \{1, \dots, 48\}$; $t = 1$ corresponds to 7:00 a.m.

k worker categories
 f shifts

Sets

G machine groups, $G = \{\text{AFCS, MLOCR, BCS-OSS, DBCS, DBCS-OSS}\}$
 $G(n)$ machine groups capable of performing the operation at node n
 I, O input and output mail streams for the facility
 N nodes
 K worker categories
 F shifts
 $G(k)$ machine groups a worker in category k can operate
 $N(g)$ nodes whose operations can be performed by machines in group g
 $P(n)$ nodes immediately preceding node n
 $S(n)$ nodes immediately succeeding node n
 $I(n)$ input mail streams to node n
 $O(n)$ output mail streams from node n
 $T(n)$ periods during which an operation at node n can be performed
 $T(i)$ periods during which mail arrivals i may be processed; generally,
 $T(i) \subseteq T(n)$ where n is the node that accepts the arrivals of input stream i
 $T(o)$ periods during which mail stream o can be dispatched from the facility;
generally, $T(n) \subseteq T(o)$ where n is a node that dispatches to output stream o

Data and parameters

u_n amount of mail left unprocessed at node n from the previous day
 $a_i(t)$ amount of external mail of type $i \in I$ arriving at the facility during period t ; $t \in T(i)$
 $m_g(t)$ number of machines available in group g during period t
 ρ_n processing rate for operation at node n (pieces/period)
 l_{pn} time in periods required to transfer mail processed at operation p to operation n
 f_{ns} fraction of mail processed at node n that is sent to successor node s or dispatched from the facility, $s \in N \cup O$
 τ_1, τ_2 the amount of time required to start up or clear a machine respectively; currently both times are set at 10 minutes
 s_f, e_f starting and ending periods for shift f
 r_{kg} number of workers of type k required to run a machine in group g
 ψ_{1n} amount of ending inventory allowed for each operation n at the end of the day
 ψ_2 total number of shifts available during the day
 ε small positive weight
 θ objective function weight

Decision variables

$v_n(t)$ inventory level of mail at node n at the end of period t ; $t \in T$
 $w_{ng}(t)$ amount of mail processed at node n by machine group g during period t ; $t \in T(n)$

- $Y_{ng}(t)$ number of machines devoted to the operation that is performed at node n by machine group g during period t ; $t \in T(n)$
- $Z_{ng}^1(t)$ number of startups in machine group g for operation n at the beginning of period; $t \in T(n)$
- $Z_{ng}^2(t)$ number of clearance operations performed at node n in machine group g at the end of period t ; $t \in T(n)$
- ω_{kf} number of workers of type k assigned to shift f

The mathematical formulation is as follows:

$$\text{Minimize } \sum_{n \in N} \sum_{t \in T(n)} \sum_{g \in G(n)} [\theta(1 - \epsilon t)Y_{ng}(t) + (1 - \theta)Z_{ng}^1(t)] \tag{1a}$$

subject to

$$u_n + \sum_{i:i \in I(n), 1 \in T(i)} a_i(1) - \sum_{g \in G(n)} w_{ng}(1) = v_n(1) \quad \forall n \in N, \tag{1b}$$

$$v_n(t-1) + \sum_{i:i \in I(n), t \in T(i)} a_i(t) + \sum_{p \in P(n)} f_{pn} \sum_{g \in G(p)} w_{pg}(t-l_{pn}) - \sum_{g \in G(n)} w_{ng}(t) = v_n(t) \tag{1c}$$

$$\forall t \in T \setminus \{1\}, n \in N,$$

$$w_{ng}(t) + \frac{\tau_1}{30} \rho_n Z_{ng}^1(t) + \frac{\tau_2}{30} \rho_n Z_{ng}^2(t) \leq \rho_n Y_{ng}(t) \quad \forall g \in G(n), t \in T(n), n \in N, \tag{1d}$$

$$\sum_{n \in N(g)} Y_{ng}(t) \leq m_g(t) \quad \forall t \in T, g \in G, \tag{1e}$$

$$Z_{ng}^1(t) - Z_{ng}^2(t-1) = Y_{ng}(t) - Y_{ng}(t-1) \quad \forall t \in T(n), n \in N(g), g \in G, \tag{1f}$$

$$v_n(48) \leq \psi_{1n} \quad \forall n \in N, \tag{1g}$$

$$\sum_{g \in G(k)} r_{kg} \sum_{n \in N(g)} Y_{ng}(t) \leq \sum_{f:s_f \leq t \leq e_f} \omega_{kf} \quad \forall t \in T, k \in K, \tag{1h}$$

$$\sum_{k \in K} \sum_{f \in F} \omega_{kf} \leq \psi_2, \tag{1i}$$

$$v_n(t), w_{ng}(t) \geq 0 \quad \forall t \in T(n), n \in N, g \in G(n), \tag{1j}$$

$$Y_{ng}(t) \geq 0 \text{ and integer } \quad \forall g \in G(n), t \in T(n), n \in N, \tag{1k}$$

$$\omega_{kf} \geq 0 \quad \forall k \in K, f \in F, \tag{1l}$$

$$Z_{ng}^1(t), Z_{ng}^2(t) \geq 0 \quad \forall t \in T(n), n \in N, g \in G(n). \tag{1m}$$

The objective function (1a) is comprised of two terms weighted by the parameter $\theta \in [0, 1]$. The first term, itself, is the weighted sum of working periods. To gain a better understanding of this term, let us

first look at the unweighted sum of working periods.

$$\sum_{n \in N} \sum_{t \in T(n)} \sum_{g \in G(n)} Y_{ng}(t).$$

Given the volume of mail associated with an operation, the smaller the value of the total number of working periods, the more likely that a machine will be running close to its capacity. The weighted sum is obtained by multiplying $Y_{ng}(t)$ by the coefficient $(1 - \varepsilon t)$ which decreases with time (previous testing found that $\varepsilon=0.01$ provided the best results). More specifically, when t is small, the coefficient is relatively large (close to 1) so processing in earlier periods is penalized more than processing in later periods. This has the effect of pushing an operation to as late in the operation window as possible, and implicitly serves to shorten the working intervals of an operation, a quality preferred by the managers in the facility.

The second term in (1a) is the total number of startups. A startup occurs whenever a machine changes operations from one period to the next. Likewise, a machine must be cleared whenever a different operation is scheduled on it in the upcoming period. Minimizing the sum of startups reduces the time that the machines are switched on and off, and is designed to ensure batch production. Note that the number of startups equals the number of clearances so it is not necessary to include both in the objective function.

The constraints can be classified as follows.

Multi-level inventory balance constraints (1b) and (1c): Constraint (1b) stipulates that for each operation n at the beginning of the day in period 1, the mail remaining from the previous day, plus the sum of external arrivals, minus the sum of mail processed during this period, equals the ending inventory on the right-hand side. Similarly, constraint (1c) stipulates that for all other time periods, the starting inventory, plus the sum of external arrivals, plus mail transferred from predecessor nodes, minus the sum of mail processed during this period, equals the ending inventory. The transfer delay is typically set so that $l_{pn} \geq 1$.

Production capacity limitation (1d): Constraint (1d) stipulates that in period t , the workload at node n allocated to group g , plus the lost capacity during startup and clearance, must be less than or equal to the processing capacity of the machines in group g assigned to operation n at time t .

Machine capacity limitation (1e): Constraint (1e) ensures that the number of machines in group g operating in time period t is less than or equal to the number of machines available in group g .

Definition of startup and clearance activities (1f): Constraint (1f) defines the startup and clearance activities, each of which takes a certain amount of time, denoted by τ_1 and τ_2 , respectively. As mentioned, the lost production capacity associated with these activities is taken into account in constraint (1d).

Maximal ending inventory allowed (1g): Constraint (1g) specifies that the maximum inventory that is allowed to remain unprocessed at the end of the day.

