IIE Transactions

ISSN: 0740-817X (Print) 1545-8830 (Online) Journal homepage: www.tandfonline.com/journals/uiie20

'Just-In-Time' Production of Large Assemblies

A. Agrawal, G. Harhalakis, I. Minis & R. Nagi

To cite this article: A. Agrawal, G. Harhalakis, I. Minis & R. Nagi (1996) 'Just-In-Time' Production of Large Assemblies, IIE Transactions, 28:8, 653-667, DOI: 10.1080/15458830.1996.11770710

To link to this article: https://doi.org/10.1080/15458830.1996.11770710

Published online: 13 Sep 2016.

IIE Transactions (1996) **28**, 653–667

'Just-in-time' production of large assemblies

A. AGRAWAL[1], G. HARHALAKIS[1], I. MINIS[1] and R. NAGI[2]

[1]*Department of Mechanical Engineering & Institute for Systems Research, University of Maryland, College Park, MD 20742, USA*
[2]*Department of Industrial Engineering, 342 Bell Hall, SUNY at Buffalo, Buffalo, NY 14260, USA*

Received August 1994 and accepted February 1995

The problem of scheduling operations in a manufacturing facility that produces large assemblies is considered. Given the due dates of the end items, the objective is to determine a schedule that minimizes the cumulative production lead time. The lot sizes of the end items are considered known and 'just-in-time' (JIT) production is assumed. The scheduling problem within such an environment has been formulated and is NP-hard. Its similarities to and differences from the well studied resource-constrained project scheduling problem have been established. An efficient heuristic is developed that uses the precedence network of operations to prioritize execution of those operations that belong to a dynamically computed critical path. The effectiveness of the proposed heuristic is assessed in terms of cumulative lead-time and work-in-process inventory (WIP) costs.

1. Introduction

The production environment addressed by this study is a manufacturing facility that produces large and complex assemblies. Within such an environment, planning and scheduling involve a host of decisions related to lot-sizing, timing, and stockpiling (Zahorik *et al.*, 1984). The study focuses specifically on scheduling the production of a small set of large assemblies under multiple capacity constraints.

The scheduling objective is to minimize the cumulative lead time of the production schedule (total makespan). This is a critical issue for large assembly manufacturing in which finished product cycle times range between 6 months and 2 years. In addition, a robust scheduling capability is important within this environment to support production planning decisions, such as estimation of product lead times and cost. A 'just-in-time' (JIT) production strategy is adopted to develop the production schedule by delaying operations to be as late as possible. Although the strategy considered is similar to that employed by MRP (Orlicky, 1975) and MRP II, these systems do not explicitly consider detailed capacity constraints. They plan production simply by backtracking from the due date using lead times that are defined *a priori* (Billington *et al.*, 1983). This often results in infeasible schedules and subsequent delays in completion times. Some production planning and scheduling systems that perform the functions of MRP II while considering capacity constraints have also been proposed. Excluding Goldratt's algorithms (1980), which are proprietary, these systems (Reiter, 1966; Godin and Jones, 1969; Goldratt, 1980) do not perform satisfactorily in a multistage environment that includes more than one capacity constraint (Billington *et al.*, 1983). An interesting production planning and scheduling tool is the OPT system (Levulis, 1985), which also employs proprietary algorithms. It identifies bottleneck resources through simulation and uses heuristic procedures to load these resources during periods when they are starved. It is noted, however, that comparisons of these proprietary systems with conventional production scheduling strategies are not available.

The following discussion of the literature focuses on (i) the scheduling of multilevel products within a job-shop and (ii) selected work in project scheduling and network programming. Both these areas are relevant to the problem of producing large assemblies within the shortest possible makespan.

1.1. *Job-shop scheduling of multilevel products*

The vast literature in the scheduling field has been reviewed and classified by a number of authors. Panwalkar and Iskander (1976) present and critique a comprehensive list of different priority rules proposed in the literature. Rodammer and White (1988) survey different approaches to the production scheduling problem such as machine sequencing, resource-constrained project scheduling, discrete event simulation and artificial intelligence methods. MacCarthy and Liu (1993) have tabulated the types of scheduling problem addressed in the literature along with the most important solution methods. For the n-job, m-machine, job-shop problem, for which the objective targeted is minimization of the makespan, various approaches have been proposed, including those based on mixed-integer linear program-

ming (MILP) (Greenberg, 1968) and branch-and-bound (Ashour and Hiremath, 1973; Lageweg *et al.*, 1977; Barker and McMahon, 1985; Carlier and Pinson, 1989). For practical cases the optimal methods are computationally prohibitive, given that the underlying problem is accepted to be NP-hard. Thus, various heuristic approaches have been proposed (Ashour, 1967; Adams *et al.*, 1988). Adams *et al.* (1988) present a 'shifting bottleneck' method that uses an iterative procedure inspired by the single-machine scheduling problem. This method has been applied to a number of standard scheduling examples and has shown superior performance for realistic sized problems. It is emphasized, however, that it schedules a given set of operations on their respective workcenters and does not allow for precedence relationships between routings. Therefore, although it is applicable to single-level product structures, it cannot be easily extended to multilevel products, such as large assemblies.

Billington *et al.* (1983) have classified multilevel product structures into four types and proposed a two-stage approach for the multilevel job-shop scheduling problem. The first stage uses a heuristic, which is based on bottleneck resource identification and lot-sizing, to compress the product structure and determine the critical manufacturing activities. An MILP method is employed in the second stage to find the optimal makespan for the reduced problem. No comparisons of makespan obtained from the compressed product structure with the minimum makespan of the complete product structure have been presented.

1.2. *Project scheduling and network programming*

Project scheduling, network programming, and machine sequencing methods are relevant to job-shop scheduling and have been studied extensively by numerous authors.

Mixed-model assembly line scheduling methods have been proposed by Miltenburg and Sinnamon (1989, 1992). Zahorik *et al.* (1984) have employed network programming models to solve the multistage, multi-item scheduling problem. Boctor (1990) has discussed basic optimality criteria for project scheduling and presented some rule-based heuristics to minimize the project delay. These include MINSLK, in which the activity with the smallest slack is scheduled first, RSM (resource scheduling method) and MINLFT, in which the activity with the smallest late finish time is scheduled first. The author suggested that a single heuristic may not address adequately a wide spectrum of applications. Instead, if a set of heuristics is applied to a given problem, the best solution is typically close to the optimal. Ulusoy and Ozdamar (1989) have presented a weighted resource utilization algorithm (WRUP) for the project scheduling problem, which performed slightly better than the rules presented by Boctor (1990). The main difference from previous algorithms is that WRUP is neither resource nor time oriented.

Khattab and Choobineh (1991) have listed a number of rule-based heuristics for single resource constraint project scheduling, including ROT (Elsayed, 1982), ACTIM (Bedworth and Bailey, 1982), and TIMROS (Elsayed and Nasr, 1986). These heuristics use the PERT/CPM type network to prioritize operations. The authors employed a family of eight heuristics (collectively called 'SEARCH'), each using a different prioritization rule to generate feasible schedules for a given problem. From the solutions obtained, the schedule giving the smallest makespan was selected. Makespan comparisons between 'SEARCH' and other methods were presented showing that the former performed better over the 27 different examples tested. Similar suggestions have been made in the literature advocating the use of a small set of proven heuristics to solve complex scheduling problems, rather than the computationally prohibitive enumeration and/or integer programming approaches. The above heuristics apply to mixed-model assembly line scheduling or project scheduling. The problem of sequencing a given set of operations has been extensively researched for one or two machines and some heuristics such as SPT have been shown to perform well. However, the capacity constrained job-shop scheduling problem has not been adequately addressed in the literature owing to its extreme complexity.

Manufacture of large assemblies consists of a large number of operations with precedence constraints. Conventional production scheduling approaches that do not account for such constraints cannot be applied directly to this case. In addition, even though machine sequencing methods are more relevant to this problem, they cannot be readily applied to facilities with multiple machines and capacity constraints.

The method proposed in this study addresses the case of large assembly manufacturing, in which both precedence and capacity constraints are critical, and generates a schedule of operations with a near-optimal makespan (as indicated by extensive empirical study). It can be thought of as an integrated production planning and scheduling tool, which derives the release dates of jobs and operations from the detailed schedule. The schedule is generated by considering the critical path of a precedence network of operations obtained from the Bills of Materials (BOMs) of the products to be manufactured and the operation sequences (routings) of all make items. The method has been applied to a large number of examples and the results have been compared with those obtained from other heuristics found in the project and job-shop scheduling literature. Comparisons with the optimal schedules obtained from a branch-and-bound algorithm (Agrawal *et al.*, 1994) are also presented for small dimensional problems. It is shown that for large assembly-type problems, the proposed heuristic performs better than a set of 15 heuristics, which have been

extensively employed in project scheduling and/or machine sequencing.

