AN INTEGER LINEAR-PROGRAMMING MODEL FOR MACHINE SCHEDULING*

Harvey M. Wagner
Stanford University†

I. INTRODUCTION

Several authors [1, 2, 3, 10, 12, 13] have previously considered versions of the following machine-scheduling or sequencing problem: Given n items, each to be processed on one or more of m machines, the order of processing for an item being partially or entirely specified, find the sequencing of items on the machines which minimizes the total elapsed time to complete the manufacture of all items. It is assumed that the manufacturing time of an item on a machine is specified (i.e., nonstochastic) and that in-process inventory is allowable.

In this paper we offer an integer linear-programming model which is capable of characterizing the problem. We feel it is of interest to present the model, because (1) it shows that there is a single model which encompasses a wide variety of machine-scheduling situations, (2) the linear format establishes the existence of a finite algorithm which monotonically seeks an optimum solution to such sequencing problems, (3) although the model is now of very limited computational interest, future developments in integer linear programming and methods for efficiently handling "secondary constraints" may make numerical solutions of particular problems possible, and (4) there is an indication of the possibility of constructing special algorithms to exploit the structure of certain of the "classical" scheduling problems. Needless to say, the major justification for considering such an approach is that Gomory [9] and others [5, 6, 11] have recently discovered promising methods for solving integer linear-programming problems; a related justification is that Dantzig, Fulkerson, and Johnson [8] have achieved noteworthy success in using secondary constraint techniques for solving a problem which a priori contains a mammoth number of restrictions. As will be evident below, the model in its present form is computationally unwieldy except perhaps for situations with a very few machines and a limited number of items; in such cases, a frequently recurring sequencing problem or one involving a considerable financial sum might profitably be solved by the method herein. We are more hopeful, as stated above, that this presentation might serve as a link to a more computationally feasible formulation.

In Section II we discuss a general model which is adaptable to a large number of scheduling situations. The familiar problem [3, 10, 12] of sequencing n items, each of which must be placed on all m machines, the order of manufacturing being machine 1, machine 2, ... , machine m, is considered in Section III. The further restriction, to three machines only, is explored in Section IV.

*Manuscript received November 20, 1958.
†Work reported here was sponsored by the Office of Naval Research.

II. GENERAL FORMULATION

We may picture the restrictions characterizing a specific scheduling sequence problem by means of two tabular arrays. In the first matrix (I), a row corresponds to one of the n items, and a column corresponds to a process stage in the manufacture of the item. For example, if item i must be processed on machine 1, 2, ... , m in that order, we define for the i-th row the entry at process stage 1 to be "machine 1," at process stage 2 to be "machine 2," ... , at process stage m to be "machine m." If an item is not to be placed on every machine, then the number of process stages is less than m. For the sake of expositional simplicity, we postulate that the processing specifications for any item are such that any machine does not appear more than once (if at all) in the i-th row of matrix I; a minor modification of the model may be made to allow for multiple processing. If at some point of the manufacturing process of an item, the item is to be placed on q machines out of a subset (Q) of machines, the specific order of manufacturing being unimportant, then we define the entry at that process stage to be the subset (Q) and the specification q; e.g., item i at the fifth process stage may be required to be placed on any of three (= q) machines in the group (Q) = (machine 5, machine 8, machine 26, machine 27, machine 35). Note that all items do not necessarily have the same number of process stages, and within each stage we permit several of the aforesaid restrictions. Thus we shall consider those production situations in which it is possible to analyze the manufacturing process for each item as a consecutive sequence of stages, each comprising processing on one or more machines, and such that the order of manufacturing within a process stage is irrelevant. By a straightforward extension of our analysis, we could also treat situations requiring processing "substages"; to keep the present exposition within bounds, we omit the details required for this extension.

In the second matrix (II) the rows correspond to the items and the columns to the m machines. The entry $t_i^{(k)}$ at the intersection of the i-th row and k-th column represents the manufacturing time (set-up plus production)* for item i on machine k. If matrix I indicates that under no circumstances is an item to be placed on a particular machine, then the corresponding element in matrix II will not be defined. We postulate that our time units are measured such that every $t_i^{(k)}$ is an integer.

