Mathematical Programming Solution of a Hoist Scheduling Program

L. W. PHILLIPS
Western Electric Company
Richmond, Virginia

P. S. UNGER
Bell Laboratories
Holmdel, New Jersey 07733

Abstract: This paper describes a mixed integer programming model of a process during which electronic circuit boards are chemically treated in a sequence of tanks. The boards must remain in these tanks for periods of time lying within specified bounds. A hoist mechanism is to be programmed to place boards into tanks, remove boards from tanks, and transport boards between tanks, so as to maximize the throughput of the system. Computational experience and a detailed example are given.

1. Introduction

■ During their manufacture, certain electronic circuit boards must undergo a given sequence of chemical treatment and electroplating processes. The Richmond, Virginia, Western Electric Plant performs these processes with a tank and hoist setup (see Fig. 1). Circuit boards, held in "carriers," must proceed in order from tank 0 (for loading) to tank 1, to tank 2, . . . , to tank n, and back to tank 0 for unloading. The carriers are moved by a tape-controlled programmable hoist. This hoist can remove carriers from tanks, lower carriers into tanks, transport carriers between tanks, move (empty) from tank to tank, and pause. Since a tape of finite length controls the hoist's actions and then rewinds automatically and in a negligible time, the hoist must be programmed for one "cycle" of actions to be repeated through the day. The amount of time between successive loadings of carriers into the system (departures from tank 0) will be taken to be a cycle.

Carriers must remain within the various tanks for periods of time lying between prespecified minimum and maximum numbers of seconds. Travel times for the hoist between all pairs of tanks are given. The number of carriers which can be serviced by the hoist simultaneously depends on the relative magnitudes of the minimum and maximum tank times and the hoist travel times. In typical setups in Richmond, three appears to be the maximum number of carriers which can be present in the system at the same time. This maximum number is unknown before the problem is solved.

The overall objective is to maximize the throughput of carriers per hour. This is equivalent to minimizing the cycle length, the time between successive departures of carriers from tank 0 (the load-unload step). A simulation approach to a similar problem has been described in [1]. Section 2 of this paper consists of a mixed integer programming model of this problem, along with formal procedures for interpretation of a solution. In Section 3, a numerical example is presented, along with an analysis of the solution.

Received July 1975; revised November 1975.

2. Mathematical Programming Formulation

Notation

The following are constants which are known for any particular hoist setup:

n—the number of chemical tanks. $n = N - 1$. The tanks are labeled $0, 1, 2, \ldots, n$.

