



## A FORMAL FUNCTIONAL CHARACTERIZATION OF SHOP FLOOR CONTROL

RICHARD A. WYSK and JEFFREY S. SMITH

Texas A & M University, College Station, TX 77843, U.S.A.

**Abstract**—Shop-floor control systems for discrete part manufacturing have been widely described by many researchers. These descriptions have led to several proposed frameworks to solve the overall shop floor control problem. Some terms commonly used to describe these frameworks include: centralized control, hierarchical control, heterarchical control, and hybrid control. However, these frameworks have not provided a formal generic vision of shop floor control. Rather, they have concentrated on specific implementations of the problem, or small classes of similar implementations. As a result, the frameworks which have been developed are applicable only to a subset of the problem domain. By contrast, a solution to the general shop floor control problem is highly desirable. It is only after the general shop floor problem is understood that such a general solution can be developed. In response to this need, this paper presents a formal functional characterization of the general shop floor control problem. Key to our functional characterization of the controller is the use of process plans as part of the control. We use a process plan representation model compatible with the ISO TC184/SC4/WG3/P11 proposed standard as part of the control input. The ISO process plan model is converted to a digraph which provide the task representation for control. In this paper, we illustrate how these digraphs provide the essential requirements for input to and for a functional characterization of shop floor control.

### 1. INTRODUCTION

Shop floor control for discrete part manufacturing has been widely described by many researchers [1-9]. These descriptions have led to several frameworks to solve the overall shop floor control problem. Many of these frameworks have been called control architectures, control structures, control models, etc. Examples include centralized control, hierarchical control, heterarchical control, and hybrid control. However, none of these provide an adequate description of the *general* shop floor control problem. Consequently, at best, these structures provide a solution to a single instance or a set of similar instances of the general problem.

Davis and Jones [10] suggest that, "The key to resolving these [integration] issues lies in a better understanding of each manufacturing function and how it is related to other manufacturing functions. Once we have this understanding, we can then address the questions involving the best architecture for a given manufacturing environment". We agree with this assessment and include that it is premature to describe solutions to specific instances of the shop floor control problem, and suggest that they are of little generalizable value without first creating a formal functional description of the general problem. In response, this paper presents a formal characterization of the general shop floor control problem. An important property of this description is that it is independent of the solution methodology selected to solve the problem.

The control system and related models that are described in this paper were developed for discrete manufacturing systems such as machining systems. The control system that we are working toward is intended to operate with automated equipment such as NC machines and robots; however, human assisted or manual workstations can also be included if the human responds to prompts and commands (via a CRT, for example) in a deterministic manner as the equipment would respond. In this case, the human is transparent to the other portions of the control system. Similarly, the equipment is assumed to be connected by an automated material transport system, such as an AGV or conveyor system which transports items singly or in batches between the individual pieces of equipment. Although the formalism presented is not dependent on these assumptions, they will simplify the presentation.

## 2. SHOP FLOOR CONTROL

A shop consists of several pieces of manufacturing equipment which are used to process, transport, and store *items*. In this case, an *item* can be a part, raw material, a tool, a fixture, or any other object which moves through the shop. The equipment includes NC machines tools, robots, conveyors, automated guided vehicles, automated storage systems, etc. In order to process a part on a piece of equipment, a set of other items (resources) might also be required (e.g. specific cutting tools or fixtures), depending on the operation. Furthermore, between processing tasks, some or all of these items must be transported between the different pieces of equipment. An input/output diagram for the general shop floor control problem in this environment is illustrated in Fig. 1. This figure shows the shop floor control system (SFCS) as a black box entity which receives inputs from an external (business) system(s) and generates the device-specific instructions necessary to enact the individual manufacturing tasks. This paper describes the functions performed by the black box. As shown in Fig. 1, the primary inputs to the SFCS are the *production requirements*, which describe the parts to be manufactured, and the *resources*, which describe the shared resources (items) to be used by the equipment within the shop.

Based on these inputs (which are described in more detail in the following paragraphs), the SFCS must generate a set of individual equipment processing instructions necessary to manufacture and transport the parts specified in the production requirements. These instructions include, for example, RS-274 instructions (or CLDATA) for NC machines, VAL II, AML, or Cartesian coordinates for robots, load/unload station and path descriptions for AGV's, etc. The specific format of the instructions will depend on the manufacturer/model of the equipment within the shop. Moreover, this set of individual instructions must be sequenced so that it provides a "good" or hopefully optimal, throughput, equipment utilization, or other measure; based on some performance criteria. Note, however, that we are looking for *a* set of processing instructions rather than *the* set of processing instructions, highlighting the possible existence of alternative processing routes for parts. Also note that the term *generate* as used here in the context of equipment instructions, can mean on-line creation, database retrieval, or some combination of both.

The *production requirements* input describes the technical and administrative requirements for the parts to be manufactured in the shop. The production requirements are provided by the master production schedule, the MRP system, or whatever system is used for shop-wide planning and loading. The technical requirements include the processing requirements and any special handling and/or environmental requirements which affect the parts. The processing requirements for a part are specified by the *process plan*, which includes the valid part routings with the associated tooling and fixturing requirements. We use a process plan representation compatible with the ISO TC184/SC4/WG3/P11 representation standard for process plans, and convert these plans to AND/OR digraphs for control use [11–13]. The process plan graphs define the alternative tasks and sequence requirements for the specific equipment within the shop.

