

July, 1987

LIDS-P-1682

**A HIERARCHICAL FRAMEWORK
FOR DISCRETE EVENT SCHEDULING
IN MANUFACTURING SYSTEMS**

by

Stanley B. Gershwin

Massachusetts Institute of Technology
Laboratory for Information and Decision Systems
Cambridge, Massachusetts

*Presented at the
IIASA Workshop on Discrete Event Systems:
Models and Applications*

*August 3-7, 1987
Sopron, Hungary*

A HIERARCHICAL FRAMEWORK FOR DISCRETE EVENT SCHEDULING IN MANUFACTURING SYSTEMS

Stanley B. Gershwin

Massachusetts Institute of Technology, Cambridge, Massachusetts, USA
Boston University, Boston, Massachusetts, USA

1. INTRODUCTION

Operating policies for manufacturing systems must respond to important discrete events that occur during production such as machine failures, setups, demand changes, expedited batches, etc. These feedback policies must be based on realistic models, and they must be computationally tractable. In this paper, we develop a class of hierarchical scheduling and planning algorithms whose structure is systematically based on the characteristics of the specific kind of production that is being controlled. The levels of the hierarchy correspond to classes of events that have distinct frequencies of occurrence.

Computational tractability is an important concern because of the complexity of the system. Even for a very small, deterministic, idealization of a production system, the computational effort for combinatorial optimization renders it impractical for real-time control. Any control scheme must be based on a simplified representation of the system and a heuristic solution of the scheduling problem.

There have been many hierarchical scheduling and planning algorithms, some quite practical and successful. However, there has been no systematic justification of this structure. The main contribution of this paper is a framework for studying and synthesizing such a structure.

This work extends a formulation by Kimemia and Gershwin (1983) in which only two kinds of events were considered: production operations on parts and failures and repairs of machines. Operations occurred much more often than failures, and this allowed the use of a continuous representation of material flow. A dynamic programming formulation led naturally to a feedback control policy. The state of the system had two parts: a vector of real numbers represented the surplus, the cumulative difference between production and requirements. The discrete part of the state represented the set of machines that are operational. The object was to choose the production rate vector as a function of the state to keep the surplus near 0.

The production rate (the continuous control variable) was restricted by linear inequality constraints that depended on the repair state. They represented the instantaneous capacity of the system, and they expressed the idea that no machine, while it is operational, may be busy more than 100% of the time; and no machine, while it is not operational, may be used at all. The present paper describes the extension of this work to the widest possible variety of phenomena and decisions in a manufacturing environment.

Figure 1.1 illustrates some of the issues that are considered here. It is a graph of the cumulative production and demand for one part type (j) among many that share one machine. A long term production rate (u_j^1) is specified for this part type, and its integral is represented by the solid straight line. It is not possible to follow this line exactly because the machine is set up for Type j parts only during a set of time intervals. During such intervals, the medium term production rate u_j^2 must be greater than u_j^1 , because during the other intervals -- while it is set up for other parts -- u_j^2 is 0. The integral of u_j^2 (the dashed line) is staircase-like, close to the integral of u_j^1 .

The dashed line cannot be realized either. The machine is unreliable, and while it is down, its production rate u_j^3 is 0. Consequently, while it is up and set up for Type j , it must be operated at a short term rate u_j^3 greater than that of the dashed line. The dotted line, which represents this phenomenon, is again staircase-like, and is close to the dashed line. Finally, the actual cumulative production graph (which requires too much resolution to be plotted) is a true staircase. It has vertical steps at the instants when parts are loaded, and it is flat otherwise. It is very close to the dotted line.

This paper formalizes this hierarchy, and extends it to an arbitrary number of levels and several machines.

Literature Survey

There is a large literature in scheduling (Graves, 1981). Many papers are based on combinatorial optimization/integer programming methods (Lageweg, Lenstra, and Rinnooy Kan, 1977 and 1978; Papadimitriou and Kannelakis, 1980) or mixed integer methods (Afentakis, Gavish, and Karmarkar, 1984; Newhouse, 1975a and 1975b; Wagner and Whitin, 1958). Because of the difficulty of the problem, authors are limited to analyzing computational complexity, or proposing and analyzing heuristics.

An important class of problem formulations is that of hierarchical structure (Bitran, Haas, and Hax, 1981; Dempster et al., 1981; Graves, 1982; Hax and Meal, 1975; and others). The goal is to replace one large problem by a set of many small ones because the latter is invariably easier to solve. These methods are often used but there is no general, systematic way of synthesizing hierarchies for large classes of stochastic scheduling problems.