Shift covering constraints (1h): Full-time shifts are used in the model to represent the size of the workforce. Constraint (1h) ensures that the number of active shifts during period t is sufficient to match the number required to run the equipment.

Shifts capacity limitation (1i): Constraint (1i) specifies the maximal number of full-time shifts available in a day.

A few remarks on the definition of the variables follow. First, we note that the shift variables ω_{kf} can be treated as continuous rather than discrete. This is shown in Proposition 1 Appendix A. Second, because we are trying to minimize a linear function of the startup variables, for a given integer value of $Y_{ng}(t)$, $Z_{ng}^1(t)$ and $Z_{ng}^2(t)$ will be integer valued in any feasible solution. This implies that only the production variables $Y_{ng}(t)$ need to be explicitly defined as integer.

Table 1
Breakdown of model for medium size facility

	Model components	Size
Variables	Inventory variable, $v_n(t)$	1440
	Production volume, $w_{ng}(t)$	712
	Production variable $Y_{ng}(t)$, integer	712
	Startups and clearances, $Z_{ng}^1(t), Z_{ng}^2(t)$	1424
	Shift variable, ω_{kf}	192
	Total	4480
Constraints	Inventory constraints (1b) and (1c)	1440
	Production capacity constraint (1d)	712
	Machine capacity (1e)	192
	Definition of $Z_{ng}^1(t), Z_{ng}^2(t)$, (1f)	744
	Ending inventory constraint (1g)	30
	Shift cover constraint (1h)	96
	Shifts capacity constraint (1i)	1
Total	3215	

3.1. Model size and analysis

To gain an appreciation of problem (1a)–(1m), consider a medium-size facility such as the Dallas P&DC. In a typical day, the facility performs 30 major letter operations over 48 time periods using more than 50 machines. The corresponding model contains 3768 continuous variables, 712 general integer variables and 3215 constraints. The breakdown of the model's components is given in Table 1.

3.2. Solution approaches

The P&DC equipment scheduling problem falls into the general category of multi-level capacitated lot sizing that is common in both job shops and flow shops. Here, multi-level refers to the fact that the input of an operation depends not only on external arrivals, but also on flows from upstream operations. To the best of our knowledge, there are no specialized algorithms for solving this problem exactly.

There are, however, several exact algorithms for the single-level, single-item problem which primarily rely on problem reformulation and the use of polyhedral theory to derive valid inequalities to tighten the LP relaxation (e.g., see Pocket and Wolsey [9], Wolsey [10]). Computationally, their performance has been mixed because they are only able to solve small problems. Unfortunately, the theory that supports cut generation is not applicable to multi-level problems so researchers have resorted to heuristics to obtain solutions.

Maes et al. [11] proposed three LP-based heuristics that were applicable to assembly and linear product structures. Each starts with a solution to the LP relaxation and differs only in the way they round the fractional production variable to 0 or 1. Kuik et al. [12] extended these heuristics and embedded them in both simulated annealing and tabu search algorithms, and were able to achieve considerable improvement in performance.

Table 2
Comparison of LP and IP solutions for operation 015

Time	18	19	20	21	22	23	24	25	26	27	28	29	30
LP	4.84	4.84	4.84	4.84	4.84	4.84	4.84	4.84	4.84	4.84	4.84	4.84	4.84
IP	6	6	5	5	5	5	5	5	5	5	5	4	4

Tempelmeier [13] developed a Lagrangian-based heuristic to solve the multi-level problem that has a general product structure. In his approach, both the multi-level inventory constraints and the capacity constraints were relaxed to obtain single-item lot sizing subproblems that could be solved efficiently with an $O(|T| \log |T|)$ algorithm. Feasible solutions were then constructed by a sophisticated finite scheduling procedure that shifted production away from overloaded periods.

Of these approaches, the Tempelmeier's Lagrangian heuristic seems to be the most effective. Nevertheless, several difficulties arise if such an approach is to be applied to our problem. First, both the multi-level inventory constraints (1b) and (1c) and the capacity constraints (1e), (1h) and (1i) need to be relaxed to create single-item subproblems. As seen in Table 1, these constraints comprise more than 1/2 of the total so intuitively at least, slow convergence would be expected. Second, the existence of startup and clearance times in the subproblems calls for an algorithm of complexity $O(|T|^4)$ rather than $O(|T| \log |T|)$ as pointed out in Vanderbeck [14]. This sharply increases the overall computational burden and hence undermines the efficiency of the Lagrangian approach.

To dramatize the difficulty of the P&DC equipment scheduling problem, we note our previous computational experience with XPRESS, a state-of-the-art, general branch and bound code. On average, XPRESS took anywhere from several minutes to a half-hour to find integer solutions, depending on the tightness of the shift resources. Also, it was unable to improve the lower bound after exploring tens of thousands of nodes. Our experience with CPLEX was nearly identical. Because the success of branch and bound relies heavily on good lower bounds for fathoming, this fact alone calls into question the ability of commercial codes to solve our scheduling problem.

4. A piece-by-piece LP-based heuristic

In this section, we present an LP-based heuristic for deriving feasible solutions from (1a) to (1m). The methodology is relatively simple and, as will be seen later, produces much better results than branch and bound with no visible sacrifice in solution quality.

4.1. Motivation

For MIPs that are not combinatorial in nature, it is often possible to construct high-quality integer solutions from the LP relaxation. In our case, the linear programming model can be solved within 1–2 s and captures the essence of mail flow between operations, the external arrival profiles, and the shared capacity. As such, its solution gives realistic information on a feasible production plan. To see this, let us look at the similarities between the fractional LP solutions and integer solutions to the original problem.

Tables 2 and 3 present solutions for two major operations over a portion of a day (a typical Monday at the Dallas P&DC). The integer solution was the best one found within a time limit of 600 s using the

Table 3
Comparison of LP and IP solutions for operation 881

Time	20	21	22	23	24	25	26	27	28	29	30	31	32	33
LP	3.51	3.51	3.51	3.74	3.74	3.74	3.74	3.74	3.74	3.74	3.17	2.07	0.60	0.69
IP	3	4	4	4	4	4	4	4	4	4	3	2	1	1

branch and bound code. The “Time” row refers to the time period in the day, and the “LP” and “IP” rows record the values of the production variables; i.e., the number of machines open in each solution.

As we can see, the distance between the two solutions is relatively small in most time periods. For operation 015 in Table 2, the LP solution gives a series of values of 4.84 over 13 periods; i.e., 4.84 machines to be used to process this operation for 13 consecutive periods or 6.5 h. While this solution is not feasible because it is fractional, the accompanying integer solution suggests that 6 machines be opened for the first two periods, then 5 for 9 periods, and finally 4 in the remaining 2 periods. This solution is reasonably close to the rounded schedule of running 5 machines for 13 periods.

The results presented in Table 3 for operation 881 are even more telling. The distance between the two solutions is less than 0.5 in all but the first period in which the integer solution is 3 rather than 4, the nearest integer to 3.51.

The above observation suggests the use of the linear programming solution as a basis for constructing good feasible solutions. However, the first idea that comes to mind, a procedure to round the fractional solutions to the nearest integer values is not likely to be effective. Without any consideration of shared machine or shift resources, such a procedure could easily lead to schedules with overloaded production periods that require more machines or shifts than available.

A better approach is to take the shared capacity constraints into consideration while constructing an integer solution from the relaxed solution. One way to do this is by defining an optimization problem whose objective is to minimize the sum of the absolute deviations of the integer component of the solution from a target (LP) solution. Loosely speaking, the objective is to find a feasible solution as “close” as possible to the LP solution.

From a computational point of view, the use of a target solution is believed to create an asymmetry in the problem structure that helps reduce the size of the branch and bound tree. Morton et al. [15], for example, report substantial computational improvements with the use of the L_1 -norm. We believe that one of the reasons why convergence is often so rapid is because the search is indirectly limited to a small neighborhood of the target solution. That is, we are looking for a local rather than a global solution.