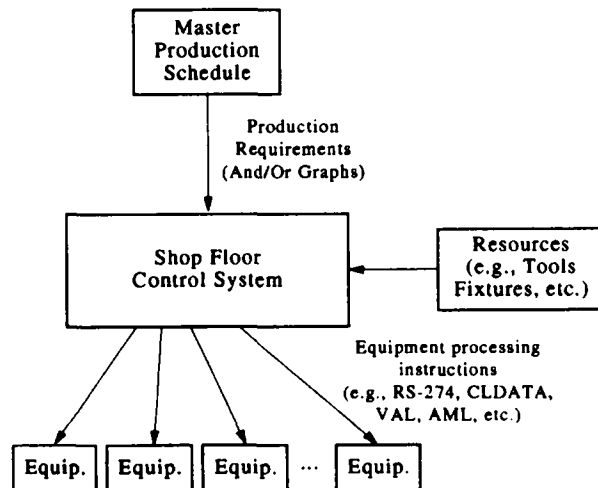


Fig. 1. Input/output diagram for the general shop floor control problem.

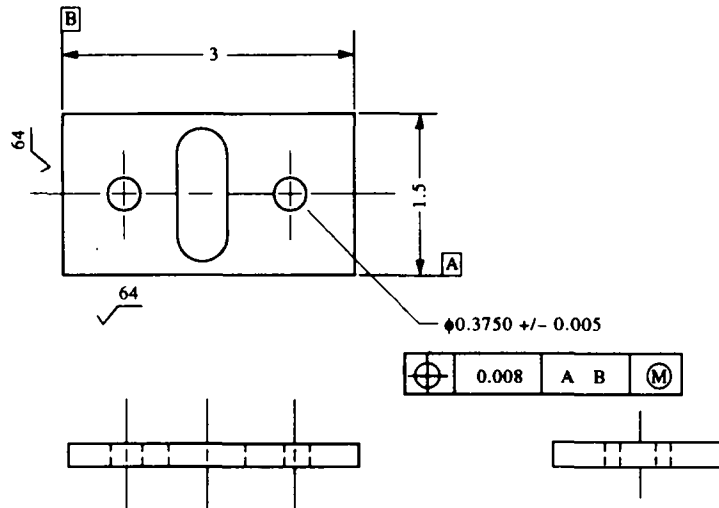


Fig. 2. Sample bracket.

Based on this structure, we define  $P = \{P_1, P_2, \dots, P_n\}$  as a set of  $n$  process plans (1 for each of  $n$  parts), where,  $P_i = \langle V_i, A_i \rangle$  for  $i = 1, 2, \dots, n$  is the process plan graph for part  $i$ .  $V_i$  is a finite set of nodes whose elements represent the individual processing steps for the part.  $A_i$  is a finite set of arcs whose elements represent precedence among processing steps. Unlike the traditional graph structure, there are two types of junctions connecting arcs and nodes in AND/OR graphs. OR junctions represent the typical case where *any one* arc within the junction may be taken out of the node. AND junctions represent the case where *all* arcs must be taken, but the sequence is unspecified [11]. So OR junctions present alternative *operations* and AND junctions present alternative *sequences* of fixed operations.

We will demonstrate the process plan representation using the relatively simple part (a bracket) shown in Fig. 2. The operation summary for the part is shown in Table 1. The operations summary indicates the required processes along with the equipment, tooling and fixturing required to perform the processes. It should be noted that operations 5a and 6a are alternative processes that can be used to finish the holes shown on the part (the holes are rough drilled and then either reamed or bored to create the finish). This operation summary is based on a manufacturing workstation which includes an NC drill, an NC machining center, and an industrial robot for loading/unloading the machines (as shown in Fig. 3). In the table, multiple lines for an operation represent alternatives for processing that operation. For example, operation 3 (twist drill hole 1) can either be performed by the machining center, or by the drill using either fixture 2 or fixture 3.

Figure 4 shows the feature precedence graph for the example part. For this part, the only feature precedents are that the holes be rough drilled before they are finished (reamed or bored). In the general case, there will be significantly more feature interactions. The important point here is that the representation not only represents precedence between operations, but also alternatives where applicable. These alternatives allow the planning and scheduling functions to incorporate up to date information when making processing decisions.

The administrative requirements included in the processing requirements include the due dates, priorities, and batch sizes for the parts. The SFCS uses these requirements to plan and schedule the production. The administrative requirements also provide some general scheduling philosophy to guide the SFCS. As an example, one general scheduling philosophy might be that meeting external customer due dates is more important than achieving high equipment utilizations. This general philosophy is used by the SFCS to select from alternative performance measures in the detailed scheduling phase.

The *resources* input describes the resources that are shared by all of the equipment within the shop. These resources include (but are not limited to) tooling, fixturing, maintenance, etc. Material transport and processing equipment is not included in this input even though these devices are shared resources from the point of view of the parts. Instead, this equipment is defined in a *factory*

Table 1. Operation task and resource summary for bracket arranged by feature

Oper.	Description	Feature	Feature Spec.	Time (Es.) (min)	Tooling	Machine	Fixture
1	Side Mill	Locat surf A	Mill Face A	0.6	1.0" endmill	Mach Cntr	Fxt F # 1 Fxt F # 2
2	Side Mill	Locat surf B	Mill Face B	0.35	1.0" end mill	Mach Cntr	Fxt F # 1 Fxt F # 2 Fxt F # 2 Fxt F # 2 Fxt F # 3
3	Twist drill	Hole # 1	Rough drl H # 1	0.33	0.3595" drill	Mach Centr	Fxt F # 2 Fxt F # 2 Fxt F # 3
5	Ream	Hole # 1	Finish H # 1	0.25	0.375" ream	Mach Cntr	Fxt F # 2 Fxt F # 2 Fxt F # 3
5a	Bore	Hole # 1	Finish H # 1	0.34	0.375" bore	Mach Cntr Drill	Fxt F # 2 Fxt F # 2 Fxt F # 3
4	Twist drill	Hole # 2	Rough drl H # 2	0.33	0.3595" drill	Mach Cntr Drill	Fxt F # 2 Fxt F # 3 Fxt F # 2 Fxt F # 2 Fxt F # 3
6	Ream	Hole # 2	Finish H # 2	0.25	0.375" ream	Mach Cntr	Fxt F # 2 Fxt F # 3
6a	Bore	Hole # 2	Finish H # 2	0.34	0.375" bore	Mach Cntr Drill	Fxt F # 2 Fxt F # 2 Fxt F # 3
7	Slotmill	Slot # 1	Mill Slot # 1	0.75	0.5" endmill	Mach Cntr	Fxt F # 1 Fxt F # 2

