European Journal of Operational Research 35 (1988) 414–425
North-Holland

Theory and Methodology

A transportation type aggregate production model with bounds on inventory and backordering

S. Selcuk ERENGUC
University of Florida, College of Business Administration, Department of Management, Gainesville, FL 32611, USA

Suleyman TUFEKCI
University of Florida, College of Engineering, Department of Industrial and Systems Engineering, Gainesville, FL 32611, USA and Bilkent University, Ankara, Turkey

Abstract: We consider a certain T period aggregate production planning model, where the two sources of production are regular and overtime. The model allows for time varying production, holding and backordering costs and includes bounds on inventory and backorders. We show that the problem has a rather interesting network structure and exploit this structure to develop a greedy algorithm to solve the problem. The procedure is easy to implement and has a computational complexity of $O(T^2)$. We report computational experience with the greedy procedure and demonstrate its superiority to a well known network simplex code, GNET, implemented on the classical network formulation of the problem.

Keywords: Aggregate scheduling, polynomial algorithm, network model

Introduction

A special class of aggregate scheduling models are those that fit into a transportation model framework. The first transportation type aggregate scheduling model was suggested by Bowman [3] and later discussed by Bishop [2], Manne [11] and Sadleir [13]. This problem involved finding the minimum cost production schedule over a T period finite planning horizon with known demands d_t, $t = 1, 2, \ldots, T$. The demand in each period must be satisfied out of production or existing inventory in that period, i.e., backordering is not allowed. In each of the T periods goods can be produced with regular time at a unit cost of 'a' and with overtime at a unit cost of 'w' where $w > a$. The items can be stored indefinitely at a cost of h per unit per period. There are also capacity restrictions on regular time and overtime production in each of the T periods.

It was subsequently shown by Johnson [8] that a simple noniterative method would suffice to solve Bowman's model. Later, an alternative solution procedure was suggested by Szwarc [14].

In a recent paper Posner and Szwarc [12] generalized Bowman's Aggregate Scheduling model to include backordering at a unit cost of b per period. It was shown in [12] that this problem can also be solved by a noniterative procedure with a computational requirement of $O(T^2)$. For other related literature see [12].

This paper will examine a generalization of Posner and Szwarc's aggregate scheduling model that includes upperbounds on inventory and backordering and allows for time varying production, inventory holding and backordering costs.

This generalized model has a well known standard minimum cost network flow representation (see for example Johnson and Montgomery [7, p. 205]). An optimal solution for this representation may be obtained by any one of the following algorithms: network simplex, out-of-kilter, flow augmentation. For further information on these algorithms see for example Bazaraa and Jarvis [1]. None of these algorithms is polynomially bounded for solving general minimum cost network flow problems. Also, whether or not any of these three algorithms is polynomially bounded for the standard network representation of the problem is not known.

In this paper we show that the problem has a different and very interesting network flow representation. We propose an algorithm which finds an optimal solution to this specially structured network flow problem by solving a sequence of shortest path problems. The algorithm requires $O(T)$ augmentations and is of computational complexity $O(T^2)$. We report some computational results which compare very favorably with the computational times obtained from a well known network simplex code, GNET [4], applied on the classical formulation of the problem.

Problem statement

In developing the optimization model we will use the following notation for $t = 1, 2, \ldots, T$:

d_t : Demand in period t;
a_t : Regular time production cost per unit in period t, $a_t > 0$;
w_t : Overtime production cost per unit in period t, $w_t > a_t$;
h_t : Inventory holding cost per unit in period t assessed on ending inventories, $h_t > 0$;
b_t : Backordering cost per unit in period t, $b_t > 0$;
R_t : Regular time production capacity in period t;
O_t : Overtime production capacity in period t;
β_t : Upper bound on the number of units that can be backordered in period t, $\beta_t \geqslant 0$;
S_t : Upper bound on inventory at the end of period t, $S_t \geqslant 0$;
r_t : Number of units produced using regular time in period t;
o_t : Number of units produced using overtime in period t;
I_t : Ending inventory of period t;
B_t : Number of units backordered in period t;

Without loss of generality we assume that $I_T = I_0 = B_T = B_0 = 0$, that is $S_T = \beta_T = 0$.

We now write the optimization problem:

$$
\begin{aligned}
\text{(P1)} \quad \text{Minimize} \quad & Z = \sum_{t=1}^{T} (a_t r_t + w_t o_t) + \sum_{t=1}^{T} h_t I_t + \sum_{t=1}^{T} b_t B_t, \\
\text{subject to} \quad & \left.\begin{aligned} & r_t + x_t = R_t, \\ & o_t + y_t = O_t, \\ & r_t + o_t + I_{t-1} + B_t - I_t - B_{t-1} = d_t, \end{aligned}\right\} \quad t = 1, 2, \ldots, T, \\
& \left.\begin{aligned} & B_t + \bar{B}_t = \beta_t, \\ & I_t + \bar{I}_t = S_t, \end{aligned}\right\} \quad t = 1, 2, \ldots, T-1, \\
& I_o = I_T = B_o = B_T = 0, \\
& \text{all variables nonnegative,}
\end{aligned}
$$

where x_t, y_t, $\bar{B}_t$ and $\bar{I}_t$ are the slack variables associated with the constraints in which they appear. A standard network representation of this problem (P1) is shown in Figure 1.