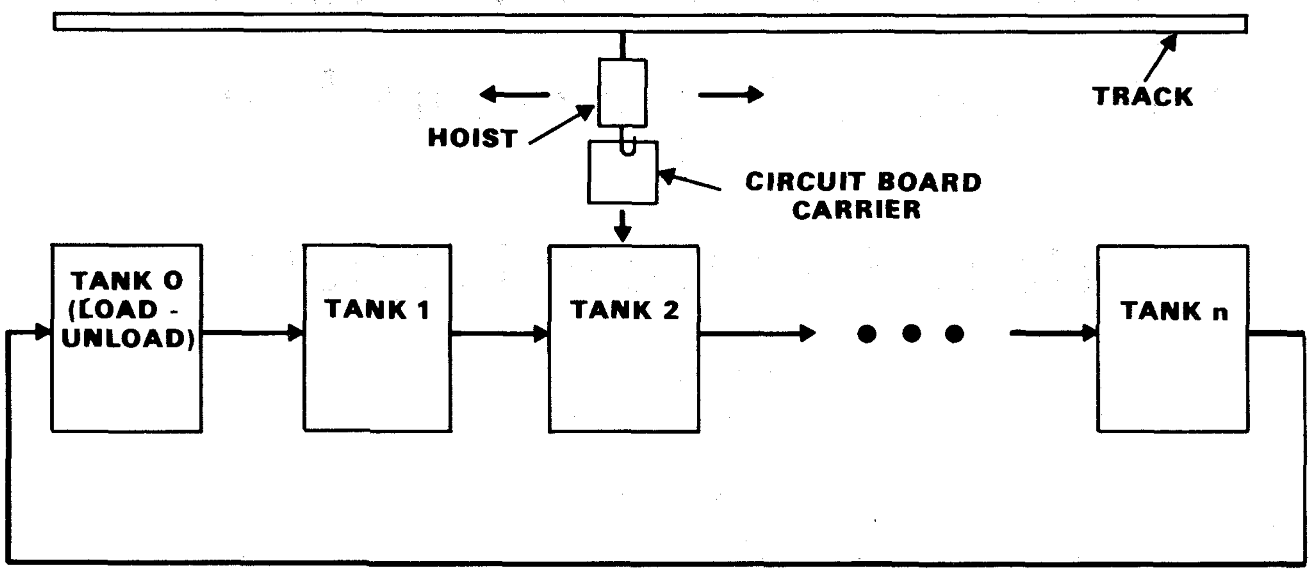

Fig. 1. Hoist setup.

a_i—the minimum amount of time a carrier must spend in tank i.

b_i—the maximum time a carrier may spend in tank i (some b_i may be $+\infty$).

c_{ij}—the travel time for the empty hoist from tank i to tank j. $c_{ii} = 0$. $c_{ji} = c_{ij}$.

$c'_{i,i+1}$—the travel time for the hoist escorting a carrier from tank i to tank i+1. $c'_{i,i+1} = c_{i,i+1} + 20$ seconds.

The reason $c'_{i,i+1} = c_{i,i+1} + 20$ is as follows. About 8 ½ seconds are required to raise (or lower) a carrier to (or from) the height required for transportation of the carrier. About 3 seconds are needed before placement into a tank for any oscillation of the carrier to die down. Whenever the hoist escorts a carrier from tank i to tank i+1, it must raise the carrier (8 ½ seconds), travel ($c_{i,i+1}$), wait for the oscillation to die down (3 seconds), and lower the carrier (8 ½ seconds). When moving empty between tanks, the hoist need not raise itself.

Consider the cycle length to be the time between time 0 (when a carrier departs tank 0) and the time when the next carrier departs tank 0. Cycles are assumed to be identical, implying that the configuration of carriers in tanks at the end of the cycle must be the same as at time 0. It follows that during each cycle, each tank has one carrier dropped in, and one removed, not necessarily in that order. That is, at the beginning of the cycle, a tank may have a carrier in it. During the cycle, that carrier is removed, and some time later, another carrier is dropped into the tank. On the other hand, the tank may be empty at the beginning of the cycle. In that case, a carrier will eventually be entered into the tank, and removed later in the cycle. At all times, there will be 3 carriers in the system. Consequently, 3 cycles must occur before any one carrier makes a complete trip through the tanks.

Following are the decision variables whose values are obtained in the solution process:

t_i— the time at which a carrier is *removed* from tank i. $t_0 = 0$, and is removed from the formulation. The n remaining t_i are continuous variables, although there will be an optimal solution with all t_i integer whenever all the a_i, b_i, and c_{ij} are integer.

$t_{\max}$—a continuous variable which will be forced by a set of constraints to be equal to the maximum of the n t_i's.

y_{ij}—$n(n-1)/2$ zero-one variables which will be forced by constraints to be equal to 1 if $t_i > t_j$ (a carrier leaves tank j before a carrier leaves tank i), and 0 if $t_i < t_j$. They are defined only for $i > j$.

z_i—n zero-one variables. Exactly one of them will be forced by constraints to be 1. That z_i will be forced to correspond to the tank which has a carrier removed from it last, i.e., $z_i = 1$ iff $t_i = t_{\max}$.

It will be shown how a complete specification of the hoist routing may be obtained from the t_i's and the other variables and constants defined above.

In addition to the above constants and variables, a "large number," M, is required. Let the subscript n+1 mean subscript 0; i.e., $c_{i,i+1} = c_{n0}$ when $i = n$.