The paper is structured as follows. The problem is formulated in Section 2. The proposed scheduling heuristic is presented in Section 3. The performance of the heuristic is evaluated and discussed in Section 4. Applications of the proposed methodology are suggested in Section 5. Finally, the conclusions are presented in Section 6.

2. Problem formulation

A manufacturing facility consisting of m work-centers $\{\mathcal{W}_1, \mathcal{W}_2, \ldots, \mathcal{W}_m\}$ is considered. Each work-center $\mathcal{W}_i$ may comprise f_i functionally identical machines. The production system transforms raw materials into finished products through a series of semi-finished (intermediate) parts. We consider a set of final assemblies, $\mathcal{P}_f = \{p_1, p_2, \ldots, p_{n_f}\}$, and a set of all make parts, $\mathcal{P}_m \supset \mathcal{P}_f$. Semi-finished or make parts are assumed to be inventory items, i.e., they contribute to inventory holding costs. Each make part $p_i \in \mathcal{P}_m$, has a unique sequence of operations (routing) denoted by $< o_{ij} >_{j=1}^{s_i}$. Further, each finished product p_i is associated with a unique bill-of-materials (BOM), which represents its assembly structure. A sample BOM of a single finished product A is shown in Fig. 1, along with the routings of its make items. A delivery schedule consists of a list of finished products, along with the associated quantities and due-dates. Let D_i and q_i represent the due date and order quantity respectively of product p_i.

In this study the back-to-back batch production strategy is adopted, i.e., the entire batch of a part is processed by a certain machine before the next operation is commenced. Furthermore, batches are not distributed between functionally identical machines. This strategy is believed to be most common in industrial practice. Finally, the following additional assumptions are made:

- a lot-for-lot strategy is employed for make items;
- machines are reliable;
- a machine can perform only one operation at a time;
- an operation can be performed on at most one machine at a given time;
- pre-emption of operations is not permitted;
- processing times of all operations and due dates of all final products are deterministic and known *a priori*.

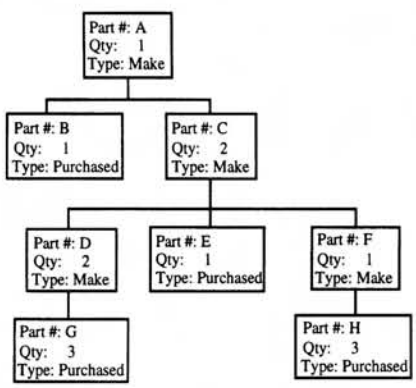
2.1. *Network representation of precedence relationships*

For each end item, the precedence relationships among operations, which may be obtained from the BOM and part routings, can be conveniently represented by a precedence network. This network may also incorporate *lead time offsets*.

Lead time offsets are made possible by the fact that production of a component p_k of part p_i may be delayed until p_k is actually required for the first time in operation o_{ij} ($j \in \{1, 2, \ldots, s_i\}$) of the routing of p_i. The lead time offset of p_k is the difference between the start times of operations o_{ij} and o_{i1}. For example, the lead time offset of component C in Fig. 1 is 5 units.

Fig. 2 combines the BOM and the routing information of Fig. 1 to obtain the precedence network of operations corresponding to product A. This network shows that operation A.20 can be initiated only after operations A.10 and C.10 are completed. Similarly, operation C.10 can be initiated only after operations D.10 and F.20 are completed. Note that if lead time offsets were not used, the network would change. The difference is indicated by the dotted arrow. This is due to the fact that part C is an immediate component of part A, and the entire kit-list of A must be available before its production can start.

To employ the operation network formulation in the practical case in which multiple final products are man-



Part	Operation	Components Required	Processing Time	Workcenter
A	A.10	B	5	WC #1
	A.20	B, C	2	WC #2
C	C.10	D, E, F	1	WC #1
D	D.10	G	3	WC #1
F	F.10	H	2	WC #2
	F.20	H	3	WC #1

Fig. 1. Bill-of-materials (BOM) of the final product A and routings of its make items. (The processing time includes both the setup and run times for the entire batch.)

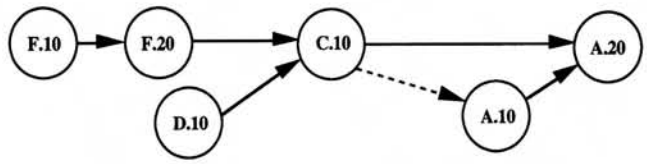


Fig. 2. Operation network for BOM shown in Fig. 1.

ufactured in the system, a single multilevel product structure is generated to combine the corresponding BOMs. Consider the set $\mathcal{P}_f$ containing n_f final assemblies. Let these products be ordered in non-increasing order of their due dates such that $D_i \geq D_j$, for $i < j$. Given the BOMs and routings of the finished products, a dummy assembly p_d is created with a due date $D_d = D_1$. Its BOM structure and routing information are shown in Fig. 3. The precedence network for the manufacture of all n_f products is then derived from the BOM and routing of product p_d.

2.2. *Scheduling model*

Scheduling in the above environment has been formulated as a mixed-integer linear programming (MILP) problem. Let there be a total of n operations to be processed $\{J_1, J_2, \ldots, J_n\}$. Let us define the set of indices of operations to be processed by work-center $\mathcal{K}$ as: $I_{\mathcal{K}} = \{i : J_i \text{ requires work-center } \mathcal{K}\}$. Let the processing time of operation J_i be denoted by t_i, and the start and completion times of J_i by S_i and C_i respectively.

A schedule may be defined by selecting a sequence of operations and assigning them to the required resources. Since this sequence defines the schedule, it also dictates the makespan. Let

$$\delta_{ij} = \begin{cases} 1 & \text{if } J_j \text{ precedes } J_i \quad (i,j \in I_{\mathcal{K}}, i \neq j;\ \mathcal{K} = 1,2,\ldots,m), \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\Delta_{ik} = \begin{cases} 1 & \text{if } J_i \text{ is performed on the } k\text{th functionally} \\ & \text{identical machine of work-center } \mathcal{W}_{\mathcal{K}}, \\ 0 & \text{otherwise} \quad (\forall\, i \in I_{\mathcal{K}},\ k = 1,2,\ldots,f_{\mathcal{K}}). \end{cases}$$

For each operation J_i, $d^{(i)}$ is defined as the immediate downstream operation or successor. For example, in the network of operations shown in Fig. 2, if A.10 is the ith operation, $d^{(i)}$ is operation A.20. Let $\mathcal{S}$ represent a schedule. The scheduling problem can be formally stated as follows (P1):

$$\text{Maximize} : \text{cost}(\mathcal{S}) = \min\{S_i:\ i = 1, 2, \ldots, n\} \quad (2.1)$$

subject to:

$$S_{d(i)} \geq C_i \qquad i = 1,2,\ldots,n; \quad (2.2)$$

$$C_i = S_i + t_i \qquad i = 1,2,\ldots,n; \quad (2.3)$$

$$C_e \leq D_e \qquad e = 1,2,\ldots,n_f,$$
$$D_e = \text{due date of } e\text{th final assembly};$$
$$C_e = \text{completion time of the last of } e\text{th final assembly}; \quad (2.4)$$

$$\delta_{ij} + \delta_{ji} = 1 \qquad i,\ j \in I_{\mathcal{K}},\ i \neq j;\ \mathcal{K} = 1,2,\ldots,m; \quad (2.5)$$

$$S_i - C_j \geq M\left(\delta_{ij} + \Delta_{ik} + \Delta_{jk} - 3\right)$$
$$i,\ j \in I_{\mathcal{K}},\ i \neq j;\ \mathcal{K} = 1,2,\ldots,m;\ k = 1,\ldots,f_{\mathcal{K}}; \quad (2.6)$$