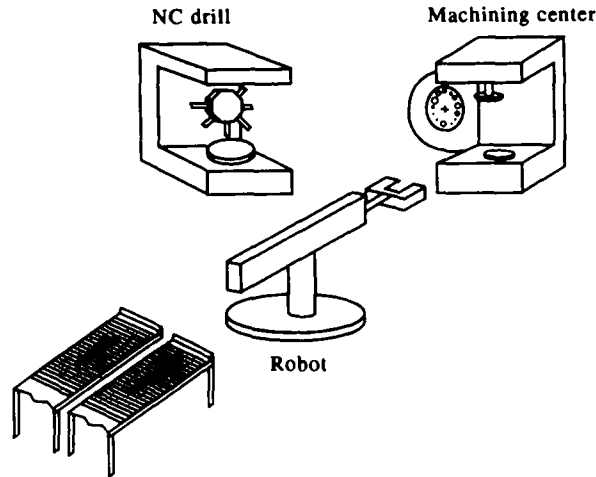


Fig. 3. Manufacturing system to illustrate control.

*model* (as described below) which is used to develop the SFCS. This distinction is made because the equipment is *controllable* and essentially permanent, whereas the shared resources are non-permanent or perishable and the individual requirements for these resources depend on the parts specified in the production requirements. Therefore the availability of the non-permanent resources (tools, fixtures, etc.) will change over a shorter time horizon than the availability of permanent resources (equipment), and we would like to create a control system independent of changes to non-permanent resource availability. In other words, introduction of new pallets, fixtures, cutting tools, etc., should not mandate extensive changes in the control system.

We define  $\mathbf{R} = \{R_1, R_2, \dots, R_s\}$  as a set of available resources, where  $R_i, i = 1, 2, \dots, s$ , describes a particular resource (e.g. a tool, or a fixture). Specification of a processing task includes a set of resources for  $\mathbf{R}$  which are required to perform that task.

The *factory model* describes the individual pieces of equipment within the shop, including the functionality and capabilities of each machine, and the relationships that exist between these individual pieces of equipment. The factory model is based on the equipment classification notation described by Smith [9]. We define  $\mathbf{E} = \{E_1, E_2, \dots, E_m\}$  as a set of controllable equipment, where  $E_i, i = 1, 2, \dots, m$ , is a specific piece of equipment. The factory model depends only on the physical equipment configuration and is independent of the set of parts currently being produced in the

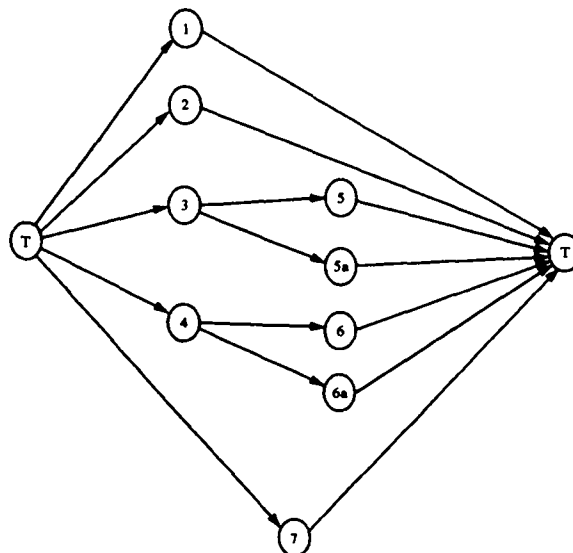


Fig. 4. Precedence graph for sample bracket.