Objective Function

The objective function which is to be minimized is the cycle length. Cycle length is the sum of several terms:

(i) t_{max} (the last time the hoist departs from a tank with a carrier), (ii) the travel time for that carrier to be dropped off in the next tank, (iii) the travel time for the (empty) hoist to return to the position it was in at time 0. Since by assumption the hoist starts each cycle at tank 0, the objective is to

$$\text{minimize}\left\{ \underbrace{t_{max}}_{\substack{\text{last time}\\ \text{the hoist}\\ \text{departs}\\ \text{from a}\\ \text{tank with}\\ \text{a carrier}}} + \sum_{i=1}^{n} \underbrace{(\underbrace{c'_{i,i+1}}_{\substack{\text{travel time}\\ \text{to drop that}\\ \text{carrier off}\\ \text{into the}\\ \text{next tank}}} + \underbrace{c_{i+1,0}}_{\substack{\text{travel time}\\ \text{for empty}\\ \text{hoist to}\\ \text{return to}\\ \text{tank 0}}})\, z_i}_{\substack{\text{only one term will appear because}\\ \text{exactly one } z_i \text{ will be 1, and all}\\ \text{others will be 0 (see constraint (3)}\\ \text{below)}}} \right\}.$$

Constraints

Type 1 Constraints

These $2n + 1$ constraints force t_{max} to be equal to the maximum t_i and force z_i to be 1 for that i such that $t_i = t_{max}$, and z_i to be 0 for all other i.

$$t_{max} \geqslant t_i \quad , \quad i = 1, 2, \ldots, n \tag{1}$$

$$t_{max} \leqslant t_i - (z_i - 1)M, \quad i = 1, 2, \ldots, n. \tag{2}$$

$$\sum_{i=1}^{n} z_i = 1 \tag{3}$$

Type 2 Constraints

These $n(n-1)$ constraints ensure that the y_{ij} are defined correctly (i.e., $y_{ij} = 1$ if $t_i > t_j$, $y_{ij} = 0$ if $t_i < t_j$). (Recall that y_{ij} are defined only for $i > j$.) In addition, the constraints provide sufficient travel time for the hoist between adjacent tanks. Constraints (5) ensure that carriers remain in tank i for at least the minimum time required (a_i). The constraints are:

$$t_j - t_i \geqslant c'_{i,i+1} + c_{i+1,j} - y_{ij}M \qquad i > j \tag{4}$$

$$t_i - t_j \geqslant c'_{j,j+1} + a_{j+1} + (y_{ij} - 1)M \qquad i = j + 1 \tag{5}$$

$$t_i - t_j \geqslant c'_{j,j+1} + c_{j+1,i} + (y_{ij} - 1)M, \qquad i > j + 1 . \tag{6}$$

There are two possibilities for given $i > j$, namely $t_i > t_j$ and $t_j > t_i$.

Case 1. $t_i > t_j$. (A carrier is removed from tank j before a carrier is removed from tank i).

The left-hand-side (LHS) of (4) is negative, while the first two terms on the right-hand-side (RHS) of (4) are positive. Therefore the last term on the RHS of (4) must be negative, implying $y_{ij} = 1$.

Case 1.1. $i = j + 1$. Constraint (5) now says that the time between removing a carrier from tank j and removing a carrier from tank $j + 1$ must be at least time enough to take a carrier from tank j to tank $j + 1$ ($c'_{j,j+1}$) plus the minimum time required in tank $j + 1$ (a_{j+1}).

Case 1.2. $i > j + 1$. Constraint (6) now says that the time between removing a carrier from tank j and removing a carrier from tank i must be at least large enough to take a carrier from tank j to tank $j + 1$ ($c'_{j,j+1}$) plus the time for the empty hoist to move from tank $j + 1$ to tank i($c_{j+1,i}$) to pick up another carrier.

Case 2. $t_j > t_i$. (A carrier is removed from tank i before a carrier is removed from tank j.)