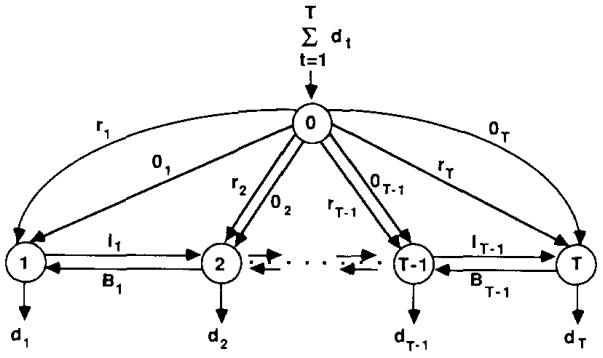

Figure 1. A standard network representation of (P1)

Hereafter we assume that (P1) has a feasible and therefore an optimal solution.

(P1) can be transformed into a balanced uncapacitated transportation problem by performing elementary row operations as summarized by Klein [10]. Defining

$$d_{T+1} \equiv \sum_{t=1}^{T} (R_t + O_t) - \sum_{t=1}^{T} d_t, \tag{1}$$

and performing the appropriate elementary row operations, we obtain the equivalent problem (P2), a balanced uncapacitated transportation problem.

$$\text{(P2)} \quad \text{minimize} \quad Z = \sum_{t=1}^{T} (a_t r_t + w_t o_t) + \sum_{t=1}^{T-1} h_t I_t + \sum_{t=1}^{T-1} b_t B_t,$$

$$\text{subject to} \quad \left.\begin{aligned} & r_t + x_t = R_t, && \text{(2.a)}\\ & o_t + y_t = O_t, && \text{(2.b)}\\ & r_t + o_t + I_{t-1} + \bar{B}_{t-1} + B_t + \bar{I}_t = d_t + S_t + \beta_{t-1}, && \text{(2.c)} \end{aligned}\right\} t = 1, 2, \ldots, T,$$

$$\sum_{t=1}^{T} (x_t + y_t) = d_{T+1}, \tag{2.d}$$

$$\left.\begin{aligned} & B_t + \bar{B}_t = \beta_t, && \text{(2.e)}\\ & I_t + \bar{I}_t = S_t, && \text{(2.f)} \end{aligned}\right\} \quad t = 1, 2, \ldots, T-1,$$

$$I_0 = I_T = B_0 = B_T = \bar{I}_0 = \bar{B}_0 = 0, \tag{2.g}$$

$$\text{all variables nonnegative.} \tag{2.h}$$

Also note that by definition $\beta_0 = S_T = 0$.

We will now put (P2) in the transportation tableau format. In the transportation tableau the first four rows will correspond to regular time production, overtime production, ending inventory and backordered quantity in period 1, respectively and the second, third, fourth and up to $(T-1)$st four rows will correspond to regular time production, overtime production, ending inventory and backordered quantity in their respective periods. The last two rows will correspond to the regular time production and overtime production in period T, respectively. Also the first T columns will represent the demand periods and column $T+1$ will be the slack column. The capacities for regular time production, overtime production, inventory and backordering are R_t, O_t, S_t and β_t, respectively. We note that demand in period 1 is $d_1 + S_1$ and in period T it is $d_T + \beta_{T-1}$. For period t, $t = 2, 3, \ldots, T-1$, demand is $d_t + S_t + \beta_{T-1}$. To clarify this representation we put a three period problem in the transportation format in Table 1.

Note in Table 1 that, the entry in the upper left hand corner of each cell is the variable represented by that cell and the entry in the upper right hand corner is the associated cost. The cells with an asterisk '*'

Table 1

	Demand periods								Capacities
	1		2		3		Slack		
Regular time 1	r_1	a_1	*		*		x_1	0	R_1
Overtime 1	o_1	w_1	*		*		y_1	0	O_1
Inventory 1	$\bar{I}_1$	0	I_1	h_1	*		*		S_1
Backordering 1	B_1	b_1	$\bar{B}_1$	0	*		*		β_1
Regular time 2	*		r_2	a_2	*		x_2	0	R_2
Overtime 2	*		o_2	w_2	*		y_2	0	O_2
Inventory 2	*		$\bar{I}_2$	0	I_2	h_2	*		S_2
Backordering 2	*		B_2	b_2	$\bar{B}_2$	0	*		β_2
Regular time 3	*		*		r_3	a_3	x_3	0	R_3
Overtime 3	*		*		o_3	w_3	y_3	0	O_3
	$d_1 + S_1$		$d_2 + S_2 + \beta_1$		$d_3 + \beta_2$		$d_4 = \sum_{t=1}^{3}(R_t + O_t) - \sum_{t=1}^{3} d_t$		

Note: all cells with a '*' have infinite cost.

indicate the excluded arcs in the corresponding transportation network. No shipments can be made on these arcs. This exclusion can be enforced by assigning sufficiently large positive costs to these arcs.