For large-scale integer programs, however, minimizing some measure of the deviation from a target may be more difficult than solving the original problem directly because of the additional constraints and variables needed to linearize the deviation terms. To circumvent this potential hurdle, we propose a three-step approach in which the target solution is updated at each step. The details follow.

4.2. Procedure

Let us define three additional parameters:

β_1 “small” weight associated with the original objective function; $\beta_1 \geq 0$

β_2 “small” weight associated with the objective of minimizing the ending inventory; $\beta_2 \geq 0$

ρ average processing rate over all operations.

Step 1: Solve the LP relaxation of the original problem (1a)–(1m). Denote the solution for the production variables by $Y_{ng}^{LP}(t)$.

Step 2: Solve the following MIP:

$$\text{Minimize } \sum_{t \in T} \sum_{n \in N} \sum_{g \in G(n)} d_{ng}(t) + \beta_1 \sum_{n \in N} \sum_{t \in T(n)} \sum_{g \in G(n)} [\theta(1 - 0.01t)Y_{ng}(t) + (1 - \theta)Z_{ng}^1(t)] \tag{2a}$$

subject to (1e)–(1f), (1h)–(1i) and

$$d_{ng}(t) \geq Y_{ng}(t) - Y_{ng}^{LP}(t) \quad \forall g \in G(n), t \in T(n), n \in N, \tag{2b}$$

$$d_{ng}(t) \geq Y_{ng}^{LP}(t) - Y_{ng}(t) \quad \forall g \in G(n), t \in T(n), n \in N, \tag{2c}$$

$$d_{ng}(t) \geq 0 \quad \forall g \in G(n), t \in T(n), n \in N. \tag{2d}$$

Constraints (2b) and (2c) together define $d_{ng}(t) = |Y_{ng}(t) - Y_{ng}^{LP}(t)|$, the L_1 -norm. The objective (2a) is to minimize a combination of the deviations (term 1) and the original objective function (term 2). Here, term 1 acts as the primary objective function and term 2 as a secondary objective. It is important to note that only the capacity constraints (1e)–(1f) and (1h)–(1i) are included in the model. Denote the integer solution by $Y_{ng}^{IP}(t)$

Step 3: Solve the following MIP:

$$\text{Minimize } \sum_{t \in T} \sum_{n \in N} \sum_{g \in G(n)} d_{ng}(t) + \beta_2 \sum_{n \in N} v_n(48)/\rho \tag{3a}$$

subject to (1b)–(1f), (1h)–(1i) and

$$d_{ng}(t) \geq Y_{ng}(t) - Y_{ng}^{IP}(t) \quad \forall g \in G(n), t \in T(n), n \in N, \tag{3b}$$

$$d_{ng}(t) \geq Y_{ng}^{IP}(t) - Y_{ng}(t) \quad \forall g \in G(n), t \in T(n), n \in N, \tag{3c}$$

$$d_{ng}(t) \geq 0 \quad \forall g \in G(n), t \in T(n), n \in N. \tag{3d}$$

Similar to constraints (2b) and (2c), constraints (3b) and (3c) define the deviation from the target, in this case, the integer solution obtained in Step 2. The objective in (3a) is to minimize a weighted sum of the deviation (term 1) and the inventory at the end of the day (term 2). Again, term 1 acts as the primary objective and term 2 as a secondary objective.

In Step 1, we solve the linear programming relaxation of the original problem. The solution provides a valid lower bound and is used as a basis for constructing integer solutions. In Step 2, we solve a MIP under the shared capacity limits to find an integer solution as close to the LP solution as possible. This is achieved by minimizing the deviation, the first term in objective function (2a). The second term in (2a) is actually the original objective function (1a). The inclusion of this term is designed to select among alternative optima the one that is closest to the original objective. As such, β_1 is set at a relative small value, 0.1 in the implementation.

Note that in Step 2, the inventory constraints (1b) and (1c), the production capacity constraints (1d), the ending capacity (1g) constraints, as well as the inventory variables, $v_n(t)$, and the production volume variables, $w_{ng}(t)$, have all been dropped from the model. This is permissible because these constraints

and variables have no effect on the objective function. Consequently, the dimension of the model is significantly reduced and the problem can be solved quickly, usually within 2 s when a 1% optimality gap is specified. By constructing an IP solution close to the LP solution, the expectation is that the production volume, the inventory levels and the flow of mail will remain close to the solution found in Step 1. In actuality, more than 90% of the mail can usually be processed using the integer solution obtained in Step 2. Nevertheless, due to the absence of constraints (1b)–(1d) and (1g), there is no guarantee that this solution is feasible to the original problem.

In Step 3, we solve another optimization problem designed to resolve this issue. In model (3a)–3(d), constraints (1b)–(1d) are added back to account for the inventory and processed volume. The goal is to find a feasible solution that is close as possible to the infeasible integer solution found in Step 2, but with less ending inventory. This is achieved in (3a) by minimizing the sum of the deviations (term 1) and the ending inventory (term 2). The integer solution to this problem is then reported as the heuristic solution to the original problem. In the implementation, $\beta_2 = 1$ provided the best results.

4.3. Justification

Our three-step approach, which we call a piece-by-piece approximation procedure, is based on the idea that many types of models can be decomposed naturally into “pieces”—subsets of decision variables and constraints whose union defines the full model (cf, Cai et al. [4]). First, we use the LP solution as the target and solve problem (2a)–(2d), which only contains variables $Y_{ng}(t)$ and $Z_{ng}(t)$. This is the first piece. The integer solution obtained serves as a new target. We then solve model (3a)–(3e), which includes constraints (1b)–(1d) and variables $v_n(t)$ and $w_{ng}(t)$. This is the second piece. Each of these problems is relatively easy to solve as empirically demonstrated in Section 6. The reason for this, we believe once again, is due to the fact that the feasible region being searched is limited to a neighborhood of the target solution.

5. A Benders decomposition algorithm

When applied to MTPs, Benders decomposition iterates between a master problem that generates trial values of the integer variables and a lower bound for a minimization objective, and a linear subproblem that provides an upper bound (e.g., see Nemhauser and Wolsey [16]). The approach is most effective when (1) the master problem is much easier to solve than the original problem with, say, branch and bound, and (2) the subproblem can be solved quickly for fixed values of the integer variables. The first condition is generally true when only a few cuts are needed in the master problem to obtain a close approximation to the convex hull of the original problem.

One way to apply Benders decomposition to our problem is to retain constraints (1e), (1f), (1h) and (1i) in the master problem. For fixed values of $Y_{ng}(t)$, the inventory and processing variables, $v_n(t)$ and $w_{ng}(t)$, can be determined easily—the second condition. Our algorithmic approach to implementing this idea is described in the remainder of this section.

5.1. Benders reformulation

Let Ω be the set of solution vectors satisfying constraints (1e), (1f), (1h) and (1i). For a given tentative solution $(\bar{Y}_{ng}(t), \bar{Z}_{ng}^1(t), \bar{Z}_{ng}^2(t)) \in \Omega$, the resulting problem to determine the continuous variables $v_n(t)$ and $w_{ng}(t)$ is called the primal subproblem and is defined as follows:

Primal subproblem

$$\Phi_P = \text{Minimize } \phi_P(v_n(t), w_{ng}(t)) = 0 \tag{4a}$$

subject to

$$u_n + \sum_{i:i \in I(n), 1 \in T(i)} a_i(1) - \sum_{g \in G(n)} w_{ng}(1) = v_n(1) \quad \forall n \in N, \tag{4b}$$

$$\begin{aligned} v_n(t-1) + \sum_{i:i \in I(n), t \in T(i)} a_i(t) + \sum_{p \in P(n)} f_{pn} \sum_{g \in G(p)} w_{pg}(t-l_{pn}) \\ - \sum_{g \in G(n)} w_{ng}(t) = v_n(t) \quad \forall t \in T \setminus \{1\}, n \in N, \end{aligned} \tag{4c}$$

$$\begin{aligned} -w_{ng}(t) \geq -\left(\rho_n \bar{Y}_{ng}(t) - \frac{\tau_1}{30} \rho_n \bar{Z}_{ng}^1(t) - \frac{\tau_2}{30} \rho_n \bar{Z}_{ng}^2(t)\right) \\ \forall g \in G(n), t \in T(n), n \in N, \end{aligned} \tag{4d}$$

$$-v_n(48) \geq -\psi_{1n} \quad \forall n \in N, \tag{4e}$$

$$v_n(t), w_{ng}(t) \geq 0 \quad \forall g \in G(n), t \in T(n), n \in N. \tag{4f}$$