The LHS of (5) (if $i = j+1$) or (6) (if $i > j+1$) is negative, while the first two terms on the RHS of (5) (if $i = j+1$) or (6) (if $i > j+1$) are positive. Hence the last term of (5) or (6) must be negative, implying $y_{ij} = 0$. Constraint (4) then says that the time between removing a carrier from tank i and removing a carrier from tank j must be at least large enough to take a carrier from tank i to tank $i + 1$ ($c'_{i,i+1}$) plus the time for the empty hoist to get from tank $i + 1$ to tank j($c_{i+1,j}$).

Type 3 Constraints

These n constraints guarantee that carriers are kept in tanks for an amount of time lying between the minimum (a_i) and maximum (b_i) acceptable amounts. To simplify the expressions, let

$$d = \sum_{i=1}^{n} (c'_{i,i+1} + c_{i+1,0}) z_i .$$

The Type 3 constraints are:

$$(t_{max} + d + t_i) - [t_{i-1} + c'_{i-1,i}] \geqslant a_i - y_{i,i-1}M \quad , \text{all } i \tag{7}$$

$$(t_{max} + d + t_i) - [t_{i-1} + c'_{i-1,i}] \leqslant b_i + y_{i,i-1}M \quad , \text{all } i \tag{8}$$

$$t_i - [t_{i-1} + c'_{i-1,i}] \leqslant b_i + (1 - y_{i,i-1})M \text{ , all } i. \tag{9}$$

Case 1: There *is* a carrier in tank i at time 0.

The carrier in tank i must be removed before any other carrier could be removed from tank $i - 1$. This is so because once the other carrier is removed from tank $i - 1$ and taken to tank i, tank i must be empty. Therefore $t_i < t_{i-1}$, and by the Type 2 constraints, $y_{i,i-1} = 0$.

Constraint (9) is now satisfied automatically. Constraints (7) and (8) now force $(t_{max} + d + t_i) - [t_{i-1} + c'_{i-1,i}]$ to lie between a_i and b_i. But $t_{max} + d + t_i$ is just the time (during the following cycle) when a carrier is removed from tank i, and this carrier is placed into tank i at time $t_{i-1} + c'_{i-1,i}$ (during the present cycle). It is clear then that what is being constrained to lie between a_i and b_i is the time a carrier remains in tank i.

Case 2: There is *no* carrier in tank i at time 0.

In this case, a carrier (not necessarily in tank i - 1 at time 0) must be removed from tank i - 1 (and placed into tank i) before it can be removed from tank i. That is, $t_i > t_{i-1}$, which implies (by the Type 2 Constraints) that $y_{i,i-1} = 1$. (7) and (8) are then automatically satisfied. Constraint (9) then says that the time a carrier spends in tank i [the time it leaves, t_i, less the time it arrives, $t_{i-1} + c'_{i-1,i}$] must be no greater than b_i. The fact that the time a carrier spends in tank i must be at least a_i was covered in constraint (5).

This model involves n+1 continuous variables (the n t_i plus $t_{\max}$) and $(n^2+n)/2$ zero-one variables (the $n(n-1)/2$ y_{ij} plus the n z_i). There are $(n+1)^2$ constraints ($2n$+1 Type 1, $n(n-1)$ Type 2, and n Type 3). The mathematical programming problem embodied in this model may be solved using MPSX/MIP [2], [3]. A numerical example is given in Section 3, following the discussion below on how the solution of the above problem should be interpreted.

Interpretation of a Solution

The process of translating an optimal solution $(t_0^*, t_1^*, \ldots, t_n^*)$ of the above model into a hoist schedule proceeds as follows.

1. List the t_i^* in increasing order, for example,

$$t_{n_1}^* < t_{n_2}^* < t_{n_3}^* < \ldots < t_{n_N}^* .$$

That is, the first tank which has a carrier removed from it is tank n_1, etc. We note that if $t_{n_1}^* < t_{n_1+1}^*$, then tank $n_1 + 1$ must have been empty at time 0, in order to have received the carrier removed from tank n_1. If $t_{n_1}^* > t_{n_1+1}^*$, then tank $n_1 + 1$ must have been full at time 0, for it to have had a carrier to be removed.

