A Concave-Cost Production Planning Problem with Remanufacturing Options

Jian Yang,[1] Boaz Golany,[2] Gang Yu[3]

[1] *Department of Industrial and Manufacturing Engineering, New Jersey Institute of Technology, Newark, New Jersey 07102*

[2] *Faculty of Industrial Engineering and Management, The Technion—Israel Institute of Technology, Haifa, 32000, Israel*

[3] *Department of Management Science and Information Systems, The University of Texas at Austin, Austin, Texas 78712*

Received October 2001; revised December 2004; accepted 31 March 2005
DOI 10.1002/nav.20089
Published online 28 April 2005 in Wiley InterScience (www.interscience.wiley.com).

Abstract: We focus on the concave-cost version of a production planning problem where a manufacturer can meet demand by either producing new items or by remanufacturing used items. Unprocessed used items are disposed. We show the NP-hardness of the problem even when all the costs are stationary. Utilizing the special structure of the extreme-point optimal solutions for the minimum concave-cost problem with a network flow type feasible region, we develop a polynomial-time heuristic for the problem. Our computational study indicates that the heuristic is a very efficient way to solve the problem as far as solution speed and quality are concerned. © 2005 Wiley Periodicals, Inc. Naval Research Logistics 52: 443–458, 2005.

Keywords: production planning; remanufacturing; heuristics

1. INTRODUCTION

We consider a single-item production system which faces periodic deterministic demand and return of used items over a finite horizon and where some (or all) of the demand in certain periods can be satisfied through remanufacturing of the returned used items. Relevant information about demand and used-item arrival rates for the entire planning horizon is known in advance and backlogging is not permitted. The production, holding (of both used and new items), remanufacturing, and disposal costs are given for each period.

In each period, there are three options for every used item on hand: discard as spoilage, keep in a "used-item storage" for future consideration, and remanufacture. Each of the options may be associated with some costs. Remanufacturing may involve testing, cleaning, disassembly with replacements of some components, and then re-assembly, etc. Holding the used items in inventory may involve some movement in and out of storage, cost of storage space, monitoring, etc. Disposal of items may incur the costs associated with transportation, disassembly and separation of hazardous materials, etc. Disposal may also be associated with negative costs (i.e., revenues) due to the salvage value of the disposed items.

Most of the production planning problems addressed before dealt with flows of materials or goods in one direction—from the manufacturer to the customers. Here we discuss a situation in which flow goes both forward (to the customers) and backward (to the manufacturer). Such situations are now denoted as "reverse logistics" (see the extensive reviews in Fleischmann et al. [5], Guide et al. [11], and the survey on remanufacturing practices by Guide [10]). In terms of modeling, the closest works to our paper were done by Richter and Sombrutzki [13] and Richter and Weber [14]. The authors exploited special properties for special cases of the problem, and developed Wagner-Whitin type algorithms. In an earlier paper [9], the current authors formulated a special case of the problem with linear cost functions, examined its complexity and developed a polynomial algorithm to solve it.

Though not necessarily confined to the production planning framework we assume here, there are numerous other

Correspondence to: B. Golany (golany@ie.technion.ac.il); J. Yang (yang@adm.njit.edu); G. Yu (yu@uts.cc.utexas.edu)

papers that addressed the issue of optimally utilizing the remanufacturing option. Teunter [16] derived EOQ type formulas for such a problem with stationary cost parameters and constant and continuous item flows. Kiesmüller [12] presented optimal control policies for a continuous, deterministic, and dynamic system with remanufacturing. Simpson [15] showed the structure of the optimal policy for a stochastic inventory control problem involving remanufacturing. van der Laan and Salomon [18] and van der Laan, Salomon, and Dekker [19] applied the extensions of two simple control policies to the control of production systems with remanufacturing options. Later, van der Laan et al. [20] developed a methodology to study the effects of these control policies and compared the performances of such systems with and without remanufacturing options. Toktay, Wein, and Zenios [17] studied a stochastic inventory management problem for a product with remanufacturing option where knowledge about the probabilistic behavior of the returning flow of used items is acquired through Bayesian learning. There, a returned used item is always remanufactured and the focus is on controlling the levels of production from scratch to reduce inventory holding and lost sales costs.

Here, we study a deterministic, discrete-time, and dynamic production planning problem with remanufacturing options and focus on the case where all cost functions are concave. Since the maximum number of remanufactured items is limited by the number of used items on hand, the problem bears some similarity to a production planning problem with production capacity limits. Consequently, existing results for the non-capacitated production planning problem cannot be readily generalized to our problem. On the other hand, the production planning problem with production capacity was found to be extremely difficult. The special case with constant capacities were first found to be polynomially solvable in $O(T^4)$ time by Florian and Klein [6]. van Hoesel and Wagelmans [21] then reduced the time complexity to $O(T^3)$. After the general problem with arbitrary capacities was shown to be NP-hard by Florian, Lenstra, and Rinnooy Kan [7], Bitran and Yanasse [3] further investigated the computational complexity of some special cases and identified polynomially solvable cases among them. Approximate solution procedures were proposed for the general capacitated problems by Bitran and Matsuo [2], Gavish and Johnson [8], and van Hoesel and Wagelmans [22], but their results could not be generalized to our case.

In this paper, we first extend our earlier work [9] by presenting an exact dynamic programming (DP) algorithm for the general-case problem. Second, we show that the concave-cost problem is NP-hard even when all the cost functions are stationary. We then concentrate on our main theme, that of developing a polynomial-time heuristic for the concave-cost problem. The heuristic is inspired by the fact that the extreme-point optimal solution of the problem, which must exist, has the special spanning-tree property and that this spanning tree can be decomposed into the well-defined flow patterns. Since the heuristic exploits all combinations of flow patterns that render its time complexity still polynomial, it might be denoted as "the best practical" procedure to our problem. To evaluate the performance of our proposed heuristic algorithm, we carried out a computational study that compares the solution speeds and qualities of the heuristic to those obtained through the DP algorithm and a MIP formulation of the aforementioned problem. The results clearly show the competitiveness of the heuristic.

We organize the rest of the paper as follows: In Section 2 we formulate the general problem and present the DP algorithm; in Section 3 we elaborate on the complexity of the concave-cost problem and, in particular, we show the NP-hardness of the special case with stationary costs; in Section 4 we explore the special structure of the extreme-point optimal solution of the concave-cost problem and introduce notions that are essential to the development of the heuristic; in Section 5 we introduce the polynomial heuristic and show why it might be called the "best practical" procedure; in Section 6 we present our computational study that demonstrates the effectiveness of the heuristic; and in Section 7 we conclude the paper.

2. THE GENERAL FORMULATION AND DP ALGORITHM

We assume that the time horizon to be considered spans over T periods ($t = 1, \ldots, T$). In each period t, B_t used items become newly available for remanufacturing and D_t new items are demanded. Our objective is to optimally utilize our resources (production, remanufacturing, and storage capacities) to satisfy demand and process returned used items over the entire time horizon. We use the following notation:

- B_t: number of used items newly available in period t for $t = 1, \ldots, T$;
- D_t: number of new items demanded in period t for $t = 1, \ldots, T$;
- u_t: number of used items held at the end of period t for $t = 1, \ldots, T - 1$;
- v_t: number of items disposed of in period t for $t = 1, \ldots, T$;
- z_t: number of items that are remanufactured in period t for $t = 1, \ldots, T$;
- x_t: number of items produced from scratch in period t for $t = 1, \ldots, T$;
- y_t: number of new items held at the end of period t for $t = 1, \ldots, T - 1$;

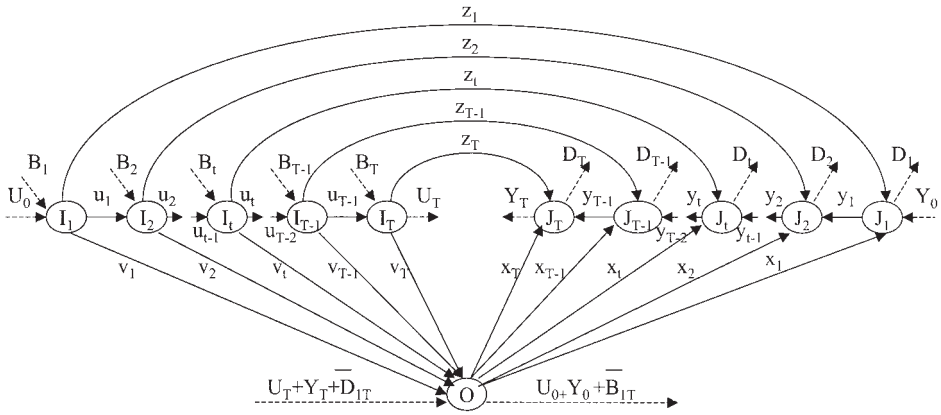

Figure 1. Model representation.

U_0: the given starting used-item inventory level;
U_T: the given ending used-item inventory level;
Y_0: the given starting new-item inventory level;
Y_T: the given ending new-item inventory level;
$W_t(u_t) \geq 0$: used-item holding cost in period t for $t = 1, \ldots, T-1$;
$S_t(v_t)$: disposal cost in period t for $t = 1, \ldots, T$;
$R_t(z_t) \geq 0$: remanufacturing cost in period t for $t = 1, \ldots, T$;
$P_t(x_t) \geq 0$: production cost in period t for $t = 1, \ldots, T$;
$H_t(y_t) \geq 0$: new-item holding cost in period t for $t = 1, \ldots, T-1$.

We always maintain that

$$\begin{cases} W_t(0) = S_t(0) = R_t(0) = P_t(0) = H_t(0) = 0, \\ W_t(-i) = S_t(-i) = R_t(-i) = P_t(-i) = H_t(-i) = +\infty \\ \qquad\qquad \forall\, i = 1, 2, \ldots. \end{cases}$$

The objective function in the proposed model minimizes the total cost and the constraints ensure material conservation. This leads to the following formulation of the production and remanufacturing planning (PRP) problem:

$$(PRP) \qquad \min \sum_{t=1}^{T-1} W_t(u_t) + \sum_{t=1}^{T} S_t(v_t) + \sum_{t=1}^{T} R_t(z_t) + \sum_{t=1}^{T} P_t(x_t) + \sum_{t=1}^{T-1} H_t(y_t) \quad (1)$$

subject to

$$u_1 - U_0 + v_1 + z_1 = B_1, \quad (2)$$

$$u_t - u_{t-1} + v_t + z_t = B_t \qquad \forall\, t = 2, \ldots, T-1, \quad (3)$$

$$U_T - u_{T-1} + v_T + z_T = B_T, \quad (4)$$

$$z_1 + x_1 + Y_0 - y_1 = D_1, \quad (5)$$

$$z_t + x_t + y_{t-1} - y_t = D_t \qquad \forall\, t = 2, \ldots, T-1, \quad (6)$$

$$z_T + x_T + y_{T-1} - Y_T = D_T, \quad (7)$$

$$v_t, z_t, x_t \geq 0 \qquad \forall\, t = 1, \ldots, T, \quad (8)$$

$$u_t, y_t \geq 0 \qquad \forall\, t = 1, \ldots, T-1. \quad (9)$$

For convenience, we define $\bar{B}_{tt'} = \sum_{\tau=t}^{t'} B_\tau$ and $\bar{D}_{tt'} = \sum_{\tau=t}^{t'} D_\tau$. The problem will be feasible if and only if the parameters satisfy both $U_T \leq U_0 + \bar{B}_{1T}$ and $Y_0 \leq Y_T + \bar{D}_{1T}$.