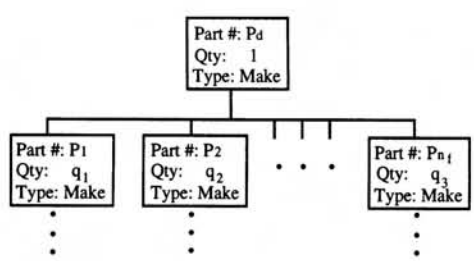
$$\sum_{s=1}^{f_k} \Delta_{is} = 1 \qquad i \in I_{\mathcal{K}};\ \mathcal{K} = 1,2,\ldots,m; \quad (2.7)$$

$$S_i \geq 0 \qquad i = 1,2,\ldots,n; \quad (2.8)$$

$$\delta_{ij} \in \{0,1\} \qquad i,\ j \in I_{\mathcal{K}};\ i \neq j;\ \mathcal{K} = 1,2,\ldots,m; \quad (2.9)$$

$$\Delta_{ik} \in \{0,1\} \qquad i \in I_k; \mathcal{K} = 1,2,\ldots,m; k = 1,2,\ldots,f_{\mathcal{K}}. \quad (2.10)$$

The objective function (2.1) indicates that the starting time of the first production operation is to be delayed as much as possible. Constraint (2.2) states that the starting time of a downstream operation is always later than the completion time of the upstream operation (respecting the 'back-to-back' scheduling rule). Constraint (2.3) provides the relationship between the starting and completion times for each operation. Constraint (2.4) states that the due dates for all final products should be respected. Note that there is at least one final product for which (2.4) holds with strict equality (see also Agrawal *et al.*, 1994). Constraint (2.5) signifies the unidirectional nature of the precedence relationship



Routing Information for Dummy Assembly P_d

Operation	Components Required	Processing Time
1	P_1	$D_1 - D_d = 0$
2	P_2	$D_2 - D_1$
•		
•		
N_f	R_{hf}	$D_{n_f} - D_{n_{f-1}}$

Fig. 3. Integration of multiple products into a single product structure.

between two operations in the scheduling sequence performed on the same work-center. Constraint (2.6) provides the relationship between the starting and completion times of any two operations on a work-center W_K containing f_K functionally identical machines. M is assumed to be a large positive number (larger than the expected value of the schedule, S), and therefore constraint (2.6) is active only when all three terms inside the parenthesis on the right-hand side are equal to unity. Constraint (2.7) indicates that if an operation J_i is performed on work-center W_K, then it must be performed on only one of its f_K identical machines. Constraint (2.8) ensures that the starting times of all operations are non-negative. Finally, constraints (2.9) and (2.10) denote that δ_{ij} and Δ_{ik} are binary variables.

Problem (P1) is a complex scheduling problem, and it is NP-hard (see also Agrawal *et al.*, 1994). The existence of a polynomial time algorithm to solve this problem is highly unlikely.

3. An effective heuristic

Explicit enumeration methods, such as the branch-and-bound algorithm presented in Agrawal *et al.* (1994) as well as mixed-integer linear programming methods, can be applied to solve problem (P1) only for small dimensional cases. To address practical problems, an effective heuristic has been developed; it exploits the critical path of the operation network presented in Section 2.1 to generate a near-optimal schedule, on the average.

3.1. *Critical path concept*

Given the BOM structure of a final assembly, a continuous sequence of operations that starts from the final operation of the finished item and terminates at a purchased item is defined as a BOM path. Among the set of BOM paths, the critical path is defined as the one along which the sum of the processing times of all operations is maximal. The latter would be the time needed to produce the final product if infinite resource capacity were assumed and lead time offsets were not used. This is, typically, the product lead time considered by MRP II. However, owing to limited resource availability, the actual product cycle time is, in general, larger than the MRP II lead time, which is computed from the routings of the make parts and the batch size.

The (infinite-capacity) critical path is fixed for a given product structure and determines the lower bound of the makespan. For example, the critical path of the product shown in Fig. 1 comprises parts F-C-A or operations F.10-F.20-C.10-A.10-A.20. The corresponding makespan is 13 units of time. Note that if lead time offsets are introduced, the infinite-capacity critical path cannot be directly calculated from the BOM, and the network of operations of Fig. 2 should be used. Referring to Fig. 2, the critical path comprises operations F.10-F.20-C.10-A.20 and the corresponding makespan is 8 units. This, however, is the lower bound of the makespan (see also Section 3.2).

The proposed heuristic uses the network of operations to generate the production schedule. Initially the first and second operations on the infinite-capacity critical path are scheduled, because they also belong to the capacitated critical path. However, scheduling subsequent operations changes, in general, the BOM path with the longest processing time. The new such path may be thought of as the *current* critical path, and can be easily computed given the BOM and the production routings of the make items. Because this path determines the lower bound of the makespan of the remaining operations, the proposed strategy schedules at each step the first operation of the path. By doing so, the procedure attempts to minimize the deviation of the schedule from the lower bound of the makespan.

As discussed in Section 1, scheduling strategies that use the critical-path concept have been employed mostly in project scheduling and include MINSLK (minimum slack) and RSM (resource scheduling method). The major difference between these methods and the proposed heuristic is that the former do not update the activity network after scheduling each activity, which leads to considerably inferior performance. Comparisons of the above methods with the proposed heuristic as well as other scheduling heuristics are presented in Section 4.

3.2. *Lead time evaluation and scheduling algorithm (LETSA)*

The proposed algorithm addresses problem (P1), and generates a feasible schedule with a near-optimal makespan as indicated by extensive numerical studies (see Section 4). It proceeds in a backward scheduling manner similar to MRP II, in which the last operation is scheduled first and the remaining operations are scheduled subsequently while respecting all precedence constraints.

The inputs to the algorithm, i.e. the delivery schedule, product structures and routing data, are used to construct an integrated operation network. Given this network, a set F is defined to include those operations that do not have a succeeding operation, i.e. $F = \{J_i : d^{(i)} = \emptyset, i = 1, 2, \ldots, n\}$. Generating the schedule consists of four steps: (i) select an operation from the set of feasible operations F, (ii) select a machine from the required work-center, (iii) schedule the selected operation, and (iv) update the operation network and the set of feasible operations.

In step (ii), the operation to be scheduled is selected from F as follows: the processing times that correspond to each path of the existing network are computed by summing the processing times of all operations along

this path. The critical path is determined, and its first operation, which also belongs to F, is selected for scheduling. The starting and completion times of the selected operation are determined from constraints (2.2), (2.3), (2.4) and (2.6) of problem (P1). The completion time C_i is determined from: (i) the starting time of its successor operation $d^{(i)}$ (see (2.2)), or the due date if $d^{(i)} = \emptyset$ in the initial network (see (2.4)); (ii) the first available time of the machine of the corresponding work-center (see (2.6)). The operation is then scheduled on that machine. Subsequently, the operation network is modified by deleting the operation just scheduled. Its predecessors are included in the feasible set of operations F, and the process is repeated until F is empty, i.e. the schedule is complete. The step-by-step algorithm is listed below and its flowchart is presented in Fig. 4.

The algorithm

1. Input manufacturing system data including: work-centers, their capacities and number of identical machines; product data, i.e., part master records, BOMs, routings, and the delivery schedule.
2. Generate the operation network from the BOM and the routing data of all make items $\mathcal{P}_m$. If lead time offsets are to be used, some additional information about the components assembled at each operation is required.
3. Define set of feasible operations as $F = \{J_i : d^{(i)} = \emptyset, i = 1, 2, \ldots, n\}$.
4. While the feasible list of operations is non-empty (i.e. $F \neq \emptyset$),
   - 4.1 For each operation in the feasible list formulate all possible network paths.
   - 4.2 Compute the length (cumulative processing time) of each path determined in step 4.1.
   - 4.3 Determine the critical (largest cumulative processing time) path and select the operation J_c of the critical path that also belongs to the feasible list F.
   - 4.4 Set its tentative completion time C_c equal to: (i) the starting time of operation $d^{(c)}$ from the partial schedule (constraint 2.2 of (P1)), or (ii) the due-date D_e if operation c is the last operation of the final assembly P_e (constraint 2.4 of (P1)).
   - 4.5 For each identical machine included in the required work-center,
     - 4.5.1 Identify the latest available starting time for operation J_c (to verify constraint (2.6) of (P1)).
     - 4.5.2 If latest available starting time $S_c = C_c - t_c$, such that the machine is available during (S_c, C_c); S_c, C_c are ideal starting and completion times, respectively. Else select $\max\{S_c\}$ as the latest available starting time such that $S_c < (C_c - t_c)$ and the machine is available during $(S_c, S_c + t_c)$; revise $C_c = S_c + t_c$.
   - 4.6 Schedule operation J_c at the latest available starting time S_c on the corresponding machine.
   - 4.7 Delete operation J_c from the operation network.
   - 4.8 Add all operations J_i, such that $d^{(i)} = J_c$, to the feasible list.
5. Compute lead times and actual lead time offsets for all make items from the resulting schedule.
6. Compute work-in-process costs from the resulting schedule.
7. Compute cost of production combining material, labor and work-in-process costs.