2. Determine the optimal times, τ_i, at which the carrier enters tank i, $i = 0, 1, 2, \ldots, n$. These times must be compatible with the optimal tank departure times, $\{t_i^*, i = 0, 1, 2, \ldots, n\}$. Most of the remainder of this section will be devoted to explaining how the τ_i's can be determined.

3. Complete the specification of the hoist movement by listing the times when the hoist is actually moving and pausing. This is straightforward once the t_i^*'s and τ_i's are determined.

We now return to explaining how the τ_i's may be determined. Let $s(i)$ be the tank to which the hoist should travel immediately after it takes a carrier from tank i - 1 to tank i. Formally,

$$s(i) = j \text{ iff } t_j^* = \min_{k=1}^{n} \{t_k^* : \; t_k^* > t_{i-1}^*\}$$

$$s(i) = 0 \text{ iff } t_{i-1}^* = t_{\max}^* .$$

Notice that $s(i) \neq i - 1$, and $s(i) = i$ iff the hoist remains at tank i and removes the carrier from tank i next.

The following theorem, whose proof is given in the Appendix, expresses the optimal τ_i as functions of the optimal t_i^*.

Theorem. Let t^*, y^* be values of t, y in an optimal solution, and let c^* be the optimal cycle length.

(i) If $y_{i,i-1}^* = 1$ (tank i is empty at time 0), then τ_i, the optimal entry time for a carrier into tank i, may take on any value in the interval

$$I_i = \left[\max\{t_i^* - b_i ; t_{i-1}^* + c'_{i-1,i}\}, \min\{t_i^* - a_i ; t_{s(i)}^* - c_{i,s(i)}\}\right].$$

(ii) If $y_{i,i-1}^* = 0$ (tank i is full at time 0), then τ_i may take any value in the interval

$$J_i = \left[\max\{t_i^* - b_i \; ; t_{i-1}^* + c'_{i-1,i}\} \bmod c^*, \right.$$

$$\left. \min\{t_i^* - a_i \; ; t_{s(i)}^* - c_{i,s(i)}\} \bmod c^*\right] .$$

Here h mod c^* is defined to be the integer g in $[0, c^*-1]$ such that $h = nc^* + g$ for some integer n.

The procedure for detailing an optimal hoist schedule has now been fully explained. A complete numerical example is given in Section 3.

Multi-function Tanks

It may be that a particular tank must be used for two (or more) steps in the overall process. That is, carriers must be placed into that tank more than once during a cycle. One way of modeling this situation is as follows.

Suppose carriers must be placed into tank i as the ith and jth tanks to be visited. Without loss of generality, we assume $j > i + 1$. Constraints (5) for i - 1 and j - 1 become

$$t_{j-1} - t_{i-1} \geq c'_{i-1,i} + a_i + c'_{i,i+1} + c_{i+1,j} + (y_{j-1,i-1} - 1)M$$

$$t_{i-1} - t_{j-1} \geq c'_{j-1,j} + a_j + c'_{j,j+1} + c_{j+1,i} - y_{j-1,i-1} M .$$

If $t_{j-1} > t_{i-1}$, these constraints force the time between one carrier's leaving tank i-1 and the next carrier's leaving tank j-1 to be at least the travel time from tank i-1 to tank i plus the minimum time in tank i plus travel time from tank i to tank i+1 plus travel time from tank i+1 to tank j-1. If $t_{i-1} > t_{j-1}$, these constraints force a similar time gap with i interchanged with j in the above sentence.

An additional constraint, ensuring that at least one of tanks i and j is empty (i.e., two carriers are not started at steps i and j at time 0) is that

$$y_{j,j-1} + y_{i,i-1} \geq 1 .$$

3. Computational Considerations and Numerical Example

The IBM MPSX/MIP [2], [3] package is a leased set of mixed integer programming programs which use a branch and bound approach. The authors' experience with MIP on hoist problems is detailed in this section. The problem sizes have been between 8 and 13 tanks, always with three as the optimal number of carriers present in the system simultaneously. Whenever possible, supplementary constraints on the t_i and the y_{ij} were used to reduce the problem size. (For example, it may be obvious that some $t_{i_1} > t_{i_2}$.) The tightness of the feasible tank times, i.e., the intervals $[a_i, b_i]$, probably contributed to the short run times by enabling efficient pruning of the solution tree.