The PRP mathematical formulation is of the network flow type, as depicted in Figure 1. We do not need to impose the integrality constraints on its variables. In the network, we have three kinds of nodes: O, $I_1, \ldots, I_T$, and $J_1, \ldots, J_T$. Here, O is the node whose total in-flow $U_T + \sum_{t=1}^{T} v_t + Y_T + \bar{D}_{1T}$ is balanced by total outflow $U_0 + \sum_{t=1}^{T} x_t + Y_0 + \bar{B}_{1T}$, every I_t is the node whose total in-flow $u_{t-1} + B_t$ is balanced by total out-flow $u_t + v_t + z_t$, and J_t is the node whose total in-flow $z_t + x_t + y_{t-1}$ is balanced by total out-flow $y_t + D_t$. We label the arc from node A to node B as (A, B). Hence, x_t is the flow on arc (O, J_t), y_t is the flow on arc (J_t, J_{t+1}), z_t is the flow on arc (I_t, J_t), v_t is the flow on arc (I_t, O), and u_t is the flow on arc (I_t, I_{t+1}). When we say a node has supply d, we mean that the difference

between its total controllable outgoing flow and total controllable incoming flow is required to be d. If we take node O's supply to be $-U_0 + U_T - Y_0 + Y_T - \bar{B}_{1T} + \bar{D}_{1T}$, node I_1's to be $U_0 + B_1$, node J_1's to be $Y_0 - D_1$, node I_T's to be $-U_T + B_T$, node J_T's to be $-Y_T - D_T$, and for any $t = 2, \ldots, T-1$, node I_t's supply to be B_t and node J_t's supply to be $-D_t$, then PRP is the minimum-cost network flow problem satisfying all the node supplies.

In Richter and Sombrutzki [13], the ending inventory levels U_T and Y_T are treated as decision variables (in our terminology, they would have been denoted as u_T and y_T, respectively), whose values/costs to future operations are reflected in the costs $W_T(u_T)$ and $H_T(y_T)$ (in [13], these are linear costs). We note that we can convert a T-period instance considered in [13] into an equivalent $(T+1)$-period PRP instance, where ending inventory levels are fixed: Let all the parameters of the PRP instance for $t = 1, \ldots, T$ be the same as provided by the given instance; let $U_{T+1} = 0$, $Y_{T+1} = 0$, $B_{T+1} = 0$, $D_{T+1} = U_0 + Y_0 + \bar{B}_{1T}$; and for $i = 1, 2, \ldots$, let the production cost $P_{T+1}(i) = 0$, the remanufacturing cost $R_{T+1}(i) = +\infty$, and the disposal cost $S_{T+1}(i) = 0$. The PRP instance will generate the same optimal solution as the given instance, because, like the latter, it allows no remanufacturing in period $T+1$ ($z_{T+1} = 0$); its decision on all other variables except x_{T+1} and v_{T+1} is solely dependent on parameters associated with periods that precede period $T+1$, which are the same as those in the given instance; and the costless production in period $T+1$ makes up the shortfall in demand ($x_{T+1} = U_0 + Y_0 + \bar{B}_{1T} - y_T$) while the costless disposal in period $T+1$ gets rid of the excess in used items ($v_{T+1} = u_T$).

There is a pseudopolynomial DP algorithm for the general-cost problem. For $t = 1, 2, \ldots, T$, $\max\{U_T - \bar{B}_{tT}, 0\} \le u_{t-1} \le U_0 + \bar{B}_{1,t-1}$, and $\max\{Y_0 - \bar{D}_{1,t-1}, 0\} \le y_{t-1} \le Y_T + \bar{D}_{tT}$, let $f_t(u_{t-1}, y_{t-1})$ be the minimum cost of any partial production plan from period t to period T when the used- and new-item inventory levels at the end of period $t-1$ are u_{t-1} and y_{t-1}, respectively. Then, for the terminal period T, for $\max\{U_T - B_T, 0\} \le u_{T-1} \le U_0 + \bar{B}_{1,T-1}$ and $\max\{Y_0 - \bar{D}_{1,T-1}, 0\} \le y_{T-1} \le Y_T + D_T$, we have

$$f_T(u_{T-1}, y_{T-1}) = \min\{S_T(u_{T-1} - U_T - z + B_T) + R_T(z) + P_T(-y_{T-1} + Y_T - z + D_T) \mid 0 \le z \le \min\{u_{T-1} - U_T + B_T, -y_{T-1} + Y_T + D_T\}\}. \quad (10)$$

While for $t = T-1, T-2, \ldots, 1$, for $\max\{U_T - \bar{B}_{tT}, 0\} \le u_{t-1} \le U_0 + \bar{B}_{1,t-1}$ and $\max\{Y_0 - \bar{D}_{1,t-1}, 0\} \le y_{t-1} \le Y_T + \bar{D}_{tT}$, we have the recursive relationship

$$\begin{aligned} f_t(u_{t-1}, y_{t-1}) = \min\{&S_t(v) + R_t(z) + P_t(x) + W_t(u_{t-1} - v - z + B_t) + H_t(y_{t-1} + z + x - D_t) + f_{t+1}(u_{t-1} - v - z + B_t, y_{t-1} + z + x - D_t) \mid 0 \le z \le \min\{u_{t-1} + \min\{-U_T + \bar{B}_{tT}, B_t\}, -y_{t-1} + Y_T + \bar{D}_{tT}\}, \max\{-U_0 + u_{t-1} - z - \bar{B}_{1,t-1}, 0\} \le v \le u_{t-1} + \min\{-U_T + \bar{B}_{tT}, B_t\} - z, \max\{-y_{t-1} + \max\{Y_0 - \bar{D}_{1,t-1}, D_t\} - z, 0\} \le x \\ &\le -y_{t-1} + Y_T - z + \bar{D}_{tT}\}. \end{aligned} \quad (11)$$

The optimal solution of the original problem can then be found via the calculation of $f_1(U_0, Y_0)$ using backward induction.

PROPOSITION 1: The above DP algorithm solves PRP in $O(T \cdot \min\{U_0 + \bar{B}_{1T}, Y_T + \bar{D}_{1T}\} \cdot (U_0 + \bar{B}_{1T})^2 \cdot (Y_T + \bar{D}_{1T})^2)$ time.

PROOF: The complexity of the DP algorithm is clearly dominated by the processing of the recursive relationships. There are $O(T)$ such steps. In each step, $O((U_0 + \bar{B}_{1T}) \cdot (Y_T + \bar{D}_{1T}))$ value functions need to be evaluated. To evaluate each of the value functions, $O(\min\{U_0 + \bar{B}_{1T}, Y_T + \bar{D}_{1T}\} \cdot (U_0 + \bar{B}_{1T}) \cdot (Y_T + \bar{D}_{1T}))$ terms need to be compared. So the overall time complexity of the DP algorithm is $O(T \cdot \min\{U_0 + \bar{B}_{1T}, Y_T + \bar{D}_{1T}\} \cdot (U_0 + \bar{B}_{1T})^2 \cdot (Y_T + \bar{D}_{1T})^2)$. □

3. COMPLEXITY OF THE CONCAVE-COST CASE

For the production planning problem without remanufacturing, good results have been found when costs are concave. Most importantly, Wagner and Whitin [25] gave an $O(T^2)$ solution to a special case in which storage costs are linear and production costs can be partitioned into setup and linear components. Federgruen and Tzur [4], Wagelmans, van Hoesel, and Kolen [23], and Aggarwal and Park [1] all provided $O(T \log T)$ algorithms for this special case. For the general concave-cost problem, Wagner [24] recognized it to be a minimum-concave-cost network flow problem and pointed out that the extreme-point optimal solution must form a spanning tree in the network. This observation led to the discovery by Zangwill [26] of an $O(T^2)$ algorithm for the problem.

However, when there are capacity limits on production, the problem is NP-hard even when demands are equal and storages are costless [7]. The concave-cost PRP (CPRP) should not be much easier than the production planning problem with capacity limits, since demands can be partially met by remanufacturing with the restriction that the

items being remanufactured should first come from the returning used-item flows. Indeed, CPRP is NP-hard, as was proved in Golany, Yang, and Yu [9].

Furthermore, we can show that, even when all costs are stationary, the problem remains NP-hard.

THEOREM 1: The stationary CPRP is NP-hard.

PROOF: We reduce the NP-hard decision version of the SUBSET SUM problem to the stationary CPRP. A description of SUBSET SUM goes as follows: There are $N + 1$ positive integers $a_1, a_2, \ldots, a_N$, and A. The problem is to know whether there is a subset $\mathcal{S}$ of $\{1, 2, \ldots, N\}$ such that $\sum_{n\in\mathcal{S}} a_n = A$.

Given any instance of SUBSET SUM, we define an instance of the stationary CPRP. In this instance, $T = N + 1$; for $n = 1, 2, \ldots, N + 1$, the production cost $P_n(x) = 2 \cdot \mathbf{1}(x \geq 1) + 4 \cdot x$, the new-item inventory holding cost $H_n(y) = y/(2NA)$, the remanufacturing cost $R_n(z) = \mathbf{1}(z \geq 1) + 3 \cdot z$, the returned-item inventory holding cost $W_n(u) = 2 \cdot u$, and the disposal cost $S_n(v) = \mathbf{1}(v \geq 1)$; for $n = 1, 2, \ldots, N$, demand level $D_n = 0$ and returned item quantity $B_n = a_n$; in the last period $N + 1$, demand level $D_{N+1} = A$ and returned item quantity $B_{N+1} = 0$; and the starting and ending inventory levels U_0, U_{N+1}, Y_0, and Y_{N+1} are all 0. Obviously, this reduction can be done in polynomial time. To complete the proof, we shall show that the answer to the SUBSET SUM instance is yes if and only if the optimal cost to the CPRP instance is no more than $N + 3A + 1/2$.

The *if* part: Suppose the CPRP can cost no more than $N + 3A + 1/2$. In view of the production and remanufacturing costs, it takes at least a $3A$ variable cost to satisfy the demand level A in period $N + 1$. Also, this minimum level is achieved when all demands are satisfied with remanufacturing, while the level is at least $3A + 1$ when this is not true. On the other hand, for every period $n = 1, \ldots, N$, it takes at least 1 in costs other than the variable remanufacturing cost to process all the returned items in that period, through remanufacturing, disposal, or holding to the next period; the sum of these N periods' costs achieves the minimum level N when all returned items in any of these periods are either remanufactured altogether or disposed of altogether, and is at least $N + 1$ when this is not true. Therefore, it must be true that the total returned-item quantity over certain periods among the first N periods equals exactly the demand level A in period $N + 1$, and these certain periods are exactly the periods where remanufacturing activities take place. These certain periods form the subset $\mathcal{S}$ that is required for the SUBSET SUM instance.