Multiple time scale problems have recently been studied in the control theory (Sakseena, O'Reilly, and Kokotovic, 1984) and Markov chain literature (Delebecque, Quadrat, and Kokotovic, 1984). We use insights from these methods to develop a systematic justification for hierarchical analysis. This paper also makes use of, and extends the work of Kimemia and Gershwin (1983). A recent survey (Maimon and Gershwin, 1987) describes this and several related papers.

Outline

Section 2 describes the manufacturing systems that we are considering. It establishes terminology and discusses the basic concepts for the present approach: capacity and frequency separation. Section 3 builds on the frequency separation to derive a small set of results that form the foundation of the hierarchy. Control in the hierarchy is described in detail in Section 4. Sections 5 and 6 present the two building blocks: the staircase strategy and the hedging point strategy. A simple example appears in Section 7, and conclusions are drawn in Section 8.

2. PRODUCTION EVENTS AND CAPACITY

In this section, we discuss the discrete events that occur during the production process. We define terminology to help describe these events. We categorize events in two ways: the frequency with which they occur; and the degree of control that decision-

makers can exert over them. We define capacity, and show how capacity is affected by production events.

2.1 Definitions

A *resource* is any part of the production system that is not consumed or transformed during the production process. Machines -- both material transformation and inspection machines, workers, pallets, and sometimes tools -- if we ignore wear or breakage -- can be modeled as resources. Workpieces and processing chemicals cannot.

An *activity* is a pair of events associated with a resource. The first event corresponds to the start of the activity, and the second is the end of the activity. Only one activity can appear at a resource at any time. For example, the *operation* of drilling the 3/8" hole in part type 12, serial number 947882927 that started at 10:03 this morning and ended at 10:07 is an activity. Other examples include machine failures, setups (i.e., changes of tooling, etc.), preventative maintenance, routine calibration, inspection, and training sessions. We use the same term to refer to a set of such pairs of events. For example, drilling 3/8" holes in type 12 parts is an activity; specifically, an operation.

Let i be a resource and j an activity. Define $\alpha_{ij}(t)$ to be the state of resource i . This is a binary variable which is 1 if resource i is occupied by activity j at time t , and 0 otherwise. Since at most only one activity may be present at a resource at a given time,

$$\sum_j \alpha_{ij}(t) \leq 1. \text{ for all } i \quad (1)$$

Every activity has a *frequency* and a *duration*. To define frequency, let $N_{ij}(T)$ be the total number of times that resource i is occupied by activity j in $(0, T)$. Then define

$$u_{ij} = \frac{1}{T} N_{ij}(T). \quad (2)$$

This is the frequency with which type j activities occur at resource i .

In the following, we do not indicate a resource (i) explicitly in the subscript of u . This allows the flexibility of either considering the index j to include a specific resource (in which case j might mean "operation 30 on a Type 12 part at Machine 5") or any resource (in which case j might mean "the required operation on a Type 12 part at the current machine"). When u has only the j subscript, (2) holds only when activity j actually takes place at resource i . If it does not take place at resource i , (2) is meaningless, and if it takes place at more than one resource i , it must hold for each i . (This implies a "conservation of flow" condition, since $u_j = u_{1j} = u_{2j}$ if j goes to both resource 1 and resource 2.)

The vector u is the *activity rate vector*. It satisfies $u_j \geq 0$. Let τ_{ij} be the average duration of activity j at resource i . Then τ is the *activity duration matrix*. It satisfies $\tau_{ij} \geq 0$. (We can now say that (2) holds only when $\tau_{ij} > 0$.) Durations may be random or deterministic, but we assume that they are not under the control of the decision-maker.

Observation: If the system is in steady state,

$$\tau_{ij} u_j = E\alpha_{ij} \quad (3)$$

Proof:

Consider a sample history of the system. The total time that resource i is occupied by activity j in $(0, T)$ is

$$\int_0^T \alpha_{ij}(t) dt. \quad (4)$$

The average duration satisfies

$$\tau_{ij} = \frac{\int_0^T \alpha_{ij}(t) dt}{N_{ij}(T)} = \frac{\frac{1}{T} \int_0^T \alpha_{ij}(t) dt}{u_j}. \quad (5)$$

If the system is in steady state, then the time average of a quantity is the same as its expected value, so the numerator is $E\alpha_{ij}$ and (3) is proved. (This can also be viewed as an instance of Little's law.) The assumption that the system is in steady state is an important one. In later sections, the dynamics of the system is divided into subsets, each considered over different time scales. Each subset has a different time period for steady state.

Since only one activity may occur at a resource at one time, the fraction of resource i 's time that is spent on activity j is $\tau_{ij}u_j$. This is called the *occupation* of resource i by activity j .

