

Scheduling manufacturing systems for delayed product differentiation in agile manufacturing

DAVID HE†* and ASTGHIK BABAYAN†

Production of customized products to respond to changing markets in a short time and at a low cost for agile manufacturing can be implemented with delayed product differentiation in a manufacturing system. The successful implementation of delayed product differentiation lies in efficient scheduling of the manufacturing system. Scheduling problems in implementing delayed product differentiation in a general flexible manufacturing system are defined, formulated and solved here. The manufacturing system consists of two stages: machining and assembly. At the machining stage, a single machine is used to produce standard component parts for assembly products. These parts are then assembled at the assembly stage by multiple identical assembly stations to form customized products. The products to be produced in the system are characterized by their assembly sequences represented by digraphs. The scheduling problem is to determine the sequence of products to be produced in the system so that the maximum completion time (makespan) is minimized for any given number of assembly stations at the assembly stage. Based on the representation of assembly sequence of the products, three production modes are defined: production of a single product with a *simple assembly sequence*; production of a single product with a *complex assembly sequence*; and production of N products. According to the three defined production modes, the associated scheduling problems are defined as G_s scheduling problems, G_c scheduling problems and N -product scheduling problems, respectively. Optimal and heuristic methods for solving the scheduling problems are developed. The computational experiment shows that the heuristics provide good solutions to the scheduling problems.

1. Introduction

Producing customized products in a short time at low cost is one of the goals of agile manufacturing. Delayed product differentiation is a design strategy used to achieve this goal in a manufacturing system. The delayed product differentiation strategy is commonly implemented in a manufacturing process that involves two stages: machining and assembly. When the delayed product differentiation strategy is implemented, common and simple parts are machined at the machining stage and delivered to the assembly stage to form the finished products. In this case, the differentiation of the products can be easily postponed to the assembly stage. This strategy allows one to standardize components and create a variety of products. The delayed product differentiation strategy also provides the following benefits for agile manufacturing: greater product customization, rapid introduction of new or modified products, easy upgradable products, dynamic reconfiguration of production

Revision received January 2002.

† Department of Mechanical and Industrial Engineering, University of Illinois-Chicago, Chicago, IL 60607, USA.

* To whom correspondence should be addressed. e-mail: davidhe@uic.edu

processes, etc. The benefits of delayed product differentiation in terms of reduced safety stock and shorter customer response time have been studied (e.g. Silver and Peterson 1985, Gupta and Krishnan 1998).

To respond to the challenge of implementing delayed product differentiation, modularity, which also enables component commonality of products, gains considerable attention. Modularity arises from the breakdown of a complex part into simple and functionally independent components which are assembled to make customized parts. Although the number of parts in the modular design is larger than that in the integral design, the total time of machining operations and manufacturing cost are more likely to decrease in the modular design. The integral designs result in complex parts that require more complicated and costly manufacturing processes. They also result in a higher inspection cost, a long processing time and are more likely to fail in use.

Since in designing the modular design designers intend to replace a unique integral part with the assembly of common components that may be manufactured in the machining stage or bought directly from suppliers, modular designs increase the number of assembly operations and the assembly time and, hence, they may require additional assembly stations in the system (He and Kusiak 1996).

Little research has been done to address the scheduling issues related to the implementation of delayed product differentiation in a manufacturing system, especially when products with complicated assembly sequences are produced. Here, the scheduling problems related to the implementation of delayed product differentiation for agile manufacturing are formulated and solved.

The structure of the manufacturing system that implements the delayed product differentiation in an agile manufacturing environment is shown in figure 1. The manufacturing process in the system consists of two stages: machining and assembly. There is *one* machine at the machining stage and there are $q > 1$ identical parallel assembly stations at the assembly stage. Examples of systems with such a shop structure can be observed in the manufacturing of automotive parts where components are machined by a multifunctional machine tool and delivered to multiple assembly stations for final assembly (Warnecke and Walther 1982). The machining

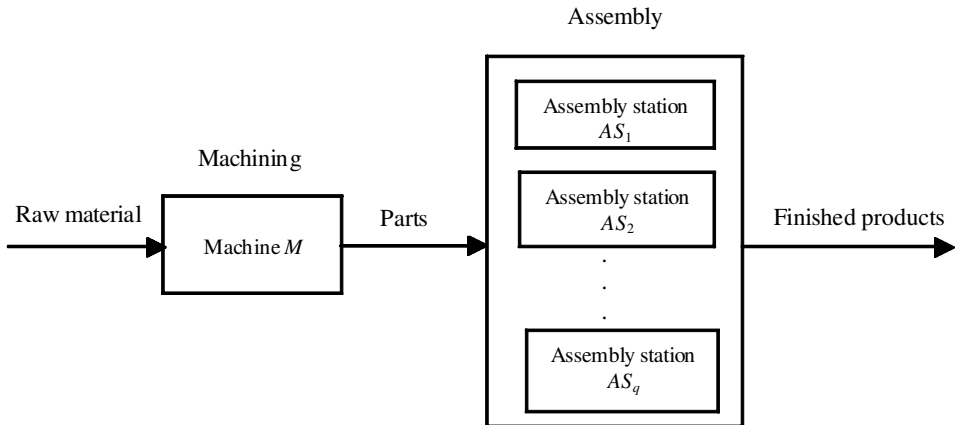


Figure 1. General structure of the manufacturing system.

operations include all activities that consist of material-removal operations. Assembly involves assembling the machined parts to form the required products.

Here, optimal solution methods for solving the scheduling problems are developed. Based on the insights gained in developing the optimal solution methods, heuristic algorithms for solving efficiently the scheduling problems are developed.

The remainder of the paper is organized as follows. In Section 2, the scheduling problems to be solved are formally defined. The literature on solving related scheduling problems is reviewed in Section 3. Section 4 focuses on the development of solution methods for solving the scheduling problems. The computational results for solving the scheduling problems with the developed solution approaches are given in Section 5. Section 6 concludes the paper.

2. Problem description

The scheduling problems for delayed product differentiation in agile manufacturing are defined here.

2.1. Scheduling problem

There is single machine at the machining stage and there are $q > 1$ numbers of assembly stations at the assembly stage. The objective of the scheduling is to assign parts and subassemblies/assemblies to the machines at the machining and assembly stages and determine the processing sequences on the machines so that the makespan (C_{\max}), i.e. the maximum completion time, is minimized.

Note that in an agile manufacturing environment, since fast delivery of products is of a major concern, C_{\max} is chosen as the scheduling objective for scheduling the manufacturing systems to minimize the manufacturing cycle times. Here, the assembly sequence of a product refers to the order in which parts and subassemblies are assembled by the assembly stations. The assembly sequence of a product to be produced in the system is represented by a digraph G and this representation follows that defined in Kusiak (1989). In a digraph G , each node represents a part or a subassembly/assembly, and an arc represents a precedence relationship between two nodes. Any node with the number of edges incident to the node equal to 1 denotes a part. Any node with the number of edges incident to the node > 1 denotes a subassembly or an assembly. The root node of a digraph always represents an assembly.