This is a feasibility problem in $O(|G| \cdot |N| \cdot |T|)$ linear constraints and $O(|G| \cdot |N| \cdot |T|)$ non-negative variables, and can be solved easily with any commercial LP code. However, rather than solving the primal subproblem, in Benders decomposition the associated dual problem is solved. Let $\pi = (\pi_n^1(1), \pi_n^1(t), \pi_{ng}^2(t), \pi_n^4(48))$ be the dual variables associated with constraints (4b)–(4e), respectively. The dual subproblem is as follows:

Dual subproblem

$$\begin{aligned} \Phi_D = \text{Maximize } \varphi_D(\pi) = & - \sum_{n \in N} \pi_n^1(1) u_n - \sum_{i \in I(n)} \sum_{t \in T(i)} \sum_{n \in N} \pi_n^1(t) a_i(t) \\ & - \sum_{g \in G(n)} \sum_{t \in T(n)} \sum_{n \in N} \rho_n \pi_{ng}^2(t) \left(\bar{Y}_{ng}(t) - \frac{\tau_1}{30} \bar{Z}_{ng}^1(t) - \frac{\tau_2}{30} \bar{Z}_{ng}^2(t) \right) \\ & - \sum_{n \in N} \pi_n^4(48) \psi_{1n} \end{aligned} \tag{5a}$$

subject to

$$-\pi_n^1(t) + \pi_n^1(t+1) \leq 0 \quad \forall t = 1, \dots, 47, n \in N, \tag{5b}$$

$$-\pi_n^1(48) - \pi_n^4(48) \leq 0 \quad \forall t = 48, n \in N, \tag{5c}$$

$$-\pi_{ng}^2(t) - \pi_n^1(t) + \sum_{s \in S(n)} f_{ns} \pi_s^1(t + l_{ns}) \leq 0 \quad \forall t \in T(n), n \in N, g \in G(n), \tag{5d}$$

$$\pi_{ng}^2(t), \pi_n^4(48) \geq 0, \pi_n^1(t) \text{ free} \quad \forall t \in T(n), n \in N, g \in G(n). \tag{5e}$$

Two characteristics of this subproblem are apparent. The first is that the null vector $\mathbf{0}$ is always feasible so the dual polyhedron is non-empty, and the second is that the constraint region (5b)–(5e) are independent of $\bar{Y}_{ng}(t), \bar{Z}_{ng}^1(t), \bar{Z}_{ng}^2(t)$ which only appear in the objective function (5a).

Now, because that the objective function (4a) of the primal subproblem has a constant value of zero, strong duality tells us that the dual subproblem, if bounded, must have an optimal value of zero. Because the null vector $\mathbf{0}$ is feasible and achieves this value, it is an optimal extreme point when the optimal solution of the dual subproblem is bounded. Thus the main purpose in solving (5) is to determine whether or not the primal subproblem is feasible. To do so, let \mathbf{R} be the set of extreme rays of the dual polyhedron. Then the primal is feasible if and only if the dual subproblem is bounded. In particular, the dual subproblem is bounded if and only if

$$\varphi_D(\pi_n^1(1), \pi_n^1(t), \pi_{ng}^2(t), \pi_n^4(48)) \leq 0$$

for every extreme ray $\mathbf{r} = (\pi_n^1(1), \pi_n^1(t), \pi_{ng}^2(t), \pi_n^4(48)) \in \mathbf{R}$.

Because we are only interested in solutions such that the resulting primal subproblem is feasible, we wish to make sure that we select only those $\bar{Y}_{ng}(t), \bar{Z}_{ng}^1(t), \bar{Z}_{ng}^2(t)$ that give rise to a bounded dual subproblem. The original problem can thus be reformulated as follows:

Master problem

$$\text{Minimize} \quad \sum_{n \in N} \sum_{t \in T(n)} \sum_{g \in G(n)} \theta(1 - 0.01t)Y_{ng}(t) + \sum_{n \in N} \sum_{t \in T(n)} \sum_{g \in G(n)} (1 - \theta)Z_{ng}^1(t) \tag{6a}$$

$$\text{subject to} \quad (1e), (1f), (1h), (1i) \text{ and} \\ \varphi_D(\pi_n^1(1), \pi_n^1(t), \pi_{ng}^2(t), \pi_n^4(48)) \leq 0 \quad \forall \mathbf{r} \in \mathbf{R}. \tag{6b}$$

Although this model contains a huge number of constraints, most will be inactive in an optimal solution so there is no need to generate them all. Benders algorithm is designed to generate only a subset of the constraints in (6b) on the fly when needed. Note that the cuts normally arising from vertex solutions to the dual problem are absent in model (6) because $\mathbf{0}$ is the only vertex in the dual feasible region and in our reformulation $\mathbf{0}$ does not produce any cut.

At each iteration, a relaxed master problem is solved by replacing the set \mathbf{R} with the subset $\mathbf{R}^k \subseteq \mathbf{R}$ of extreme rays available at iteration k . The optimal solution of this problem is used to set up the primal and dual subproblems. In our implementation, a simple polynomial algorithm is first run to check whether or not the primal subproblem is feasible for the current value of the master variables, $\bar{Y}_{ng}(t), \bar{Z}_{ng}^1(t), \bar{Z}_{ng}^2(t)$. If it is feasible, the objective value of the subproblem is known to be 0 and the combined solution is feasible to the original problem. This provides an upper bound on the master problem objective that can be used to eliminate inferior solutions in future steps. Otherwise, the primal subproblem is infeasible and the dual subproblem is unbounded along some extreme ray $\mathbf{r}^k \in \mathbf{R}$. In this case, we set up the extreme homogeneous dual problem to find an extreme ray \mathbf{r}^k that violates (6b), update the subset $\mathbf{R}^{k+1} = \mathbf{R}^k \cup \mathbf{r}^k$, and add the violated constraint to the relaxed master problem. At each iteration, the optimal solution of

the relaxed master problem provides a non-decreasing lower bound. The algorithm terminates when the lower bound and upper bound converge.

Proposition 2 in Appendix A shows that if the optimal solution to the relaxed master problem yields a feasible solution to the primal subproblem, then the combined solution is optimal to the original problem. From this observation, it follows that the first feasible solution generated by the Benders algorithm will be optimal to the original problem if the relaxed master problem is solved optimally.

5.2. Enhancements

Additional constraints: When started with $R^1 = \emptyset$, Benders decomposition can be very slow to converge. One way to improve its performance is to augment the master problem with a set of constraints that is redundant in the original problem but nevertheless is required for feasibility. With this in mind, we add the following constraints to the master problem:

$$\sum_{t \in T(n)} \sum_{g \in G(n)} Y_{ng}(t) \geq \left\lceil \frac{V_n^{\text{total}}}{\rho_n} + \frac{\tau_1}{30} Z_n^{1,\min} + \frac{\tau_2}{30} Z_n^{2,\min} \right\rceil \quad \forall n \in N, \quad (6c)$$

where V_n^{total} is the total volume to be processed, and $Z_n^{1,\min} = Z_n^{2,\min}$ are the minimum number of startups and clearances, respectively, needed to complete operation n . Proposition 3 in Appendix A provides lower bounds on these variables, while Proposition 4 confirms that (6c) is valid for the original problem. The inclusion of (6c) in the master problem works to prevent the generation of low-quality solutions and hence to reduce the total number of iterations.