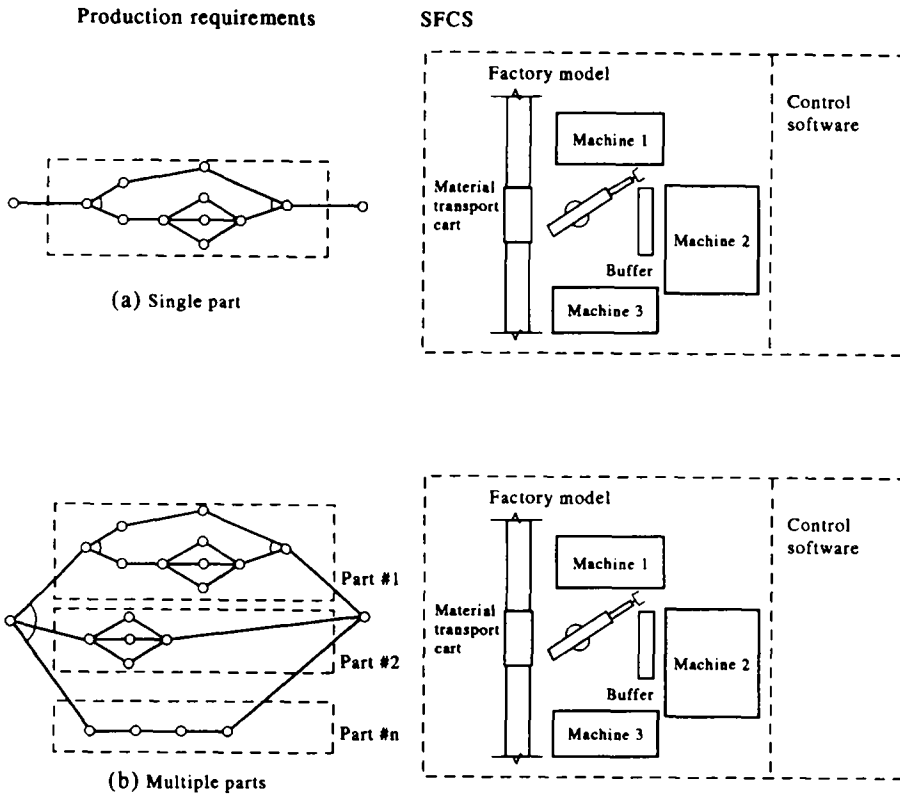


Fig. 5. Generalized shop floor control model.

shop. This independence is necessary since the product mix will change much more frequently than the physical equipment configuration.

Figure 5 illustrates the generalized model of the SFCS (the Resources input is not shown in this figure). In part (a) of this figure, a single part is to be produced. The process plan graph,  $P$ , defines the alternative processing sequences and describes the resource requirements for each processing step. Given the AND/OR graph  $P = \langle V, A \rangle$ :

$$v_j \in V,$$

and

$$v_j: \mathbf{E} \times \mathbf{R}^*, \quad j = 1, 2, \dots, j,$$

where

$$J = |V|.$$

Each node  $v$ , in the process plan graph identifies the specific piece of equipment which performs the corresponding processing task as well as the set of non-permanent resources required for the processing ( $\mathbf{R}^*$  represents zero or more elements from the set  $\mathbf{R}$ ).  $J$  is the total number of nodes in the graph. Arcs represent the precedence constraints between tasks. The set of equipment ( $\mathbf{E}$ ) is defined in the factory model described above. The alternatives for processing the part are represented using AND/OR junctions in the graph. As described above, the OR junctions represent alternative equipment which can perform a task, and AND junctions represent a series of tasks/equipment which must all be performed, but in no specific order. Therefore, any path through the graph represents a feasible processing route for the part. The objective of the controller is to select a "good" path through this graph based on the state of the shop and the current scheduling criteria and then generate the instructions for the physical equipment corresponding to each node on this path.

Figure 5(b) illustrates the more general case where multiple parts are assigned to the controller. In this case, all of the single parts' process plans are conjugated into a single graph by connecting

the individual part graphs at a single AND junction. This conjugated graph is called the *control graph* and it formalizes the requirement that all parts be processed by the controller, but does not impose an order to the processing. In the single part case, the control graph is simply that part's process plan graph. As with the single part case, the objective with the multiple part case is to find a good path through the graph and the represent and select the corresponding equipment instructions.

A key here is that process plan representation *can* be hierarchical in nature. In other words, sub graphs can be encapsulated inside single nodes to reduce the size of the graph. This modeling power can be exploited to represent the fact that decisions being made at one level of abstraction do not necessarily require the detailed operational characteristics needed at a lower level of abstraction. Therefore, these details can be encapsulated inside a single node. Plans and graphs of the plans may be aggregated based on the specific controller requirements. At this point we introduce the *task graph*. A task graph is an AND/OR graph which describes the processing requirements for the individual manufacturing features of a part. Within the control graph, the task graph is normally encapsulated inside the node for the equipment which produces the features. This concept will be more fully illustrated in our examples.

It should be noted that the control and task graphs for a high variety, multiple machine system can become exceptionally complex. Each part in a manufacturing system has its own process plan (task) graph derived from the ISO model of that part. For example, a system that contains 500 parts, each requiring ten operations, contains at least 5000 task nodes. Furthermore, if alternative plans are used to enhance the flexibility of these systems, the size of the graphs grow significantly. Graphs with tens of thousands of nodes connected by both AND or OR junctions would be common for a centralized supervisory controller of a low volume FMS.

The complexity of the process plan graphs will typically dictate the type of control architecture that will be used in a manufacturing system. An excessively complex graph would seem to dictate that some form of hierarchical control system would be in order. A reasonably straightforward graph could be controlled using most any type of architecture. We will illustrate our examples using the hierarchical control architecture of Joshi *et al.* [7], in order to illustrate that planning decisions can be aggregated based on a hierarchy. However, the hierarchical model is only used to simplify the presentation and the general characterization is independent of the control architecture.

### 3. FUNCTIONAL CONTROLLER

Previous research has suggested that the shop floor control function can be hierarchically decomposed such that each controller's functionality is partitioned into *planning*, *scheduling*, and *execution* tasks (see Fig. 6) [11, 14, 15]. Planning, in this model has been described as selecting the

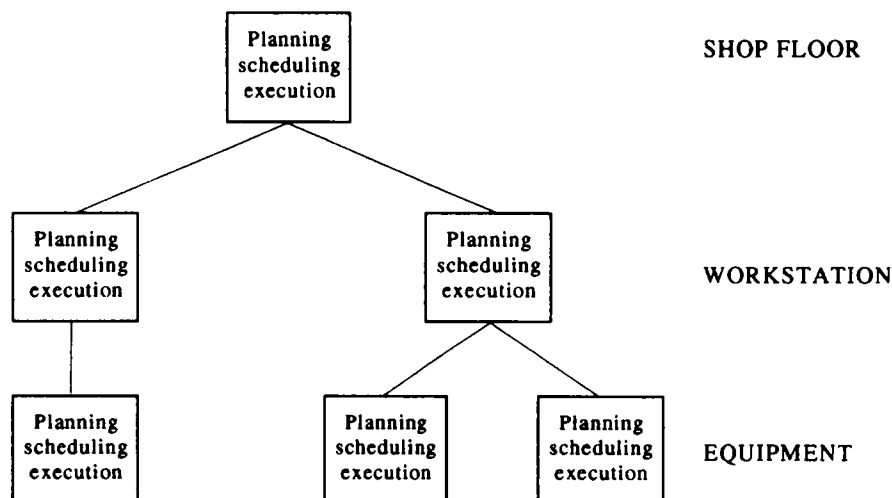


Fig. 6. Hierarchical control architecture.