We make the following definitions of nonnegative integer-valued variables

$$x_{ij}^{(k)} = \begin{cases} 1 & \text{if item } i \text{ is scheduled in order-position } j \text{ on machine } k \\ 0 & \text{if item } i \text{ is not scheduled in order-position } j \text{ on machine } k \end{cases}$$

$h_j^{(k)}$ = time at which the item scheduled in order-position j begins processing on machine k

$s_j^{(k)}$ = time elapsing on machine k between completion of processing the item scheduled in order-position j and initiation of processing the item scheduled in order-position $j+1$

$s_0^{(k)}$ = time elapsing on machine k between the initiation of production and the start of processing the item scheduled in the first position.

*We might also let the processing time be a function of the number of items previously placed on machine k; we forego this generality as it would further complicate the notation, but the interested reader may supply the necessary amendments.

By "the item scheduled in order-position j ... on machine k" we mean that, prior to this item being placed on machine k, (j - 1) items have previously been processed thereon.

We let n(k) be the maximum number of items that might ever be processed on machine k. This number may be found by scanning matrix I and counting the total number of times machine k appears in the process-stage listing. It represents an upper bound to the number of ordered positions that need be considered for machine k. We also define N(k) to be the set of items for which machine k appears at some stage.

The first collection of constraints ensures that each item i completes the necessary operations within every process stage. If for a process stage p the item i must be placed on machine k, we require

$$\sum_{j=1}^{n(k)} x_{ij}^{(k)} = 1 . \tag{1}$$

That is, the item i must appear in some ordered position on machine k; the restriction that $x_{ij}^{(k)}$ be integer-valued implies that the item will be scheduled once, and only once, on machine k. If there are Q machines listed for process stage p and the item must be placed on every one of the Q machines, then we have a set of Q equations of the form of Eq. (1), one for each such machine. A similar statement holds for the relations below.

If for a given item i and process stage p the item must be placed on one machine out of a group of machines $k_1, k_2, \ldots, k_Q$, we require

$$\sum_{j=1}^{n(k_1)} x_{ij}^{(k_1)} + \sum_{j=1}^{n(k_2)} x_{ij}^{(k_2)} + \ldots + \sum_{j=1}^{n(k_Q)} x_{ij}^{(k_Q)} = 1 . \tag{2}$$

Finally, if for a process stage p the item i must be placed on any of q machines out of a group of machines $k_1, k_2, \ldots, k_Q$, we require*

$$\sum_{j=1}^{n(k_1)} x_{ij}^{(k_1)} = 1 - \delta_1 \tag{2a}$$

$$\sum_{j=1}^{n(k_2)} x_{ij}^{(k_2)} = 1 - \delta_2 \tag{2b}$$

$$\vdots$$

$$\sum_{j=1}^{n(k_Q)} x_{ij}^{(k_Q)} = 1 - \delta_Q \tag{2Q}$$

$$\delta_1 + \delta_2 + \ldots + \delta_Q = Q - q , \tag{3}$$

*The general form of the constraint suggested is given by Dantzig [7].

where each δ_k is restricted to be 0 or 1. Equations (2) and (3) and the integer restriction on δ ensure that item i is processed on exactly q out of the Q machines.

The second set of constraints guarantees that no more than one item be assigned the j-th ordered position on a machine. For each machine k, we require

$$\sum_{i \in N(k)} x_{ij}^{(k)} \leq 1 \qquad j = 1, 2, \ldots, n(k) . \tag{4}$$

It should be noted that (1), (2), (3), and (4) and the condition that the $x_{ij}^{(k)}$ and δ_k be integer valued are sufficient to restrict these variables to the values 0 and 1, i.e., no additional upper bounds are needed.

Reviewing the above constraints, we recognize there is yet no guarantee that (1) an item be scheduled for process stage p and completed before it is scheduled for process stage p + 1, or even that (2) an item not be scheduled to be processed on two or more machines during the same time. Consequently we need a set of (linear) relations implying that the production schedule observes the process-stage restrictions and does not call for the item being placed on more than one machine at one time. Toward this end we find it convenient to suggest a "shorthand" notation and to give explicit relations for $h_j^{(k)}$.