The *only if* part: Suppose there is a subset $\mathcal{S}$ of $\{1, 2, \ldots, N\}$ such that $\sum_{n\in\mathcal{S}} a_n = A$. For periods in $\mathcal{S}$, we remanufacture all their returned items. For periods outside $\mathcal{S}$, we dispose of all their returned items. The remanufactured items, after being stored, satisfy the demands in period $N + 1$ exactly. Hence, there is no need to have any production run. So the CPRP instance has a solution which costs N in remanufacturing setup or disposal cost, $3A$ in variable remanufacturing cost, at most 1/2 in new-item inventory holding cost, and incurs no other costs; hence its optimal cost is no more than $N + 3A + 1/2$. □

4. THE OPTIMAL SOLUTION STRUCTURE

Given all the flows in the CPRP network over the time horizon $\{1, \ldots, T\}$, there always exists for some k a k-partition of the set $\{1, \ldots, T\}$ into sets $\{T_0 + 1 \equiv 1, \ldots, T_1\}, \{T_1 + 1, \ldots, T_2\}, \ldots, \{T_{k-1} + 1, \ldots, T_k \equiv T\}$, such that there is no flow on any inventory arc crossing between periods T_i and $T_i + 1$ for any i between 1 and $k - 1$ while there is a positive flow on at least one inventory arc crossing between periods t and $t + 1$ for any other t. So, we may say that an optimal solution for CPRP optimally partitions the original time horizon into subhorizons and, in each subhorizon, it solves the corresponding subproblem optimally.

In a given subhorizon $\{t, \ldots, t'\}$, where $t' \geq t$, our subproblem is that of finding the optimal subsolution for periods $t, \ldots, t'$, where it is required that there is no flow on any inventory arc through either boundary of the subhorizon and there is a positive flow on at least one inventory arc in any internal interval in the subhorizon. Due to the absence of flows on inventory arcs through its boundaries, the cost of each subproblem that is to be optimized is well defined.

When the subproblem in every subhorizon $\{t, \ldots, t'\}$ is solved and the optimal cost is found to be $C(t, t')$, we may find the optimal solution using dynamic programming. Define $F(t)$ for t between 1 and $T + 1$ to be the optimal cost for CPRP being constrained to periods $t, t + 1, \ldots, T$. Then, we have

$$F(T + 1) = 0 \tag{12}$$

and

$$F(t) = \min_{t'=t}^{T}\{C(t, t') + F(t' + 1)\} \qquad \forall\, t = T, T - 1, \ldots, 1. \tag{13}$$

$F(1)$ gives the optimal cost, and the way to find the optimal partition of the entire horizon is obvious. This process takes $O(T^2)$ time. However, we will see that it requires an amount

of time that is exponential to $t' - t$ to solve the subproblem on subhorizon $\{t, \ldots, t'\}$ to obtain $C(t, t')$.

As a minimum concave-cost network flow problem, a minimization problem with a concave objective function and a convex feasible region, CPRP must have a spanning tree as the basis of an optimal solution since these spanning trees correspond to extreme points of the convex feasible region. A spanning tree is a subgraph of the CPRP network that constitutes a tree when the directions of its arcs are ignored. In the following, we elaborate on a removal procedure which associates every node with a unique arc in any spanning tree of the CPRP network except the node O. The associations in the same periods then lead to the formation of flow patterns, the building blocks of extreme-point solutions to CPRP.

For a tree, it is easy to see that there are at least two nodes with degree 1. After the removal of one of the nodes along with the unique arc attached to it, the remaining graph is still a tree. This removal procedure can keep on going until what remains is only one single node. Since, at each stage during the procedure, there are always at least two candidate nodes for removal, we can designate beforehand any node to be the one that remains at the end. For a removal procedure to be operated on a spanning tree of the CPRP network, we may let O be this node. This procedure then associates any other node with one unique arc in the spanning tree. Each node I_t may be associated with one of four possible arcs: z_t, u_{t-1}, u_t, and v_t; and each node J_t may also be associated with one of four possible arcs: x_t, y_{t-1}, y_t, and z_t. Thus, there are $4 \times 4 - 1 = 15$ ways, or 15 flow patterns, in which an (I_t, J_t) pair can be associated with different pairs of arcs. They are:

(zx): I_t with z_t and J_t with x_t;
$(z\bar{y})$: I_t with z_t and J_t with y_{t-1};
(zy): I_t with z_t and J_t with y_t;
$(\bar{u}x)$: I_t with u_{t-1} and J_t with x_t;
$(\bar{u}\bar{y})$: I_t with u_{t-1} and J_t with y_{t-1};
$(\bar{u}y)$: I_t with u_{t-1} and J_t with y_t;
$(\bar{u}z)$: I_t with u_{t-1} and J_t with z_t;
(ux): I_t with u_t and J_t with x_t;
$(u\bar{y})$: I_t with u_t and J_t with y_{t-1};
(uy): I_t with u_t and J_t with y_t;
(uz): I_t with u_t and J_t with z_t;
(vx): I_t with v_t and J_t with x_t;
$(v\bar{y})$: I_t with v_t and J_t with y_{t-1};
(vy): I_t with v_t and J_t with y_t;
(vz): I_t with v_t and J_t with z_t.

A spanning-tree solution for the whole-horizon network is merely a combination of T flow patterns—one for each period. The subgraph resulting from limiting this spanning tree to the nodes I_t, J_t, $\ldots$, $I_{t'}$, $J_{t'}$, O of a subhorizon

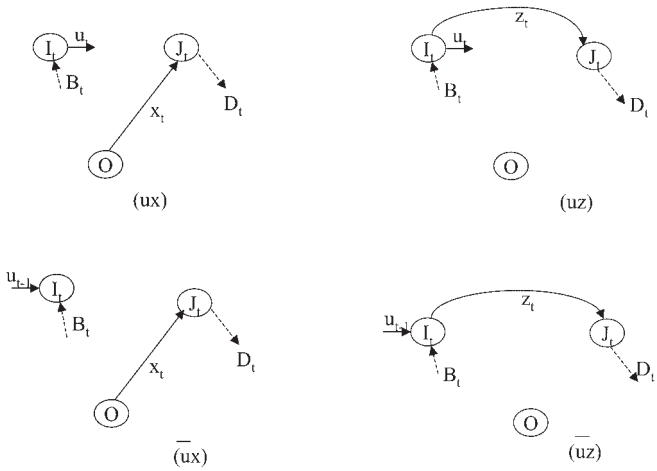

Figure 2. Flow patterns I.

$\{t, \ldots, t'\}$ must form a spanning tree as well, since by the definition of a subhorizon, there can be no connection between it and the rest of the tree. So, the basic subsolution on subhorizon $\{t, \ldots, t'\}$ can be described by a sequence of $t' - t + 1$ flow patterns. However, not all $15^{t'-t+1}$ possible sequences can be counted as basic subsolutions.

Figures 2–5 show all the patterns. From now on, by a sequence, we mean a series of flow patterns residing at consecutive time periods that, when no cycles are formed (acyclic), make up a spanning tree, i.e., the basis of a subsolution of CPRP, on a subhorizon. Here again, as in the definition of a spanning tree, we ignore the directions of arcs when judging whether a cycle has been formed. From the above definitions, two flow patterns can appear in a common sequence consecutively if and only if they are connected by at least one inventory arc and have no redundancy

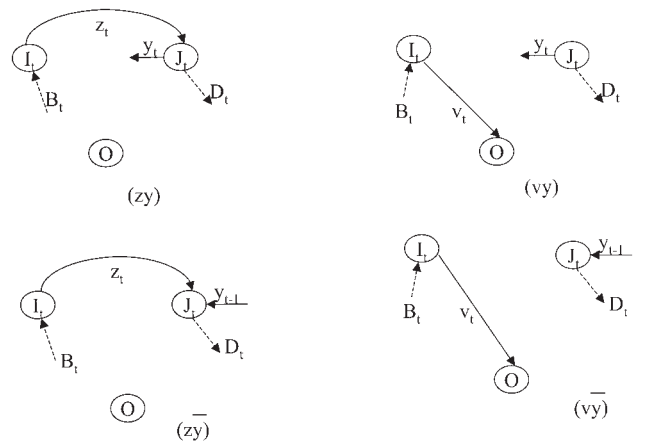

Figure 3. Flow patterns II.

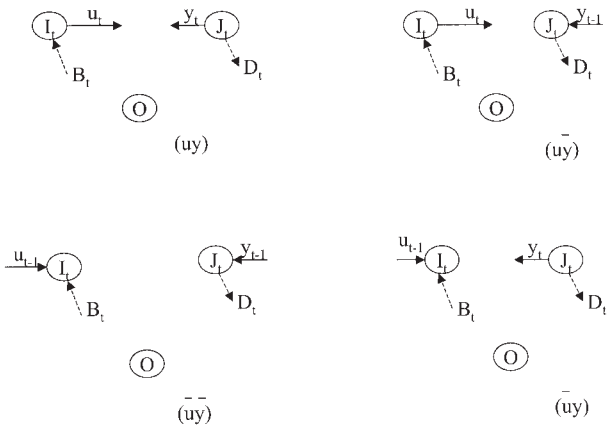

Figure 4. Flow patterns III.

in arcs.[1] Table 1 presents the exhaustive result regarding whether a column pattern can immediately follow a row pattern in a sequence. In the table, a "Y" entry means this is possible; a "d" entry means the two periods are not connected by any inventory arc from the two patterns; and an "r" entry means the two patterns have redundant arcs.

Not all flow patterns can start or end a sequence by our definition of a subhorizon. Hence we have the following.

PROPOSITION 2: A sequence must start with one of the patterns (ux), (uz), (zy), (vy), (uy), (zx), (vx), or (vz) and end with one of the patterns $(\bar{u}x)$, $(\bar{u}z)$, $(z\bar{y})$, $(v\bar{y})$, $(\bar{u}\bar{y})$, (zx), (vx), or (vz). However, the (vy) or (uy) pattern can only start the first sequence in the case of $Y_0 \geq D_1 + 1$, while the $(\bar{u}x)$ or $(\bar{u}\bar{y})$ pattern can only end the last sequence in the case of $U_T \geq B_T + 1$.

PROOF: Any pattern with a $\bar{u}$ or $\bar{y}$ element cannot start a sequence, since it imports items from preceding periods; while any pattern with a u or y element cannot end a sequence, since it exports items to subsequent periods. However, were its level positive, the current-period demand cannot be satisfied when a sequence is started with a (vy) or (uy) pattern, while were its level positive, the current-period returned item supply cannot be depleted when a sequence is ended with a $(\bar{u}x)$ or $(\bar{u}\bar{y})$ pattern. □

From now on, we consider only typical sequences which neither start with a (vy) or (uy) pattern nor end with a $(\bar{u}x)$ or $(\bar{u}\bar{y})$ pattern. Now, we let I^0 be the set of all 15 flow patterns, I^0_S the set of patterns that can start a sequence, i.e., $I^0_S = \{(ux), (uz), (zy), (zx), (vx), (vz)\}$, and I^0_E the set of patterns that can end a sequence, i.e., $I^0_E = \{(\bar{u}z), (z\bar{y}), (v\bar{y}), (zx), (vx), (vz)\}$.

[1] Redundancy in arcs occurs when two consecutive patterns happen to use the same arc. For instance, if a pattern at t is (ux) and pattern at $t + 1$ is $(\bar{u}x)$, then both patterns use the u_t arc.

5. THE "BEST PRACTICAL" POLYNOMIAL-TIME HEURISTIC

We now propose a polynomial-time heuristic for CPRP using the flow patterns and the sequence structure. The rationale behind this heuristic is that a similar structure also appears, as we have seen, in CPRP's extreme-point optimal solutions. In a nutshell, our heuristic procedure optimally divides a problem into subproblems defined over subhorizons, and in each subhorizon, it seeks the best acyclic sequence made up of flow patterns that can be found in polynomial time.