tasks that the manufacturing system will perform. Scheduling then identifies a “good” sequence for these planned tasks based on some performance criteria. Execution performs the scheduled tasks through the direct interfaces with the physical equipment and other external (business) systems. Information flow within the controller during normal system operation occurs in a top-down fashion from planning, to scheduling, to execution. During error recovery, the information flow is reversed and goes from execution, to scheduling, to planning (bottom-up).

The partitioning between scheduling and execution activities has been described in detail in a previous paper [15]. The execution function is dependent only on the physical configuration of equipment, and is therefore relatively static. The scheduling function, however, is also dependent on the production requirements, and is therefore dynamic. Consequently, the performance criteria can change independently from the physical configuration in response to changes in the production requirements. By explicitly separating the scheduling and execution modules, different schedulers can be “plugged into” the execution module based on the production requirements. Using the formalism described above, the execution is responsible for performing the functions associated with each node in the process plan graph independent of the specific path through the graph. So for each  $v_j \in V$ , where  $v_j: \mathbf{E} \times \mathbf{R}^*$ , the execution module is responsible for verifying the availability of each element of  $R_p$  and for implementing the processing instructions on the device  $E_j$ .

The distinction between planning and scheduling, on the other hand, has remained convoluted throughout the manufacturing research community. Joshi *et al.* [7] and Jones and Saleh [14] provide similar qualitative distinctions between planning and scheduling, but neither provides a *formal* description of the distinction. Using the formalism described above, planning and scheduling together are responsible for determining a “good” path through the control graph. Once this path has been determined, execution is responsible for performing the individual tasks associated with each node in the path.

We partition the overall planning/scheduling problem as follows. Planning is responsible for selecting specific resources for the individual parts’ process plan graphs. By selecting the specific resources, we mean removing all of the OR junctions from the process plan graph for each part (“DeORing” the graph). This would typically be performed by using a solution technique for some variant of the resource assignment problem. Thus allocation of resources would remove the OR junctions from the graph. The input to scheduling is then a list of individual part processing routes with no operation alternatives, with all parts connected by a single AND junction. At this point, all resources have been *committed* to specific tasks, but the sequence and times of the allocation have not been specified.

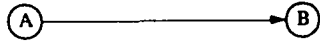
For example, it is not uncommon to have a queue of parts at a single equipment resource. In the case where a machinist is responsible for overseeing operations, the machinist chooses the sequence that is used to process the product(s). This is the function performed by our scheduler. In the case where several parts may be loaded onto a tombstone-type fixture or an automatic pallet exchanger, the processing equipment can select from any of the parts to begin processing. The ordering of the process can significantly affect the efficiency. To formally model this phenomenon, the planned tasks for each part are joined to a single source and sink node in order to create an AND graph of the sequence possibilities. This process is illustrated in Fig. 5(b). The scheduler uses this graph to make good sequencing decisions to order the parts (“DeANDING” the graph). The input to execution is then a linear sequence of tasks (a DeORED and DeANDed control graph).

#### 4. ILLUSTRATIVE EXAMPLES

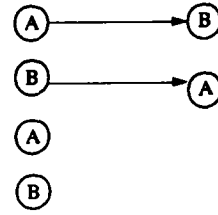
The functional model for planning, scheduling and execution described above must be able to accommodate virtually all control circumstances encountered in manufacturing. We would like to illustrate the model first with a couple of traditional scheduling examples (specifically Johnson’s and Jackson’s algorithms for the two machine problem) and then provide a more detailed problem consistent with those encountered in everyday manufacturing. The examples illustrated here will be described within the three-level hierarchical control architecture illustrated in Fig. 6 [7].

Johnson [16] and Jackson [17] provided optimal solutions for the two machine scheduling problem under several limiting assumptions, Johnson first posed a solution for problems where parts visit two machines in the same sequence (essentially a two machine flow shop). The digraph

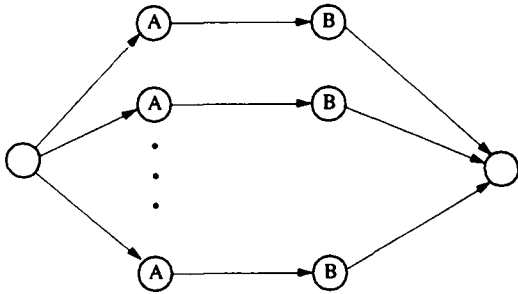




(a) Single part task graph

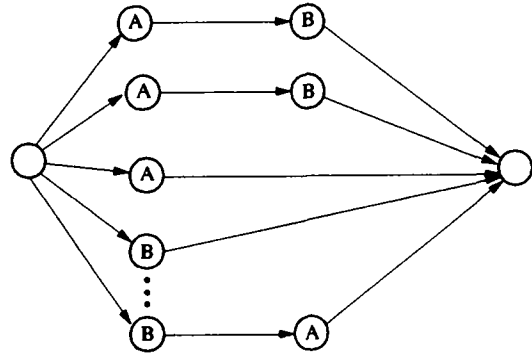


(a) Single part task graph



(b) Multiple part task graph

Fig. 7. Johnson's model task graph.