It is a well known fact that the addition of a constant to any row or column of the cost matrix of a balanced transportation problem leaves the optimal solution unchanged. We now perform the following operations on the cost matrix of (P2) (Table 1). By convention let the slack column be column $T+1$, and also let any sum, $\sum_{i=k}^{l} z_i \equiv 0$ if $l < k$.

(1) To each column j, $j = 2, 3, \ldots, T$, add $(-\sum_{k=1}^{j-1} h_k)$.
(2) Add $-a_j + \sum_{k=1}^{j-1} h_k$ to the j-th regular time row, $j = 1, 2, \ldots, T$.
(3) Add $-w_j + \sum_{k=1}^{j-1} h_k$ to the j-th overtime row, $j = 1, 2, \ldots, T$.
(4) Add $\sum_{k=1}^{j-1} h_k$ to each inventory row j, $j = 2, 3, \ldots, T-1$.
(5) Add $\sum_{k=1}^{j} h_k$ to each backordering row j, $j = 1, 2, \ldots, T-1$.

It is instructive to see how similar operations were performed in [12]. After performing these operations we obtain the equivalent transportation problem (P3).

$$\text{(P3)} \quad \text{Minimize} \quad Z = \sum_{t=1}^{T} c_t x_t + \sum_{t=1}^{T} \bar{c}_t y_t + \sum_{t=1}^{T-1} (b_t + h_t) B_t,$$

$$\text{subject to} \quad \text{(2.a), (2.b), (2.c), (2.d), (2.e), (2.f), (2.g), (2.h),}$$

where

$$c_t = \begin{cases} -a_t & \text{if } t = 1, \\ -a_t + \sum_{k=1}^{t-1} h_k & \text{if } 1 < t \leqslant T, \end{cases} \quad \text{and} \quad \bar{c}_t = \begin{cases} -w_t & \text{if } t = 1, \\ -w_t + \sum_{k=1}^{t-1} h_k & \text{if } 1 < t \leqslant T. \end{cases}$$

Table 2

0	*	*	$-a_1$	R_1
0	*	*	$-w_1$	O_1
0	0	*	*	S_1
$b_1 + h_1$	0	*	*	β_1
*	0	*	$-a_2 + h_1$	R_2
*	0	*	$-w_2 + h_1$	O_2
*	0	0	*	S_2
*	$b_2 + h_2$	0	*	β_2
*	*	0	$-a_3 + h_1 + h_2$	R_3
*	*	0	$-w_3 + h_1 + h_2$	O_3
$d_1 + S_1$	$d_2 + S_2 + \beta_1$	$d_3 + \beta_2$	d_4	

The transportation tableau (Table 2) summarizes these operations for the three period example given in Table 1.

Note that in the objective function of (P3) only x_t, y_t and B_t have nonzero coefficients. In accordance with this observation we will rewrite (P3) only in terms of variables x_t, y_t and B_t. Upon performing a series of algebraic manipulations (involving only additions and subtractions) on (P3) we obtain (P4). For space considerations the details of this transformation are not discussed here. However, the interested reader can find these details along with a constructive proof of the equivalence of (P4) and (P3) and therefore (P4) and (P1) in [6].

$$\text{(P4)} \quad \text{Minimize} \qquad Z = \sum_{t=1}^{T} c_t x_t + \sum_{t=1}^{T} \bar{c}_t y_t + \sum_{t=1}^{T-1} e_t B_t' + \sum_{t=1}^{T-1} e_t \theta_t,$$

$$\text{subject to} \qquad \left.\begin{aligned} &\sum_{i=1}^{t} (x_i + y_i) - \varepsilon_t \leqslant B_t' \leqslant \Psi_t, \\ &v_t - S_t \leqslant \sum_{i=1}^{t} (x_i + y_i) \leqslant \Psi_t + \varepsilon_t, \end{aligned}\right\} \quad t = 1, 2, \ldots, T-1, \tag{3.a, 3.b}$$

$$\sum_{t=1}^{T} (x_t + y_t) = d_{T+1}, \tag{3.c}$$

$$\left.\begin{aligned} &0 \leqslant x_t \leqslant R_t, \\ &0 \leqslant y_t \leqslant 0_t, \end{aligned}\right\} \quad t = 1, 2, \ldots, T, \tag{3.d}$$

$$B_t' \geqslant 0, \quad t = 1, 2, \ldots, T-1, \tag{3.e}$$

where for each $t \in \{1, 2, \ldots, T-1\}$, $e_t = b_t + h_t$; $v_t = \sum_{i=1}^{t}(R_i + O_i) - \sum_{i=1}^{t} d_i$; $\theta_t = \max[0, -v_t]$; $\Psi_t = \beta_t - \theta_t$; $\varepsilon_t = v_t + \theta_t$ and $B_t' = B_t - \theta_t$. Also note that the term $(\sum_{t=1}^{T-1} e_t \theta_t)$ in the objective function of (P4) is a constant.