Let

$$T x_j^{(k)} = \sum_{i \in N(k)} t_i^{(k)} x_{ij}^{(k)} \qquad j = 1, 2, \ldots, n(k) . \tag{5}$$

Given (4), $T x_j^{(k)}$ represents processing time of the j-th ordered item on machine k. Then for each machine k it may be verified that

$$h_1^{(k)} = s_0^{(k)} \tag{6a}$$

$$h_j^{(k)} = T x_1^{(k)} + T x_2^{(k)} + \ldots + T x_{j-1}^{(k)} + s_0^{(k)} + s_1^{(k)} + \ldots + s_{j-1}^{(k)} . \qquad j = 2, 3, \ldots, n(k) \tag{6b}$$

Equation (6) states that the item in order-position j on machine k commences processing at a time equal to the sum of the manufacturing and idle periods accumulated from the initial commencement of production.

First we consider the restriction that for a particular item i any machining taking place in process stage p + 1 may commence only after all machining is completed in process stage p. We suppose in the i-th row of matrix I machine k_1 appears in the entry for column p and machine k_2 appears in the entry for column p + 1. Suppose further that $x_{ij'}^{(k_1)} = 1$ and $x_{ij''}^{(k_2)} = 1$, i.e., item i is scheduled in order-position j' on machine k_1 and in order-position j'' on machine k_2. For the schedule to be feasible

$$h_{j'}^{(k_1)} + t_i^{(k_1)} x_{ij'}^{(k_1)} \leq h_{j''}^{(k_2)} . \tag{7}$$

Given our hypothesis about the specific order of production for item i, (7) guarantees that machining on (k_2) is not commenced until machining on (k_1) is completed. But we cannot add (7) in its present form as a constraint, since it requires too much, viz., that the starting time of the j''-positioned item on machine k_2 never precedes the finishing time of the j'-positioned item on machine k_1; we want (7) to hold only whenever item i happens to be the item scheduled in both of these ordered positions. Therefore we merely need to require

$$h_{j'}^{(k_1)} + t_i^{(k_1)} x_{ij'}^{(k_1)} \leq h_{j''}^{(k_2)} + M\left(1 - x_{ij'}^{(k_1)}\right) + M\left(1 - x_{ij''}^{(k_2)}\right), \tag{8}$$

where M is a large positive integer and $h_j^{(k)}$ is evaluated by (6). Observe that (8) is a binding constraint only if $x_{ij'}^{(k_1)} = x_{ij''}^{(k_2)} = 1$.

In general, for item i, each pair of process stages p and p + 1, each ordered couple (machine k_1 in process stage p, machine k_2 in process stage p + 1), and $j' = 1, 2, \ldots, n(k_1)$, $j'' = 1, 2, \ldots, n(k_2)$, we have a constraint of the form (8).* Although the total number of such constraints for the model is obviously enormous, only a relatively few of the relations will be binding in any solution. This fact suggests the technique of treating relations of the type (8) as secondary constraints [4, 8, 14]; in other words we might attempt to solve a scheduling problem by a series of trial solutions in which constraints are introduced only as needed to eliminate infeasibilities.

Secondly we consider for item i the restriction that any machining taking place within process stage p must be on only one machine at a time. We suppose machines k_1 and k_2 appear in the i-th row and p-th column of matrix I. Under the assumption $x_{ij'}^{(k_1)} = x_{ij''}^{(k_2)} = 1$, we require that one of the relations below must hold in order that the schedule be feasible:

$$h_{j'}^{(k_1)} + t_i^{(k_1)} x_{ij'}^{(k_1)} \leq h_{j''}^{(k_2)}, \tag{9a}$$

$$h_{j''}^{(k_2)} + t_i^{(k_2)} x_{ij''}^{(k_2)} \leq h_{j'}^{(k_1)}. \tag{9b}$$

Analogous to the reasoning in (8), we want either (9a) or (9b) to hold only whenever item i happens to be the item scheduled in both of these ordered positions. Our linear constraint then is the pair†