(b) Multiple part task graph

Fig. 8. Jackson's model task graph.

for a set of parts that comply with the assumptions of Johnson is shown in Fig. 7. As can be seen from the task graph in this figure, all equipment resources are determined explicitly in the initial graph. Planning, given the assumptions of Johnson, is essentially detailing the two tasks nodes for each part so that detailed processing times can be determined and conjuncting the individual part graphs with a source node and a sink node (with no OR junctions, there is no DeORing). Scheduling is then responsible for sequencing the individual tasks for each part by removing the single AND junction. Johnson's rule can be applied to develop the optimal sequence of tasks for all parts (under the assumptions of all material handling and setup times equal to zero and all parts available for processing at time zero).

Jackson's algorithm further developed the two machine problem to include parts that can be processed at both machines in either an A-B sequence or a B-A sequence, or on a single machine (essentially a two machine job shop). The task graph for a random set of parts for Jackson's rules is shown in Fig. 8. Again the sequence of operations at machines is fixed, and planning serves only to detail the tasks and provide a source and sink node for a cumulative parts graph.

The detail of both Jackson's and Johnson's rules is rather interesting in the functional context that we have developed. Their rules apply to an equipment level controller where parts would be sequenced via a pallet exchanger or similar part transfer mechanism at a processing station. Inter-equipment material handling is not considered, and is assumed to take zero time. While in a manual environment this might be a reasonable assumption, in an automated environment, all material handling tasks must be explicitly specified in order to control the system. In order to include material handling within the control framework, the process graph would have to be modified, and the resource model may also require change. In this case, the system becomes essentially a three machine system where a robot (or some other transfer device) moves parts from one machine to another, rather than a two machine system. The single-part graph for this modified Johnson's system is shown in Fig. 9 (*M*-nodes represent material handling operations). As can be



Fig. 9. Johnson's model task graph with material handling.

seen in the figure, a material handling operation is required between each processing operation. The material handler becomes another equipment resource that must also be scheduled in the system. Johnson's and Jackson's rules do not provide a means for sequencing systems of this type. Therefore, while Jackson and Johnson provide optimal solutions for the two machine case under certain restrictions, they do not provide the detail required to control these systems.

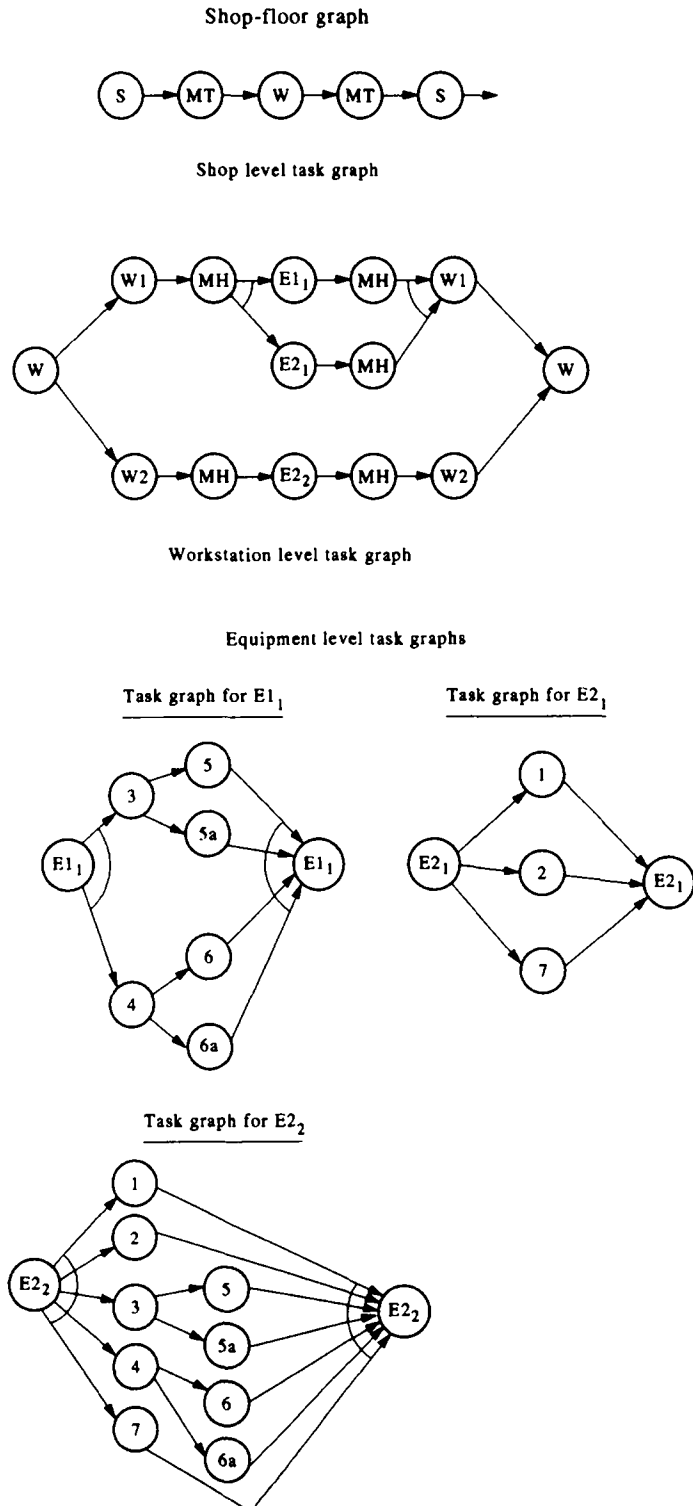


Fig. 10. Task graphs for part and system.