Example: Type 1 parts arrive at Machine 1 at a rate of 1 per hour (u_1). They undergo operations that take 20 minutes (τ_{11}). Therefore Machine 1 is occupied by making Type 1 parts for 1/3 of its time.

2.2 Capacity

From (1),

$$1 \geq E \sum_j \alpha_{ij}(t) = \sum_j \tau_{ij}u_j \text{ for all resources } i. \quad (6)$$

This is the fundamental capacity limitation: no resource can be occupied more than 100% of the time.

Example: In addition to the Type 1 parts, we wish to send Type 2 parts to Machine 1 for an operation that takes 25 minutes (τ_{12}). There is a demand of one Type 2 part every 35 minutes (u_2). This is not possible because it violates (6).

The set of all activity rate vectors u that satisfies (6) is the *capacity set* Ω . It is important to observe that *capacity is a set* -- a polyhedron -- and not a scalar. Here we have defined capacity as a constant set. In later sections, capacity is described as a function of the state of the system. This means that *capacity is a time-varying, stochastic set*.

2.3 Frequency Separation

Dynamic models always have two parts: a constant part and a time-varying part. In all dynamic models, there is something that is treated as unchanging over time: some parameters, and, most often, the structure of the model. For example, the model described in Sections 2.1 and 2.2 is a conventional one in which there are static quantities (u_j , τ_{ij}), a static structure, and dynamic quantities ($\alpha_{ij}(t)$, $N_{ij}(t)$).

Recently, the dichotomy between static and dynamic has been extended to systems with multiple time scales, modeled as differential equations or Markov chains. At one end of the scale, there are quantities that are treated as static. The other variables are divided into groups according to the speed of their dynamics. Because of this grouping, it is possible to simplify the computation of the behavior of these systems. Approximate but accurate techniques have been developed to calculate the effects of the slower and faster dynamics of adjacent groups on each group of variables.

The essential idea is: when treating any dynamic quantity, treat quantities that vary much more slowly as static; and model quantities that vary much faster in a way that ignores the details of their variations (such as replacing fast-moving quantities by their averages; or by Brownian noises with appropriate parameters.) This is the central assumption of the hierarchical decomposition presented here.

Assumption 1: The activities can be grouped into sets J_1, J_2, \dots such that for each set J_k , there is a characteristic frequency f_k satisfying

$$f_1 \ll f_2 \ll \dots \ll f_k \ll f_{k+1} \ll \dots \quad (7)$$

The activity rates satisfy

$$j \in J_k \Rightarrow f_{k-1} \ll u_j \ll f_{k+1}. \quad (8)$$

Figure 2.1 represents two kinds of production that satisfy this assumption. The horizontal axis represents frequency and the vertical axis represents occupation of some critical resource. Because of Assumption 1, all the event frequencies occur at distinct clusters.

The time period over which a component of the system reaches steady state depends on the frequency classes of the activities that affect that component. It is on the order of $1/f_{k-1}$ if the lowest frequency activity is a member of J_k .

A capacity set can be associated with each time scale k . Consequently, *capacity is a set of time-varying, stochastic sets.*

2.4 Slow variation

In 2.1 and 2.2, u_j is treated as constant. However, it is convenient to allow u_j to be slowly varying. That is, u_j is not constant, but it changes slowly compared to the changes in α_{ij} . An important special case is where u_j is piecewise constant, and its changes occur much less often than those of α_{ij} . Equation (3) is now

$$\tau_{ij} u_j(t) = E\alpha_{ij}(t). \quad (9)$$

This is established in the same manner as (3), but the bounds of the integral (4) are t_1 and t , where t_1 is the time of the most recent change in u_j , and t is the current time. The quantity $u_j(t)$ satisfies

$$N_{ij}(t) = \int_0^t u_j(s) ds \text{ for } \tau_{ij} > 0, \text{ or}$$

$$N_{ij}(t) - N_{ij}(t_1) = \int_{t_1}^t u_j(s) ds = (t - t_1)u_j(t_1). \quad (10)$$

The assumption here is that many occupations of resource i by activity j occur in the interval (t_1, t) : enough so that

$$E\alpha_{ij}(t) = \frac{1}{t-t_1} \int_{t_1}^t \alpha_{ij}(s) ds. \quad (11)$$

2.5 Degree of control

Events may or may not be under the control of the decision-maker. Figures 2.2-2.4 represent a variety of activities with different degrees of control. Figure 2.2 shows the two repair states of a machine: operational and down. In this case, the times at which the transitions occur are beyond the control of the production personnel.