The level of assembly in a digraph is assigned as follows: a 1 is assigned to the root node (assembly) and working backward from the root node, values of increment 1 are assigned to each subassembly node. Part nodes have the same assembly level as the corresponding subassembly or assembly nodes. Two products and the corresponding digraph representations of their assembly sequences are shown in figure 2. The digraphs representing the assembly sequences of products can be classified into two types: simple and complex. A simple digraph is one in which at most one subassembly node can be found at every assembly level (figure 2(a)). A simple digraph represents a linear assembly sequence of a product design. In a complex digraph, more than one subassembly node can be found on at least one assembly level (figure 2(b)). Throughout, an assembly sequence of a product represented by a simple digraph is referred to as a *simple assembly sequence* and an assembly sequence of a product represented by a complex digraph is referred to as a *complex assembly sequence*.

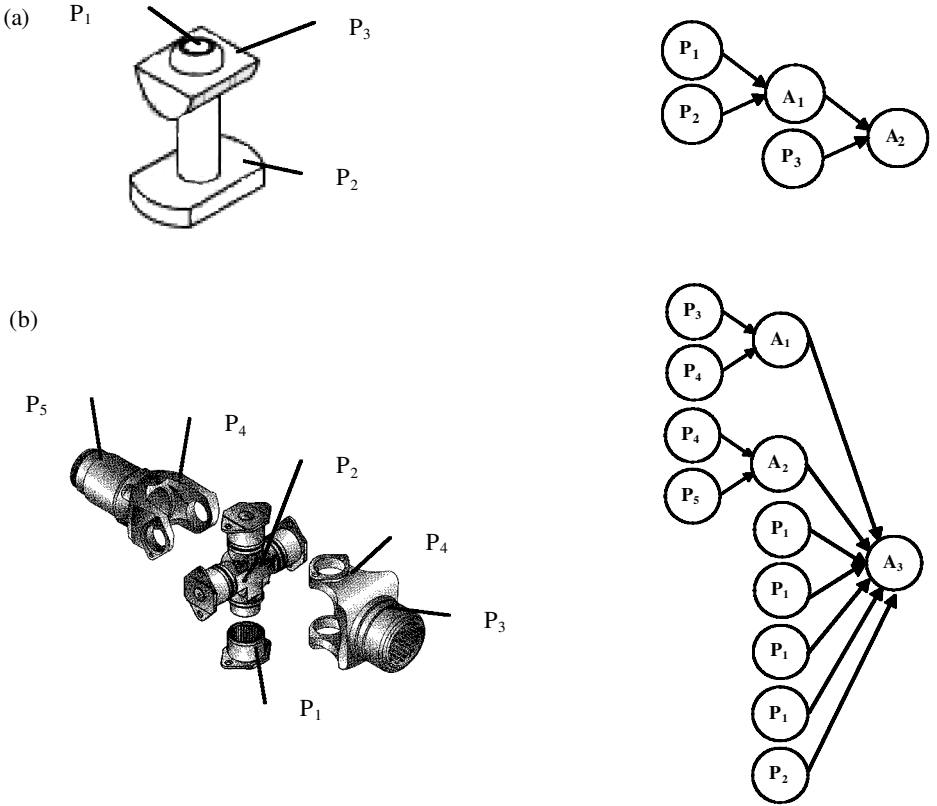


Figure 2. Example of (a) a product and its simple digraph and (b) a product and its complex digraph.

The complete formulation of the scheduling problems requires some assumptions to be made about the operations, resources and processing times. A collection of standard assumptions applicable to the scheduling problem discussed in this paper are given below.

- Every machine processes only one part at a time.
- Every part is processed on one machine at a time.
- Operations are not pre-emptive.
- There is unlimited buffer space between assembly stations in assembly stage.
- There is unlimited buffer space between machining and assembly stage.
- Set-up times for the operations are included in the processing times.
- Problem is a deterministic scheduling problem, i.e. fixed processing times of operations are known.

The scheduling problems will be solved according to the production modes in the system. Based on the representation of assembly sequence of the products, three production modes are defined: production of a single product with a *simple assembly sequence*; production of a single product with a *complex assembly sequence*; and production of *N products*. According to the three defined production modes, the

associated scheduling problems are defined as G_s scheduling problems, G_c scheduling problems and N -product scheduling problems, respectively.

3. Literature review

In general, there are types of problems solved in the literature that are related to the scheduling problems discussed here. One type of related problems are those of scheduling products represented by simple and complex digraphs in a two-stage flexible manufacturing system first solved as an aggregate scheduling problem in Kusiak (1989). Here, the aggregate scheduling problem is modelled as the two-machine flow shop scheduling problems. Optimal scheduling algorithms for solving both single product and N -product scheduling problems were developed. However, the aggregate scheduling problems solved in Kusiak (1989) assume that only one processing unit is available at both machining and assembly stages and this assumption does not reflect the real situation in implementing product differentiation strategies in an agile manufacturing environment.

Another type of related scheduling problems solved in the literature are the flow shop problems with parallel machines (FSPM). FSPM is considered as an extension of two classical scheduling problems: flow shop scheduling problem and parallel machine scheduling problem. FSPM is also a basic model for flexible flow line scheduling problems solved in the literature. Hunsucker and Shah (1994) reviewed industrial applications of scheduling in chemical engineering, computer systems, telecommunication networks, flexible manufacturing systems (FMSs), etc. Brah and Hunsucker (1991) developed a branch-and-bound algorithm to solve the make-span FMPM scheduling problem. Mathematical models of FSPM have been discussed in Brah *et al.* (1991). Some special cases of FSPM have also been studied in the literature. Gupta (1988) developed a heuristic algorithm for a two-stage problem with one machine at the second stage. Sriskandarajah and Sethi (1989) developed heuristics for a two-stage case and established the worst case bounds. Two heuristic algorithms generating high-quality solutions for a two-stage case with one machine at stage one and several machines at stage two have been developed by Gupta and Tunc (1991). Chen (1995) developed heuristics to solve the special cases for systems that have only two centres or systems where only one of the centres has parallel machines. Extensions of FSPM that incorporate buffers and transporters between centres have been studied by Wittrock (1988) and Sawik (1993).

Flow shop scheduling problems and parallel machine scheduling problems represent another class of related problems. The two-machine flow shop-scheduling problem can be solved by Johnson's algorithm (1954). However, in general, if the number of machines in a flow shop is more than two, then the scheduling problem is known to be NP-complete (e.g. Gonzalez and Sahni 1978). While the majority of research into flow shop scheduling is focused on the serial-type flow shop, a three-machine assembly-type flow shop-scheduling problem was studied by Lee *et al.* (1993).

Parallel machine scheduling problem ($P||C_{\max}$) has been proved by Garey and Johnson (1978) as NP-hard in a strong sense when the number of machines is unlimited. However, the problem is solvable in pseudo-polynomial time when the number of machines is fixed and thus NP-hard only in the ordinary sense. A recent survey on parallel machine scheduling problems was by Cheng and Sin (1990). Blazewicz *et al.* (1991) proposed optimal algorithms. However, most algorithms developed for solving $P||C_{\max}$ are heuristics (e.g. Graham 1969, Goffman *et al.* 1978, Friesen and Langston 1986).

Except for the scheduling problems solved by Kusiak (1989), none of the related scheduling problems solved in the literature considered product structures represented by the simple and complex digraphs even though they provide the best structural information of the products.