At this point we would like to emphasize the fact that, (P4) can be directly obtained from (P1) by some simple transformations on the original data. Therefore the users of our procedure do not have to go through the problem transformations given in this paper and in [6] to obtaine (P4).

Upon solving (P4), one obtains the optimal values of x_t, y_t and B_t. Once these optimal values are determined, once can go back to the transportation tableau (Table 2) and compute the optimal values of r_t, o_t, $\bar{B}_t$, I_t and $\bar{I}_t$. This is done easily since in each row of the transportation tableau only two assignments can be made. For the regular time, overtime and backordering rows, the solution to (P4) yields one of these assignments. After the optimal assignments are placed in these rows, obtaining the optimal values of I_t and $\bar{I}_t$ is trivial.

Problem (P4) has a very interesting and special network flow representation which is amenable to solution by a greedy flow augmenting algorithm. To be specific this problem is called a minimum cost maximum flow problem.

In what follows we will give the network flow representation implied by (P4), comment on some of the properties of this representation and give an $O(T^2)$ algorithm to solve the production scheduling problem.

A network flow representation of (P4)

A network flow representation of (P4) is given in Figure 2. We will call this network G.

The lower and upper bounds indicated by constraints (3.b) are denoted LB_t and UB_t, $t = 1, 2, \ldots, T-1$, respectively. Therefore

$$LB_t = \max[0, v_t - S_t] \quad \text{and} \quad UB_t = \Psi_t + \varepsilon_t.$$

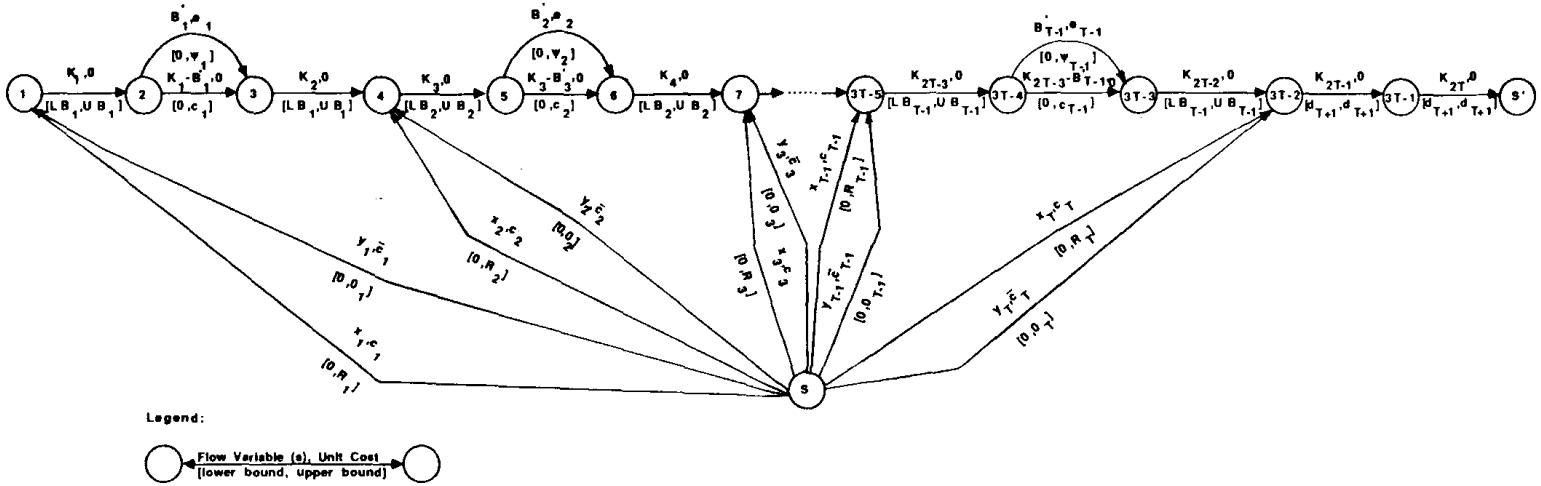

Figure 2. A network representation of (P4)