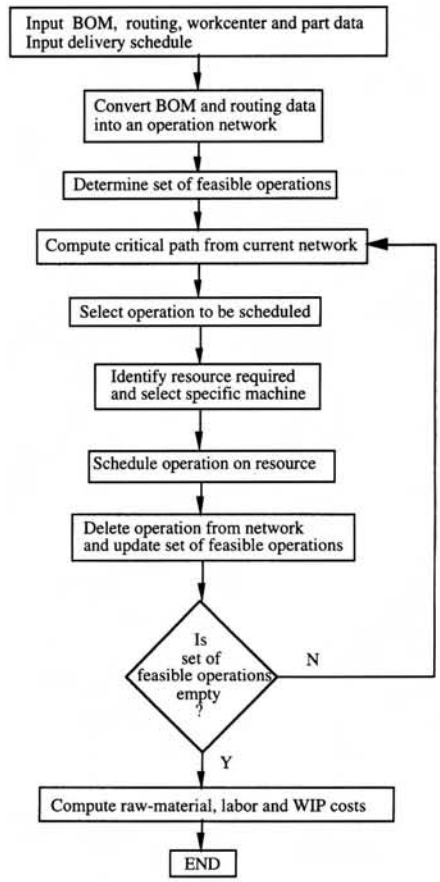


Fig. 4. LETSA flow chart.

The following example illustrates the algorithm.

Example

Consider the product shown in Fig. 1, and the corresponding network of operations of Fig. 2 which incorporates the appropriate lead-time offset. Let one unit of A be due at 14 time units. Assuming that the shop includes only one machine per work-center, we present below and in Table 1 the iterations of LETSA for determining the production schedule.

At the beginning of the algorithm (step 3), the feasible list of operations is F = {A.20}.

Iteration 1: Because there is only one operation in the feasible list A.20 (column 4 of Table 1), step 4.4 (ii) of the algorithm schedules the tentative completion time of operation A.20 to match the due date 14 (column 5 of Table 1). This completion time is tentative because constraint (2.6) of (P1) has not yet been verified. At step 4.5 the latest starting time is determined as 12 and the completion time is finalized as 14 (column 6 of Table 1). At step 4.6 operation A.20 is scheduled on work-center 2 (column 7 of Table 1; see also Fig. 5). Finally, at steps 4.7 and 4.8 the list of feasible operations is determined as F = {C.10, A.10} (column 8 of Table 1).

Iteration 2: At steps 4.1 and 4.2, the alternative network paths corresponding to the operations in F, and their lengths are (columns 2 and 3 of Table 1, respectively):

- Path 1 for operation C.10, (F.10-F.20-C.10) = 2 + 3 + 1 = 6,
- Path 2 for operation C.10, (D.10-C.10) = 3 + 1 = 4,
- Path 3 for operation A.10, (A.10) = 5.

At step 4.3 the critical path is determined as path 1, and operation C.10 is selected for scheduling (column 4 of Table 1), because it belongs to the critical path and the list of feasible operations. Step 4.4 (i) sets the tentative completion time of operation C.10 equal to 12 (column 5 of Table 1), which is the starting time of its successor operation A.20. At step 4.5 the latest starting time is determined as 11 and the completion time is finalized as 12 (column 6 of Table 1). At step 4.6 operation C.10 is scheduled on work-center 1 (column 7 of Table 1; see also Fig. 5). Finally, at steps 4.7 and 4.8 the list of feasible operations is determined as F = {A.10, F.20, D.10} (column 8 of Table 1).

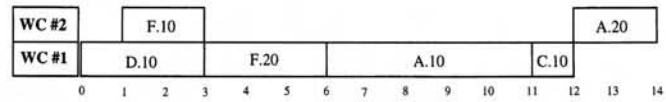


Fig. 5. Gantt chart of LETSA (and optimal) schedule for the example given in Fig. 1.

Iteration 3: At steps 4.1 and 4.2, the alternative network paths corresponding to the operations in F, and their lengths are (columns 2 and 3 of Table 1, respectively):

- Path 1 for operation A.10, (A.10) = 5,
- Path 2 for operation F.20, (F.10-F.20) = 2 + 3 = 5,
- Path 2 for operation D.10, (D.10) = 3.

At step 4.3 the critical path is determined as path 1, and we select the operation A.10 (column 4 of Table 1), because it belongs to the critical path and the list of feasible operations (the tie is broken randomly). Step 4.4 (i) sets the tentative completion time of operation A.10 equal to 12 (column 5 of Table 1), which is the starting time of its successor operation A.20. At step 4.5 the latest starting time is determined as 6 and the completion time is finalized as 6 + 5 = 11 (column 6 of Table 1). Now, constraint (2.6) of (P1) has been verified. At step 4.6 operation A.10 is scheduled on work-center 1 (column 7 of Table 1; see also Fig. 5). Finally, at steps 4.7 and 4.8 the list of feasible operations is determined as F = {F.20, D.10} (column 8 of Table 1).

Table 1. Steps of LETSA for scheduling the product shown in Fig. 1

	LETSA steps						
	4.1	*4.2*	*4.3*	*4.4*	*4.5*	*4.6*	*4.7 and 4.8*
Iteration number	*Network paths*	*Length of path*	*Operation selected*	*Tentative completion*	(S_c, C_c)	*Work-center selected*	*Feasible list, F*
1	F.10-F.20-C.10-A.20	8					
	D.10-C.10-A.20	6	A.20	14	(12, 14)	2	{C.10, A.10}
	A.10-A.20	7					
2	F.10-F.20-C.10	6					
	D.10-C.10	4	C.10	12	(11, 12)	1	{A.10,F.20,D.10}
	A.10	5					
3	A.10	5					
	F.10-F.20	5	A.10	12	(6, 11)	1	{F.20, D.10}
	D.10	3					
4	F.10-F.20	5					
	D.10	3	F.20	11	(3, 6)	1	{D.10, F.10}
5	D.10	3					
	F.10	2	D.10	11	(0, 3)	1	{F.10}
6	F.10	2	F.10	3	(1, 3)	2	{}

This process is repeated till the end of the sixth iteration, when the list of feasible operations becomes empty and the schedule is complete (see Table 1 for detailed steps). The resulting schedule corresponds to the minimal makespan of 14 time units; the complete Gantt Chart is presented in Fig. 5.

Given the schedule of operations, the cost of production as well as work-in-process (WIP) costs are computed. For the WIP costs, a rate of interest per annum and a certain compounding period are assumed. Fig. 6 illustrates the method used to compute the cost of a part processed through two operations with idle time between them (see also Appendix B). Note that the accumulation of labor cost during an operation is assumed to be linear, and interest during the operation is compounded assuming that the principal consists of the mean value of the part and labor costs. Thus for operation 10 shown in Fig. 6 the total cost comprises two components, the labor cost (linear component), and the interest (nonlinear component). During the idle time between operations 10 and 20, the value of the part in incremented by its WIP cost. The latter is computed assuming that the principal is the value at the end of operation 10. This procedure is repeated for all operations to obtain the total WIP cost.

Note that, in general, there is more than one schedule that corresponds to the makespan obtained by LETSA. This is due to the infinite number of ways of inserting idle time for non-critical path tasks. Maintaining the same sequence of tasks determined by LETSA, step 4.5 of the algorithm selects between the possible alternatives, the schedule that yields the minimal WIP cost.

4. Performance: numerical results and discussion

The performance of LETSA was evaluated considering two cases of examples. The first example case included small BOMs with corresponding networks of 12–25 operations. Although small BOMs are not of direct interest in this study, it is only for these examples that the optimal solution could be obtained. For this purpose we developed a branch-and-bound (B&B) algorithm (see Agrawal *et al.*, 1994). This B&B implementation employs simple heuristic rules (such as those included in Section 4.1) to obtain a 'good' starting schedule and depth-first search to improve this schedule until all feasible solutions have been implicitly explored. The critical path is used as a lower-bound of the time-to-complete during the search process.

The second example case included large BOMs corresponding to large assemblies of three different types that are of direct interest. Typical parameters of the BOMs corresponding to these assembly types are presented in Table 2. Each example in either case consisted of determining the production schedule of an assembly given its BOM, the routings of its make items and the relevant characteristics of the production system. The data for all examples were generated randomly. Routings for the make parts were made up of one to four operations. For each BOM structure, there were an average of four work-centers in the facility, each with one or two identical machines.