Lee and Vairaktarakis (1998) considered a similar system structure in their paper to the problem discussed in this paper and developed heuristics with worst-case error bounds. However, the products/jobs in their paper include only two sequential operations: one to be performed in the machining station, the other succeeding operations to be performed in the assembly station. The multilevel assembly structure of a product referred in this paper changes the essence of the problem considerably. This structure allows the implementation of the product differentiation concept.

A three-machine assembly-type flow shop-scheduling problem was studied by Lee *et al.* (1993). In their model, a final product is simply an assembly of two components. In the three-machine assembly-type flow shop, there are two parallel machines at the first stage and there is one assembly station at the second stage. When both components of a job are completed at the first stage, they are delivered to assembly stage, where a single assembly station assembles the components.

4. Development of solution methods

Methods developed for solving the scheduling problems with implementation of delayed product differentiation for agile manufacturing are presented here. The presentation consists of three parts. The first part covers the development of optimal solution methods for solving the scheduling problems. The second part covers the development of heuristic methods. In the third part, a lower bound is presented.

4.1. Development of optimal solution methods

Before the optimal solution methods for solving the G_s , G_c and N -product scheduling problems are presented the following notations and variables are defined.

- q is the number of assembly stations at the assembly stage,
- N is the number of assembly operations,
- A_i is the assembly operation i ,
- P_i is the machining operation i ,
- $t(A_i)$ is the assembly time of assembly operation A_i ,
- $t(P_i)$ is the machining time of machining operation P_i ,
- AP is the set of assembly operations with preceding part nodes,
- $IP(A_i)$ is the set of all assembly operations immediately preceding A_i ,
- $NA(A_i)$ is the set of assembly operation neither preceding nor succeeding to A_i ,
- l is the index of assembly level in a digraph,
- L is the maximum assembly level of a digraph, and
- M is the arbitrary large number.

Variables:

- $CT(A_i)$ is the completion time of assembly operation A_i ,
- $ST(A_i) = CT(A_i) - t(A_i)$, is the starting time of assembly operation A_i ,
- $x_{ij} = \begin{cases} 1, & \text{if assembly operation } i \text{ is assigned to machine } j \\ 0, & \text{otherwise,} \end{cases}$

$$q_{ij} = \begin{cases} 1, & \text{if machining operation } i \text{ is scheduled before} \\ & \text{machining operation } j; \\ 0, & \text{otherwise,} \end{cases}$$

$$CT_j(A_i) = \begin{cases} CT(A_i), & \text{if assembly operation } i \text{ is assigned to machine } j; \\ 0, & \text{otherwise,} \end{cases}$$

$y_{ijk}, z_{ij} = 0$ or 1 are auxiliary variables.

4.1.1. Solving the G_s scheduling problem

Here, the structure of the manufacturing system is represented by a tuple (m, q) , where m is the number of machines at the machining stage and q is the number of assembly stations at the assembly stage. Note that the G_s scheduling problem involves scheduling a single product with a simple assembly sequence in the manufacturing system with a structure $(m = 1, q > 1)$. The following theorem provides the optimal solution for solving the G_s scheduling problems.

Theorem 1 (see proof in Appendix A): Solving the G_s scheduling problems for system structure $(m = 1, q = 1)$ gives an optimal solution for solving the G_s scheduling problems for system structure $(m = 1, q > 1)$. The optimal solution can be obtained by applying the maximum level of depth first (MLDF) rule.

4.1.2. Solving the G_c scheduling problem

When a single product with a complex assembly sequence is produced in the system, the scheduling problem is a G_c scheduling problem.

Any complex digraph can be converted into a ‘standard’ digraph representation where *only* the assembly nodes with the highest assembly level have preceding parts. The digraph standardization procedure is described in the following.

4.1.2.1. *Digraph standardization procedure (DSP)*. Identify the set of part nodes with the assembly level l less than highest assembly level $(l < L)$. For each part node P_i from that set define dummy subassembly nodes $A_{d,i}^L, A_{d,i}^{L-1}, \dots$, and $A_{d,i}^{l+1}$ (the lower case letter d indicates *dummy*, index i indicates relation with part node P_i , the upper index indicates the level of assembly), with processing time equal to zero. Connect the part node to the subassembly node $A_{d,i}^L$, connect $A_{d,i}^L$ with $A_{d,i}^{L-1}$, and so on, until connecting $A_{d,i}^{l+1}$ with the subassembly/assembly node immediately succeeding to part node P_i in the original digraph.

Figure 3 shows a simple example of converting a complex digraph into a ‘standard’ digraph representation using DSP. Note that subassembly node $A_{d,5}^2$ added to the digraph is a dummy assembly node with assembly time equal to zero.

Note that the manufacturing system under the investigation has only one machine at the machining stage. Therefore, without loss of generality, it is convenient to aggregate all the part nodes that precede the same subassembly node into a single ‘aggregated’ part node whose machining time is the sum of the machining times of all the part nodes. The result of this aggregation is an ‘aggregated’ digraph representation of the assembly sequence. Figure 4 shows a simple example of the aggregation of a digraph.

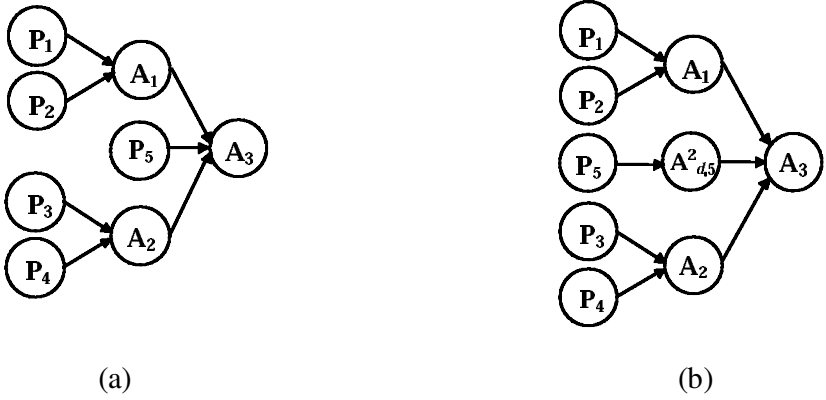


Figure 3. (a) Original complex digraph, where A_3 has a preceding part node; (b) converted digraph, where only assemblies with the highest assembly level have preceding part nodes.

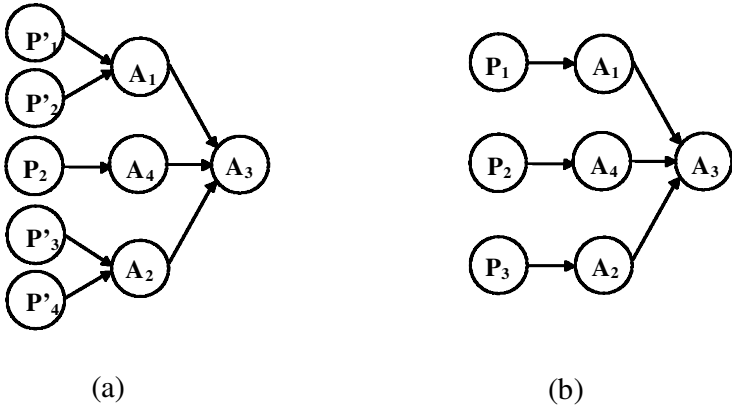


Figure 4. (a) Original digraph; (b) 'aggregated' digraph, where, $t(P_1) = t(P'_1) + t(P'_2), t(P_3) = t(P'_3) + t(P'_4)$.