Table 2. A list of task/operations derived from the operation routing summary

Resource	Tasks/operations	Additional resources
E1 <sub>1</sub> -Drill	3, 4, 5, 5a, 6, 6a	Standard fixture (F2)
E2 <sub>1</sub> -MachCntr	1, 2, 7	Standard fixture (F1)
E2 <sub>2</sub> -MachCntr	1, 2, 3, 4, 5, 5a, 6, 6a, 7	Magnetic fixture (F3)

W1 = {E1, E2<sub>1</sub>}; W2 = {E2<sub>2</sub>}.

For our next example, we will illustrate a single part (with multiple pieces in the same batch) flowing through the manufacturing system shown in Fig. 3. This system consists of a single workstation containing three equipment level devices: an NC drill (E1), an NC machining center (E2), and a robot (E3).

The process plan task graphs for the part are shown in Fig. 10. MT nodes indicate inter-workstation material handling; while MH nodes denote intra-workstation material handling. The resource requirements for each node are not shown on the graph, but are described in Table 2. The equipment level task graph shows the alternative equipment operations (task sequences) that can be used and the workstation task graph shows the alternative equipment sequences (Workstation Options) for the part. The operation routing summary is used to develop detailed plans which appear in the plan graphs. There are several interesting aspects of this example. We will discuss how the functional framework that was presented can be used to develop the necessary shop-floor control required to produce a single part on a three machine manufacturing system (where the robot is considered to be a machine in this context).

Although the part and system being illustrated are rather simple, there are a number of interesting aspects associated with the part/system combination. For instance, we will assume that the part can be located and secured with a magnetic vise that makes all of the features accessible

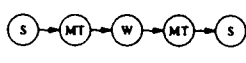

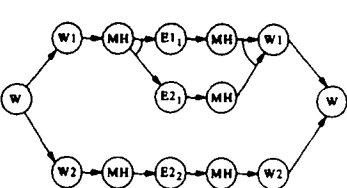
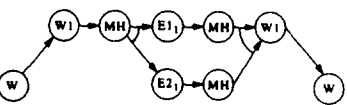
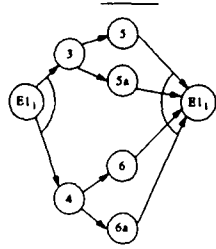
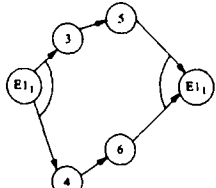
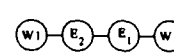

Activity	Shop	Workstation	Equipment
Planning	 <p>No alternatives appear in the graph. The graph need not be altered</p> <p>DeOR</p> 	 <p>Depending on the criteria and procedure(s), alternatives are removed from the graph.</p> 	<p>For E1</p>  <p>Depending on the criteria and procedure(s), alternatives are selected.</p> 
Scheduling	<p>No AND nodes here. Planning produced a serial graph.</p>	<p>Depending on the criteria and scheduling procedures, tasks must be sequenced (serialized).</p> 	<p>Depending on the criteria and scheduling procedures, tasks are sequenced at the machine.</p> 

Fig. 11. Control specifics for multiple parts in the sample system.

to the machine spindle. We will also assume that each machine has all the necessary tooling to produce the features. Even given both of these broad rather sweeping assumptions, the system has some rather interesting characteristics. We will begin by constructing the required control tasks for both the workstation and equipment system.

From Table 2 and Fig. 10, we can infer that all of the part's tasks can be performed by the machining center (Workstation Task 2 [W2]), or the drilling, reaming and/or boring can be performed by the NC drill after the machining center has performed the preliminary machining (Workstation Task 1 [W1]). If our manufacturing objective is to produce the parts as quickly as possible, we may want to maintain both possibilities, i.e. produce some parts only on the machining center and others on both the machining center and then on the NC drill. The route selected for any given part will be determined just prior to processing based on the shop status. The task graphs required for this breakdown are shown in Fig. 10 and conjuncted as defined by the physical workstation. The bottom-most graph of the figure (Workstation Task 2) illustrates producing the part only at the machining center. The graphs immediately above (Workstation Task 1) in the figure illustrate the sequence requirements to produce the part on the machine center and then on the NC drill. It should be noted that many other feasible visit possibilities exist which would entail more machine visits. These will not be illustrated.

We will begin our illustration of the decision making of the formal functional control system at the workstation level graph where our first decisions must be made. As can be seen from the workstation level task graph, the part can be produced using equipment E1 in conjunction with E2, or using E2 alone. The workstation planning function will decide which alternative to use. When the plan graph is disaggregated, Workstation Task 1 (W1) is composed of equipment tasks E1 and E2 which occur at two separate pieces of equipment. Workstation Task 2 (W2) only requires the machining center. We commit (plan) our resources based on some criteria and related procedures (e.g. resource utilization, workload balance, etc.). For the sake of our example we will fabricate a simplistic procedure to DeOR the workstation level graph as shown below.

```

procedure Workstation_DeOR
  if machine_center is idle and magnetic_fixture available
    use Workstation Option 2
  otherwise
    use Workstation Option 1
  endif
endproc

```

The control process continues as defined by the formalization—planning, then scheduling, and then execution. According to the hierarchical control structure, the shop plans are first formulated and scheduled to be executed on the workstation. The workstation then plans (DeORs) its tasks and then schedules (DeANDs) the tasks to be executed at the equipment. The schema is illustrated in Fig. 11. The planning (DeORing) and scheduling (DeANDing) procedure used is admittedly arbitrary. The intent here is to illustrate the functional separation and integration of these activities (nothing inhibits the system from planning and scheduling simultaneously). Sophisticated planning and scheduling procedures can be used to operate within this functional structure. They can even be evoked recursively. The key here is the functional structure.