5.1. More about Patterns and Sequences

For any set A, we use $|A|$ to denote its cardinality. For any $i, j \in I^0$, let $a(i, j)$ be 1 when the (i, j) entry in Table 1 is "Y" and 0 otherwise. We let a semisequence be an ordered set $P = [i^P_1, i^P_2, \ldots, i^P_{|P|}]$ of patterns that are connected and have no redundancy in arcs, i.e., an ordered set P with $\prod_{k=1}^{|P|-1} a(i^P_k, i^P_{k+1}) = 1$. For a semisequence P and a set $I \subset I^0$, we write $P \subset I$ when all patterns in P belong to I. Now, any sequence P is merely a semisequence whose first member pattern can start a sequence and last member pattern can end a sequence: $i^P_1 \in I^0_S$ and $i^P_{|P|} \in I^0_E$.

Let $\mathcal{I}_1$ be all subsets of I^0 whose member patterns can form at least one sequence. For any set $I \in \mathcal{I}_1$, we say a given member pattern $i \in I$ is contributing if and only if it helps form a sequence, i.e., there exists a sequence $P \subset I$ and an integer $k \in \{1, \ldots, |P|\}$ such that $i = i^P_k$. For a given $I \in \mathcal{I}_1$, we say I is compact if and only if every pattern belonging to it is contributing. Also, for a given $I \in$

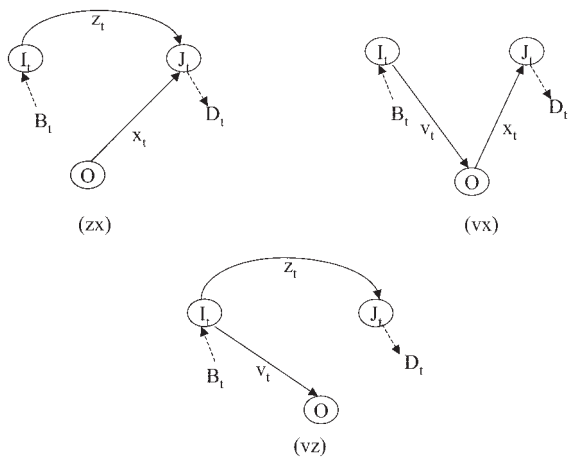

Figure 5. Flow patterns IV.

Table 1. Feasible pattern transitions for CPRP.

	(ux)	(uz)	$(\bar{u}x)$	$(\bar{u}z)$	(zy)	(vy)	$(z\bar{y})$	$(v\bar{y})$	(uy)	$(u\bar{y})$	$(\bar{u}\bar{y})$	$(\bar{u}y)$	(zx)	(vx)	(vz)
(ux)	Y	Y	r	r	Y	Y	Y	Y	Y	Y	r	r	Y	Y	Y
(uz)	Y	Y	r	r	Y	Y	Y	Y	Y	Y	r	r	Y	Y	Y
$(\bar{u}x)$	d	d	Y	Y	d	d	Y	Y	d	Y	Y	Y	d	d	d
$(\bar{u}z)$	d	d	Y	Y	d	d	Y	Y	d	Y	Y	Y	d	d	d
(zy)	Y	Y	Y	Y	Y	Y	r	r	Y	r	r	Y	Y	Y	Y
(vy)	Y	Y	Y	Y	Y	Y	r	r	Y	r	r	Y	Y	Y	Y
$(z\bar{y})$	d	d	Y	Y	d	d	Y	Y	d	Y	Y	Y	d	d	d
$(v\bar{y})$	d	d	Y	Y	d	d	Y	Y	d	Y	Y	Y	d	d	d
(uy)	Y	Y	r	r	Y	Y	r	r	Y	r	r	r	Y	Y	Y
$(u\bar{y})$	Y	Y	r	r	Y	Y	Y	Y	Y	Y	r	r	Y	Y	Y
$(\bar{u}\bar{y})$	d	d	Y	Y	d	d	Y	Y	d	Y	Y	Y	d	d	d
$(\bar{u}y)$	Y	Y	Y	Y	Y	Y	r	r	Y	r	r	Y	Y	Y	Y
(zx)	d	d	Y	Y	d	d	Y	Y	d	Y	Y	Y	d	d	d
(vx)	d	d	Y	Y	d	d	Y	Y	d	Y	Y	Y	d	d	d
(vz)	d	d	Y	Y	d	d	Y	Y	d	Y	Y	Y	d	d	d

$\mathcal{I}_1$ and a positive integer τ, we let $N_I(\tau)$ be the total number of distinct sequences of length τ that patterns in I can form. For any compact $I \in \mathcal{I}_1$, we say it is polynomial when for a sufficiently large $\alpha \in [0, +\infty)$, it follows that

$$\limsup_{\tau \to +\infty} \frac{N_I(\tau)}{\tau^\alpha} < +\infty;$$

and we say I is exponential when the opposite is true: No matter how large $\alpha \in [0, +\infty)$ is, it always follows that

$$\limsup_{\tau \to +\infty} \frac{N_I(\tau)}{\tau^\alpha} = +\infty.$$

The following proposition gives an easy-to-check sufficient and necessary condition for a compact set I to be exponential or polynomial.

PROPOSITION 3: Given a compact $I \in \mathcal{I}_1$, the following three statements are equivalent:

1. I is exponential.
2. There exist $i \in I$ and two different semisequences with members in I, both starting and ending at i.
3. There exist i, j_1, $j_2 \in I$, such that $j_1 \neq j_2$, $a(i, j_1) = a(i, j_2) = 1$, and there exist semisequences $P_1 = [i_1^{P_1} = j_1, i_2^{P_1}, \ldots, i_{|P_1|}^{P_1} = i] \subset I$ and $P_2 = [i_1^{P_2} = j_2, i_2^{P_2}, \ldots, i_{|P_2|}^{P_2} = i] \subset I$.

PROOF: $1 \rightarrow 2$: We prove this by contradiction. Suppose that, for any $i \in I$, there is at most one semisequence with members in I that starts from and ends at i.

Let us introduce an artificial pattern i^* outside of I, such that $a(i^*, i^*) = 1$, and, for any $i \in I$, $a(i, i^*) = 1$ and $a(i^*, i) = 0$. We can append to the end of a semisequence an arbitrary number of i^* patterns. Now, for $n = 1, \ldots, |I|$, let $N_I(n, \tau)$ be the maximum number of semisequences of length τ that any n patterns in I along with i^* can form while without the i^* pattern being the leading pattern. We apparently have $N_I(\tau) \leq N_I(|I|, \tau)$ and $N_I(n, 1) = n$. For n distinct patterns $i_1, i_2, \ldots,$ and i_n in I, let us consider a semisequence that starts from i_1 and in total contains τ patterns in the set $\{i_1, \ldots, i_n, i^*\}$. In this semisequence, either i_1 reappears somewhere after the first position or i_1 never appears again after the first position. In the former case, the portion of the semisequence between the first two i_1 patterns is uniquely determined due to the result just derived in the above, and the portion from the second i_1 on can be expanded to a semisequence that starts from i_1 and in total contains $\tau - 1$ patterns in the set $\{i_1, \ldots, i_n, i^*\}$, while, in the latter case, the portion of the semisequence from the second pattern on is either a semisequence that starts from one of the patterns in $\{i_2, \ldots, i_n\}$ and in total contains $\tau - 1$ patterns in the set $\{i_2, \ldots, i_n, i^*\}$ or a semisequence of $\tau - 1$ consecutive i^* patterns. Since, by definition,

$$N_I(n, \tau) = \max_{\{i_1, \ldots, i_n\} \subset I} \sum_{k=1}^{n} (\text{number of semisequences that start from } i_k \text{ and in total contain } \tau \text{ patterns in } \{i_1, \ldots, i_n, i^*\}),$$

we can conclude from the above that

$$N_I(n, \tau) \leq N_I(n, \tau - 1) + (n - 1) \times N_I(n - 1, \tau - 1) + 1 \qquad \forall\, n = 2, \ldots, |I|. \quad (14)$$

Define $f(n, \tau)$ for $n, \tau = 1, 2, \ldots$ so that

$$\begin{cases} f(n, 1) = f(1, n) = n & \forall\, n = 1, 2, \ldots, \\ f(n, \tau) = f(n, \tau - 1) & \\ \quad + (n - 1) & \\ \quad \times f(n - 1, \tau - 1) + 1 & \forall\, n, \tau = 2, 3, \ldots. \end{cases} \tag{15}$$

Using induction, we can easily show that $N_I(n, \tau) \le f(n, \tau)$. After some meticulous effort, we further find that

$$f(n, \tau) = \sum_{m=0}^{n-2} \left(\frac{(n-1)!}{(n-m-1)!} \times \left(\frac{(n-m)\cdot(\tau-1)!}{m!\cdot(\tau-m-1)!} + \sum_{l=m}^{\tau-2} \frac{l!}{m!\cdot(l-m)!} \right) \right) + \sum_{m=n-1}^{\tau-1} \frac{m!}{(m-n+1)!} \tag{16}$$

when $\tau \ge n$, and

$$f(n, \tau) = \sum_{m=0}^{\tau-2} \left(\frac{(n-1)!}{(n-m-1)!} \times \left(\frac{(n-m)\cdot(\tau-1)!}{m!\cdot(\tau-m-1)!} + \sum_{l=m}^{\tau-2} \frac{l!}{m!\cdot(l-m)!} \right) \right) + \frac{(n-\tau+1)\cdot(n-1)!}{(n-\tau)!} \tag{17}$$

when $\tau < n$. So when τ is large enough, we have $f(n, \tau) \le (n+1)! \cdot \tau^{n-1}$. For such a τ,

$$N_I(\tau) \le N_I(|I|, \tau) \le f(|I|, \tau) \le (|I|+1)! \cdot \tau^{|I|-1}. \tag{18}$$

Therefore, as long as $\alpha \ge |I| - 1$, we have

$$\limsup_{\tau \to +\infty} \frac{N_I(\tau)}{\tau^\alpha} < +\infty. \tag{19}$$

$2 \to 3$: Suppose for some $i \in I$, there are two different semisequences $P_1 = [i_1^{P_1} = i, \ldots, i_{|P_1|}^{P_1} = i]$ and $P_2 = [i_1^{P_2} = i, \ldots, i_{|P_2|}^{P_2} = i]$. Then, there must be a $k = 1, \ldots, \min\{|P_1|, |P_2|\} - 2$ such that $i_l^{P_1} = i_l^{P_2}$ for $l = 1, \ldots, k$ and yet $i_{k+1}^{P_1} \ne i_{k+1}^{P_2}$. We then have two semisequences starting differentially at $i_{k+1}^{P_1}$ and $i_{k+1}^{P_2}$ while ending at the same $i_k^{P_1(2)}$; these two semisequences are $[i_{k+1}^{P_1}, i_{k+2}^{P_1}, \ldots, i_{|P_1|}^{P_1} = i = i_1^{P_1(2)}, i_2^{P_1(2)}, \ldots, i_k^{P_1(2)}]$ and $[i_{k+1}^{P_2}, i_{k+2}^{P_2}, \ldots, i_{|P_2|}^{P_2} = i = i_1^{P_1(2)}, i_2^{P_1(2)}, \ldots, i_k^{P_1(2)}]$.