Following is a numerical example modeling an actual hoist setup at the Richmond Western Electric plant. This setup has thirteen tanks, listed in Fig. 2 with maximum and minimum times (in seconds). Figure 3 displays the empty hoist travel times (in seconds) between tanks. An MPSX/MIP computer run was made, using these data and the mixed integer programming model of Section 2. An optimal solution was found and proved to be optimal in 0.17 minutes of CPU time on the IBM 370/168 computer.

The first analysis of the output should be of the optimal tank removal times, the t_i^*'s, which may be found in Fig. 4. Ordering these variables yields

$$0 = t_0^* < t_9^* < t_4^* < t_5^* < t_1^* < t_{10}^* < t_6^* < t_2^* < t_{11}^* < t_7^* < t_{12}^* < t_8^* < t_3^* = t_{\max}.$$

Since t_0^* is the smallest t_i^*, tank 0 must be full at time 0, and tank 1 must be empty. Since t_9^* is the next smallest t_i^*, tank 9 must be full, and tank 10 empty. Similarly, tank 4 must be full and tank 5 empty. $t_5^* < t_6^* < t_7^* < t_8^*$ implies tanks 6, 7, and 8 empty. $t_1^* < t_2^* < t_3^*$ implies tanks 2 and 3 empty. $t_9^* < t_{10}^* < t_{11}^* < t_{12}^*$ implies tanks 10, 11, and 12 empty. In summary, the full tanks just prior to time 0 are 0, 4, and 9.

The times carriers are placed into tanks, the τ_i, are derived by a simple computer program implementing the theorem of Section 2. Figure 4 lists the data used and intermediate numbers derived in obtaining the τ_i. The detailed hoist schedule is given in Fig. 5.

4. Summary

A mathematical programming model of a chemical treatment process for electronic circuit boards has been given. A numerical example of a particular hoist setup was solved and analyzed, using the MPSX/MIP mixed integer programming package. Hopefully this model can serve as a prototype for modeling and solving analogous assembly-line problems.

Tank (i)	0	1	2	3	4	5	6	7	8	9	10	11	12
Minimum Time (a_i)	120	150	90	120	90	30	60	60	45	130	120	90	30
Maximum Time (b_i)	∞	200	120	180	125	40	120	120	75	∞	∞	120	60

Fig. 2. Steps of hoist setup.

	Tank i												
c_{ij}	0	1	2	3	4	5	6	7	8	9	10	11	12
Tank j 0	0	11	14	16	14	19	22	24	26	29	6	8	10
1	11	0	2	5	2	8	10	13	15	17	10	3	1
2	14	2	0	2	0	5	8	10	13	15	12	6	3
3	16	5	2	0	2	3	5	8	10	13	15	8	6
4	14	2	0	2	0	5	8	10	13	15	12	6	3
5	19	8	5	3	5	0	3	5	7	10	18	11	9
6	22	10	8	5	8	3	0	2	5	7	20	14	11
7	24	13	10	8	10	5	2	0	2	5	23	16	14
8	26	15	13	10	13	7	5	2	0	2	25	19	16
9	29	17	15	13	15	10	7	5	2	0	27	21	19
10	6	10	12	15	12	18	20	23	25	27	0	7	9
11	8	3	6	8	6	11	14	16	19	21	7	0	2
12	10	1	3	6	3	9	11	14	16	19	9	2	0

Fig. 3. Empty hoist travel times between tanks.

i	a_i	b_i	$c'_{i-1,i}$	t_i^*	$s(i)$	$c_{i,s(i)}$	$y^*_{i,i-1}$	I_i	J_i
0	120	∞	30	0	8	26	0	-	{450}
1	150	200	31	195	9	17	1	{31}	-
2	90	120	22	316	10	12	1	{217}	-
3	120	180	22	511	11	8	1	{338}	-
4	90	125	22	107	0	14	0	-	[562, 566]
5	30	40	25	162	5	0	1	{132}	-
6	60	120	23	284	1	10	1	{185}	-
7	60	120	22	382	2	10	1	{306}	-
8	45	75	22	476	12	16	1	{404}	-
9	130	∞	22	48	3	13	0	-	{498}
10	120	∞	47	229	4	12	1	{95}	-
11	90	120	27	346	6	14	1	{256}	-
12	30	60	22	420	7	14	1	{368}	-

Fig. 4. Calculations involved in obtaining the τ_i.