$$h_{j'}^{(k_1)} + t_i^{(k_1)} x_{ij'}^{(k_1)} \leq h_{j''}^{(k_2)} + M\left(1 - x_{ij'}^{(k_1)}\right) + M\left(1 - x_{ij''}^{(k_2)}\right) + \delta M, \tag{10a}$$

$$h_{j''}^{(k_2)} + t_i^{(k_2)} x_{ij''}^{(k_2)} \leq h_{j'}^{(k_1)} + M\left(1 - x_{ij'}^{(k_1)}\right) + M\left(1 - x_{ij''}^{(k_2)}\right) - M(\delta - 1), \tag{10b}$$

where δ is either 0 or 1.

*As the reader may verify, we need not consider constraints between anything more remote than consecutive pairs of process stages.

†The general form of the δ constraint suggested is given in [7].

In general, for item i, each process stage p, each couple (machine k_1 in process stage p, machine k_2 in process stage p), and $j' = 1, 2, \ldots, n(k_1)$, $j'' = 1, 2, \ldots, n(k_2)$, we have a constraint of the form (10).*

We finally come to the question of the objective form. Let h^* represent the earliest point in time at which processing on all items has been completed according to a given trial schedule; we desire to find that schedule which minimizes h^*. If, for example, all items must be "finished-off" on machine m, then the optimizing form is simply minimize $\left(h_n^{(m)} + T x_n^{(m)}\right)$, where (6) is used to evaluate $h_n^{(m)}$. If a "finishing" machine does not exist and there is no a priori reason to believe that a particular machine will be the last one in operation, then we must allow for the possibility of any machine being the last in operation. One method of solving the problem is to minimize h^*, subject to the additional constraints

$$h_{n(k)}^{(k)} + T x_{n(k)}^{(k)} \leq h^* \qquad k = 1, 2, \ldots, m, \tag{11}$$

where, as in (8) and (10), the term on the extreme left hand side of (11) is evaluated by (6).†

To summarize, we have formulated a scheduling model capable of handling, theoretically, a wide class of machine-sequencing problems. In spite of the fact that the total number of constaints to be satisfied is staggering, most of them are inoperative, and the technique of secondary constraints may prove useful. In addition to the complexity of the problem due to its size alone, a further computational difficulty is introduced by the requirement that an integer linear-programming algorithm of some sort be employed. Nevertheless, the demonstration that there does exist a finite procedure which monotonically searches for an optimal solution‡ affords some encouragement that eventually a practicable algorithm may be developed.

III. THE CLASSICAL PROBLEM OF THE m-STAGE PROCESS

A well-known special case of the above model is the problem: Given n items, each to be processed on m machines, the order of processing being specified as machine 1, machine 2, ..., machine m, find the sequencing of items on the machines that minimizes the time at which the last item is completed on machine m [2, 3, 10, 12].

Now all items must go through m process stages, machine k is specified for process stage k, and n(k) = n. Constraints (1) and (4) for each machine $k = 1, 2, \ldots, m$ become

*As the reader may verify, we need consider only all pairs (k_1, k_2) to ensure feasibility within process stage.

†An alternative device would be to define a ficticious "finishing" machine with process time equaling zero.

‡A comment is called for in explaining the meaning of a "monotonic" search. The modus operandi of most extant integer programming algorithms is to ignore the integer constraints and to find an optimal solution to the linear-programming model by a standard technique; if a nonintegral solution is thereby obtained, constraints are added one by one which eventually force the enlarged problem to yield an optimal integer solution to the original model. Consequently the value of the optimizing form is first (monotonically) brought, say, to a minimum, and then monotonically increased by the addition of new constraints. Similarly in our model, if both the integer restrictions as well as the feasibility constraints are added as needed, the optimizing form would first be minimized and then increased monotonically as feasibility is established. The important claim for a finite monotonic procedure, of course, is that in most problems it eliminates having to enumerate completely all possible solutions in order to find an optimal one.

$$(1') \qquad \sum_{j=1}^{n} x_{ij}^{(k)} = 1 \qquad i = 1, 2, \ldots, n$$

$$(4') \qquad \sum_{i=1}^{n} x_{ij}^{(k)} = 1 . \qquad j = 1, 2, \ldots, n$$

Notice that for each value of k, the relations (1') and (4') describe a standard "assignment-problem" model.