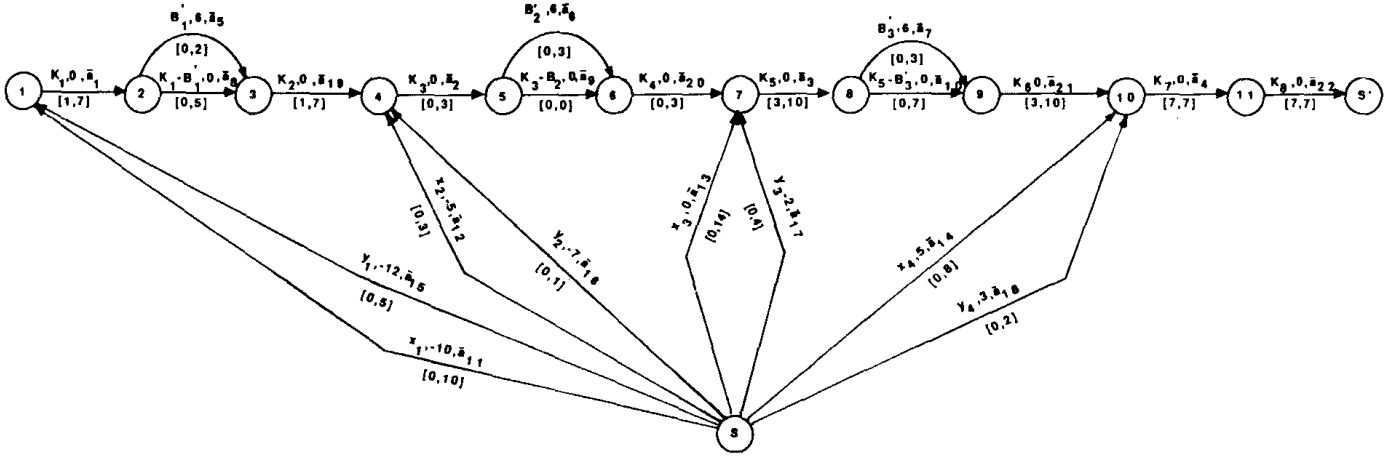

Figure 3. A network representation of the numerical example

Note that since $x_t,\ y_t \geqslant 0,\ t = 1, 2, \ldots, T$, we define $\mathrm{LB}_t = \max[0,\ v_t - S_t]$. Also the constraint (3.c) is replaced by

$$\mathrm{LB}_T = d_{T+1} \leqslant \sum_{t=1}^{T} (x_t + y_t) \leqslant d_{T+1} = \mathrm{UB}_T.$$

The two arcs between the source node S and node $3t-2,\ t = 1, 2, \ldots, T$, correspond to regular time and overtime slack variables. These arcs will be called regular slack and overtime slack arcs in their respective periods. Between each pair of nodes $3t-1$ and $3t,\ t = 1, 2, \ldots, T-1$, there are two arcs; the top one will be labelled the backordering arc and the bottom one will be called the free arc, for their respective periods. For period $t,\ t = 1, 2, \ldots, T-1$, constraints (3.a), (3.b) and (3.e) are represented by arcs $(3t-2,\ 3t-1)$, $(3t,\ 3t+1)$ and the backordering and free arcs in that period. Note that if for any period $t,\ v_t \leqslant 0$, then $\varepsilon_t = 0$ implying that the free arc for that period is closed. Constraints (3.d) are represented by the regular and overtime slack arcs and the arc $(3T-1,\ S')$ represents constraint (3.c). We also note that arcs $(3t,\ 3t+1)$ are not actually needed, however we introduced them for visual convenience.

As clearly illustrated Figure 2, the objective is to send exactly d_{T+1} units $(x_t,\ y_t)$ from the source node S, to the sink node S' at a minimum cost while observing the lower and upper bounds on the arcs. It is also instructive to note that the only arcs that may have nonzero cost are the regular and overtime slack arcs and the backordering arcs.

An $O(T^2)$ algorithm for solving (P4)

In this section we present an algorithm for solving (P4) by exploiting the network structure given in Figure 2. The algorithm makes assignments to x_t and $y_t,\ t = 1, 2, \ldots, T$, in a greedy fashion by augmenting flow over a sequence of shortest paths, in the corresponding network until flow equals d_{T+1}. The computational complexity of the procedure is $O(T^2)$.

Let the arcs $\bar{a}_t$ be numbered as follows

$\bar{a}_t = (3t-2,\ 3t-1),\ t = 1, 2, \ldots, T;$
$\bar{a}_{T+i} =$ backordering arc in period $i,\ i = 1, 2, \ldots, T-1;$
$\bar{a}_{2T-1+i} =$ free arc in period $i,\ i = 1, 2, \ldots, T-1;$
$\bar{a}_{3T-2+i} =$ regular time slack arc in period $i,\ i = 1, 2, \ldots, T;$
$\bar{a}_{4T-2+i} =$ overtime slack in period $i,\ i = 1, 2, \ldots, T;$
$\bar{a}_{5T-2+t} = (3t,\ 3t+1),\ t = 1, 2, \ldots, T-1;$
$\bar{a}_{6T-2} = (3T-1,\ S').$

Also let c_k and $(l_k,\ u_k)$ represent the cost, and lower and upper bounds on arc $\bar{a}_k,\ k = 1, 2, \ldots, 6T-2$, respectively. We note that $c_k = +\infty$ if $l_k = u_k = 0$ for any arc $\bar{a}_k$. Note here that $l_T = u_T = d_{T+1} = \mathrm{UB}_T = \mathrm{LB}_T$.

The algorithm

for $t = 1$ to T **do**
begin
  while $l_t > 0$ **do**
  begin
    Find the shortest path P_1 from node S to node $3t-2$.
    Find the shortest path P_2 from node $3t-1$ to S'
    $\Delta_1 = \min_{\bar{a}_k \in P_1} \{u_k\}$
    $\Delta_2 = \min_{\bar{a}_k \in P_2} \{u_k\}$

$\Delta = \min\{\Delta_1, \Delta_2, l_t\}$
Augment Δ units along the path $P = P_1 \cup \bar{a}_t \cup P_2$.
Update l_k, u_k and c_k along P as

$$l_k \leftarrow \begin{cases} 0 & \text{if } l_k \leq \Delta \\ l_k - \Delta & \text{if } l_k > \Delta, \text{ and} \end{cases}$$

$u_k \leftarrow u_k - \Delta$. For all $l_k = u_k = 0$, $\bar{a}_k \in P$ set $c_k = +\infty$.
end
endwhile
end
endfor

The rest of this section is devoted to the proof of validity and the computational complexity of the algorithm. We need the following results to establish the computational complexity of the algorithm.