$3 \to 1$: Let M be the least common multiplier of $|P_1|$ and $|P_2|$. There are at least two semisequences of length M that involve only patterns in I: that of repeating path P_1 for $M/|P_1|$ times and that of repeating path P_2 for $M/|P_2|$ times. Since I is compact, i must be contributing. Let $P = [i_1^P, \ldots, i_k^P = i, \ldots, i_{|P|}^P]$ be a sequence which defines that i is contributing. Then, for any nonnegative integer m, we can construct 2^m different sequences of length $|P| + m \times M$ using patterns belonging to I: The first k members of these sequences all agree with the first k members of P, then there are 2^m variations of using the two types of aforementioned semisequences of length M to cover a length of $m \times M$, and the last $|P| - k$ members of all the sequences agree with the last $|P| - k$ members of P. Therefore,

$$N_I(|P| + m \times M) \ge 2^m. \tag{20}$$

Hence,

$$\limsup_{\tau \to +\infty} N_I(\tau) \ge 2^{-|P|/M} \times (2^{1/M})^\tau, \tag{21}$$

and, for any $\alpha \ge 0$, we have

$$\limsup_{\tau \to +\infty} \frac{N_I(\tau)}{\tau^\alpha} = +\infty. \qquad \square \tag{22}$$

A corollary of Proposition 3 is immediately in order.

COROLLARY 1: If a compact I is polynomial, then no two-element subset $\{i, j\}$ of I can be such that $a(i, i) = a(i, j) = a(j, i) = 1$.

For any $I \subset I^0$, we use $A(I)$ to denote the matrix that comprises all the $a(i, j)$'s for $i, j \in I$. Also, define $I^1 = \{(ux), (uz), (zy), (vy), (uy)\}$ and $I^2 = \{(\bar{u}x), (\bar{u}z), (z\bar{y}), (v\bar{y}), (\bar{u}\bar{y})\}$.

PROPOSITION 4: Suppose I is a polynomial subset of I^0; then:

1. At most one pattern in I^1 can be in I and at most one pattern in I^2 can be in I.
2. Neither $(u\bar{y})$ nor $(\bar{u}y)$ can be in I.

PROOF: According to Table 1, $A(I^1)$ and $A(I^2)$ are both all 1's. We can reach 1 by Corollary 1.

To prove 2, first note that by Table 1 and Corollary 1, for the polynomial I to contain $(u\bar{y})$, it cannot contain patterns other than $(\bar{u}x)$, $(\bar{u}z)$, (zy), (vy), (uy), $(\bar{u}\bar{y})$, or $(\bar{u}y)$. On the other hand, to start a sequence with, I must contain (zy), while to end a sequence with, it must contain $(\bar{u}z)$. However, using Table 1 and Corollary 1 again, we find that I cannot contain $(\bar{u}z)$ were it to contain $(\bar{u}x)$, $(\bar{u}\bar{y})$, or $(\bar{u}y)$, and cannot contain (zy) were it to contain (vy), (uy), or

$(\bar{u}y)$. Therefore, I can only contain the three patterns (zy), $(u\bar{y})$, and $(\bar{u}z)$. But, from checking Table 1, we know that no sequence can be made out of it.

Similarly, by Table 1 and Corollary 1, we know that for the polynomial I to contain $(\bar{u}y)$, it cannot contain patterns other than (ux), (uz), $(z\bar{y})$, $(v\bar{y})$, (uy), $(u\bar{y})$, or $(\bar{u}\bar{y})$. On the other hand, to start a sequence with, I must contain either (ux) or (uz), but not both; while to end a sequence with, it must contain either $(z\bar{y})$ or $(v\bar{y})$, but not both. However, using Table 1 and Corollary 1 again, we find that I cannot contain (ux) or (uz) were it to contain (uy) or $(u\bar{y})$, and cannot contain $(z\bar{y})$ or $(v\bar{y})$ were it to contain $(u\bar{y})$ or $(\bar{u}\bar{y})$. Therefore, I can only contain three patterns: one from between (ux) and (uz), $(\bar{u}y)$, and one from between $(z\bar{y})$ and $(v\bar{y})$. But from checking Table 1, we know that no sequence can be made out of it. □

Proposition 4 helps to answer our earlier question as to why there exist an exponential number of basic subsolutions, i.e., acyclic sequences, on a given subhorizon: Just consider all the sequences that can be formed by the three patterns (ux), (zy), and $(z\bar{y})$. Both (ux) and (zy) can start a sequence, they are both in I^1, $(z\bar{y})$ can end a sequence, $a((ux), (z\bar{y})) = 1$, and the sequences these patterns form are all acyclic. Hence there are an exponential number of basic subsolutions formed by the (ux), (zy), and $(z\bar{y})$ patterns alone.

5.2. Maximally Polynomial Sets and the Heuristic

We say that I is maximally polynomial if any compact I' containing I is exponential. We are now in a position to identify all maximally polynomial sets.

PROPOSITION 5: CPRP has exactly nine maximally polynomial sets. Each such set contains five patterns, one of (ux), (uz), and (zy), all (zx), (vx), and (vz), and one of $(\bar{u}z)$, $(z\bar{y})$, and $(v\bar{y})$.

PROOF: Denote a maximally polynomial set by I. By Proposition 4, I does not contain either $(u\bar{y})$ or $(\bar{u}y)$. By Proposition 2, I must contain at least one pattern in I_S^0 to start a sequence with. By Proposition 4, if I contains any one of (ux), (uz), or (zy), it cannot contain any other member in I^1. Symmetrically, I must contain at least one pattern in I_E^0, and if I contains any one of $(\bar{u}z)$, $(z\bar{y})$, or $(v\bar{y})$, it cannot contain any other member in I^2. By Table 1, a sequence started with a number of uniform (ux), (uz), or (zy) patterns can be followed by one pattern of (zx), (vx), or (vz), and then be ended with a number of uniform $(\bar{u}z)$, $(z\bar{y})$, or $(v\bar{y})$ patterns. Also, the (ux) patterns can be directly followed by the $(z\bar{y})$ or $(v\bar{y})$ patterns, the (uz) patterns can be directly followed by the $(z\bar{y})$ or $(v\bar{y})$ patterns, and the (zy) patterns can be directly followed by the $(\bar{u}x)$ or $(\bar{u}z)$ patterns. However, according to Table 1 and Corollary 1, $(\bar{u}x)$ cannot be followed by any pattern that can be contained in a polynomial set that has already contained (zy), while it itself is not in I_E^0. For the special case where the length of the sequence is only 1, the sequence can only be made up of one pattern of (zx), (vx), or (vz). Since we want I to be maximal, we therefore have the nine possibilities outlined in the statement. □

On the subhorizon from period t to t', noting that the two types of sequences produced by following (uz) patterns with $(z\bar{y})$ patterns and following (zy) patterns with $(\bar{u}z)$ patterns, respectively, are cyclic, the nine maximally polynomial sets can generate 30 types of different acyclic sequences when $t' \geq t + 1$. In the following lines we present all these types. For each type, we list in series the flow patterns and the corresponding starting and ending periods these patterns reside at. The types of acyclic sequences are:

Type 1, the $(ux)[t, t + k - 1](zx)[t + k, t + k](\bar{u}z)[t + k + 1, t']$ type;
Type 2, the $(ux)[t, t + k - 1](vx)[t + k, t + k](\bar{u}z)[t + k + 1, t']$ type;
Type 3, the $(ux)[t, t + k - 1](vz)[t + k, t + k](\bar{u}z)[t + k + 1, t']$ type;
Type 4, the $(ux)[t, t + k - 1](zx)[t + k, t + k](z\bar{y})[t + k + 1, t']$ type;
Type 5, the $(ux)[t, t + k - 1](vx)[t + k, t + k](z\bar{y})[t + k + 1, t']$ type;
Type 6, the $(ux)[t, t + k - 1](vz)[t + k, t + k](z\bar{y})[t + k + 1, t']$ type;
Type 7, the $(ux)[t, t + k - 1](zx)[t + k, t + k](v\bar{y})[t + k + 1, t']$ type;
Type 8, the $(ux)[t, t + k - 1](vx)[t + k, t + k](v\bar{y})[t + k + 1, t']$ type;
Type 9, the $(ux)[t, t + k - 1](vz)[t + k, t + k](v\bar{y})[t + k + 1, t']$ type;
Type 10, the $(uz)[t, t + k - 1](zx)[t + k, t + k](\bar{u}z)[t + k + 1, t']$ type;
Type 11, the $(uz)[t, t + k - 1](vx)[t + k, t + k](\bar{u}z)[t + k + 1, t']$ type;
Type 12, the $(uz)[t, t + k - 1](vz)[t + k, t + k](\bar{u}z)[t + k + 1, t']$ type;
Type 13, the $(uz)[t, t + k - 1](zx)[t + k, t + k](z\bar{y})[t + k + 1, t']$ type;
Type 14, the $(uz)[t, t + k - 1](vx)[t + k, t + k](z\bar{y})[t + k + 1, t']$ type;
Type 15, the $(uz)[t, t + k - 1](vz)[t + k, t + k](z\bar{y})[t + k + 1, t']$ type;
Type 16, the $(uz)[t, t + k - 1](zx)[t + k, t + k](v\bar{y})[t + k + 1, t']$ type;
Type 17, the $(uz)[t, t + k - 1](vx)[t + k, t +$

$k](v\bar{y})[t + k + 1, t']$ type;

Type 18, the $(uz)[t, t + k - 1](vz)[t + k, t + k](v\bar{y})[t + k + 1, t']$ type;

Type 19, the $(zy)[t, t + k - 1](zx)[t + k, t + k](\bar{u}z)[t + k + 1, t']$ type;

Type 20, the $(zy)[t, t + k - 1](vx)[t + k, t + k](\bar{u}z)[t + k + 1, t']$ type;

Type 21, the $(zy)[t, t + k - 1](vz)[t + k, t + k](\bar{u}z)[t + k + 1, t']$ type;

Type 22, the $(zy)[t, t + k - 1](zx)[t + k, t + k](z\bar{y})[t + k + 1, t']$ type;

Type 23, the $(zy)[t, t + k - 1](vx)[t + k, t + k](z\bar{y})[t + k + 1, t']$ type;

Type 24, the $(zy)[t, t + k - 1](vz)[t + k, t + k](z\bar{y})[t + k + 1, t']$ type;

Type 25, the $(zy)[t, t + k - 1](zx)[t + k, t + k](v\bar{y})[t + k + 1, t']$ type;

Type 26, the $(zy)[t, t + k - 1](vx)[t + k, t + k](v\bar{y})[t + k + 1, t']$ type;

Type 27, the $(zy)[t, t + k - 1](vz)[t + k, t + k](v\bar{y})[t + k + 1, t']$ type;

Type 28, the $(ux)[t, t + k](z\bar{y})[t + k + 1, t']$ type;

Type 29, the $(ux)[t, t + k](v\bar{y})[t + k + 1, t']$ type;

Type 30, the $(uz)[t, t + k](v\bar{y})[t + k + 1, t']$ type.

In the above, k is an integer, ranging from 0 to $t' - t$ for the first 27 types, and from 0 to $t' - t - 1$ for the last three types. Therefore, in total, we have $30 \times (t' - t) + 3$ different acyclic sequences. When $t' = t$, we have three different acyclic sequences: $(zx)[t, t]$, $(vx)[t, t]$, and $(vz)[t, t]$. We let $C^H(t, t)$ be the minimum cost among the costs of the three sequences. For the more general case where $t' \geq t + 1$, we let $C_i^H(t, t', k)$ be the cost of the Type i sequence at k where either $i = 1, 2, \ldots, 27$ and $k = 0, 1, \ldots, t' - t$, or $i = 28, 29, 30$ and $k = 0, 1, \ldots, t' - t - 1$. We define $C^H(t, t')$ as follows:

$$C^H(t, t') = \min\left\{ \min_{i=1}^{27} \min_{k=0}^{t'-t} C_i^H(t, t', k), \min_{i=28}^{30} \min_{k=0}^{t'-t-1} C_i^H(t, t', k) \right\}. \quad (23)$$

The reader may refer to the appendix for the expressions of the $c^H(t, t)$'s and the $c_i^H(t, t', k)$'s for $t' \geq t + 1$.