By the use of (6) and (1') evaluated for machine m, the minimizing form reduces as follows

$$(12a) \quad h_n^{(m)} + Tx_n^{(m)} = Tx_1^{(m)} + Tx_2^{(m)} + \ldots + Tx_n^{(m)} + s_0^{(m)} + s_1^{(m)} + \ldots + s_{n-1}^{(m)}$$

$$= t_1^{(m)} \left(x_{11}^{(m)} + x_{12}^{(m)} + \ldots + x_{1n}^{(m)} \right) + \ldots + t_n^{(m)} \left(x_{n1}^{(m)} + x_{n2}^{(m)} + \ldots + x_{nn}^{(m)} \right) + \ldots$$

$$(12b) \qquad + s_0^{(m)} + \ldots + s_{n-1}^{(m)}$$

$$(12c) \quad = \text{constant} + s_0^{(m)} + s_1^{(m)} + \ldots + s_{n-1}^{(m)} .$$

In (12) we have the commonly recognized relation [2, 10] that the optimizing problem may be stated as minimizing the total amount of idle time on machine m.

We may, of course, use constraints of the form (8) to ensure a feasible processing sequence. But we may also formulate another set of restrictions, based on this model's special assumptions, which encompasses many (although not all) of the restrictions implied by the full set of relations (8). To do this we define nonnegative integer valued variables

$u_j^{(k)}$ = time elapsing between the completion of processing the item scheduled in order-position j on machine k and the start of processing the item scheduled in order-position j on machine k + 1.

We know the item scheduled in order-position j on machine k + 1 cannot commence processing before the item scheduled in order-position j on machine k is completed; for, otherwise, there would be a violation of the restriction that the machine order of production is the same for every item. Note that the previous statement does not imply that the item in order-position j on machine k is the same item as that in order-position j on machine k + 1; the restriction is solely one of the time interdependence among process stages.

Explicitly for period $j (> 1)$ and machine $k (> 1)$ we have

$$(13) \qquad h_j^{(k)} = h_j^{(k-1)} + Tx_j^{(k-1)} + u_j^{(k-1)} .$$

That is, the time at which the processing of the item at order-position j on machine k begins must not be earlier than the time at which the item at order-position j on machine k - 1 is completed.

We may also write (6) in the form

$$h_j^{(k)} = h_{j-1}^{(k)} + T x_{j-1}^{(k)} + s_{j-1}^{(k)} . \tag{6'}$$

Subtracting (13) from (6'), we get

$$0 = h_{j-1}^{(k)} - h_j^{(k-1)} + T x_{j-1}^{(k)} - T x_j^{(k-1)} + s_{j-1}^{(k)} - u_j^{(k-1)} . \tag{14}$$

Subtracting (14) evaluated at j' and k' from (14) evaluated at j' + 1 and k' and using (6'), we derive the constraints

$$0 = T x_{j'}^{(k')} - T x_{j'+1}^{(k'-1)} + s_{j'}^{(k')} - s_{j'}^{(k'-1)} + u_{j'}^{(k'-1)} - u_{j'+1}^{(k'-1)} . \tag{15}$$

$$1 < k' \leq m \quad , \quad 1 < j'+1 \leq n$$

It may be verified that the (m-1)(n-1) relations in (15) comprise a modified transportation problem; each variable appears in no more than two of the relations, although the $x_{ij}^{(k)}$ have coefficients other than 1 or -1.

The structure of the model [2, 10] also permits us to set

$$s_j^{(1)} = u_1^{(1)} = u_1^{(m-1)} = 0 \tag{16}$$

without loss of optimality.