Lemma 1. *Finding the shorest path between any pair of nodes* (i, j) *in* G *requires* $O(T)$ *additions and comparisons.*

Proof. From Figure 2, it is clear that if $i \neq S$ we need at most T additions and comparisons to find the shortest path between nodes i and j. If $i = S$, then we need at most $3T$ additions and comparisons to find the shortest path between node S and a node j, $j \neq S$. □

Remark 1. In the execution of the algorithm no flow is reduced on an arc and at each augmentation either a lower bound is satisfied or an arc is saturated.

As an immediate consequence of Remark 1 we have:

Lemma 2. *The algorithm terminates after at most* $6T - 2$ *augmentations.*

Proof. There are at most $5T - 2$ positive upper bounds and at most T positive lower bounds in this problem. From Remark 1, it may take as many as $6T - 2$ augmentations for the algorithm to terminate.

Theorem 1. *The* ***algorithm*** *is of complexity* $O(T^2)$.

Proof. The proof immediately follows from Lemmas 1 and 2, the fact that augmenting along the shortest path requires $O(T)$ additions and comparisons.

Theorem 2. *The complexity of the procedure to solve the* ***aggregate production*** *planning problem* (P1) *is* $O(T^2)$.

Proof. Obtaining (P4) from the original data is of complexity $O(T)$ (see the details of obtaining (P4) from (P3) in [6]). Solving (P4) yields the optimal values of x_t, y_t, $t = 1, 2, \ldots, T$ and B_t, $t = 1, 2, \ldots, T - 1$. Once these values are obtained, one has to go back to the transportation tableau (Table 2) and find the optimal values of r_t, o_t $\bar{B}_t$, I_t and $\bar{I}_t$ as discussed in the previous section. Evidently, completing the transportation tableau is also of complexity $O(T)$. Combining these observations with Theorem 1 yields the desired result.

Let f_i, for each arc $\bar{a}_i$ in G be the flow obtained by the algorithm. In proving the validity of the algorithm we will make use of the following G' derived from G as follows. (This is the procedure used in Klein's [9] method for finding minimum cost maximum flow by negative cycles.)

G' has the same node set G. Let A and A' represent arc sets of G and G' respectively. We construct the arcs of G' as follows.

If $\bar{a}_j \in A$ and $f_j < u_j$ then place $\bar{a}_j$ in A' with capacity $u'_j = u_j - f_j$, and cost $c'_j = c_j$. Also if $\bar{a}_j \in A$ and $f_j > l_j$ then place $\bar{\bar{a}}_j$ in A' with capacity $u'_j = f_j - l_j$ and cost $c'_j = -c_j$,

where $\bar{\bar{a}}_j$ is an arc exactly in opposite direction to $\bar{a}_j$. i.e., if $\bar{a}_j = (p, q)$ then $\bar{\bar{a}}_j = (q, p)$. Let $C = \{a_1, a_2, \ldots, a_k\}$ be a directed cycle (circuit) in G' where the head node of $a_1 = (p, q)$ is the tail node of arc $a_2 = (q, r)$. Moreover the head node of arc $a_k = (z, p)$ is the tail node of a_1. Let

$$c(C) = \sum_{a_i \in C} c'_i.$$

A circuit C with $c(C) < 0$ is called a negative circuit. Klein [9] suggested a method for finding a minimum cost maximum flow by using negative circuits. First the maximum flow is found. Given the maximum flow this algorithm then proceeds on finding negative circuits in G' and circulating as much flow as possible along this circuit. The G' network, u'_i and c'_i are updated and the process is repeated until all negative circuits are eliminated from G'. When G' does not contain any negative circuits then the current maximum flow is in fact the minimum cost maximum flow. The proof of optimality of a given maximum flow to be the minimum cost maximum flow when G' does not contain any negative circuits is given in [9].

Theorem 3. *The algorithm produces an optimal solution to* (P4).