The last characteristic that will be illustrated in the paper is one of scheduling complexity. Suppose that the milling operations required for the part require two different setups—one for the side milling and one for the slot milling (as might be the case if the magnetic vise were not available). We further redefine the equipment level task graphs and the workstation level tasks graphs as shown in Fig. 10. As can be seen from the figure, Workstation Task 1 W1 now consists of three tasks which we will define to require the fixed sequence  $E1_1-E2-E1_2$ . Parts following this sequence will first visit the machining center, then the NC drill, and finally the machining center, with the material handling being handled by the robot. The scheduling function now becomes far more complicated because if two parts are arbitrarily allowed in the system, a deadlock can occur [18]. For example, if the first part has completed task E1 and now resides at the NC drill. A second part with the same planned (DeORed) task graph is being processed (E2,) at the machining center. Since the robot can only pick-up a single part at a time, the system is in a state of deadlock. This

provides an interesting case in that the deadlock can either be avoided in the planning (selecting between Workstation Tasks 1 and 2) of an arriving part or through the scheduling of the parts.

## 5. CONCLUSION AND SUMMARY

In this paper, we have developed a formal functional control system for discrete part manufacturing systems. The formal functional model is predicated on process plans that can be represented as AND/OR digraphs as defined in the evolving ISO process plan model. We have also illustrated how this functional model can be used in a simple manufacturing environment. Although the illustrations presented in the paper are somewhat simple, we feel that the framework that has been created defines a formalization of shop-floor control previously missing in manufacturing research. Furthermore, the model is "scaleable", in that it can be directly applied to more complex systems. For a more detailed discussion of the scaleability of the graphical process plan representation and the factory resource model, see Lee *et al.* [19] and Wysk *et al.* [20]. We feel that this formal model can be used as the platform to coordinate shop-floor control research and development; thereby allowing new tools and research to plug directly into the control system. The contribution is in the ease and elegance that the model provides for both finite state/automata activities as well as the production engineering elements, e.g. planning, scheduling and process plan representation.

## REFERENCES

1. J. Albus, A. Barbera and N. Nagel. Theory and practice of hierarchical control. *Proceedings of the 23rd IEEE Computer Society International Conference*, Washington D.C., pp. 18–39 (1981).
2. A. T. Jones and C. R. McLean. A proposed hierarchical control architecture for automated manufacturing systems. *J. Manuf. Syst.* **5**, 15–25 (1986).
3. D. Beeckman. CIM-OSA: computer integrated manufacturing—open systems architecture. *Int. J. Computer Integr. Manuf.* **2**, 94–105 (1989).
4. A. W. Naylor and R. A. Volz. Design of integrated manufacturing control software. *IEEE Trans. Syst. Man. Cybernet.* **SMC-17**, 881–897 (1987).
5. N. A. Duffie and R. S. Piper. Non-hierarchical control of a flexible manufacturing cell. *Robot. Computer Integr. Manuf.* **3**, 175–179 (1987).
6. E. G. Mettala. Automatic generation of control software in computer integrated manufacturing. Ph.D thesis, The Pennsylvania State University (1989).
7. S. B. Joshi, R. A. Wysk and A. Jones. A scaleable architecture for CIM shop floor control. *Proceedings of Cimcon '90* (Edited by A. Jones), pp. 21–33. National Institute of Standards and Technology (1990).
8. M. K. Senehi, E. Barkmeyer, M. Luce, S. Ray, E. Wallace and S. Wallace. Manufacturing systems integration initial architecture document. National Institute of Standards and Technology, NIST Interagency Report NISTIR 4682, Gaithersburg, MD (1991).
9. J. S. Smith. A formal design and development methodology for shop floor control in computer integrated manufacturing. Ph.D. thesis, The Pennsylvania State University (1992).
10. W. Davis and A. Jones. A functional approach to designing architectures for CIM. *IEEE Trans. Syst. Man. Cybernet.* **19**, 164–189 (1989).
11. S. Joshi, E. G. Mettala and R. A. Wysk. Formal models for control of flexible manufacturing cells: physical and system model. *IIE Trans.* **24**, No. 3 (1992).
12. B. A. Catron and S. R. Ray. ALPS: a language for process specification. *Int. J. Computer Integr. Manuf.* **4**, 105–113 (1991).
13. H. Cho, A. Derebail, T. Hale and R. A. Wysk. A formal approach to integrating computer aided process planning and shop floor control. *J. Engng Ind.* Nov. (1993).
14. A. T. Jones and A. Saleh. A multi-level/multi-layer architecture for intelligent shopfloor control. *Int. J. Computer Integr. Manuf.* **3**, 60–70 (1990).
15. J. S. Smith, W. C. Hoberecht and S. B. Joshi. A shop floor control architecture for computer integrated manufacturing. Industrial Engineering Working Paper #INEN-MS-WP-13-11-92, Texas A & M University, College Station (1992).
16. S. M. Johnson. Optimal two and three stage production schedules with setup times included. *Nav. Res. Logist. Q.* **1**(1) (1954).
17. J. R. Jackson. Networks of waiting lines. *Opns Res.* **5**, 518–521 (1957).
18. R. A. Wysk, N. S. Yang and S. Joshi. Detection of deadlocks in flexible manufacturing systems. *IEEE Trans. Robot. Automat.* **7**, 853–859 (1991).
19. S. Lee, R. A. Wysk and J. S. Smith. Process planning interface for a shop floor control architecture for computer integrated manufacturing. *Int. J. Prod. Res.* In press.
20. R. A. Wysk, B. A. Peters and J. S. Smith. A formal process planning schema for shop floor control. Texas A & M Industrial Engineering Department Working Paper Series (1994).