The following four performance measures were employed in the numerical study:

1. Mean percentage variation from optimum (P_1), where

$$\text{Percentage variation} = v = \frac{(\text{Makespan})_{\text{heuristic}} - (\text{Makespan})_{\text{optimum}}}{(\text{Makespan})_{\text{optimum}}} \times 100.$$

$P_1 = E\{v\}$ was evaluated considering all examples within an example type.

2. P_2 is defined such that the values of v corresponding to 90% of the cases lie in the interval $[0,P_2]$.

3. The percentage of cases heuristic finds the optimal solution (P_3).

4. The percentage of cases heuristic yields solution within 5% of the optimal solution (P_4).

Note that because the optimal solution is not computed for the examples of the second case, the heuristic that yielded the lowest makespan was used to replace 'optimum' in the performance measures; thus the measures used in the second example case will be referred to as P'_1 through P'_4, respectively.

Similar measures were used to assess the performance of the solutions obtained in terms of WIP costs. It is important to emphasize that the reference solution for the WIP measures, P^w1 through P^w4, is not the optimal WIP cost, but the minimum of the WIP costs obtained from all methods.

4.1. *Other heuristics*

A number of heuristic rules, commonly employed in the literature, were used to assess the performance of LETSA. Each rule replaced the critical-path procedure in the backward scheduling algorithm; i.e. it was used to select and schedule one operation between the operations of the feasible list F, as in Section 3.2; ties are resolved randomly in each case. The following rules were employed:

Table 2. BOM parameters for large numerical examples

Parameter	*BOM type*		
	Long	*Wide*	*Long and wide*
Levels	14	6	9
Branches	110	90	200
Make parts	100	110	130
Operations	130	150	300

- SPT – Shortest Processing Time.
  Selects from F the operation with the shortest processing time.
- SIT – Shortest Idle Time Rule.
  Selects from F the operation which introduces the minimum idle time on the resources.
- MPO – Maximal Predecessor Operations Rule.
  Selects from F the operation with the maximal number of predecessor operations.
- RANDOM – Random Rule.
  Selects an operation at random.
- LPT – Longest processing time.
  Selects from F the operation with the longest processing time.
- EDD – Earliest due date.
  Selects from F the operation with the most imminent due-date; the due dates are generated from an uncapacitated backward schedule based on the operation network.
- FCFS – First-come-first-served.
  Selects from F the operation that was released for processing first.
- FOPNR – Fewest operations remaining.
  Selects from F the operation belonging to the routing with the minimum number of operations remaining to be performed.
- MOPNR – Maximum operations remaining.
  Selects from F the operation belonging to the routing with the maximum number of operations remaining to be performed.
- LRM – Longest remaining processing time.
  Selects from F the operation belonging to the routing with maximum cumulative processing time of the remaining operations, excluding the operation under consideration.
- LWKR – Least work remaining.
  Selects from F the operation belonging to the routing with minimum cumulative processing time of the remaining operations, excluding the operation under consideration.
- MWKR – Maximum work remaining.
  Selects from F the operation belonging to the routing with maximum cumulative processing time of the remaining operations.
- TWORK – Total work content in routing.
  Selects from F the operation belonging to the routing with largest total work content.
- NINQ – Next operation goes to work-center with the shortest queue.
  Selects from F the operation that if scheduled, would cause the next operation in the routing to be assigned to the work-center with the shortest queue.
- WINQ – Next operation goes to work-center with the least work content in queue.
  Selects from F the operation that if scheduled, would cause the next operation in the routing to be assigned to the work-center with the least work content in its queue.

In addition to the above heuristics, a commercially available shop-floor simulation tool (SIMFACTORY) was employed to evaluate a 'push' type production strategy. The inputs to this tool include part master records, BOM structure, routings for all make items as well as the parameters of the production environment. The simulation proceeds by attempting to schedule the operations as soon as the corresponding resources are available, in effect 'pushing' the parts through the system. The FCFS rule was chosen for selecting operations to be scheduled on the facility.

4.2. *Results and discussions*

Example Case I: A study of 217 small BOMs, each yielding a 12–25 operation network, was conducted. The optimal solution for each case was obtained by using the branch-and-bound algorithm (Agrawal *et al.*, 1994). The results of the study are presented in Table 3 and are prioritized with respect to the performance factor P_1. The values in the last row of Table 3, 'Best heuristic', have been evaluated with, for each example, the shortest makespan determined by any heuristic. Thus the values of the measures in each row quantify the performance of the system that applies all scheduling heuristics to each example and selects the best result.

The first two rows of Table 3 indicate that the first-come first-served based heuristics have the best performance in the case of small operation networks. This is the case for both the 'pull' and 'push' FCFS implemen-

Table 3. Makespan performance of scheduling heuristics for Example Set I

Heuristic	P_1	P_2	P_3	P_4
FCFS	6.34	16.35	37.50	57.87
SIMFACTORY	6.73	19.35	39.00	57.50
LETSA	8.29	21.40	28.11	45.62
MOPNR	11.39	27.47	17.05	35.02
MWKR	11.80	29.23	23.04	37.79
SIT	11.88	27.89	18.43	35.02
LPT	12.18	29.04	21.66	35.48
LRM	12.34	27.96	19.82	37.33
EDD	12.93	30.58	21.20	35.48
RANDOM	13.14	29.77	16.13	30.88
TWORK	14.17	31.67	20.28	32.26
MPO	15.15	32.76	17.05	27.19
SPT	16.35	37.69	14.75	27.65
LWKR	17.10	36.84	13.82	25.35
FOPNR	17.49	37.18	13.36	24.88
NINQ	23.03	44.74	4.15	12.44
WINQ	23.03	44.74	4.15	12.44
Best heuristic	1.50	5.66	70.97	88.02

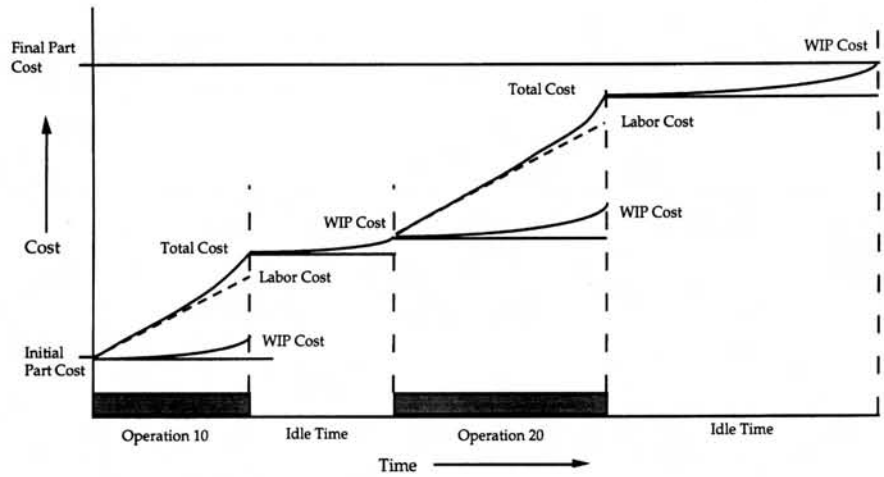


Fig. 6. Costing method.

Table 4. WIP cost performance of 'pull' and 'push' scheduling strategies for Example Set I

Heuristic	P^w1	P^w2	P^w3	P^w4
LETSA	12.54	24.58	14.35	37.04
SIMFACTORY	28.12	48.14	0.99	4.46

tations. The fact that both implementations yield similar results is expected, because the 'push' and 'pull' scheduling problems for minimum makespan are equivalent for the case of a single final product (see Agrawal *et al.*, 1994). Both methods perform slightly better than LETSA. This is attributed to the structure and size of the operations network. In this case, the network consists of very few branches with comparable lead times and, thus, the infinite-capacity critical path identified by LETSA may not always coincide with the finite capacity one. This is because LETSA considers only the product structure and does not account for the changes in the facility due to scheduling the selected operation.

Table 4 illustrates the difference in WIP costs between the just-in-time 'pull' production scheduled by LETSA and the 'push' production scheduled by SIMFACTORY. It is noted that the WIP costs corresponding to all 'pull' schedules were comparable. In contrast with the makespan results of LETSA and SIMFACTORY, which are comparable, Table 4 shows that the mean WIP cost for the 'pull' strategy (LETSA) is substantially less than the corresponding value for the 'push' strategy. The other performance measures also support this observation.