Proof. We will simply show that at the termination of the algorithm no negative cycles are present in the network. Note that because of the structure of the network the only cycles which do not include node S are the cycles formed by the backordering arcs and free arcs at each period $1, 2, \ldots, T-1$. Any other cycle in G must include node S. From the construction of the algorithm, if there is any allocation to slack arcs in a given period it is always first to the cheapest slack arc. Therefore a negative cycle containing node S will imply an augmentation not done on a shortest path which contradicts the construction of the algorithm. In a very similar manner it is easy to show that a negative cycle not including node S implies that the backordering arc is used instead of the available free arc at some period t, which also is contradictory to the construction of the algorithm. □

A numerical example

We now give a four period numerical example to illustrate the algorithm:

$$a_j = a = 10, \quad w_j = w = 12, \quad d_j = d = 10, \quad j = 1, 2, 3, 4;$$
$$h_j = h = 5, \quad b_j = b = 1, \quad e_j = 6, \quad j = 1, 2, 3;$$
$$S_1 = 4, \quad S_2 = 3, \quad S_3 = 4, \quad \beta_1 = 2, \quad \beta_2 = 4, \quad \beta_3 = 3;$$
$$R_1 = 10, \quad R_2 = 3, \quad R_3 = 14, \quad R_4 = 8, \quad O_1 = 5, \quad O_2 = 1, \quad O_3 = 4, \quad O_4 = 2;$$
$$v_1 = 5, \quad v_2 = -1, \quad v_3 = 7;$$
$$\theta_1 = \theta_3 = 0, \quad \theta_2 = 1, \quad \Psi_1 = 2, \quad \Psi_2 = 3, \quad \Psi_3 = 3, \quad \varepsilon_1 = 5, \quad \varepsilon_2 = 0, \quad \varepsilon_3 = 7;$$
$$B_1 = B'_1, \quad B_3 = B'_3, \quad B_2 = B'_2 + 1.$$

Table 3
Summary of computations

Augmentation stage	Assignment (variable, units)	Arcs on the shortest path [a] from S to S', $P = P_1 \cup a_k^* \cup P_2$	Length of the shortest path
1	$(y_1, 1)$	$(\bar{a}_{15}, \bar{a}_1^*, \bar{a}_8, \bar{a}_{19}, \bar{a}_2, \bar{a}_6, \bar{a}_{20}, \bar{a}_3, \bar{a}_{10}, \bar{a}_{21}, \bar{a}_4, \bar{a}_{22})$	-6
2	$(y_1, 2)$	same as above, except $\bar{a}_k^* = \bar{a}_3$	-6
3	$(y_3, 4)$	$(\bar{a}_{17}, \bar{a}_3, \bar{a}_{10}, \bar{a}_{21}, \bar{a}_4^*, \bar{a}_{22})$	-2

[a] a_k^* is the arc with the smallest index such that $l_k > 0$ in the i-th augmentation stage

We now formulate (P4).

$$
\begin{aligned}
\text{Minimize} \quad & Z = 6 + (-10x_1 - 5x_2 + 0x_3 + 5x_4) + (-12y_1 - 7y_2 - 2y_3 + 3y_4) \\
& \quad + 6B_1' + 6B_2' + 6B_3', \\
\text{subject to} \quad & x_1 + y_1 - 5 \leqslant B_1' \leqslant 2, \\
& x_1 + x_2 + y_1 + y_2 - 0 \leqslant B_2' \leqslant 3, \\
& x_1 + x_2 + x_3 + y_1 + y_2 + y_3 - 7 \leqslant B_3' \leqslant 3, \\
& 1 \leqslant x_1 + y_1 \leqslant 7, \\
& 0 \leqslant x_1 + x_2 + y_1 + y_2 \leqslant 3, \\
& 3 \leqslant x_1 + x_2 + x_3 + x_4 + y_1 + y_2 + y_3 \leqslant 10, \\
& 7 \leqslant x_1 + x_2 + x_3 + y_1 + y_2 + y_3 + y_4 \leqslant 7, \\
& 0 \leqslant x_1 \leqslant 10, \quad 0 \leqslant x_2 \leqslant 3, \quad 0 \leqslant x_3 \leqslant 14, \quad 0 \leqslant x_4 \leqslant 8, \\
& 0 \leqslant y_1 \leqslant 5, \quad 0 \leqslant y_2 \leqslant 1, \quad 0 \leqslant y_3 \leqslant 4, \quad 0 \leqslant y_4 \leqslant 2.
\end{aligned}
$$

Figure 3 is the network representation of the numerical example, where $\bar{a}_i$, $i = 1, \ldots, 6T - 2$, are the arcs as defined in the previous section. Table 3 summarizes the flow augmentation for this example. Note that the optimal solution to the network flow problem has $B_1' = B_3' = 0$ and $B_2' = 3$. Therefore from $B_t = B_t' + \theta_t$, $t = 1, 2, 3$, we get $B_1 = 0$, $B_2 = 4$, $B_3 = 0$.

The optimal solution to the production scheduling problem is given in the following transportation tableau (Table 4).

Computational experience

In this section we present some computational experience with the algorithm presented in this paper (referred to as SCHED hereafter). We tested SCHED against one of the fastest network simplex codes in the literature, GNET [4] [1]. GNET was used to solve the standard minimum cost network flow representation of problem (P1). These two algorithms were tested on 6 sets of randomly generated problems with number of planning periods $T \in \{8, 12, 16, 20, 24, 30\}$. Each set contained 12 test problems and all of the 72 problems had feasible solutions [2]. All the data were integer as required by GNET.

SCHED was coded in FORTRAN and computational experiments were conducted on an IBM 3090 computer. Both GNET and SCHED were complied with EXTENDED H FORTRAN compiler. Computational results are presented in Table 5. CPU times include the transformation operations from (P1) to (P4) for SCHED and exclude input and output operations both for GNET and SCHED.