A mixed-integer programming formulation for solving the G_c scheduling problem is given in equations (1)–(12).

Mixed-integer programming formulation:

$$\min Z = CT(A_N) \tag{1}$$

subject to

$$CT(A_i) - t(A_i) \geq \sum_{\substack{j=1 \\ j \neq i}}^k q_{ji} t(P_j) + t(P_i) \quad \text{for } i, A_i \in AP, k = |AP| \tag{2}$$

$$q_{ij} + q_{ji} = 1 \quad \text{for } i, j, A_i \in AP, A_j \in AP, i \neq j \tag{3}$$

$$CT(A_i) \geq CT(A_j) + t(A_i) \quad \text{for } i = 1, \dots, N, A_j \in IP(A_i) \tag{4}$$

$$\sum_{j=1}^q x_{ij} = 1 \quad i = 1, \dots, N \tag{5}$$

$$\sum_{j=1}^q CT_j(A_i) = CT(A_i) \quad i = 1, \dots, N - 1 \tag{6}$$

$$x_{ij} \leq M(1 - z_{ij}) \tag{7}$$

$$CT(A_i) - CT_j(A_i) \leq Mz_{ij} \quad \text{for } i = 1, \dots, N - 1, j = 1, \dots, q \tag{8}$$

$$CT_j(A_i) - CT_j(A_k) + x_{kj}t(A_k) \leq My_{ikj} \tag{9}$$

$$CT_j(A_k) + x_{ij}t(A_i) - CT_j(A_i) \leq M(1 - y_{ikj}) \tag{10}$$

for $i = 1, \dots, N - 1, j = 1, \dots, q, A_k \in NA(A_i)$

$$x_{ij}, q_{ij}, y_{ikj} \text{ and } z_{ij} = 0 \text{ or } 1. \tag{11}$$

$$\text{All other variables are non-negative.} \tag{12}$$

The objective function (1) minimizes the completion time of the final assembly operation A_N . Constraint (2) ensures that each assembly operation at the highest assembly level will start after the corresponding machining operation is completed. Constraint (3) ensures that for any two part nodes P_i and P_j either P_i is assigned before P_j or P_j is assigned before P_i . Constraint (4) ensures that assembly/subassembly operations having immediately preceding subassemblies will not start before the completion time of preceding assembly operations. Constraint (5) ensures that each assembly operation will be assigned to only one assembly station. Constraints (6)–(8) ensure that if assembly A_i is assigned to station j , then the completion time of assembly A_i on assembly station j is equal to the completion time of that assembly operation A_i . Constraints (9) and (10) ensure that no overlapping of assembly operations occur on any assembly station. Constraints (11) and (12) ensure integrity and non-negativity.

4.1.3. Solving the N -product scheduling problem

The N -product scheduling problem involves scheduling N multiple products with either simple or complex assembly sequences. The approach for solving the N -product scheduling problem is to construct a complex digraph by connecting the assembly nodes of N products to a dummy final assembly node, A_d . An assembly time of 0 is assigned to A_d , i.e. $t(A_d) = 0$. Then solving the N -product scheduling problem is equivalent to solving a G_c scheduling problem. Figure 5 shows the transformation of N digraphs into a single complex digraph.

Note that optimal solution for G_s scheduling problem can be obtained easily. However, the model (1–12) for G_c scheduling problems is difficult to solve optimally for large size problems due to its computational complexity as the G_c scheduling problem can be considered as an extended parallel machine makespan scheduling problem. As an N -product scheduling problem can be always converted into an equivalent G_c scheduling problem, the computational complexity for solving a N -product scheduling problem is the same as for solving a G_c scheduling problem. To solve the scheduling problems efficiently, heuristic algorithms for solving the G_c and N -product scheduling problems are developed.

4.2. Development of heuristic algorithms

Consider scheduling an assembly product that has a complex assembly sequence. Without loss of generality, it is assumed that the assembly sequence of the product

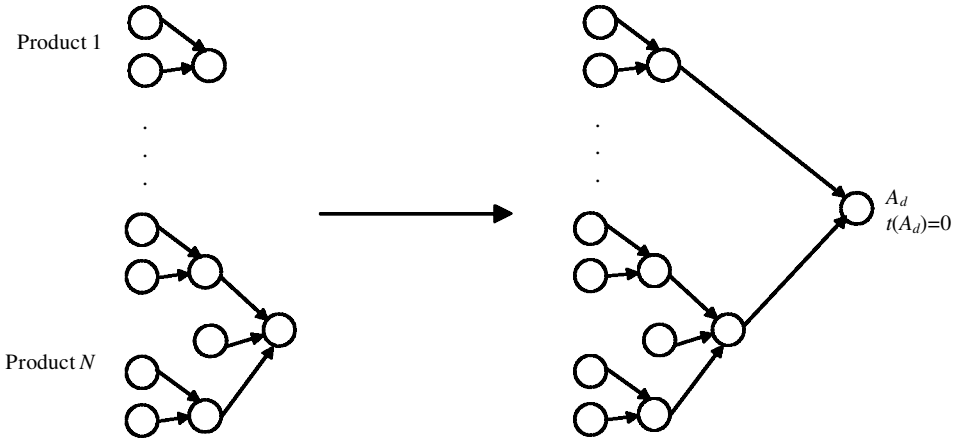


Figure 5. Transforming an N -product scheduling problem to a G_c scheduling problem.

can be represented by a ‘standard’ aggregated digraph with total number of aggregated part nodes equal to n .

Before introducing the terms to be used in this section, note that the makespan does not actually depend on the last assembly operation. So, the processing time of the last assembly is omitted for the convenience.

Define:

- N number of assembly nodes in a converted digraph,
- N_l number of assembly/subassembly nodes at assembly level l ,
- n number of part nodes in the converted digraph, which is same as number of subassemblies having preceding part node,

$$TM = \sum_{i=1}^n t(P_i) \quad \text{total machining time,}$$

$$TA = \sum_{i=1}^N t(A_i) \quad \text{total assembly time,}$$

$$TSA = \sum_{i=1}^{N-1} t(A_i) \quad \text{total subassembly time,}$$

$$\bar{t} = \frac{TM}{n} \quad \text{average machining time per part,}$$

$$\bar{T} = \frac{TSA}{N-1} \quad \text{average subassembly time per subassembly operation.}$$

In developing the heuristics, three cases are considered based on the relationship between total machining times and assembly times. For each case a heuristic is developed. To define each case, consider a general representation of assembly sequence in figure 6.