Table 5. Performance of scheduling heuristics for Example Set II (long BOMs)

Heuristic	P'_1	P'_2	P'_3	P'_4
LETSA	1.87	5.90	36.80	88.00
FCFS	4.83	14.76	42.80	65.20
MOPNR	10.26	23.36	2.80	38.00
SIT	11.77	26.02	4.00	30.80
RANDOM	12.53	26.57	2.80	28.00
EDD	12.67	27.28	1.20	26.00
MPO	14.34	30.66	2.80	24.00
MWKR	15.30	32.19	0.80	21.20
SPT	15.37	32.80	0.00	16.80
TWORK	16.11	32.21	1.20	18.80
LWKR	16.29	28.58	0.40	16.00
LPT	17.62	35.00	0.40	17.60
FOPNR	17.74	32.79	0.80	13.20
LRM	18.70	36.93	0.40	11.60
NINQ	26.77	51.22	0.80	6.80
WINQ	26.77	50.00	0.00	6.80
Best others	1.39	5.32	60.40	89.60

Example Case II: This performance evaluation test examined 250 examples of long BOMs, 250 examples of wide BOMs, and 500 examples of long and wide BOMs.

The results are presented in Tables 5, 6, and 7, and are prioritized with respect to the performance factor P'_1. The last row of each table, 'Best other', has been determined using, for each example, the shortest makespan determined by any heuristic excluding LETSA.

Tables 5, 6, and 7 indicate that LETSA performs significantly better as the number of levels of the BOM structure increases. Table 6 indicates that LETSA performs well even in the case of wide product structures for which there is a greater possibility of branches having comparable processing times. Table 7 shows that for the long and wide BOMs, which are of direct interest in this study, LETSA performs better than all other heuristics combined. It is also of interest to note that the mean WIP cost (P'_1) for LETSA was found to be the lowest for the long and wide BOMs of example set II.

Table 6. Performance of scheduling heuristics for Example Set II (wide BOMs)

Heuristic	P'_1	P'_2	P'_3	P'_4
LETSA	1.40	3.82	26.80	93.20
FCFS	1.65	5.78	64.80	87.60
MOPNR	5.57	12.23	4.00	58.00
RANDOM	6.24	14.90	2.00	53.60
MWKR	6.88	15.15	1.60	54.00
EDD	8.21	16.10	0.80	37.20
SIT	8.74	17.02	8.40	33.60
MPO	8.83	18.11	0.00	33.60
LRM	8.89	18.87	0.80	33.60
TWORK	9.02	21.71	0.80	42.40
LPT	11.16	22.74	0.40	26.40
SPT	13.16	23.99	0.00	14.00
WINQ	13.95	29.45	6.80	23.20
NINQ	14.06	30.15	6.80	24.40
FOPNR	14.34	26.00	0.00	11.60
LWKR	15.26	25.30	0.40	7.60
Best others	0.43	1.56	76.40	99.20

These tables also indicate that the FCFS heuristic is the next best alternative to LETSA; in fact, it performs almost as well as LETSA in the case of wide BOMs (Table 6). Surprisingly the RANDOM heuristic does not perform that poorly; it yields performance comparable to MOPNR and MWKR. The earliest due date (EDD) heuristic and the shortest idle time (SIT) heuristics seem to decay in performance as the BOMs become wider and longer. Finally, the work-center queue-based heuristics

Table 7. Performance of scheduling heuristics for Example Set II (long and wide BOMs)

Heuristic	P'_1	P'_2	P'_3	P'_4
LETSA	0.40	1.10	51.10	99.20
FCFS	2.12	5.83	42.08	87.78
RANDOM	5.69	12.14	2.00	58.52
MOPNR	5.81	11.15	0.40	52.30
MWKR	6.72	15.11	0.40	52.00
MPO	7.56	15.43	0.00	40.88
TWORK	7.85	17.42	0.00	42.08
EDD	9.98	17.30	0.00	18.24
LRM	10.16	20.05	0.20	25.60
SIT	11.11	20.02	0.80	17.03
LPT	11.31	22.69	0.00	22.85
FOPNR	13.93	23.80	0.00	6.61
SPT	15.06	23.94	0.00	4.01
LWKR	16.36	26.41	0.00	2.00
WINQ	26.64	45.45	0.00	2.81
NINQ	26.71	43.91	0.00	2.40
Best others	0.99	2.95	45.80	96.40

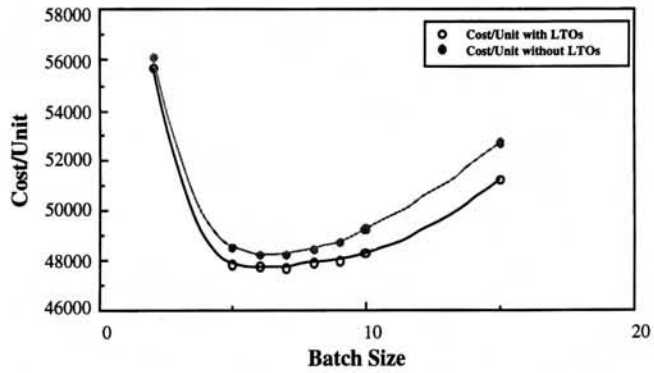


Fig. 7. Application of LETSA to batch sizing.

NINQ and WINQ are among the poorest for ('pull') scheduling of multilevel products in job-shops; this result is important for practitioners.

The computation time required for all heuristics is comparable. All methods are capable of yielding real-time solutions to the large problems used in this case, because the CPU time for a typical large example was about 30 seconds of CPU time on a SUN Sparcstation IPX.

Our numerical study has indicated that LETSA performs well over a large range of sample problems. This is consistent with the literature in which critical path scheduling similar to LETSA has been applied to a number of multiprocessor (single work-center with multiple machines of identical or different speeds) problems (Coffman, 1976). The level strategy of Hu (1961) uses a tree structure to represent the precedence relationships between tasks. The algorithm starts from the highest level in the tree and assigns a task to a processor that becomes available; the process repeats until the assembly is scheduled. This method is known to be optimal when all tasks have equal processing times. The level-strategy algorithm is very similar to a critical-path algorithm, and can be considered as a 'push-scheduling' version of LETSA.

Our analysis, however, has revealed cases for which LETSA can perform as poorly as possible. In Appendix A we provide two examples that show that the schedule determined by LETSA can be as poor as the worst non-idle time schedule; that is, a backward schedule in which a machine is not idle as long as there is an operation to be processed by it. The results indicate, for a shop defined by a total of m machines (of m or less work-centers), the makespan determined by LETSA can be m times longer than the optimal makespan. Because this ratio is the worst possible for a non-idle time schedule, LETSA can perform as poorly as the worst non-idle time algorithm. Practitioners must keep in mind that worst-case problem instances are contrived and are often not reflective of real-life product data.

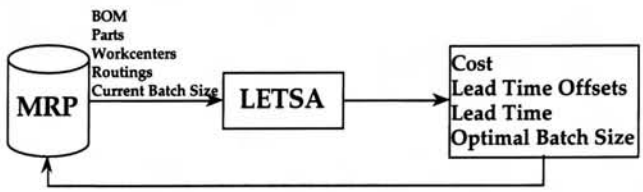


Fig. 8. Application of LETSA to improve and update MRP II data.

5. Practical applications of LETSA

Although LETSA has been developed to schedule operations within a manufacturing environment that produces large assemblies, it may be also employed to support production planning decisions. Below we have discussed some interesting LETSA applications.

1. Scheduling production operations: The schedule of operations generated by LETSA may be directly implemented on the shop floor. Owing to the low computational burden, the algorithm can be used on a rolling horizon basis to account for changes in the product mix, as well as changes in the environment concerning resource and labor availability. As new orders arrive in the system, the previous operation network(s) of the previous scheduling period is (are) updated by deleting from them the operations already completed, and appending the networks of incoming products. Thus the algorithm can be applied to a new network at the beginning of every scheduling period. A similar updating process can be employed when a resource fails or labor levels vary. Note that the software implementation of the algorithm can accommodate time-dependent resource availabilities.

2. Production planning support: Our industrial partners have found LETSA useful in determining critical production attributes for newly launched product designs, such as batch size, lead times and lead-time offsets of make items. The evaluation procedure requires the product's manufacturing BOM and product routings of all make parts. In addition, our partners estimate the fraction of the capacity of resources that is available for the manufacture of the product. This estimate may be resource-specific or a global fraction employed for the entire shop, and it depends on average resource workload history. Given this information, LETSA is run iteratively for various batch sizes. The resulting WIP cost is evaluated from the schedule of operations as described in Section 3.2. The optimum batch size is selected to be the one that corresponds to the lowest cost per unit, which is a trade-off between WIP and set-up costs (see Appendix B).