A two-phase approach: A major difficulty with Benders decomposition lies in the solution of the relaxed master problem, which is a general integer problem. Fortunately, this problem does not have to be solved to optimality and cuts generated from any admissible integer solution or even from the linear programming fractional solution are still valid. In view of this, we have developed a two-phase approach. In the first phase, the linear programming relaxation of the relaxed master problems is solved and the fractional solution is sent to the dual subproblem to generate a new constraint (cut). The process is repeated for several iterations. The stopping criteria are discussed below.

With these constraints in place, the second phase reintroduces the integrality requirements and returns to solving the integer problem. However, solving it to optimality is extremely difficult so we found that it is more efficient to stop when the first integer solution is found.

The idea of relaxing the integrality requirements in the master problem and generating cuts from the fractional solution was first proposed in McDaniel and Devine [17], and has since become common place. Theoretically, solving the LP relaxation of the full master problem is equivalent to solving the LP relaxation of the original model. In our implementation, we initialize the restricted master problem with the solution obtained from the LP relaxation of the original problem. We terminate the first phase when the linear programming objective function value of the relaxed master problem converges to, or is within 0.5% of, the relaxed solution of the original problem (1a)–(1m). The idea is to try to generate a plurality of the constraints that will be binding in an optimal solution by solving an LP rather than an IP in the early stages of the computations.

Note that because we do not solve the relaxed master problem to optimality, Proposition 2 no longer holds. A feasible solution to the relaxed master problem that is not optimal does not provide a lower

Table 4
Number of shifts under loose and tight resource constraints

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Tight	250	246	255	260	268	156	206
Loose	263	259	268	273	283	163	217

bound to the original problem. Nevertheless, the LP relaxation or best bound achieved when the first integer solution is obtained still provides a valid lower bound.

6. Computational results

To test the effectiveness of the proposed algorithms, a comprehensive set of experiments was conducted. Both algorithms were implemented using Mosel, the modeling language from Dash [18] Optimization, and all the optimization problems were solved with Xpress, their general linear and integer programming software. For comparison purposes, the branch and bound results are also presented. All computations were performed on a 1.13 GHz Pentium III PC with 256 MB of RAM.

Recall that the minimum number of full-time shifts needed to run a P&DC is calculated in the three-step framework proposed by Zhang and Bard. Two scenarios were created based on this number. In the first scenario, we allow a 5% increase in the number of full-time shifts in the solution. This implies that the labor resource constraint (1i) is fairly loose and that the problem will be relatively easy to solve. In the second scenario, no additional shifts are permitted so the labor resource constraint is much tighter and the problem is harder to solve. Table 4 lists the number of full-time shifts available under each scenario for each day of the week. In all cases, the value of ψ_{1n} in constraint (1g) was set to the minimum amount of inventory remaining at operation n at the end of each day. These values were derived from the branch and bound solution.

6.1. Computational results for the benders algorithm

Fig. 2 depicts two lower bound sequences for a typical Monday under loose shift requirements using Xpress and the Benders algorithm, respectively. The LP relaxation value corresponds to the solution at the root node of the branch and bound tree created by Xpress, and remained constant throughout. For Benders, the calculations were as follows: in phase I, the lower bound is the optimal value of the relaxed master problem with no integrality requirement; in phase II, the lower bound equals the best bound obtained when the first integer solution was identified. As mentioned, because it is difficult to solve the master problem to optimality, we stopped as soon as an integer solution was found. For these runs, no feasible solution to the original problem was ever found within the imposed time limit of 2000 s. Upon termination, 31 phase I iterations and 100 phase II iterations were executed. The accumulated time for the computations is plotted in Fig. 3.

Several observations are apparent from the results. First, Fig. 2 indicates that the Benders lower bound increases at an extremely slow rate. Although the inclusion of constraint (6c) gave an impressive lower bound of 854.67 initially, the best lower bound upon termination was only 872.95, a 2% increase. Almost

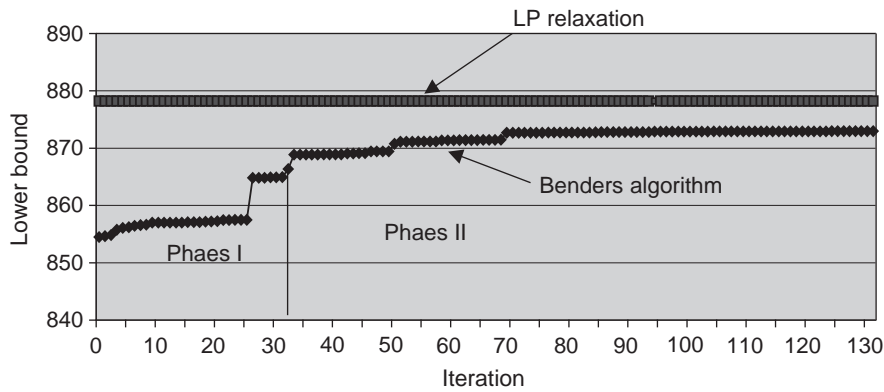


Fig. 2. Results for the Benders decomposition algorithm.

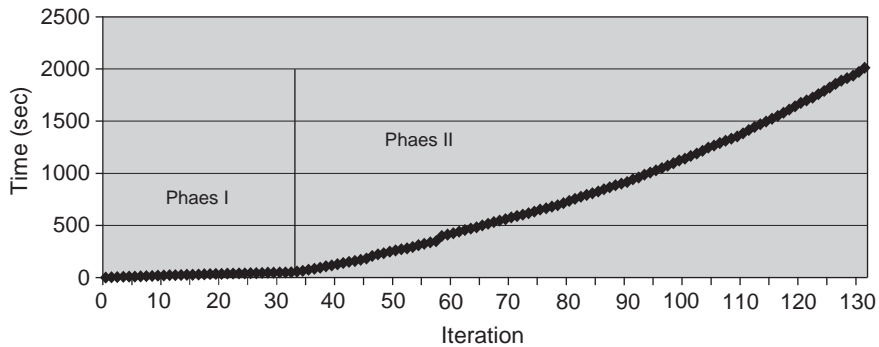


Fig. 3. Accumulated time for the Benders decomposition algorithm.

all the improvement was achieved during the first 60–70 iterations. The final bound, 872.95, is still below 878.22, the lower bound obtained with Xpress within 3 s.

Second, as more constraints in the form of (6b) are added, the master problem becomes increasingly more difficult to solve. This can be inferred from the slope of the curve in Fig. 3. While it took only 2 s to solve the master problem to optimality when no dual cuts were present, it took 7, 15, 24, and 30 s to simply get the first integer solution at iterations 33, 60, 100, and 130, respectively. The most compelling reason why performance slowed is tied directly to number of cuts generated per iteration and their density. The primal subproblem, with constraints (4b)–(4f), does not decompose by operation due to the mail flows between upstream and downstream nodes. Therefore, the solution of the dual subproblem yields only one cut per iteration, and that cut often contains variables associated with each operation at each point in time. As more cuts were added, the master problem became increasingly difficult to solve. Unfortunately, no measurable improvement in the lower bound accompanied the later solutions, and none yielded a feasible solution to the primal subproblem in the time allotted.

When the time limit was raised, a feasible solution with objective value 909.34 was obtained at iteration 314 after 12,159 s and another with objective value 898.60 at iteration 351 after 18,703 s. The latter solution

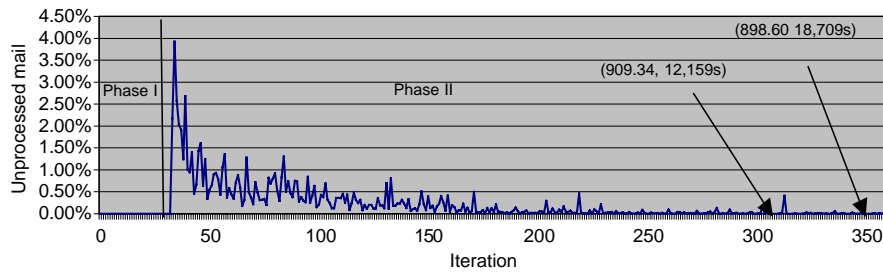


Fig. 4. Percentage of unprocessed mail associated with the master problem solution.

was the best found by Benders. Branch and bound was able to get a feasible solution with objective value 909.30 after 118 s while the best found had objective value 902.56 and was obtained after 18,000 s.