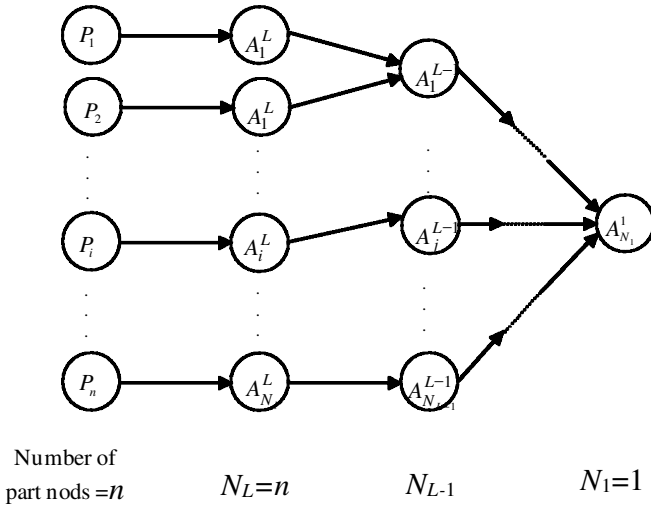


Figure 6. General representation of assembly digraph that indicates the assembly nodes at each assembly level.

Note that the maximum number of assembly nodes at assembly level l can be calculated as

$$N_l = \left\lceil \frac{n}{2^{L-l}} \right\rceil,$$

where $\lceil \bullet \rceil$ is the nearest integer of \bullet . Hence, the total maximum number of assembly operations in a digraph with n part nodes is:

$$\begin{aligned}
 N &= \sum_{l=1}^L N_l = n + \left\lceil \frac{n}{2^1} \right\rceil + \left\lceil \frac{n}{2^2} \right\rceil + \left\lceil \frac{n}{2^3} \right\rceil + \dots + \left\lceil \frac{n}{2^{L-1}} \right\rceil \\
 &= n + \left\lceil \frac{n}{2^1} + \frac{n}{2^2} + \frac{n}{2^3} + \dots + \frac{n}{2^{L-1}} \right\rceil \\
 &\leq n \left(1 + \frac{1}{2^1} + \frac{1}{2^2} + \frac{1}{2^3} + \dots + \frac{1}{2^{L-1}} \right) < 2n.
 \end{aligned} \tag{13}$$

Based on the relationship of the total machining and assembly times in a digraph, three separate cases in solving a G_c scheduling problem are considered in the following.

- Case 1 : $TM < \frac{TSA}{q}$
- Case 2 : $TM \geq \frac{TSA}{q}$ and $\bar{t} \geq 2\bar{T}$
- Case 3 : $TM \geq \frac{TSA}{q}$ and $\bar{t} < 2\bar{T}$.

Next, heuristic algorithms for solving the G_c scheduling problems for these cases are presented.

Case1 : $TM < \frac{TSA}{q}$.

In this case, from (13) it can be shown that

$$n\bar{t} = TM < \frac{TSA}{q} < \frac{N\bar{T}}{q} < \frac{2n\bar{T}}{q} \text{ or } \bar{t} < \frac{2\bar{T}}{q}, \text{ since } q \geq 2, \bar{t} < \bar{T}. \tag{14}$$

From (14) one can see that the average assembly time per part is greater than the average machining time per assembly operation. In this case, the scheduling priority should be given to the subassembly operations. Starting subassemblies as soon as possible will improve the makespan. Hence, the SPTF (shortest processing time first) rule should provide good solutions to the scheduling problems. The heuristic algorithm developed for Case 1 is presented next.

Heuristic algorithm 1 (HA1):

- Step 1. Apply the DSP to obtain a ‘standard’ aggregated digraph *G*.
- Step 2. Schedule the machining operations at the machining stage with the SPTF rule.
- Step 3. Schedule assembly operations on available assembly stations as soon as possible (ASAP).

Example 1 is given illustrate the application of HA1.

Example 1: Consider scheduling a product with ‘standard’ aggregated assembly sequence presented in figure 7 in a manufacturing system with a structure ($m = 1, q = 2$). The machining and assembly times are given in table 1.

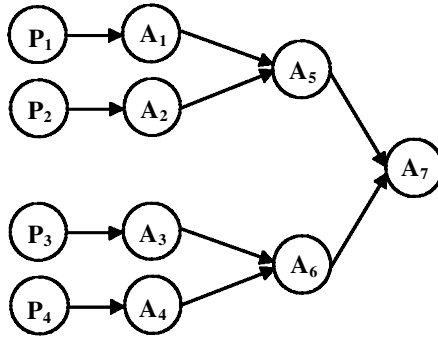


Figure 7. Assembly sequence of the product.

Part number	P ₁	P ₂	P ₃	P ₄	–	–	–
Machining time	3	4	2	5	–	–	–
Subassembly number	A ₁	A ₂	A ₃	A ₄	A ₆	A ₆	A ₇
Assembly time	7	11	9	8	10	10	6

Table 1. Machining and assembly times.

$$TM = 14$$

$$TSA = 55.$$

Since

$$TM < \frac{TSA}{2},$$

apply heuristic algorithm 1.

Since the digraph in figure 7 is a ‘standard’ aggregated digraph representation of the assembly sequence, the first step in HA1 is skipped.

Step 2. Applying the SPTF rule provides the following sequence of the machining operations: P₃, P₁, P₂, P₄.

Step 3. After scheduling the assembly operations on available assembly stations as soon as possible (ASAP), the schedule is obtained in figure 8.

$$\text{Case 2 : } TM \geq \frac{TSA}{q} \quad \text{and} \quad \bar{t} \geq 2\bar{T}.$$

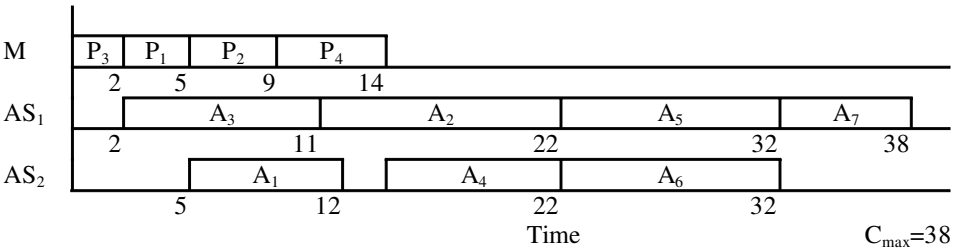


Figure 8. Gantt chart of schedule obtained from heuristic HA1.

In this case, on average, each machining operation takes twice as much times as an assembly operation. Applying inequality (13) we get:

$$TM = n\bar{t} \geq 2n\bar{T} \geq (N - 1)\bar{T} = TSA \geq \frac{TSA}{q}.$$

Since $\bar{t} \geq 2\bar{T}$ implies that

$$TM \geq \frac{TSA}{q},$$

then the condition

$$TM \geq \frac{TSA}{q}$$

can be omitted in Case 2.

Physically this means machining station is strictly busier than assembly station. Of course, the condition given by Case 2 is much stronger than this. Therefore, most of the assembly operations can be scheduled on the first assembly stations at the assembly stage. Hence, a schedule obtained by solving a G_c scheduling problem in a system where there is only one assembly station at the assembly stage provides a very good approximation of the optimal schedule.

Heuristic algorithm 2 (HA2):

Step 1. Apply the DSP to obtain a ‘standard’ aggregated digraph G .

- Step 2. Apply Theorem 2 in Kusiak (1989) to obtain an aggregate minimum make-span schedule. Schedule part nodes on the machine at the machining stage according to the aggregate schedule.
- Step 3. Schedule assembly operations on first available assembly stations as soon as possible (ASAP).