SCHED uniformly outperformed GNET in all of the 72 problems solved. On the average, for the 72 problems solved, SCHED was about 8 times faster than GNET.

[1] We would like to express our sincere thanks to G. Bradley and his coauthors for giving us permission to use GNET.

[2] A listing of SCHED and the test problems are available from the authors.

Table 4
Optimal solution to the numerical example

Sources	Periods					Capacities
	1	2	3	4	Slack	
RT_1	10	*	*	*	0	10
OT_1	2	*	*	*	3	5
INV_1	2	2	*	*	*	4
BO_1	0	2	*	*	*	2
RT_2	*	3	*	*	0	3
OT_2	*	1	*	*	0	1
INV_2	*	3	0	*	*	3
BO_2	*	4	0	*	*	4
RT_3	*	*	14	*	0	14
OT_3	*	*	0	*	4	4
INV_3	*	*	4	0	*	4
BO_3	*	*	0	3	*	3
RT_4	*	*	*	8	0	8
OT_4	*	*	*	2	0	2
Demands	14	15	18	13	7	

Before closing this section we would like to emphasize on the importance of fast algorithms for solving the problem (P1). Consider a problem (P*) identical to (P1) except for the objective function which is concave. It is demonstrated in [5] that branch and bound procedure is an efficient way for solving (P*). In this branch and bound procedure one needs to solve a very large number of subproblems which are of the form (P1). Therefore, using algorithm SCHED as opposed to GNET will produce an optimal solution in a significantly smaller CPU time.

Concluding remarks

In this paper we presented a one pass greedy procedure for solving a popular aggregate production model. This model is a generalization of the model presented in [9] in that it allows for time varying production, inventory and backordering costs and includes bounds on inventory and backorders. The procedure is of computational complexity $O(T^2)$ where T is the number of periods in the planning horizon. We showed that the problem can be transformed into a minimum cost-maximum flow problem with a very special structure. We exploited this structure to develop the greedy procedure. The procedure was computationally tested against a well known network simplex code, GNET, on a set of randomly generated problems. The computational results demonstrated the superiority of the greedy procedure to GNET.

Table 5

		SCHED		GNET		
(1) Problem set	(2) Number of periods	(3) Average number of augmentations	(4) Average CPU time (miliseconds)	(5) Average number of simplex pivots	(6) Average CPU time (miliseconds)	(7) Ratio of (6)/(4)
1	8	5.00	0.27	28.08	1.47	5.44
2	12	2.92	0.28	45.50	3.07	10.96
3	16	3.67	0.40	62.25	3.98	9.95
4	20	5.08	0.62	77.42	5.62	9.06
5	24	10.67	1.45	90.25	6.83	4.71
6	30	5.41	0.98	102.90	8.25	8.42

In closing, we note that although the model included two sources of production (regular and overtime), it can be easily extended to more than two sources.

References

[1] Bazaraa, M.S., and Jarvis, J.J., *Linear Programming and Network Flows*, Wiley, New York, 1977.

[2] Bishop, G.T., "On a problem of production scheduling", *Operations Research* 5/1 (1957) 97–103.

[3] Bowman, E.H., "Production scheduling by the transportation method of linear programming", *Operation Research* 4/1 (1956).

[4] Bradley, G.H., Brown, G.G., and Graves, G.M., "Design and implementation of large-scale transshipment algorithms", *Management Science*, 24/1 (1977) 1–34.

[5] Erenguc, S.S., and Tufekci, S., "A branch and bound algorithm for a single-item multi-source dynamic lot sizing problem with capacity constraints", *IIE Transactions* 19/1 (1987) 73–80.

[6] Erenguc, S.S., and Tufekci, S., "A transportation type aggregate production model with bounds on inventory and backordering", *Research Report 84-29*, Industrial and Systems Engineering Department, University of Florida, Gainesville, FL, 1986.

[7] Johnson, L.A., and Montgomery, D.C., *Operations Research in Production Planning, Scheduling and Inventory Control*, Wiley, New York, 1974.

[8] Johnson, S.M., "Sequential production planning over time at minimum cost", *Management Science* 3 (1957) 435–437.

[9] Klein, M., "A primal method for minimal cost flows with applications to the assignment and transportation problems", *Management Science* 14/2 (1967) 205–220.

[10] Klein, M., "A transportation model for production planning with convex costs", *AIIE Transactions* 15/3 (1983) 272–274.

[11] Manne, A., "A note on the Modigliani–Hohn production smoothing model", *Management Science* 3 (1957) 371–379.

[12] Posner, M.E., and Szwarc, W., "A transportation type aggregate production model with backordering", *Management Science* 29/2 (1983) 188–199.

[13] Sadleir, C.D., "Use of the transportation method of linear programming in production planning: A case study", *Operational Research Quarterly* 21/4 (1970) 393–402.

[14] Szwarc, W., "Instant transportation solution", *Naval Research Logistic Quarterly* 22/3 (1975) 427–440.