With the addition of cuts, the percentage of unprocessed mail associated with the master problem solution decreased sharply at first and then gradually approached zero (see Fig. 4).

An implication of this result is that Benders algorithm could be used as an approximation scheme by halting it early. However, due to the slow decrease in the percentage of unprocessed mail and the long run times experienced, we have not pursued this idea. In the next section, we show that the piece-by-piece LP heuristic serves as an approximation algorithm and runs much faster than Benders.

An alternative decomposition approach is to retain constraints (1e), (1h) and (1i) in the master problem and generate trial values for $Y_{ng}(t)$ only. In this scheme, the master problem is reduced in size due to the absence of constraints (1f). However, the subproblem is still not decomposable. When implemented, we found that more iterations were required for the LP objective function value of the relaxed master problem to converge to the relaxed solution of the original model (1a)–(1m). In addition, as the number of dual cuts increased the master problem was even harder to solve; i.e., the overall performance was worse than the results depicted in Figs. 2 and 3. We also tried to add an appropriately weighted term of the form $\varepsilon \sum_{n \in N} v_n$ (48) to objective function (1a) and hence (4a), but the results were about the same.

Although there is still room for improvement due to the drawbacks mentioned above, we believe that the Benders approach is not capable of providing any tighter lower bounds, and therefore, cannot be effectively adapted to solve the P&DC equipment scheduling problem. To gain an advantage by defining a decomposable subproblem, the multi-level inventory constraints should be included in the master problem. As noted in Section 3.2, Tempelmeier [13] showed that such an approach would involve dualizing about half of the constraints in model (1). Intuitively, this would imply very quite slow convergence.

6.2. Computational results for LP-based heuristic

Loose resource scenario: Tables 5 and 6 present the computational results for both branch and bound with Xpress, and the proposed heuristic under the loose resource scenario. For branch and bound, the time limit was set at 600 s because no significant improvement was observed thereafter. For the heuristic, the computations were stopped when the relative optimality gap fell below 1%.

Solution times and objective function values are reported in Table 5 for both the first and the best integer solutions found with Xpress. “Time” is given in seconds. The objective function values and the corresponding optimality gap from the best bound are listed in the “Value” and “Gap” columns,

Table 5
Computational results for branch and bound under loose scenario

Day	Bound (root)	Best bound	Value (first)	Time (first)	Gap (first) (%)	Value (best)	Time (best)	Gap (best) (%)
Monday	878.22	878.37	920.62	89	4.81	909.30	118	3.52
Tuesday	843.33	843.49	884.09	59	4.80	875.74	177	3.82
Wednesday	876.73	876.99	935.19	127	6.64	910.37	536	3.81
Thursday	922.27	922.41	960.25	102	4.10	955.92	530	3.63
Friday	927.94	928.62	953.29	124	2.66	952.58	140	2.58
Saturday	595.38	595.52	646.72	85	8.60	630.73	565	5.91
Sunday	561.53	561.72	580.78	10	3.39	572.28	540	1.88
Average	—	—	—	85	5.00	—	372	3.59

Table 6
Computational results for LP-based heuristic under loose scenario

Day	Bound (root)	Value	Gap (%)	Time (s)	Volume processed	Mail remaining (%)
Monday	878.22	888.53	1.17	15	99.85%	0.15
Tuesday	843.33	861.92	2.20	14	99.72%	0.28
Wednesday	876.73	898.32	2.46	10	99.79%	0.21
Thursday	922.27	938.42	1.75	12	99.86%	0.14
Friday	927.94	949.80	2.36	9	99.93%	0.07
Saturday	595.38	613.32	3.01	16	99.52%	0.48
Sunday	561.53	567.66	1.09	7	99.95%	0.05
Average	—	—	2.01	12	99.74%	0.26

respectively. The “Best bound” column refers to the best bound obtained at the point of termination, while the “Bound (root)” column lists the bound at the root node after initial cuts were generated and added to the problem. To achieve the tightest bound possible, we chose the aggressive cut generation strategy in Xpress. A slight improvement of about 1% in the lower bound compared to the LP bound was observed.

For the LP-based heuristic, the “Value” reported in the third column is not the deviation from the LP solution (3a), but the corresponding objective value of the original problem (1a). The bound at the root node is the same value listed in Table 5 and is again used for comparative purposes. The last two columns in the Table 6 indicate the percent of mail that arrived that day and was processed, and the percentage with respect to the branch and bound solution that was left to the next day. Several statistics concerning the average solution quality are provided at the bottom of each table.

As the data in Table 6 show, the LP-based heuristic yields high quality solutions in only a fraction of time required by branch and bound to solve the original model (1a)–(1m). More specifically, Xpress took 85 s on average to find the first solution that was within 5% of the lower bound. An average of 372 s was required to find the best solution. The corresponding optimality gap was 3.59% (note that additional testing revealed that several hours are needed to bring the gap down to 2%). The heuristic, on the other hand, was able to find solutions within an average of 2.01% of the lower bound associated with the original

Table 7
Computational results for branch and bound under tight resource scenario

Day	Bound (root)	Best bound	IP value	Time (s)	Gap (%)
Monday	885.23	885.97	937.91	420	5.86
Tuesday	857.64	857.84	935.94	330	9.10
Wednesday	878.52	878.77	949.58	366	8.06
Thursday	925.41	925.86	994.48	320	7.41
Friday	930.16	930.79	973.29	436	4.57
Saturday	596.73	596.87	691.10	760	15.8
Sunday	564.46	564.56	582.13	100	3.11
Average	—	—	—	378	7.70

Table 8
Computational results for LP-based heuristic under tight resource scenario

Day	Bound (root)	IP value	Time (s)	Gap (%)	Volume processed	Mail remaining (%)
Monday	885.23	899.20	32	1.58	99.81%	0.19
Tuesday	857.64	872.45	28	1.73	99.71%	0.29
Wednesday	878.52	887.79	42	1.06	99.80%	0.20
Thursday	925.41	950.45	38	2.71	99.82%	0.18
Friday	930.16	937.25	29	0.77	99.92%	0.08
Saturday	596.73	612.38	41	2.62	99.38%	0.62
Sunday	564.45	575.91	23	2.03	99.94%	0.06
Average	—	—	33	1.78	99.70%	0.30

objective function (1a) in only 12 s. For these solutions, only about 0.26% of the mail volume that could have been processed had to be shifted to the next day.

With branch and bound, the best bound remained virtually unchanged when the algorithm was terminated after 600 s. The actual increase over the root node solution was only 0.04%. This type of performance is the principle reason why commercial MIP codes are unlikely to prove effective in solving the scheduling problems that arise in P&DCs. Weak bounds usually mean large search trees. As was the case here, they may also foreshadow an inability to find good feasible solution quickly. Not surprisingly, the tighter the resource constraints the more difficult it is to simply find feasible solutions.

Tight resource scenario: Tables 7 and 8 report the computation results for branch and bound and the LP-based heuristic under the tight resource scenario. Again, the time limit was set at 600 s for Xpress and the heuristic was halted when the optimality gap fell below 1%.

The data in the tables indicate that when the labor resource constraint is tightened, the problem gets much harder to solve. For the branch and bound algorithm, the time to arrive to the first solution increased dramatically from an average of 85 to 378 s. Actually, for Saturday no feasible solution was obtained within the time limit so extra time was allocated. Except for Sunday, no more than one feasible solution was obtained. The average optimality gap after 600 s was 7.70%.