A typical plot of the relationship between batch size and cost per unit is shown in Fig. 7. The assembly tested in this case is the chassis of a large antenna, which includes 200 make items (and the total cycle time ranged from 57 days to 522 days for a batch size of one to fifteen). The two curves in the figure correspond to the conditions during scheduling; the top curve was obtained when no lead-time offsets were used, whereas the bottom curve was obtained by using lead time offsets. Fig. 7 clearly illustrates that the introduction of lead-time offsets results in lower WIP costs.

The optimal batch-size determined, the corresponding lead times and the corresponding lead-time offsets of all make items are entered into the part master records of the MRP II system, and are used thereafter for production planning. As a result, the planning function of MRP II has access to product data that accurately reflect the current production reality, and thus shop floor delays and errors are minimized. Fig. 8 illustrates the application of LETSA in updating the production planning data within MRP II.

3. Shipping schedule validation: In the case of new customer orders or manufacture of totally new products, it is important to determine the ability of the manufacturing facility to accommodate the increased workload and still meet product delivery schedules. Accurate information about the production capacity is critical in evaluating whether the order can be accepted or some changes in the shipping schedule are required. Traditional capacity planning, which accounts only for the set-up and run times of operations, may not be accurate, because the precedence relationships directly affect idle time and thus capacity utilization. LETSA can be employed to obtain more precise delivery estimates, and thus promised due-dates, because it calculates these data based on the actual production requirements and the actual workload on the shop floor.

6. Conclusions

This paper addressed an important practical problem: the scheduling of operations in a manufacturing facility that produces large assemblies. Relevant objectives within such an environment include minimization of the makespan of the schedule, and minimization of production costs. The problem of minimizing the makespan has been formulated as an integer program and is solved optimally, for small-dimensional problems, by a branch-and-bound algorithm (Agrawal *et al.*, 1994). In this paper an effective heuristic (LETSA) has been developed to solve problems of practical size in realistic times. The heuristic uses the precedence network of operations to prioritize execution of those operations that belong to a dynamically computed critical path. Furthermore, the schedule is derived in a 'pull' manner, to limit work-in-process costs.

An extensive numerical study has indicated that LETSA provides schedules with near-optimal makespans. The quality of the solutions seems to improve substantially for large (long and wide) precedence networks, which are of particular interest in this study. In fact, in

such cases LETSA yielded results that outperformed the combined results obtained from a large set of practical scheduling rules. The numerical study also showed that first-come-first-served type heuristics also yield satisfactory results of this type of precedence networks, which is not the case for work-center queue-based scheduling rules. The production costs resulting from the schedules generated by LETSA were shown to be substantially lower than 'push'-type production strategies. In practice, the proposed method has shown to be useful in determining the batch size and realistic lead time and lead-time offsets of new products.

7. Acknowledgements

This work was supported in part by Westinghouse ESG and the Maryland Industrial Partnerships under grant no. 05-4-30816 and by the Institute for Systems Research of the University of Maryland under grant no. NSFD CD 8803012. The assistance of Mr Ron Palmer and Mr Dave Delaney of Westinghouse ESG was invaluable to the completion of this project.

References

Adams, J., Balas, E. and Zawack, D. (1988) The shifting bottleneck procedure for job shop scheduling. *Management Science*, **34**(3), 391–401.

Agrawal, A., Harhalakis, G., Minis, I. and Nagi, R. (1994) Just-In-Time production of large assemblies. Technical Report No. TR 94-51, Institute for Systems Research, University of Maryland, College Park, MD.

Ashour, S. (1967) A decomposition approach for the machine scheduling problem. *International Journal of Production Research*, **6**, 109–122.

Ashour, S. and Hiremath, S.R. (1973) A branch and bound approach to the job shop scheduling problem. *International Journal of Production Research*, **11**, 47–58.

Barker, J.R. and McMahon, G.B. (1985) Scheduling the general job shop. *Management Science*, **31**, 594–598.

Bedworth, D.D. and Bailey, J.E. (1982) *Integrated Production Control Systems*, Wiley, New York.

Billington, P.J., McClain, J.O. and Thomas, L.J. (1983) Mathematical programming approaches to capacity-constrained MRP systems: review, formulation and problem reduction. *Management Science*, **29**(10), 1126–1141.

Boctor, F.F. (1990) Some efficient multi-heuristic procedures for resource-constrained project scheduling. *European Journal of Operational Research*, **49**, 3–9.

Carlier, J. and Pinson, E. (1989) An algorithm for solving the job shop problem. *Management Science*, **35**(2), 164–176.

Coffman, E.G., Jr. (1976) *Computer and Job-Shop Scheduling Theory*, John Wiley, New York.

Elsayed, E.A. (1982) Algorithms for project scheduling with resource constraints. *International Journal of Production Research*, **20**, 95–103.

Elsayed, E.A. and Nasr, N.Z. (1986) Heuristics for resource-constrained scheduling. *International Journal of Production Research*, **24**, 299–310.

Godin, V. and Jones C. (1969) The interactive shop supervisor. *Industrial Engineering*, **1**, 16–22.

Goldratt, E. (1980) Optimized production timetable: a revolutionary program for industry, in *APICS 23rd Annual Conference Proceedings* Oct 14–17, Los Angeles, CA, 172–176.

Greenberg, H.H. (1968) A branch and bound solution to the general scheduling problem. *Operations Research*, **16**, 353–361.

Hu, T.C. (1961) Parallel sequencing and assembly line problems. *Operations Research*, **9**(6), 841–848.

Khattab, M.M. and Choobineh, F. (1991) A new approach for project scheduling with a limited resource. *International Journal of Production Research*, **29**(1), 185–198.

Lageweg, B.J., Lenstra, J.K. and Rinnooy Kan, A.H.G. (1977) Job shop scheduling by implicit enumeration. *Management Science*, **24**, 441–450.

Levulis, R.J. (1985) Finite capacity scheduling and simulation systems. Manufacturing Technology Information Analysis Center, Chicago, Illinois, Rep. MTIAC TA-85-04 (October).

MacCarthy, B.L. and Liu, J. (1993) Addressing the gap in scheduling research: a review of optimization and heuristic methods in production scheduling. *International Journal of Production Research*, **31**(1), 59–79.

Miltenburg, J. and Sinnamon, G. (1989) Scheduling mixed-model multilevel just-in-time production systems. *International Journal of Production Research*, **27**(9), 1487–1509.

Miltenburg, J. and Sinnamon, G. (1992) Algorithms for scheduling multilevel just-in-time production systems. *IIE Transactions*, **24**(2), 121–130.

Orlicky, J. (1975) *Material Requirements Planning*, McGraw Hill, New York.

Panwalkar, S.S. and Iskander, W. (1977) A survey of scheduling rules. *Operations Research*, **25**(1), 45–61.

Reiter, S. (1966) A system for managing job shop production. *J. Business*, **39**(3), 371–393.

Rodammer, F.A. and White, K.P., Jr (1988) A recent survey of production scheduling. *IEEE Transactions on Systems, Man, and Cybernetics*, **18**(6), 841–851.

Ulusoy, G. and Ozdamar, L. (1989) Heuristic performance and network/resource characteristics in resource-constrained project scheduling. *Journal of the Operational Research Society*, **40**(12), 1149–1152.

Zahorik, A., Thomas L.J. and Trigeiro, W.W. (1984) Network programming models for production scheduling in multi-stage, multi-item capacitated systems. *Management Science*, **30**(3), 308–325.

Appendix A. Worst Case Examples for LETSA

In this appendix we present two examples where LETSA performs as poorly as a non-idle time algorithm possibly can. For easier understanding, these examples are presented in Figs A1 and A2. In each figure the problem instance is shown in part (a), the LETSA schedule is shown in part (b), and the optimal schedule is shown in part (c). Operations are labeled as N/W/T, where N denotes the operation identifier, W denotes the work-center required and T denotes the processing time. The precedence relations can be read from the corresponding networks, and ε is a sufficiently small positive number.