Time	Action of Hoist	Time	Action of Hoist
0	Remove C3 from TO (C3 now loaded)	306	Drop C2 into T7.
0-31	Take C3 from T0 to T1.	306-316	Go from T7 to T2 empty.
31	Drop C3 into T1.	316	Remove C3 from T2.
31-48	Go from T1 to T9 empty.	316-338	Take C3 from T2 to T3.
48	Remove C1 from T9.	338	Drop C3 into T3.
48-95	Take C1 from T9 to T10	338-346	Go from T3 to T11 empty.
95	Drop C1 into T10.	346	Remove C1 from T11.
95-107	Go from T10 to T4 empty.	346-368	Take C1 from T11 to T12.
107	Remove C2 from T4.	368	Drop C1 into T12.
107-132	Take C2 from T4 to T5.	368-382	Go from T12 to T7 empty.
132	Drop C2 into T5.	382	Remove C2 from T7.
132-162	Dwell at T5.	382-404	Take C2 from T7 to T8.
162	Remove C2 from T5.	404	Drop C2 into T8.
162-185	Take C2 from T5 to T6.	404-420	Go from T8 to T12 empty.
185	Drop C2 into T6.	420	Remove C1 from T12.
185-195	Go from T6 to T1 empty.	420-450	Take C1 from T12 to T0.
195	Remove C3 from T1.	450	Unload C1, load C4.
195-217	Take C3 from T1 to T2.	450-476	Go from T0 to T8 empty.
217	Drop C3 into T2.	476	Remove C2 from T8.
217-229	Go from T2 to T10 empty.	476-498	Take C2 from T8 to T9.
229	Remove C1 from T10.	498	Drop C2 into T9.
229-256	Take C1 from T10 to T11.	498-511	Go from T9 to T3 empty.
256	Drop C1 into T11.	511	Remove C3 from T3.
256-270	Go from T12 to T6 empty.	511-544	Dwell at T3.
270-284	Dwell at T6.	544-566	Take C3 from T3 to T4.
284	Remove C2 from T6.	566	Drop C3 into T4.
284-306	Take C2 from T6 to T7.	566-580	Go from T4 to T0 empty.

Fig. 5. Optimal hoist operations.

References

[1] Khan, M. Usman, "Simulation Model of an Overhead Crane System," *Industrial Engineering*, **3** 9, 13-17 (September 1971).

[2] Mathematical Programming System–Extended (MPSX), and Generalized Upper Bounding (GUB), IBM Program Product SH20-0968-1.

[3] Mathematical Programming System Extended (MPSX) Mixed Integer Programming (MIP), IBM Program Product SH20-0908-1.

Appendix

Proof of Theorem: (i) It must be shown that the intervals I_i are nonempty and that the range of values which they allow for the τ_i's are consistent with the t_i^*'s. To motivate the proof, we indicate the origin of the expression for the I_i's. There are three basic restrictions on the τ_i. Tank i empty at time 0 implies that $\tau_i \leqslant t_i^*$. First, the carrier must remain in tank i for a period of time lying between a_i and b_i. Formally,

$$a_i \leqslant t_i^* - \tau_i \leqslant b_i . \qquad (10)$$

Second, there must be sufficient time to bring the carrier from tank i-1 to tank i; that is,

$$\tau_i \geqslant t_{i-1}^* + c'_{i-1,\,i} . \qquad (11)$$

Third, there must be sufficient travel time, after leaving the carrier in tank i, to arrive at tank $s(i)$ to remove a carrier by time $t_{s(i)}^*$. That is,

$$\tau_i + c_{i,\,s(i)} \leqslant t_{s(i)}^* . \qquad (12)$$

It is easy to check that $\tau_i \in I_i$ if and only if τ_i satisfies (10), (11), and (12).