In contrast, the heuristic was able to provide much better results. The average time spent to solve each of the seven daily problems was only 33 s and the optimality gap averaged only 1.78%, a sharp reduction

Table 9
Daily arrival volume

Day	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
# of pieces	5,053,301	5,521,714	5,240,858	5,336,317	5,645,789	1,350,672	2,086,580

compared to branch and bound. The data in Table 8 also indicate that about 99.70% of the mail that could be processed on the same day was processed.

Implementation issues: Recall the ending inventory constraint (1g) is dropped from the two optimization problems in Steps 2 and 3. As a consequence, not all the constraints in the original model are satisfied and, as we have seen, not all the mail is processed. A natural concern is how important is this “infeasibility.” To address this issue we refer to Table 9 which identifies the number of letters that arrived daily at the facility.

On a typical weekday, more than 5 million letters arrive. The heuristic was able to process 99.75% of the actual number under the loose workforce constraints and 99.70% under the tight workforce constraints. Around 10,000 letters per day were left over. In reality, these mail pieces could be easily handled, especially when one notes that several of the machines are not always running at full capacity.

While sacrificing some feasibility, the schedule obtained from the LP-based heuristic exhibited fewer startups than the branch and bound solution, and therefore would be preferred by facility supervisors. A detailed examination of the schedules associated with the fractional LP solution to the original problem and the solution obtained with the LP-based heuristic reveals that the two are a close match.

Finally, we would like to point out that the LP-based heuristic offers an alternative to the sequential methodology proposed by Zhang and Bard [8]. Using branch and bound to solve three integer programs in sequence, they found that several hours were required to achieve an optimality gap of 2–3%. If the heuristic proposed in this paper for the third model were adopted, there would be no need to insist that the solutions to the first two problems be integral. Instead, the LP relaxations of models 1 and 2 would be solved, and the LP-based heuristic would be used for the batch production problem (model 3) as we have proposed here. As a result, only two linear programs and the heuristic would have to be solved. This can be done in less than a minute compared to several hours.

7. Conclusion

The focus of this paper has been on a batch production problem that arises in scheduling equipment at USPS mail processing and distribution centers. The problem falls into the general category of multi-level lot sizing and is well known to be difficult to solve with exact methods.

In the paper, we investigate two specialized algorithms for obtaining solutions. The first is a heuristic that is based on the LP relaxation of the original model and a modularization of the feasible region. The second is a Benders decomposition. While the latter failed to provide satisfactory results, the heuristic proved to be an effective alternative. The central idea of the heuristic is to use the LP solution as a target and then construct an integer solution that is as close to it as possible. This is achieved by minimizing the distance between the two.

Test results using data provided by the Dallas P&DC indicate that the heuristic was able to find high quality, implementable solution, orders of magnitude faster than a commercial branch and bound code. This is particularly important in an operational environment where equipment scheduling is a daily activity. More generally, the proposed piece-by-piece approach is applicable to a wide range of multi-level, capacitated lot-sizing and related problems.

Appendix A. Theoretical results

Theorem 1. *The equipment scheduling problem (ESP) given by (1a)–(1m) is NP-complete in the strong sense.*

Proof. The two-machine, flow shop scheduling problem $F2|r_i, ptmn|L_{\max}$ with n jobs is known to be NP-complete in the strong sense [19]. The problem is stated as follows. Each job i has release time r_i , due date d_i , and processing times p_{i1} and p_{i2} on the two machines, respectively. No job can commence before its release time, and all jobs have to be completed by their respective due dates. The problem is to determine the existence of feasible pre-emptive schedules.

Given an instance of the two-machine scheduling problem, an instance of ESP can be constructed as follows. We have n types of mail, where each must be processed sequentially on two machines. Mail type i arrives at time r_i , is due at d_i , and requires p_{i1} time slots (30 min each) on the first machine and p_{i2} time slots on the second (the processing rates on the two machines are different). The question is whether all the mail can be processed by the given due dates, or equivalently, whether there exists a schedule such that the total leftover mail is not more than 0.

Evidently, the constructed instance of ESP has a feasible solution if and only if the two-machine scheduling problem has a feasible solution. Because the transformation is immediate and any schedule can be checked for feasibility in $O(n)$, we have the desired result. \square

Proposition 1. *The shift variables ω_{kf} will always be integral in an optimal solution to model (1a)–(1m) for r_{kg} and ψ_2 integer.*

Proof. For every feasible solution to (1a)–(1m), $Y_{ng}(t)$ is integer as is the left-hand side of constraint (1h). Now, because a full-time shift is defined as a set of continuous periods totaling 8 h, the shift variables, ω_{kf} for all k and f , in constraints (1h) possess the “consecutive 1’s property”; that is, the variable ω_{kf} in each row in constraint (1h) has either a 0 or 1 coefficient and the “1” coefficients appear consecutively in the model. Let A be the coefficient matrix of ω_{kf} in (1h) with $Y_{ng}(t)$ fixed. This matrix is well known

to be totally unimodular (TU) [16]. The inclusion of constraint (1i) leads to a matrix of the form $\begin{bmatrix} A \\ \dots \\ \mathbf{1} \end{bmatrix}$,

where $\mathbf{1}$ is a row vector of 1’s. This augmented matrix is not, in general, TU. However, if the length of the shifts cover an even number of periods, such as 16 in our case, we claim that the augmented matrix is still totally unimodular.

To see this, partition the rows referenced by the index set $Q = \{1, 2, \dots, 48, 49\}$ into two subsets $Q_1 = \{1, 3, \dots, 49\}$ and $Q_2 = Q \setminus Q_1$, the even and odd integers, respectively. Because each shift covers an even number of periods, we have $\sum_{i=Q_1 \setminus \{49\}} a_{ij}(t) - \sum_{i=Q_2} a_{ij}(t) = 0$, which implies that

$$\left| \sum_{i=Q_1} a_{ij}(t) - \sum_{i=Q_2} a_{ij}(t) \right| = 1 \quad \text{for all } j.$$

This is a sufficient condition for the matrix $\begin{bmatrix} \mathbf{A} \\ \dots \\ \mathbf{1} \end{bmatrix}$ to be TU. As a result, ω_{kf} will always be integral in any feasible solution to (1a)–(1m). \square

Proposition 2. *If the optimal solution to the relaxed master problem yields a feasible solution to the primal subproblem, then the combined solution is optimal to the original problem.*

Proof. At Step k of the Benders algorithm, let $(\bar{Y}_{ng}(t), \bar{Z}_{ng}^1(t), \bar{Z}_{ng}^2(t))$ be the optimal solution to the relaxed master problem. Then the corresponding objective value provides a lower bound to the original problem. Suppose that when $(\bar{Y}_{ng}(t), \bar{Z}_{ng}^1(t), \bar{Z}_{ng}^2(t))$ is substituted into (4d), the vector $(\bar{v}_n(t), \bar{w}_{ng}(t))$ is feasible to (4a)–(4f). Because the objective value for problem (4) is 0, the vector $(\bar{Y}_{ng}(t), \bar{Z}_{ng}^1(t), \bar{Z}_{ng}^2(t), \bar{v}_n(t), \bar{w}_{ng}(t))$ is not only feasible to the original problem but also achieves the lower bound. Thus it is optimal to the original problem. \square

Corollary 1. *The first feasible solution generated by the Benders algorithm will be optimal to the original problem if the relaxed master problem is solved optimally.*

Proposition 3. *For each n , $Z_n^{1,\min}$ (and $Z_n^{2,\min}$) $\geq \lceil V_n^{\text{total}} / (\rho_n |T(n)|) \rceil$, where $|T(n)|$ is the length of the operation window.*