Example 1

Consider the problem in Fig. A1, where the shop includes two work-centers ($m = 2$) with one machine each. The ratio of the makespan found by LETSA (ω_{LETSA}) to

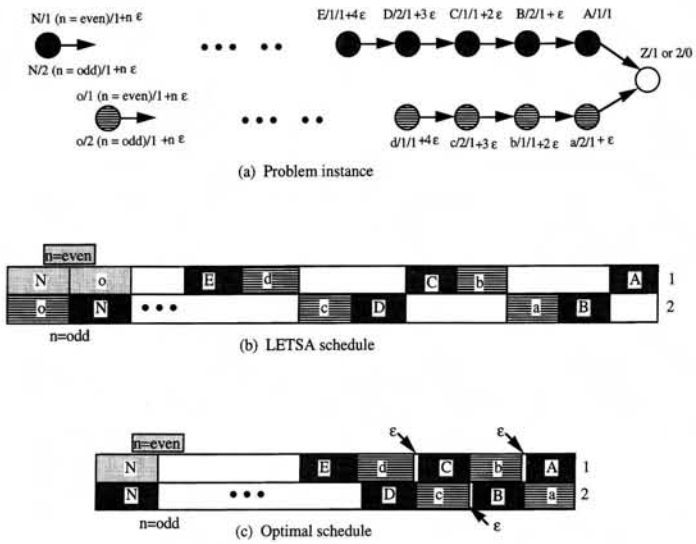


Fig. A1. Example 1.

the optimal makespan (ω_0) is

$$\frac{\omega_{\text{LETSA}}}{\omega_0} = \frac{1 + 2n + 2\varepsilon(n(n+1)/2)}{1 + n\varepsilon + n + \varepsilon(n(n+1)/2)}; \lim_{n\to\infty} \frac{\omega_{\text{LETSA}}}{\omega_0} = 2.$$

Note that the LETSA schedule of Fig. A1 (b) has been generated by 'poor' tie breaking of two operations on equal-length critical paths, whereas the optimal schedule of Fig. A1 (c) has been generated by 'smart' tie breaking. Of course, the values of ε in operation processing times can always be adjusted such that LETSA is forced to generate a similar schedule without any need for tie breaking.

Example 2

Consider the problem in Fig. A2, where the shop includes m work-centers with one machine each. The ratio of the makespan found by LETSA (ω_{LETSA}) to the optimal makespan (ω_0) is

$$\frac{\omega_{\text{LETSA}}}{\omega_0} = \frac{n\ m\ (1+\varepsilon)}{n\ (1+\varepsilon) + (m-1)(1+\varepsilon)}; \lim_{n\to\infty} \frac{\omega_{\text{LETSA}}}{\omega_0} = m.$$

Further still, for m work-centers with multiple machines, it can be shown by a similar example that the ratio of ω_{LETSA} to ω_0 can approach $\sum_{\mathcal{K}=\infty}^{m} f_{\mathcal{K}}$, where $f_{\mathcal{K}}$ is the number of functionally identical machines in work-center $\mathcal{W}_{\mathcal{K}}$. Note that all these ratios are the worst possible for a non-idle time schedule.

Appendix B. Optimal batch sizing problem

As mentioned in the last paragraph of Section 3.2, LETSA attempts to: (i) minimize the makespan of the schedule and (ii) construct the schedule that minimizes WIP costs for the resulting sequence of tasks. Within this context, the optimal batch sizing problem can be formulated in a straightforward manner.

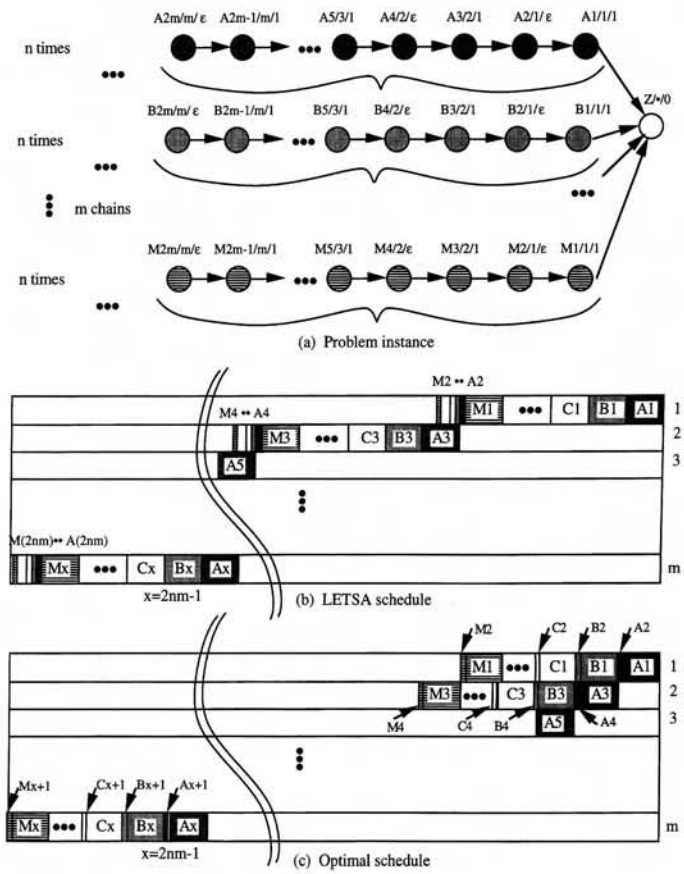


Fig. A2. Example 2.

To do this we introduce batch size (B) as a variable in the scheduling model of Section 2.2, and we employ a method to compute WIP costs for the LETSA schedule. As a result of the former, the processing time for each operation in the network is a function of the batch size (B). Let the processing time of operation J_i be denoted by $t_i(B) = s_i + B \times q_i, \times r_i$, where s_i is the set-up time for the batch at operation J_i, q_i is the number of operations J_i per assembly (found from the BOM) and r_i is the run-time per operation J_i (found from the routing).

To compute WIP costs, we introduce: (i) raw-material costs at the beginning of all starting operations, i.e. operations that do not have an immediate predecessor in the operation network; let P_i denote this cost for starting operation J_i, (ii) labor rate per unit time, u (assumed equal for all operations, for simplicity), (iii) interest rate, r (assumed to be 25% APR and compounded per hour). Let $L(x) = u[(1+r)^x - 1]/r$ be the function for compounding interest when *only* value is being added at a rate of u (equal payment series compound amount factor), and let $I(x) = (1+r)^x$ be the

function for compounding interest when *no* value is being added (single payment series compound amount factor). From the Gantt-chart of the schedule (i.e. from the operation completion times $C_i(B)$), the total costs for the (sub-)assembly till operation J_i can be computed by the following recursive expressions:

$$T_i(B) = \begin{cases} L(t_i(B)) + \sum_{j|d^{(j)}=J_i} T_j(B)\, I(C_i(B) - C_j(B)) & \text{if } \exists j \mid d^{(j)} = J_i, \\ L(t_i(B)) + P_i I(t_i(B)) & \text{otherwise.} \end{cases} \quad \text{(B1)}$$

Because the labor and material costs are constants, to minimize the WIP costs it is sufficient to determine B such that $T_e(B)$ is minimized. Therefore the batch-sizing problem is (PB):

Minimize: Cost(B) = $T_e(B)$

subject to: (B1), (2.2), (2.3), (2.4),

given δ_{ij} and Δ_{ik}; B is a positive integer.

The optimal batch size B that solves (PB) can be evaluated by a simple enumeration procedure that calculates $T_e(B)$ for different values of B and plots a graph to determine the optimal value (see Fig. 7).

Biographies

Ashutosh Agrawal is currently enrolled in the Ph.D. program in Decision Sciences and Engineering Systems at Rensselaer Polytechnic Institute and is conducting research in the area of Agile Manufacturing for Electronics. He holds an M.S. degree in Mechanical Engineering from the University of Maryland, College Park, and a B.E. degree in Mechanical Engineering from the University of Roorkee, India.

Ioannis Minis is an Associate Professor of the University of Maryland at College Park, where he holds a joint appointment with the Department of Mechanical Engineering and the Institute for Systems Research. He received his undergraduate degree from the National Technical University of Athens in 1982, the M.S. from Clarkson University, 1983, and the Ph.D. from the University of Maryland at College Park, all in Mechanical Engineering. Dr Minis is the recipient of the 1993 Earl E. Walker Outstanding Young Manufacturing Engineer Award of the Society of Manufacturing Engineers. He also received the Best Paper Award in the area of Engineering Database Management at the 1992 ASME International Conference on Computers in Engineering. His main research interests are in the areas of machine dynamics and control, and production systems.

Rakesh Nagi is an Assistant Professor of Industrial Engineering at the State University of New York, Buffalo. His B.E. (1987) degree in Mechanical Engineering is from University of Roorkee, India. He received his M.S. (1989) and Ph.D. (1991) degrees in Mechanical Engineering from the University of Maryland at College Park, he worked at the Institute for Systems Research and INRIA, France, and received the outstanding graduate student award. His research interests lie mainly in the areas of design of production system, hierarchical production planning and scheduling. His more recent interests are in agile and lean manufacturing. He is a member of the IIE.