Now, we define $F^H(t)$ as $F(t)$ is in (12) and (13), except that we replace the optimal subhorizon cost $C(t, t')$ with $C^H(t, t')$. Solving for $F^H(1)$ to obtain a heuristic solution for CPRP constitutes our heuristic. We consider the heuristic to be the "best practical" for CPRP because the acyclic sequences it examines in each subhorizon are all that the maximally polynomial sets can generate, while any meaningful addition to a maximally polynomial set will result in an exponential number of different sequences.

PROPOSITION 6: The heuristic runs in $O(T^4)$ time.

PROOF: Each $C^H(t, t)$ can be evaluated in $O(1)$ time. For $t' \geq t + 1$, each $C_i^H(t, t', k)$ can be evaluated in $O(t' - t)$ time. So each $C^H(t, t')$ can be calculated in $O((t' - t)^2)$ time, and the $C^H(t, t')$'s can be calculated in $O(T^4)$ time. Since the DP algorithm runs only in $O(T^2)$ time, we see that the entire heuristic runs in $O(T^4)$ time. □

6. A COMPUTATIONAL STUDY

6.1. The Benchmark Solution and a MIP Formulation

To put the results obtained from the different methods in perspective, we introduce a "benchmark" solution—the best solution that can be obtained without invoking remanufacturing. We generate the benchmark solution through two Wagner-Whitin style dynamic programming algorithms where the first algorithm finds the cheapest way to satisfy demands using production and the second one finds the cheapest way to get rid of used items through disposal. The sum of the two costs constitutes the benchmark cost. In particular, redefine D_1, D_T, B_1, and B_T to be $D_1 - Y_0$, $D_T + Y_T$, $B_1 + U_0$, and $B_T - U_T$, respectively; compute the $\bar{D}_{tt'}$'s and $\bar{B}_{tt'}$'s based on the new parameters; and then let

$$\begin{cases} C_D(t, t') = P_t(\bar{D}_{tt'}) + \sum_{\tau=t}^{t'-1} H_\tau(\bar{D}_{\tau+1,t'}), \\ C_B(t, t') = \sum_{\tau=t}^{t'-1} W_\tau(\bar{B}_{t\tau}) + S_{t'}(\bar{B}_{tt'}), \end{cases} \quad (24)$$

and define $F_D(t)$ and $F_B(t)$ so that the $F_D(t)$'s and $C_D(t, t')$'s and, respectively, the $F_B(t)$'s and $C_B(t, t')$'s have the same relations as those described earlier for the $F(t)$'s and $C(t, t')$'s, which are expressed by (12) and (13). The benchmark cost is simply $F_D(1) + F_B(1)$.

In each of our test instances, the production, remanufacturing and disposal costs are setup-linear, and the two storage costs are linear. Such a problem can be formulated as an integer programming problem (CPRPIP) with $3T$ binary variables:

$$(CPRPIP) \qquad \min \sum_{t=1}^{T} (\Delta_t^P \cdot \alpha_t + \Delta_t^R \cdot \beta_t + \Delta_t^S \cdot \gamma_t + P_t \cdot x_t + R_t \cdot z_t + S_t \cdot v_t) + \sum_{t=1}^{T-1} (H_t \cdot y_t + W_t \cdot u_t) \quad (25)$$

subject to

$$-(\bar{D}_{1T} + \bar{B}_{1T}) \cdot \alpha_t + x_t \le 0 \qquad \forall\, t = 1, \ldots, T, \tag{26}$$

$$-(\bar{D}_{1T} + \bar{B}_{1T}) \cdot \beta_t + z_t \le 0 \qquad \forall\, t = 1, \ldots, T, \tag{27}$$

$$-(\bar{D}_{1T} + \bar{B}_{1T}) \cdot \gamma_t + v_t \le 0 \qquad \forall\, t = 1, \ldots, T, \tag{28}$$

$$x_t + z_t + y_{t-1} - y_t = D_t \qquad \forall\, t = 1, \ldots, T, \tag{29}$$

$$z_t + v_t - u_{t-1} + u_t = B_t \qquad \forall\, t = 1, \ldots, T, \tag{30}$$

$$y_0 = Y_0, \qquad y_T = Y_T, \qquad u_0 = U_0, \qquad u_T = U_T, \tag{31}$$

$$\alpha_t, \beta_t, \gamma_t \in \{0, 1\} \qquad \forall\, t = 1, \ldots, T, \tag{32}$$

$$x_t, z_t, v_t \ge 0 \qquad \forall\, t = 1, \ldots, T, \tag{33}$$

$$y_t, u_t \ge 0 \qquad \forall\, t = 1, \ldots, T - 1. \tag{34}$$

In the above formulation, we have denoted the production setup cost by Δ_t^P, the unit variable production cost by P_t, the remanufacturing setup cost by Δ_t^R, the unit variable remanufacturing cost by R_t, the disposal setup cost by Δ_t^S, the unit variable disposal cost by S_t, the unit per period new item storage cost by H_t, and the unit per period used item storage cost by W_t. Also, α_t is the 0–1 variable indicating whether there is production, β_t the 0–1 variable indicating whether there is remanufacturing, and γ_t the 0–1 variable indicating whether there is disposal, in period t. The rest of the formulation is almost the same as CPRP, except that the cost functions are explicit in (25), and there are the additional constraints (26), (27), and (28), which spell out for production, remanufacturing, and disposal the rule of no-setup-no-activity. We employed the commercial MIP solver CPLEX to solve CPRPIP.

6.2. More about the Computation Setup

For a common instance, we denote the costs obtained from the four methods as c_{BM} (benchmark), c_{DP} (dynamic programming), c_{CP} (CPLEX solving), and c_{HR} (heuristic), respectively. And, we denote the running times of these methods in seconds as t_{BM}, t_{DP}, t_{CP}, and t_{HR}, respectively.

In our study, we let $Y_0 = Y_T = U_0 = U_T = 0$, and all other problem parameters be statistically independent of each other. The production setup cost Δ_t^P is uniformly distributed in the interval $[\bar{\Delta}_L^P, \bar{\Delta}_U^P]$; the unit production cost P_t is uniformly distributed in the interval $[\bar{P}_L, \bar{P}_U]$; the remanufacturing setup cost Δ_t^R is uniformly distributed in $[\bar{\Delta}_L^R, \bar{\Delta}_U^R]$; the unit remanufacturing cost R_t is uniformly distributed in the interval $[\bar{R}_L, \bar{R}_U]$; the disposal setup cost Δ_t^S is uniformly distributed in the interval $[\bar{\Delta}_L^S, \bar{\Delta}_U^S]$; the unit disposal cost S_t is uniformly distributed in the interval $[\bar{S}_L, \bar{S}_U]$; the unit per period new item storage cost H_t is uniformly distributed in the interval $[\bar{H}_L, \bar{H}_U]$; and the unit per period used item storage cost W_t is uniformly distributed in the interval $[\bar{W}_L, \bar{W}_U]$. The demand level D_t is generated from the Poisson random distribution with mean λ_D and the returned item level B_t is generated from the Poisson random distribution with mean λ_B.

At their default levels, we let $\bar{\Delta}_L^P = 10.0$, $\bar{\Delta}_U^P = 15.0$, $\bar{P}_L = 3.0$, $\bar{P}_U = 5.0$, $\bar{\Delta}_L^R = 3.0$, $\bar{\Delta}_U^R = 5.0$, $\bar{R}_L = 1.0$, $\bar{R}_U = 2.0$, $\bar{\Delta}_L^S = 1.0$, $\bar{\Delta}_U^S = 2.0$, $\bar{S}_L = 0.5$, $\bar{S}_U = 1.0$, $\bar{H}_L = 1.0$, $\bar{H}_U = 2.0$, $\bar{W}_L = 0.5$, $\bar{W}_U = 1.0$, $\lambda_D = 10.0$, and $\lambda_B = 5.0$. Our computation was conducted on a Dell Inspiron 4000 Laptop with a 1 GHz CPU speed and the CPLEX version which we used was 7.0. Being extremely conservative about the danger of running out of computing space, we set the $f_t(u, y)$ in the DP algorithm for the general-cost case (Section 2) at $+\infty$ whenever $u > 1000$ or $y > 1000$. Note that, for our ranges of λ_D, λ_B, and T, it is next to impossible that in an optimal solution a storage of no less than 1000 items will be needed for either the new or the used item inventories. So the above limitation almost certainly has no effect on the optimality of our DP implementation.

We first make a rough-cut comparison among all four methods. For different T's, we apply the four methods to the common 10 independently-generated instances. The results are presented in Table 2, where all entries are the sample means from the 10 runs.

Table 2 shows, as expected, that the benchmark method produced poor solutions in very short time. The DP algorithm and CPLEX both produced optimal solutions, while in so doing the former took much more time than the latter (the more so as T gets larger). So, we should avoid using the DP algorithm whenever possible. Note that the equality in the solution costs produced by these two methods helps verify the correctness of the coding in this study. Finally, the heuristic is faster than both of the aforementioned methods and yet produces solutions that are much closer in quality to those produced by the two exact procedure than those obtained by the benchmark method.

Table 2. A comparison among the four methods.

T	t_{BM}	t_{DP}	t_{CP}	t_{HR}	c_{BM}	c_{DP}	c_{CP}	c_{HR}
10	0.00	18.05	0.06	0.01	545.4	355.9	355.9	361.4
12	0.00	48.97	0.07	0.02	658.7	423.8	423.8	431.1
14	0.00	117.93	0.10	0.04	771.2	504.3	504.3	511.3
16	0.00	274.22	0.19	0.07	865.7	560.3	560.3	569.7
18	0.00	587.41	0.25	0.11	966.7	629.2	629.2	639.1
20	0.00	896.08	0.28	0.17	1075.0	698.9	698.9	709.4

6.3. A Computational Study of the Heuristic Method

Now, we concentrate on our main study of comparing the heuristic against CPLEX. Besides the conventional gap of optimality $\epsilon = (c_{HR} - c_{CP})/c_{HR}$, we will also use a new measure $\eta = (c_{HR} - c_{CP})/(c_{BM} - c_{CP})$, whose purpose is to compare the closeness between the heuristic solution and the optimal solution produced by CPLEX to the closeness between the optimal solution without remanufacturing produced by the benchmark method and the optimal solution produced by CPLEX. This measure can be interpreted as the proportion of the benefit of having the remanufacturing option that is still left unexploited by the heuristic. Therefore, we call η the "underexploitation ratio."

In this study, we fixed the relationships $\lambda_D = 2\lambda_B$, $\bar{\Delta}_L^R = \bar{\Delta}_U^R = 3\bar{R}_L = 3\bar{R}_U$, and $\bar{H}_L = \bar{H}_U = 2\bar{W}_L = 2\bar{W}_U$, and left all other parameters regarding the costs of production and disposal at their default values: $\bar{\Delta}_L^P = 10.0$, $\bar{\Delta}_U^P = 15.0$, $\bar{P}_L = 3.0$, $\bar{P}_U = 5.0$, $\bar{\Delta}_L^S = 1.0$, $\bar{\Delta}_U^S = 2.0$, $\bar{S}_L = 0.5$, and $\bar{S}_U = 1.0$. In Table 3, we present, for each set of parameters, the sample averages and sample standard deviations for t_{CP}, t_{HR}, $(100 \times \epsilon)\%$, and $(100 \times \eta)\%$ over 40 independent runs. Each entry for one of the four measures is written in the form of *average* ± *standard deviation*. Also note that we have treated $T = 30$, $\lambda_B = 5$, $\bar{R}_U = 1.5$, and $\bar{W}_U = 0.75$ as "default" parameter values around which all other parameter values revolve.