Proof. Let us first look at $Z_n^{1,\min}$. Note that constraint (1f), which defines $Z_{ng}^1(t)$ and $Z_{ng}^2(t)$, can be written alternatively as two inequalities

$$\begin{aligned} Z_{ng}^1(t) &\geq Y_{ng}(t) - Y_{ng}(t - 1) \quad \forall t \in T(n), \quad g \in G(n), \quad n \in N, \\ Z_{ng}^2(t) &\geq Y_{ng}(t) - Y_{ng}(t + 1) \quad \forall t \in T(n), \quad g \in G(n), \quad n \in N, \\ Z_{ng}^1(t) &\geq 0, \quad Z_{ng}^2(t) \geq 0 \quad \forall t \in T(n), \quad g \in G(n), \quad n \in N. \end{aligned}$$

This gives the equivalent definition of $Z_{ng}^1(t)$ as

$$Z_{ng}^1(t) \geq \max\{0, Y_{ng}(t) - Y_{ng}(t - 1)\} \quad \forall t \in T(n), \quad g \in G(n), \quad n \in N.$$

Summing this inequality over g and t , we obtain

$$\sum_{t \in T(n)} \sum_{g \in G(n)} Z_{ng}^1(t) \geq \sum_{t \in T(n)} \sum_{g \in G(n)} \max\{0, Y_{ng}(t) - Y_{ng}(t - 1)\} \quad \forall n \in N.$$

Now, given that $Y_{ng}(\tau - 1) = 0$ for τ , the first period in $T(n)$, we have

$$\sum_{t \in T(n)} \sum_{g \in G(n)} \max\{0, Y_{ng}(t) - Y_{ng}(t - 1)\} \geq Y_n^{\max},$$

where $Y_n^{\max} \geq \sum_{g \in G(n)} Y_{ng}(t)$ for all $t \in T(n)$, is the maximum number of machines that are performing operation n at any time. Taking into account the processing rate and volume of operation n , we have $Y_n^{\max} \geq \lceil V_n^{\text{total}} / (\rho_n |T(n)|) \rceil$, implying that

$$Z_n^{1,\min} \equiv \sum_{t \in T(n)} \sum_{g \in G(n)} Z_{ng}^1(t) \geq Y_n^{\max} \geq \lceil V_n^{\text{total}} / (\rho_n |T(n)|) \rceil.$$

Finally, because the number of startups equals the number of clearances, we have

$$Z_{ng}^{2,\max} \geq \lceil V_n^{\text{total}} / (\rho_n |T(n)|) \rceil.$$

which completes the proof. \square

Proposition 4. *The constraints in (6c) are valid for the original problem.*

Proof. To see why this is true, let us sum Eq. (1d) over g and t to obtain

$$\begin{aligned} & \sum_{t \in T(n)} \sum_{g \in G(n)} w_{ng}(t) + \sum_{t \in T(n)} \sum_{g \in G(n)} \frac{\tau_1}{30} \rho_n Z_{ng}^1(t) + \sum_{t \in T(n)} \sum_{g \in G(n)} \frac{\tau_2}{30} \rho_n Z_{ng}^2(t) \\ & \leq \sum_{t \in T(n)} \sum_{g \in G(n)} \rho_n Y_{ng}(t) \quad \forall n \in N. \end{aligned}$$

Dividing both sides by ρ_n and letting $V_n^{\text{total}} = \sum_{t \in T(n)} \sum_{g \in G(n)} w_{ng}(t)$ gives

$$\frac{V_n^{\text{total}}}{\rho_n} + \sum_{t \in T(n)} \sum_{g \in G(n)} \frac{\tau_1}{30} Z_{ng}^1(t) + \sum_{t \in T(n)} \sum_{g \in G(n)} \frac{\tau_2}{30} Z_{ng}^2(t) \leq \sum_{t \in T(n)} \sum_{g \in G(n)} Y_{ng}(t) \quad \forall n \in N.$$

Next, using Proposition 3 we replace $\sum_{t \in T(n)} \sum_{g \in G(n)} Z_{ng}^1(t)$ and $\sum_{t \in T(n)} \sum_{g \in G(n)} Z_{ng}^2(t)$ with their lower bounds $Z_n^{1,\min}$ and $Z_n^{2,\min}$, to obtain

$$\frac{V_n^{\text{total}}}{\rho_n} + \frac{\tau_1}{30} Z_n^{1,\min} + \frac{\tau_2}{30} Z_n^{2,\min} \leq \sum_{t \in T(n)} \sum_{g \in G(n)} Y_{ng}(t) \quad \forall n \in N.$$

Because $\sum_{t \in T(n)} \sum_{g \in G(n)} Y_{ng}(t)$ has to be integer in a feasible solution, we have

$$\left\lceil \frac{V_n^{\text{total}}}{\rho_n} + \frac{\tau_1}{30} Z_n^{1,\min} + \frac{\tau_2}{30} Z_n^{2,\min} \right\rceil \leq \sum_{t \in T(n)} \sum_{g \in G(n)} Y_{ng}(t) \quad \forall n \in N$$

which is exactly (6c). This completes the proof. \square

References

- [1] Berman O, Larson RC, Pinker E. Scheduling workforce and workflow in a high volume factory. *Management Science* 1997;43(2):158–72.
- [2] Jarrah AIZ, Bard JF, deSilva AH. Equipment selection and machine scheduling in general mail facilities. *Management Science* 1994;40(8):1049–68.
- [3] Zhang X, Bard JF. Equipment scheduling at USPS processing and distribution centers. *IIE Transactions on Scheduling and Logistics*, 2005, to appear.
- [4] Cai X, McKinney DC, Lasdon LS. Piece-by-piece approach to solving large nonlinear water resources management models. *Journal of Water Resources Planning and Management* 2001;127(6):363–8.
- [5] Blömer F, Günther H. Scheduling of a multi-product batch process in the chemical industry. *Computers in Industry* 1998;36:245–59.
- [6] Reklaitis GV, Sunol AK, Rippin DWT, Hortacsu O, editors. *Batch processing systems engineering: fundamentals and applications for chemical engineering*. Berlin; Springer: 1996.
- [7] Bard JF, Binici C, deSilva AH. Staff scheduling at the United States postal service. *Computers & Operations Research* 2003;30(5):745–71.
- [8] Zhang X, Bard JF. A multi-period machine assignment problem. *European Journal of Operational Research*, 2005, to appear.
- [9] Pocket Y, Wolsey LA. Polyhedra for lot sizing with Wagner–Whitin cost. *Mathematical Programming* 1994;67:297–324.
- [10] Wolsey LA. Solving multi-item lot-sizing problems with an MIP solver using classification and reformulation. *Management Science* 2002;48(12):1587–602.
- [11] Maes J, McClain JO, van Wassenhove LN. Multilevel capacitated lot sizing complexity and LP-based heuristics. *European Journal of Operational Research* 1991;52:131–48.
- [12] Kuik R, Salomon M, van Wassenhove LN, Maes J. Linear programming simulated annealing and tabu search heuristics for lot sizing in bottleneck assembly systems. *IEE Transactions on Design & Manufacturing* 1993;25(1):62–72.
- [13] Tempelmeier H, Derstroff M. A Lagrangean-based heuristic for dynamic multi-level multi-item constrained lot sizing with setup times. *Management Science* 1996;42(5):738–57.
- [14] Vanderbeck F. Lot-sizing with start-up times. *Management Science* 1998;44(10):1409–25.
- [15] Morton DP, Popova E, Popova I, Zhong M. Optimizing benchmark-based utility functions. *Bulletin of the Czech Econometric Society* 2003;10:1–18.
- [16] Nemhauser GL, Wolsey LA. *Integer and combinatorial optimization*. New York: Wiley; 1988.
- [17] McDaniel D, Devine M. A modified benders partitioning algorithm for mixed integer programming. *Management Science* 1977;24:312–79.
- [18] Dash Optimization, Xpress-mosel: user guide. Englewood Cliffs, NJ: Prentice-Hall; 2002.
- [19] Cho Y, Sahni S. Preemptive scheduling of independent jobs with release and due times on open, flow and job shops. *Operations Research* 1981;29(3):511–22.