Example 2 is given to illustrate the application of HA2.

Example 2: Consider scheduling a product with assembly sequence presented in figure 7 in a manufacturing system with a structure ($m = 1, q = 2$). The machining and assembly times are given in table 2.

$$TM = 87, TSA = 57, \bar{t} = 87/4 = 21.75, \bar{T} = 57/6 = 9.5.$$

Since $\bar{t} \geq 2\bar{T}$, apply heuristic algorithm 2.

Since the digraph in figure 7 is a ‘standard’ aggregated digraph representation of the assembly sequence, the first step in HA2 is skipped.

- Step 2. Applying Theorem 2 in Kusiak (1989) provides the following sequence of the machining operations: P_3, P_4, P_2, P_1 .
- Step 3. After scheduling assembly operations on the first available assembly stations as soon as possible (ASAP), the schedule is obtained in figure 9.

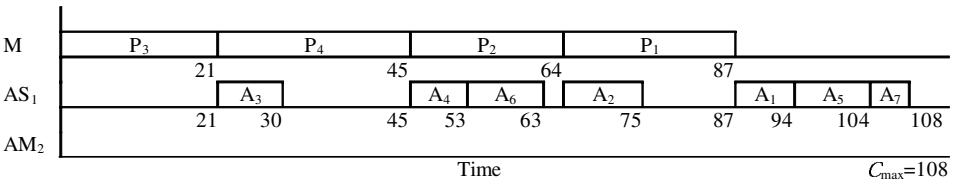


Figure 9. Gantt chart of schedule obtained using heuristic HA2.

$$\text{Case 3 : } TM \geq \frac{TSA}{q} \text{ and } \bar{t} < 2\bar{T}.$$

This is the opposite case of the combination of previous 2 cases. In this case, one can see that there are some ‘frequent’ cases of idle time on the assembly stations. Each assembly station is less busy, on average, than the machine at the machining station. Therefore, instead of assigning the scheduling priority to the subassembly operations as in Case 1, attention should be given to the total subassembly time belonging to a direct path from the part nodes to the assembly node. The heuristic algorithm for solving the G_c scheduling problems for Case 3 is presented next.

Heuristic algorithm 3 (HA3):

- Step 1. Apply the DSP to obtain a ‘standard’ aggregated digraph G .

Part number	P_1	P_2	P_3	P_4	–	–	–
Machining time	23	19	21	24	–	–	–
Subassembly number	A_1	A_2	A_3	A_4	A_6	A_6	A_7
Assembly time	7	11	19	10	10	10	4

Table 2. Machining and assembly times.

- Step 2. For each part node P_i calculate total subassembly time T_i of subassembly nodes belonging to the direct path from P_i to the root node A_N .
- Step 3. Schedule part nodes on the machine at the machining stage with the descending order of T_i .
- Step 4. Schedule assembly operations on available assembly stations as soon as possible (ASAP).

Example 3 is given to illustrate the application of HA3.

Example 3: Consider scheduling a product with assembly sequence presented in figure 7 in a manufacturing system with a structure ($m = 1, q = 2$). The machining and assembly times are given in table 3.

$$TM = 41, TSA = 64, \bar{t} = 41/4 = 10.25, \bar{T} = 64/6 = 10.67.$$

Since

$$TM > \frac{TSA}{2} \quad \text{and} \quad \bar{t} < 2\bar{T},$$

apply heuristic algorithm 3.

Since the digraph in figure 7 is a ‘standard’ aggregated digraph representation of the assembly sequence, the first step in HA3 is skipped.

Step 2. Calculate $T_i, i = 1, \dots, 4$:

$$\begin{aligned} T_1 &= t(A_1) + t(A_5) = 20 \\ T_2 &= t(A_2) + t(A_5) = 21 \\ T_3 &= t(A_3) + t(A_6) = 25 \\ T_4 &= t(A_4) + t(A_6) = 24. \end{aligned}$$

- Step 3. Schedule the part nodes at the machining stage according to the P_3, P_4, P_2, P_1 sequence.
- Step 4. After scheduling assembly operations on available assembly station as soon as possible (ASAP), the schedule is obtained in figure 10.

Finally, based on the heuristics developed for the G_c scheduling problems, the heuristic for solving N -product scheduling is presented next.

Heuristic algorithm 4 (HA4):

- Step 1. Convert the N -product scheduling problem into an equivalent G_c scheduling problem by constructing a complex digraph by connecting the assembly nodes of N products to a dummy final assembly node, A_d . Let $t(A_d) = 0$.
- Step 2. Solving the equivalent G_c scheduling problems using HA1, HA2 or HA3.

Part number	P ₁	P ₂	P ₃	P ₄	–	–	–
Machining time	10	8	11	12	–	–	–
Subassembly number	A ₁	A ₂	A ₃	A ₄	A ₆	A ₆	A ₇
Assembly time	8	9	11	10	12	14	4

Table 3. Machining and assembly times.

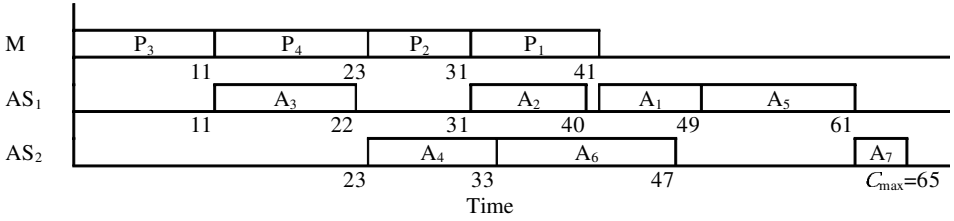


Figure 10. Gantt chart of schedule obtained using heuristic HA3.

Since all these heuristics were developed based on the relationships between the average machining and assembly times, their performance is expected to be good with a production scenario of larger number of parts to be assembled, and small standard deviations of machining and assembly times. This type of production scenario represents the situation in implementing the delayed product differentiation strategies in an agile manufacturing environment.

4.3. Development of a lower bound

To evaluate the performance of the heuristics a tight lower bound was developed. In developing the lower bound, the following terms are defined:

PH_i set of the subassembly nodes belonging to the direct path from the part node

P_i to the root node $A_N, i = 1, \dots, n$

$\min^{(k)}\{(\bullet)\}$ = the k th smallest number in the set (\bullet) .

To develop a tight lower bound, two lower bounds are constructed and the tight bound is the largest one between the two lower bounds. In general, two types of relations can be established between the machining and assembly times in the system where we have a single machining and multiple assembly stations. First, the average of the total subassembly time (TSA/q) is less than or equal to the total machining time TM , and second, the average of the total subassembly time (TSA/q) is larger than the total machining time TM . The developed lower bound LB_1 is effective in the first case, while the lower bound LB_2 is effective in the second case. To develop a tight lower bound, largest one between the two lower bounds is chosen for the general case.

$$LB_1 = t(A_N) + TM + \min_{i=1}^n \left\{ \sum_{A_j \in PH_i} t(A_j) \right\}$$

$$LB_2 = t(A_N) + \frac{TSA + \sum_{k=1}^q (q - k + 1) \min_{i=1}^{n(k)} n\{t(P_i)\}}{q}$$

$$LB = \max\{LB_1, LB_2\}$$

Example 4 is given to illustrate the calculation of the lower bound.