From Table 3, we see that in most situations, the difference between solutions generated by our heuristic and the true optimal solution is within a 3% factor. The underexploitation ratio of the heuristic is also within the 5% mark for most cases.

Table 3. Comparing the heuristic against CPLEX.

T	λ_B	$\bar{R}_U$	$\bar{W}_U$	t_{CP}	t_{HR}	$(100 \times \epsilon)\%$	$(100 \times \eta)\%$
30	5.0	1.5	0.75	1.06 ± 0.81	0.54 ± 0.01	(1.67 ± 0.69)%	(3.34 ± 1.39)%
30	5.0	1.5	0.25	1.85 ± 1.18	0.54 ± 0.01	(4.94 ± 1.07)%	(10.41 ± 2.01)%
30	5.0	1.5	0.50	2.40 ± 2.22	0.54 ± 0.00	(2.59 ± 0.68)%	(5.24 ± 1.46)%
30	5.0	1.5	1	1.56 ± 3.57	0.54 ± 0.00	(1.12 ± 0.57)%	(2.28 ± 1.18)%
30	5.0	1.5	1.25	1.07 ± 1.13	0.54 ± 0.00	(0.69 ± 0.36)%	(1.42 ± 0.73)%
30	5.0	0.5	0.75	0.75 ± 0.44	0.54 ± 0.00	(1.27 ± 0.83)%	(1.45 ± 0.82)%
30	5.0	1	0.75	1.01 ± 0.94	0.54 ± 0.00	(1.48 ± 0.48)%	(2.28 ± 0.80)%
30	5.0	2	0.75	2.14 ± 2.54	0.55 ± 0.01	(1.40 ± 0.59)%	(3.84 ± 1.62)%
30	5.0	2.5	0.75	1.80 ± 1.17	0.54 ± 0.00	(1.53 ± 0.51)%	(5.96 ± 1.73)%
30	3.0	1.5	0.75	2.77 ± 3.43	0.54 ± 0.00	(3.05 ± 0.91)%	(6.85 ± 2.21)%
30	8.0	1.5	0.75	0.85 ± 0.79	0.54 ± 0.00	(0.91 ± 0.38)%	(1.79 ± 0.76)%
30	10.0	1.5	0.75	0.65 ± 0.80	0.54 ± 0.00	(0.63 ± 0.26)%	(1.18 ± 0.50)%
30	15.0	1.5	0.75	0.46 ± 0.53	0.54 ± 0.00	(0.46 ± 0.21)%	(0.85 ± 0.38)%
40	5.0	1.5	0.75	7.80 ± 10.89	1.64 ± 0.00	(1.49 ± 0.50)%	(3.13 ± 1.06)%
40	5.0	1.5	0.25	11.09 ± 13.46	1.89 ± 0.29	(4.98 ± 0.78)%	(10.60 ± 1.81)%
40	5.0	1.5	0.50	17.24 ± 23.35	1.70 ± 0.00	(2.65 ± 0.76)%	(5.37 ± 1.42)%
40	5.0	1.5	1	5.36 ± 9.22	1.70 ± 0.01	(1.01 ± 0.38)%	(2.04 ± 0.71)%
40	5.0	1.5	1.25	6.22 ± 8.96	1.71 ± 0.01	(0.71 ± 0.28)%	(1.48 ± 0.64)%
40	5.0	0.5	0.75	1.79 ± 1.66	1.71 ± 0.01	(1.17 ± 0.44)%	(1.47 ± 0.56)%
40	5.0	1	0.75	3.19 ± 2.98	1.71 ± 0.01	(1.55 ± 0.50)%	(2.38 ± 0.84)%
40	5.0	2	0.75	8.97 ± 10.33	2.04 ± 0.31	(1.52 ± 0.49)%	(4.00 ± 1.26)%
40	5.0	2.5	0.75	23.90 ± 26.39	2.22 ± 0.24	(1.64 ± 0.47)%	(6.22 ± 1.69)%
40	3.0	1.5	0.75	14.47 ± 14.54	2.16 ± 0.41	(2.82 ± 0.72)%	(6.45 ± 1.78)%
40	8.0	1.5	0.75	2.62 ± 3.42	1.72 ± 0.09	(0.93 ± 0.34)%	(1.78 ± 0.69)%
40	10.0	1.5	0.75	1.60 ± 1.31	1.71 ± 0.01	(0.64 ± 0.32)%	(1.20 ± 0.57)%
40	15.0	1.5	0.75	1.02 ± 1.29	1.71 ± 0.01	(0.39 ± 0.19)%	(0.73 ± 0.36)%
50	5.0	1.5	0.75	35.70 ± 53.09	4.89 ± 0.61	(1.59 ± 0.51)%	(3.24 ± 1.00)%
60	5.0	1.5	0.75	408.62 ± 882.59	11.29 ± 0.04	(1.58 ± 0.41)%	(3.17 ± 0.84)%
70	5.0	1.5	0.75	2318.60 ± 5969.33	20.75 ± 1.12	(1.57 ± 0.44)%	(3.16 ± 0.85)%

As $\bar{W}_U$ grows along with other inventory holding costs, both ϵ and η visibly decrease. A possible reason for this is that, as inventory holding costs increase, in an optimal solution there tend to be more setups and more sequences with shorter lengths; as a consequence, the difference between the (exponential) number of all allowable sequences and the (polynomial) number of sequences considered by the heuristic on each subhorizon tends to be smaller, and therefore the polynomial heuristic mimic the optimal solution even better.

As $\bar{R}_U$ increases along with other costs involving remanufacturing, both t_{CP} and η tend to increase while ϵ remains quite stable. A possible reason for the increasing of t_{CP} is that higher remanufacturing costs lead to fewer remanufacturing setups which in turn causes many more production and disposal setups, and further result in longer computation times. We have η increasing because, when remanufacturing becomes more costly, the benchmark solution becomes closer to the optimal solution; and hence it becomes harder for the heuristic solution to beat the benchmark solution.

As λ_B increases along with λ_D, we see that ϵ and η all drop fairly quickly. We believe that, this is because higher supply and demand levels make four of the six patterns not considered by the heuristic, (uy), $(\bar{u}y)$, $(u\bar{y})$, and $(\bar{u}\bar{y})$, which allow only inventory transfers, less likely to appear in the optimal solution. Hence, the heuristic solution can resemble the true optimal solution more closely.

As T grows, t_{CP} grows much faster than t_{HR}. Also, t_{CP} becomes much more unpredictable (as expressed by the increase in its standard deviation). At the same time, t_{HR} grows at a much slower rate and remains very predictable.

Noting further that CPLEX works only under the setup-linear setting while the heuristic works under all settings, we conclude this section by saying that the heuristic is a stable, efficient, and effective method for solving CPRP. The advantage of the heuristic is particularly evident when the planning horizon is long, system throughput is high, or inventory handling is costly.

7. CONCLUDING REMARKS

We have modeled the production planning problem with remanufacturing using a network flow type formulation and examined its complexity when costs are concave. To overcome the computational difficulty, we have provided an effective polynomial-time heuristic which shares common grounds with the extreme-point optimal solutions. To the best of our knowledge, the various notions which we have used in the construction of the heuristic are novel. Due to their effectiveness in our current case and the prevalence of network flow type formulations, we speculate that these notions may play an important role in future development of algorithms for problems other than ours. As for future research we note that the possible correlations between the returning flows of used items and demand flows await to be exploited.

APPENDIX: THE CALCULATION OF $C^H(T, T)$'S AND $C_I^H(T, T', K)$'S

First, we redefine D_1, D_T, B_1, and B_T to be $D_1 - Y_0$, $D_T + Y_T$, $B_1 + U_0$, and $B_T - U_T$, respectively. Then, we spend $O(T^2)$ time to calculate $\bar{B}_{tt'}$ and $\bar{D}_{tt'}$ for every (t, t') pair. We now have

$$C^H(t, t) = \min\{P_t(D_t - B_t) + R_t(B_t), P_t(D_t) + S_t(B_t), R_t(D_t) + S_t(B_t - D_t)\}. \tag{35}$$

For $t' \geq t + 1$, we have the expression

$$C_i^H(t, t', k) = \sum_{\tau=t}^{t'} R_\tau(z_\tau) + \sum_{\tau=t}^{t'-1} W_\tau(u_\tau) + \sum_{\tau=t}^{t'} S_\tau(v_\tau) + \sum_{\tau=t}^{t'} P_\tau(x_\tau) + \sum_{\tau=t}^{t'-1} H_\tau(y_\tau), \tag{36}$$

which can be evaluated in $O(t' - t)$ time. Hence, the key in calculating $C_i^H(t, t', k)$ lies in evaluating the x_τ, y_τ, z_τ, w_τ, v_τ values, which, as we will see, can be done also in $O(t' - t)$ time. In the following, we list all the nonzero arguments for the 30 types.

For $i = 1$, $x_\tau = -D_\tau$ for $\tau = t, \ldots, t + k - 1$; $x_{t+k} = \bar{D}_{t+k,t'} - \bar{B}_{tt'}$; $z_{t+k} = \bar{B}_{tt'} - \bar{D}_{t+k+1,t'}$; $z_\tau = D_\tau$ for $\tau = t + k + 1, \ldots, t'$; $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; and $u_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$.

For $i = 2$, $x_\tau = D_\tau$ for $\tau = t, \ldots, t + k$; $z_\tau = D_\tau$ for $\tau = t + k + 1, \ldots, t'$; $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $u_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; and $v_{t+k} = \bar{B}_{tt'} - \bar{D}_{t+k+1,t'}$.

For $i = 3$, $x_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_\tau = D_\tau$ for $\tau = t + k, \ldots, t'$; $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $u_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; and $v_{t+k} = \bar{B}_{tt'} - \bar{D}_{t+k,t'}$.

For $i = 4$, $x_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $x_{t+k} = \bar{D}_{t+k,t'} - \bar{B}_{tt'}$; $y_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_{t+k} = \bar{B}_{t,t+k}$; $z_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$; and $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$.

For $i = 5$, $x_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $x_{t+k} = \bar{D}_{t+k,t'} - \bar{B}_{t+k+1,t'}$; $y_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$; $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; and $v_{t+k} = \bar{B}_{t,t+k}$.

For $i = 6$, $x_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $y_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_{t+k} = \bar{D}_{t+k,t'} - \bar{B}_{t+k+1,t'}$; $z_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$; $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; and $v_{t+k} = \bar{B}_{tt'} - \bar{D}_{t+k,t'}$.

For $i = 7$, $x_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $x_{t+k} = \bar{D}_{t+k,t'} - \bar{B}_{t,t+k}$; $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_{t+k} = \bar{B}_{t,t+k}$; $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; and $v_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$.

For $i = 8$, $x_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $x_{t+k} = \bar{D}_{t+k,t'}$; $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $v_{t+k} = \bar{B}_{t,t+k}$; and $v_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$.

For $i = 9$, $x_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_{t+k} = \bar{D}_{t+k,t'}$; $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $v_{t+k} = \bar{B}_{t,t+k} - \bar{D}_{t+k,t'}$; and $v_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$.