It remains to show that I_i is nonempty, i.e., that

$$\max\{t_i^* - b_i ; t_{i-1}^* + c'_{i-1,\,i}\} \leqslant \min\{t_i^* - a_i ; t_{s(i)}^* - c_{i,\,s(i)}\}$$

Subcase 1. $t_i^* - b_i \geqslant t_{i-1}^* - c'_{i-1,\,i} ; t_i^* - a_i \leqslant t_{s(i)}^* - c_{i,\,s(i)}$.

Here $I_i = [\, t_i^* - b_i , t_i^* - a_i \,]$, clearly nonempty since $a_i \leqslant b_i$.

Subcase 2. $t_{i-1}^* + c'_{i-1,\,i} > t_i^* - b_i ; t_i^* - a_i \leqslant t_{s(i)}^* - c_{i,\,s(i)}$.

Here $I_i = [t_{i-1}^* + c'_{i-1,\,i} , t_i^* - a_i]$. Constraint (5) immediately ensures that I_i is nonempty.

Subcase 3. $t_{i-1}^* + c'_{i-1,\,i} > t_i^* - b_i ; t_{s(i)}^* - c_{i,\,s(i)} > t_i^* - a_i$.

Here $I_i = [t_{i-1}^* + c'_{i-1,\,i} , t_{s(i)}^* - c_{i,\,s(i)}]$. For $i > j$, define $y_{ji}^* = 1 - y_{ij}^*$. Since $t_{s(i)}^* > t_{i-1}^*$ [by the definition of $s(i)$], it is clear that $y_{s(i),\,i-1}^* = 1$. There are two possibilities to be considered, namely $i - 1 > s(i)$ and $s(i) > i - 1$ [$s(i)$ can never be $i - 1$]. When $i - 1 > s(i)$, replace j by $s(i)$ and i by $i - 1$ in constraint (4) to obtain $t_{i-1}^* + c'_{i-1,\,i} \leqslant t_{s(i)}^* - c_{i,s(i)}$. When $s(i) > i - 1$, replace i by $s(i)$ and j by $i - 1$ in constraint (6), to obtain the same result. But this is precisely what was needed to show I_i nonempty.

Subcase 4. $t_i^* - b_i \geqslant t_{i-1}^* + c'_{i-1,\,i} ; t_{s(i)}^* - c_{i,s(i)} < t_i^* - a_i$.

Here $I_i = [t_i^* - b_i , t_{s(i)}^* - c_{i,\,s(i)}]$. Constraint (9) gives $t_i^* - b_i \leqslant t_{i-1}^* + c'_{i-1,\,i}$, since $y_{i,\,i-1}^* = 1$. Following the same procedure as in Subcase 3 yields $t_{i-1}^* + c'_{i-1,\,i} \leqslant t_{s(i)}^* - c_{i,s(i)}$. Combining these two inequalities gives $t_i^* - b_i \leqslant t_{s(i)}^* - c_{i,\,s(i)}$, proving I_i nonempty.

(ii) The proof that J_i is nonempty is directly analogous to the proof that I_i is nonempty, except that one must take care of negative expressions by adding c^* to them.

Q.E.D.

Mr. Larry W. Phillips is a Planning Engineer at Western Electric Company in Richmond, Virginia. He received his BS Degree in Electrical Engineering from Virginia Polytechnic Institute and State University. His work is currently involved with the planning and development of engineering computer systems. He is a member of the Institute of Electrical and Electronic Engineers.

Dr. Philip S. Unger is a Member of Technical Staff at Bell Laboratories. His research activities lie in mathematical programming. He received the BES Degree from The Johns Hopkins University, and the MS and PhD Degrees from Northwestern University, all in Operations Research–related fields. He is a member of ORSA, the Mathematical Programming Society, and various honor societies.