Example 4: Find the lower bound for the scheduling problem in Example 1. To find the lower bound one needs to first calculate TM, TSA, $PH_i, i = 1, \dots, 4$.

$$TM = \sum_{i=1}^4 t(P_i) = 14, \quad TSA = \sum_{i=1}^6 t(A_i) = 55$$

$$PH_1 = \{A_1, A_5\}, \text{ the total subassembly time on path } PH_1 \text{ is } \sum_{A_j \in PH_1} t(A_j) \\ = t(A_1) + t(A_5) = 17$$

$$PH_2 = \{A_2, A_5\}, \text{ the total subassembly time on path } PH_2 \text{ is } \sum_{A_j \in PH_2} t(A_j) \\ = t(A_2) + t(A_5) = 21$$

$$PH_3 = \{A_3, A_6\}, \text{ the total subassembly time on path } PH_3 \text{ is } \sum_{A_j \in PH_3} t(A_j) \\ = t(A_3) + t(A_6) = 19$$

$$PH_4 = \{A_4, A_6\}, \text{ the total subassembly time on path } PH_4 \text{ is } \sum_{A_j \in PH_4} t(A_j) \\ = t(A_4) + t(A_6) = 18$$

$$LB_1 = t(A_7) + TM + \min_{i=1}^4 \left\{ \sum_{A_j \in PH_i} t(A_j) \right\} = 6 + 14 + \min\{17; 21; 19; 18\} = 37$$

$$LB_2 = t(A_7) + \frac{TSA + \sum_{k=1}^2 (2 - k + 1) \min_{i=1}^{4_{(k)}} \{t(P_i)\}}{q} = 6 + (55 + 2 \times 2 + 3)/2 = 37$$

$$LB = \max\{LB_1, LB_2\} = \max\{37; 37\} = 37.$$

Similarly, the lower bounds for Examples 2 and 3 were calculated and the result are $LB = 65$ and 108 , respectively.

By comparing the corresponding lower bounds with the results obtained by heuristic algorithms HA1, HA2 and HA3, one can conclude that the heuristics obtained optimal solutions for the three examples.

5. Computational experience

To evaluate the performance of the heuristics developed to solve the scheduling problems, a computational experiment was conducted. In the computational experiment, makespans of the schedules obtained by the heuristics were compared with the lower bounds. Eleven types of problems with different assembly sequences were tested. For each testing problem, 10 instances were generated, and for each instance, the assembly sequence, the number of part nodes, the number of subassembly nodes, the maximum level of assembly, the machining times and the assembly times were randomly generated. The data were generated based on the real assembly application information from industrial assembly handbooks (e.g. Lotter 1989, Boothroyd 1992, Nof *et al.* 1996). The average size of the problems corresponds to 35 part nodes, 54

Relation between \bar{t} and \bar{T}	Problem no.	C_{max}^1 (HA1)	$C_{max}^{2/3}$ (HA2 or HA3)	LB (lower bound on C_{max})	$\left(\frac{C_{max}^1 - LB}{C_{max}^1}\right)$	$\left(\frac{C_{max}^{2/3} - LB}{C_{max}^{2/3}}\right) \times 100\%$
$\bar{t} < \bar{T}$	1	353	365	350	2.25	4.11
	2	365	365	348	0.81	4.66
	3	397	399	388	0.00	2.76
	4	375	392	355	2.11	9.44
	5	403	411	375	0.88	8.76
$\bar{t} \geq \bar{T}$	6	533	521	521	0.85	0.00
	7	494	490	490	4.66	0.00
	8	670	670	670	2.27	0.00
	9	522	522	511	5.33	2.11
	10	678	679	672	6.95	1.03
	11	677	658	658	2.81	0.00

Table 4. Results of the computational experiment.

assembly nodes and six assembly levels. The machining times and assembly times were generated using uniform distribution $U(2, 25)$ and $U(1, 30)$, respectively. The test problems were further divided into two categories based on the relationship between \bar{t} and \bar{T} : $\bar{t} < \bar{T}$ and $\bar{t} \geq \bar{T}$. For problems in both categories of $\bar{t} < \bar{T}$, heuristic algorithm HA1 was applied. For problems in the category $\bar{t} \geq \bar{T}$, either HA2 or HA3 was applied depending on the relationship between \bar{t} and $2\bar{T}$. Since solving the N -product scheduling problem is equivalent to solving a G_c scheduling problem and heuristics developed for solving the G_c scheduling problems are applied directly to solve the N -product scheduling problems, the performance of HA1, HA2 and HA3 should indicate the same level of performance of HA4. For this reason, HA4 was not tested in the experiment. The results of the comparison are given in table 4.

As one can see from table 4, the relative differences between the makespans of the HA1 solutions and the lower bounds are small for all the tested problems that satisfy the $\bar{t} < \bar{T}$ condition. In addition, for all the problems with $\bar{t} < \bar{T}$, the solutions obtained by HA1 are better than the solutions obtained by HA2. Similarly, the relative differences between the makespans of the HA2 or HA3 solutions and the lower bounds are small for all the tested problems that satisfy $\bar{t} \geq \bar{T}$ condition. For the problems satisfying the $\bar{t} \geq \bar{T}$ condition, HA2 or HA3 outperform HA1. For many problems tested in the experiment, the makespans were the same as the lower bounds, which indicate that the solutions are optimal. The results of the computational experiment suggest that for better scheduling performance, HA1 should be used when $\bar{t} < \bar{T}$, and HA2 or HA3 when $\bar{t} \geq \bar{T}$.

6. Conclusions

Production of customized products to respond to changing markets in a short time and at a low cost for agile manufacturing can be implemented with delayed product differentiation in a manufacturing system. The successful implementation of delayed product differentiation lies in efficient scheduling of the manufacturing system.

Here, scheduling problems associated with implementing delayed product differentiation in a general flexible manufacturing system are defined, formulated and

solved. The manufacturing system consists of two stages: machining and assembly. At the machining stage, one machine produces standard component parts for assembly products. These parts are then assembled at the assembly stage by multiple identical assembly stations to form customized products. The products to be produced in the system are characterized by their assembly sequences, which are represented by different digraphs. The scheduling problem is to determine the sequence of products to be produced in the system so that the maximum completion time (makespan) is minimized for any given number of assembly stations at the assembly stage. Based on the representation of assembly sequence of the products, three production modes are defined: production of a single product with a *simple assembly sequence*; production of a single product with a *complex assembly sequence*; and production of N products. According to the three defined production modes, the associated scheduling problems are defined as G_s scheduling problems, G_c scheduling problems and N -product scheduling problems, respectively. Optimal and heuristic methods for solving the scheduling problems are developed.

To show the effectiveness of developed solution approaches, test problems were randomly generated based on the assembly application information from industrial assembly handbooks. The computational results show that the developed heuristics provide good solutions to the scheduling problems. It is believed that these randomly generated problems can be good representatives of real industrial problems and the conclusions drawn from the heuristic performances are valid for real environments.