For $i = 10$, $x_{t+k} = \bar{D}_{tt'} - \bar{B}_{tt'}$; $z_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_{t+k} = \bar{B}_{tt'} - \bar{D}_{tt'} + D_{t+k}$; $z_\tau = D_\tau$ for $\tau = t + k + 1, \ldots, t'$; $u_\tau =$

$\bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; and $u_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$.

For $i = 11$, $x_{t+k} = D_{t+k}$; $z_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$ and $\tau = t + k + 1, \ldots, t'$; $u_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $u_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; and $v_{t+k} = \bar{B}_{tt'} - \bar{D}_{tt'} + D_{t+k}$.

For $i = 12$, $z_\tau = D_\tau$ for $\tau = t, \ldots, t'$; $u_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $u_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; and $v_{t+k} = \bar{B}_{tt'} - \bar{D}_{tt'}$.

For $i = 13$, $x_{t+k} = \bar{D}_{tt'} - \bar{B}_{tt'}$; $y_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_{t+k} = \bar{B}_{t,t+k} - \bar{D}_{t,t+k-1}$; $z_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$; and $u_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$.

For $i = 14$, $x_{t+k} = \bar{D}_{t+k,t'} - \bar{B}_{t+k+1,t'}$; $y_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$; $u_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; and $v_{t+k} = \bar{B}_{t,t+k} - \bar{D}_{t,t+k-1}$.

For $i = 15$, $y_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_{t+k} = \bar{D}_{t+k,t'} - \bar{B}_{t+k+1,t'}$; $z_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$; $u_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; and $v_{t+k} = \bar{B}_{tt'} - \bar{D}_{tt'}$.

For $i = 16$, $x_{t+k} = \bar{D}_{tt'} - \bar{B}_{t,t+k}$; $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_{t+k} = \bar{B}_{t,t+k} - \bar{D}_{t,t+k-1}$; $u_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; and $v_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$.

For $i = 17$, $x_{t+k} = \bar{D}_{t+k,t'}$; $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $u_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $v_{t+k} = \bar{B}_{t,t+k} - \bar{D}_{t,t+k-1}$; and $v_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$.

For $i = 18$, $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_{t+k} = \bar{D}_{t+k,t'}$; $u_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $v_{t+k} = \bar{B}_{t,t+k} - \bar{D}_{tt'}$; and $v_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$.

For $i = 19$, $x_{t+k} = \bar{D}_{tt'} - \bar{B}_{tt'}$; $y_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $z_\tau = B_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_{t+k} = \bar{B}_{t+k,t'} - \bar{D}_{t+k+1,t'}$; $z_\tau = D_\tau$ for $\tau = t + k + 1, \ldots, t'$; and $u_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$.

For $i = 20$, $x_{t+k} = \bar{D}_{t,t+k} - \bar{B}_{t,t+k-1}$; $y_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $z_\tau = B_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_\tau = D_\tau$ for $\tau = t + k + 1, \ldots, t'$; $u_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; and $v_{t+k} = \bar{B}_{t+k,t'} - \bar{D}_{t+k+1,t'}$.

For $i = 21$, $y_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $z_\tau = B_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_{t+k} = \bar{D}_{t,t+k} - \bar{B}_{t,t+k-1}$; $z_\tau = D_\tau$ for $\tau = t + k + 1, \ldots, t'$; $u_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; and $v_{t+k} = \bar{B}_{tt'} - \bar{D}_{tt'}$.

For $i = 22$, $x_{t+k} = \bar{D}_{tt'} - \bar{B}_{tt'}$; $y_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $y_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; and $z_\tau = B_\tau$ for $\tau = t, \ldots, t'$.

For $i = 23$, $x_{t+k} = \bar{D}_{tt'} - \bar{B}_{tt'} + B_{t+k}$; $y_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $y_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = B_\tau$ for $\tau = t, \ldots, t + k - 1$ and $\tau = t + k + 1, \ldots, t'$; and $v_{t+k} = B_{t+k}$.

For $i = 24$, $y_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $y_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = B_\tau$ for $\tau = t, \ldots, t + k - 1$ and $\tau = t + k + 1, \ldots, t'$; $z_{t+k} = \bar{D}_{tt'} - \bar{B}_{tt'} + B_{t+k}$; and $v_{t+k} = \bar{B}_{tt'} - \bar{D}_{tt'}$.

For $i = 25$, $x_{t+k} = \bar{D}_{tt'} - \bar{B}_{t,t+k}$; $y_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = B_\tau$ for $\tau = t, \ldots, t + k$; and $v_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$.

For $i = 26$, $x_{t+k} = \bar{D}_{tt'} - \bar{B}_{t,t+k-1}$; $y_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = B_\tau$ for $\tau = t, \ldots, t + k - 1$; and $v_\tau = B_\tau$ for $\tau = t + k, \ldots, t'$.

For $i = 27$, $y_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = B_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_{t+k} = \bar{D}_{tt'} - \bar{B}_{t,t+k-1}$; $v_{t+k} = \bar{B}_{t,t+k} - \bar{D}_{tt'}$; and $v_\tau = B_\tau$ for $\tau = t + k + 1, \ldots, t'$.

For $i = 28$, $x_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $x_{t+k} = \bar{D}_{t+k,t'} - \bar{B}_{tt'}$; $y_{t+k} = \bar{D}_{t+k+1,t'} - \bar{B}_{tt'}$; $y_\tau = \bar{D}_{\tau+1,t'} - \bar{B}_{\tau+1,t'}$ for $\tau = t + k + 1, \ldots, t' - 1$; $z_{t+k+1} = \bar{B}_{t,t+k+1}$; $z_\tau = B_\tau$ for $\tau = t + k + 2, \ldots, t'$; and $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k$.

For $i = 29$, $x_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $x_{t+k} = \bar{D}_{t+k,t'}$; $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $u_\tau = \bar{B}_{t\tau}$ for $\tau = t, \ldots, t + k$; $v_{t+k+1} = \bar{B}_{t,t+k+1}$; and $v_\tau = B_\tau$ for $\tau = t + k + 2, \ldots, t'$.

For $i = 30$, $y_\tau = \bar{D}_{\tau+1,t'}$ for $\tau = t + k, \ldots, t' - 1$; $z_\tau = D_\tau$ for $\tau = t, \ldots, t + k - 1$; $z_{t+k} = \bar{D}_{t+k,t'}$; $u_\tau = \bar{B}_{t\tau} - \bar{D}_{t\tau}$ for $\tau = t, \ldots, t + k - 1$; $u_{t+k} = \bar{B}_{t,t+k} - \bar{D}_{tt'}$; $v_{t+k+1} = \bar{B}_{t,t+k+1} - \bar{D}_{tt'}$; and $v_\tau = B_\tau$ for $\tau = t + k + 2, \ldots, t'$.

ACKNOWLEDGMENTS

Jian Yang's research is supported in part by New Jersey Institute of Technology under Grant No. 421830. Boaz Golany's research is partially supported by Fund for the Promotion of Research at the Technion. Gang Yu's research is supported in part by the National Natural Science Foundation of China under Grant No. 79928001, and by the FADRC Grant and the Special Research Grant from the University of Texas at Austin.

REFERENCES

[1] A. Aggarwal and J.K. Park, Improved algorithm for economic lot size problems, Oper Res 41 (1993), 549–571.

[2] G.R. Bitran and H. Matsuo, Approximation formulations for the single-product capacitated lot size problem, Oper Res 34 (1986), 63–74.

[3] G.R. Bitran and H.H. Yanasse, Computational complexity of the capacitated lot size problem, Management Sci 28 (1982), 1174–1186.

[4] A. Federgruen and M. Tzur, A simple forward algorithm to solve general dynamic lot sizing models with n periods in $O(n \log n)$ or $O(n)$ time, Management Sci 37 (1991), 909–925.

[5] M. Fleischmann, J.M. Bloemhof-Ruwaard, R. Dekker, E. van der Laan, J.A.E.E. van Nunen, and L.N. van Wassenhove, Quantitative models for reverse logistics: A review, European J Oper Res 103 (1997), 1–17.

[6] M. Florian and M. Klein, Deterministic production planning with concave costs and capacity constraints, Management Sci 18 (1971), 12–20.

[7] M. Florian, J.K. Lenstra, and A.H.G. Rinnooy Kan, Deterministic production planning: Algorithm and complexity, Management Sci 26 (1980), 669–679.

[8] B. Gavish and R.E. Johnson, A fully polynomial approximation scheme for single product scheduling in a finite capacity facility, Oper Res 38 (1990), 70–83.

[9] B. Golany, J. Yang, and G. Yu, Economic lot-sizing with remanufacturing options, IIE Trans 33 (2001), 995–1003.

[10] V.D.R. Guide, Production planning and control for remanufacturing: Industry practice and research needs, J Oper Management 18 (2000), 467–483.

[11] V.D.R. Guide, V. Jayaraman, R. Srivastava, and W.C. Benton, Supply chain management for recoverable manufacturing systems, Interfaces 30 (2000), 125–142.

[12] G.P. Kiesmüller, Optimal control of one product recovery system with lead-times, Int J Prod Econom 81–82 (2003), 333–340.
[13] K. Richter and M. Sombrutzki, Remanufacturing planning for the reverse Wagner/Whitin models, European J Oper Res 121 (2000), 304–315.
[14] K. Richter and J. Weber, The reverse Wagner/Whitin model with variable manufacturing and remanufacturing cost, Int J Prod Econom 71 (2001), 447–456.
[15] V.P. Simpson, Optimum solution structure for a repairable inventory problem, Oper Res 26 (1978), 270–281.
[16] R.H. Teunter, Economic ordering quantities for recoverable item inventory systems, Naval Res Logist 48 (2001), 484–495.
[17] L.B. Toktay, L.M. Wein, and S.A. Zenios, Inventory management of remanufacturable products, Management Sci 46 (2000), 1412–1426.
[18] E. van der Laan and M. Salomon, Production planning and inventory control with remanufacturing and disposal, European J Oper Res 102 (1997), 264–278.
[19] E. van der Laan, M. Salomon, and R. Dekker, Investigation of lead-time effects in manufacturing/remanufacturing systems under simple PUSH and PULL control strategies, European J Oper Res 115 (1999), 195–214.
[20] E. van der Laan, M. Salomon, R. Dekker, and L. van Wassenhove, Inventory control in hybrid systems with remanufacturing, Management Sci 45 (1999), 733–747.
[21] C.P.M. van Hoesel and A.P.M. Wagelmans, An $O(T^3)$ algorithm for the economic lot sizing problem with constant capacities, Management Sci 42 (1996), 142–150.
[22] C.P.M. van Hoesel and A.P.M. Wagelmans, Fully polynomial approximation schemes for single-item capacitated economic lot-sizing problems, Math Oper Res 26 (2001), 339–357.
[23] A.P.M. Wagelmans, S. van Hoesel, and A. Kolen, An $O(n \log n)$ algorithm that runs in linear time in the Wagner-Whitin case, Oper Res 40 (1992), S145–S156.
[24] H.M. Wagner, A postscript to "Dynamic problems in the theory of the firm," Naval Res Logist Quart 7 (1960), 7–12.
[25] H.M. Wagner and T.M. Whitin, Dynamic version of the economic lot size model, Management Sci 5 (1959), 89–96.
[26] W.I. Zangwill, A backlogging model and a multiechelon model of a dynamic economic lot size production system—a network approach, Management Sci 15 (1969), 506–527.