Clearly (15) imposes restrictions only on the timing of processing the item scheduled in order-position j, but does not refer to any timing constraints on the particular items themselves. Consequently if we were to use a secondary constraint technique to solve a specific problem, we would need certain of the relations (8) in addition to those in (15) to reach a feasible solution.

IV. THE CASE OF THREE MACHINES

A particularly interesting case of the model in Section III is with m = 3. This class of problems is distinguished by the fact that, without loss of optimality, we may confine our attention to schedules which sequence the n items in the same order on all three machines [2, 10]. Mathematically, this means that we may restrict the values of the unknowns by the relations

$$x_{ij}^{(1)} = x_{ij}^{(2)} = x_{ij}^{(3)} \qquad i = 1, 2, \ldots, n \quad \text{and} \quad j = 1, 2, \ldots, n .$$

As a consequence, we need only impose (1') and (4') for k = 1, comprising a system of 2n relations and n^2 unknowns (in the format of an assignment problem, as we noted above). Similarly, (15) may be rewritten in terms of $x_{ij}^{(1)}$, yielding a set 2(n-1) equations.

It may be verified that the minimizing form can be written as

$$\text{minimize} \quad Tx_1^{(1)} + Tx_1^{(2)} + s_1^{(3)} + s_2^{(3)} + \ldots + s_{n-1}^{(3)} .$$

Finally we note that, in this special case, the only timing restrictions necessary are those imposed by (15), i.e., there is no need for any additional constraints of the form (8). This conclusion follows from the observation that now there is no ambiguity between "the item scheduled in order-position j on the machine k" and the j-th item in the scheduling sequence.

Thus we have derived a fundamental system of the order of 4n equations which probably can be solved* for $n \leq 25$ on high-speed computing machinery; but it of course remains questionable whether or not such a computational proposal for finding an optimal solution is economically sound.

REFERENCES

[1] Akers, S. B., Jr., and J. Friedman, "A Non-Numerical Approach to Production Scheduling Problems," Operations Research 3:429-442 (1955).

[2] Bellman, R., "Mathematical Aspects of Scheduling Theory," J. Soc. Indus. and Appl. Math. 4:168-205 (1956).

[3] Bellman, R., "Formulation of Recurrence Equations for Shuttle Process and Assembly Line," Nav. Res. Log. Quart. 4:321-334 (1957).

[4] Dantzig, G. B., "Upper Bounds, Secondary Constraints, and Block Triangularity in Linear Programming," Econometrica 23:174-183 (1955).

[5] Dantzig, G. B., "Solving Linear Programs in Integers," RAND Corp. P-1359, May 1958.

[6] Dantzig, G. B., "On Integer and Partial Integer Linear Programming Problems," RAND Corp. P-1410, June 1958.

[7] Dantzig, G. B., "On the Significance of Solving Linear Programming Problems with Some Integer Variables," RAND Corp. P-1486, Sept. 1958.

[8] Dantzig, G. B., D. R. Fulkerson, and S. Johnson, "Solution of a Large-Scale Traveling Salesman Problem," Operations Research 2:393-410 (1954).

[9] Gomory, R. E., "Outline of an Algorithm for Integer Solutions to Linear Programs," Am. Math. Soc. Bull. 64:275-278 (1958).

[10] Johnson, S., "Optimal Two- and Three-Stage Production Schedules with Setup Times Included," Nav. Res. Log. Quart. 1:61-68 (1954).

[11] Markowitz, H. M., and A. S. Manne, "On the Solution of Discrete Programming Problems," Econometrica 25:84-110 (1957).

*Recall that additional constraints will be added to ensure an integer valued solution [9].

[12] Pollack, M., "Some Studies on Shuttle and Assembly-Line Processes," Nav. Res. Log. Quart. 5:125-136 (1958).

[13] Salveson, M. E., "The Assembly Line Balancing Problem," National Bureau of Standards, Second Symposium on Linear Programming 1:55-101 (1955).

[14] Wagner, H. M., "A Practical Guide to the Dual Theorem," Operations Research 6:364-384 (1958).

* * *