The assumptions made in formulating the problems constrain the scheduling problems tightly and also predict and facilitate the generation of the feasible solutions. Future work can be addressed to develop more sophisticated algorithms that consider set-up times, buffer spaces, etc. and which are applicable for more complex industrial environments. It is also aimed to produce a suite of mechanisms that collectively cover a broad range of problems using dominance rules suitable for different systems. Ideally, it would choose a mechanism for a given problem and generate a desired solution.

Appendix: Proof of Theorem 1

Define:

A^l is a subassembly/assembly node at assembly level l , $l = 1, \dots, L$, in a simple digraph.

By the definition of a simple digraph, for any two subassembly/assembly nodes in a simple digraph, the following condition holds:

$$ST(A^{l+1}) \geq CT(A^l). \quad (1)$$

From (1), the followings can be derived:

$$\begin{aligned} ST(A^{l+1}) + t(A^{l+1}) &\geq CT(A^l) + t(A^{l+1}) \\ CT(A^{l+1}) &\geq CT(A^l) + t(A^{l+1}). \end{aligned} \quad (2)$$

Since relation (1) holds for any subassembly/assembly nodes in a simple digraph, for subassembly/assembly nodes A^{l+1} and A^{l+2} in a simple digraph the following

should hold:

$$\text{ST}(A^{l+2}) \geq \text{CT}(A^{l+1}). \quad (3)$$

By substituting $\text{CT}(A^{l+1})$ in (2) with $\text{ST}(A^{l+2})$ in (3), the following relation can be derived:

$$\text{ST}(A^{l+2}) \geq \text{CT}(A^l) + t(A^{l+1}) \text{ or } \text{ST}(A^{l+2}) - \text{CT}(A^l)t(A^{l+1}). \quad (4)$$

Relation (4) indicates that for any three consecutive assembly operations in a feasible sequence, there is always enough time to start and complete the second assembly operation between the starting time of the third assembly operation and completion time of the first assembly operation. Since relation (1) indicates that for any two consecutive assembly operations in a feasible sequence one can always schedule the second assembly operation on the same machine as the first assembly operation, from relation (4) one can always schedule the third assembly operations on the same machine as the first and second assembly operations. Thus, by starting with the first assembly operations in a feasible sequence, one can always schedule the remaining assembly operations on the same machine as the first assembly operation. Therefore, for any feasible schedule of a G_s scheduling problem in a system structure $(m = 1, q > 1)$, there is always an equivalent feasible schedule of the G_s scheduling problem in the system structure $(m = 1, q = 1)$.

Since the optimal schedule of a G_s scheduling problem in system structure $(m = 1, q = 1)$ can be obtained by the maximum level of depth first (MLDF) rule (Kusiak 1989), an optimal schedule of the G_s scheduling problem for a system structure $(m = 1, q > 1)$ can also be obtained by the MLDF rule. Thus, the proof is completed.

References

- BLAZEWICZ, J., DROR, M. and WEGLARZ, J., 1991, Mathematical programming formulations for machine scheduling: a survey. *European Journal of Operational Research*, **51**, 283–300.
- BOOTHROYD, G., 1992, *Assembly Automation and Product Design* (New York: Marcel Dekker).
- BRAH, S. A. and HUNSUCKER, J. L., 1991, Branch and bound algorithm for the flow shop with multiple processors. *European Journal of Operational Research*, **51**, 88–99.
- BRAH, S. A., HUNSUCKER, J. L. and SHAH, J. R., 1991, Mathematical modeling of scheduling problems. *Journal of Information and Optimization Sciences*, **12**, 113–137.
- CHEN, B., 1995, Analysis of classes of heuristics for scheduling a two-stage flow shop with parallel machines at one stage. *Journal of Operational Research Society*, **46**, 234–244.
- CHENG, T. C. E. and SIN, C. C. S., 1990, A state-of-the-art review of parallel-machine scheduling research. *European Journal of Operational Research*, **47**, 271–292.
- FRIESEN, D. K. and LANGSTON, M. A., 1986, Evaluation of a multifit-based scheduling algorithm. *Journal of Algorithms*, **7**, 35–59.
- GAREY, M. R. and JOHNSON, D. S., 1978, Strong NP-completeness results: motivations, examples and implications. *Journal of Association of Computing and Mathematics*, **25**, 499–508.
- GOFFMAN, E. G., GAREY, M. R. and JOHNSON, D. S., 1978, An application of bin-packing to multiprocessor scheduling. *SIAM Journal of Computing*, **7**, 1–17.
- GONZALEZ, T. and SAHNI, S., 1978, Flowshop and jobshop schedules: complexity and approximation. *Operations Research*, **26**, 36–52.
- GRAHAM, R. L., 1969, Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, **17**, 416–429.
- GUPTA, J. N. D., 1988, Two-stage hybrid flowshop scheduling problem. *Journal of Operational Research Society*, **34**, 359–364.

- GUPTA, J. N. D. and TUNC, E. A., 1991, Schedules for a two-stage hybrid flowshop with parallel machines at the second stage. *International Journal of Production Research*, **29**, 1489–1502.
- GUPTA, S. and KRISHNAN, V., 1998, Product family-based assembly design methodology. *IIE Transactions*, **30**, 933–945.
- HE, D. W. and KUSIAK, A., 1996, Performance analysis of modular products. *International Journal of Production Research*, **34**, 253–272.
- HUNSUCKER, J. L. and SHAH, J. R., 1994, Comparative performance analysis of priority rules in a constrained flow shop with multiple processors environment. *European Journal of Operational Research*, **72**, 102–114.
- JOHNSON, S. M., 1954, Optimal two and three-stage production schedule with setup times included. *Naval Research Logistics Quarterly*, **1**, 61–68.
- KUSIAK, A., 1989, Aggregate scheduling of a flexible machining and assembly system. *IEEE Transactions on Robotics and Automation*, **5**, 451–459.
- LEE, C. Y., CHENG, T. C. E. and LIN, B. M. T., 1993, Minimizing the makespan in the 3-machine assembly-type flowshop scheduling problem. *Management Science*, **39**, 616–625.
- LEE, C. Y. and VAIRAKTARAKIS, G. L., 1998, Performance comparison of some classes of flexible flow shops and job shops. *International Journal of Flexible Manufacturing Systems*, **10**, 379–405.
- LOTTER, B., 1989, *Manufacturing Assembly Handbook* (London: Butterworths).
- NOF, S. Y., WILHELM, W. E. and WARNECKE, H.-J., 1996. *Industrial Assembly* (New York: Chapman Hall).
- SAWIK, T., 1993, A scheduling algorithm for flexible flow lines with limited intermediated buffers. *Applied Stochastic Models and Data Analysis*, **9**, 127–138.
- SILVER, E. A. and PETERSON, R., 1985, *Decision Systems for Inventory Management and Production Planning*, 2nd edn (New York: Wiley).
- SRISKANDARAJAH, C. and SETHI, S. P., 1989, Scheduling algorithms for flexible flowshop: worst and average case performance. *European Journal of Operational Research*, **43**, 143–160.
- WARNECKE, H. J. and WALTHER, J., 1982, Automatic assembly-state of the art. *Proceedings of the 3rd International Conference of Assembly Automation*, Boeblingen, pp. 1–14.
- WITTROCK, R. J., 1988, An adaptable scheduling algorithm for flexible flow lines. *Operations Research*, **36**, 445–453.