Figure 2.3 represents the operation states of a flexible machine. It can work on a family of four parts, and setup is not required. That is, after doing an operation on one part, the time required to do an operation on another part depends only on the new part, and not the identity of the part that preceded it. While the machine is in the idle state, it may be used to do an operation on any of the parts. When to make the transition, and what state to visit next, are entirely at the discretion of the manager. Once that decision has been made, however, the manager loses control. The time required to perform the operation may or may not be known, but it cannot be chosen, and the next state must be the idle state.

Figure 2.4 displays the configuration states of a machine which can do operations on three families of parts. There is a substantial setup time to switch the machine from operations on one family to another. While the system is set up for any one family, it can remain that way indefinitely. The manager can choose when to switch out of the current family and which family to switch into next. However, the system then goes to the appropriate setup state. (While it is there, tools are changed, calibration is performed, test parts are made, etc.) It stays in that state for a length of time which is not under the control of the manager. (Again, it may or may not be known, but it cannot be chosen.) After that, the system goes to the new family state, and the series of events repeats.

2.6 Effects of events

The goal of the factory is to produce in a way that satisfies demand at least cost. The only events that directly further this goal are the production events, and only if they are chosen correctly. The direct effects of all the other events work against this goal.

When any activity occurs, it prevents all other activities from occurring at the same resource. Thus a low frequency, high occupation activity is a major disruption to the system. During such an activity, the resource it occupies is unavailable for a very long time (as seen by the high frequency events). This may not simply shut down all production; instead, it may temporarily restrict only some kinds of production. Such disruptions greatly complicate the scheduling problem.

2.7 Purpose of the decomposition

It is possible to represent the scheduling problem as an integer programming problem, particularly if time is discretized. However, this almost always leads to a problem which cannot practically be solved even in the absence of random events. The goal of the approach described below is to formulate the problem in a way that will provide an approximate feedback solution for the stochastic scheduling problem.

The solution approach is based on a reformulation of the problem in which the large set of binary variables that indicate the precise times when events occur is replaced by a small set of real variables representing the rates that events occur. This is a good approximation because of the large difference in frequencies of these events. Eventually, the binary variables are calculated, but by a much simplified procedure.

3. THE SPECTRUM AND THE HIERARCHY

In this section, we define the variables of the hierarchy and what calculations take place at each of the levels. In the following sections, we propose problem formulations for those calculations.

3.1 Definitions

The structure of the hierarchy is based on Assumption 1: that events tend to occur on a discrete spectrum. Classes of events have frequencies that cluster near discrete points

on the spectrum. The control hierarchy is tied to the spectrum. Each level k in the hierarchy corresponds to a discrete point on the spectrum and thus to a set of activities. This point is the characteristic frequency f_k (and $1/f_k$ is the characteristic time scale) of those activities.

At each level of the hierarchy, events that correspond to higher levels (i.e., lower frequencies, and lower values of k) can be treated as discrete and constant or slowly varying. Events that correspond to lower levels can be described by continuous (real) variables. These variables can be treated as though they are deterministic.

The approach is to define a set of rate or frequency variables for every activity. These quantities represent the behavior of the system in an aggregated way. At each level, we calculate optimal values for those aggregate variables. Optimal, here, means that they must be close, on the average, to the corresponding values chosen at the higher levels. However, they must respond to events that occur at their level.

Define the *level* $L(j)$ of activity j to be the value of k in Assumption 1 associated with this activity. That is,

$$L(j) = k \text{ if } j \in J_k \quad (12)$$

in (8). We choose the convention that less frequent activities are higher level activities and have lower values of k ; lower levels have higher values of k .

In the following we define Level k quantities. These are values of system states as perceived by an observer who is not able to distinguish individual events -- that is, changes in α -- that happen much more frequently than f_k . The frequencies of high frequency events, as perceived by this observer, depend on the current states of low frequency activities, and expectations must similarly be conditioned on the current states of low frequency activities.

Let $\alpha_{ij}^k(t)$ be the *level k state of resource i* . This is defined only for activities j whose level is k or higher. If $L(j) \leq k$,

$$\alpha_{ij}^k(t) = \alpha_{ij}(t). \quad (13)$$

Define α^k as the vector whose components are α_{ij}^k . Its dimensionality depends on k .

Let E_k be the *level k expectation operator*. It is the conditional expectation, given that all level m quantities ($m \leq k$) remain constant at their values at time t . That is, we treat $\alpha_{ij}^m(t)$ as constant.

Let u_j^k be the *level k rate of activity j* . This is defined only if the level of activity j is lower than k , that is, $L(j) > k$. The level k rate of activity j is the frequency that a level k observer would measure that activity j occurs while all level m events ($m \leq k$) are held constant at their current values. This rate satisfies

$$u_j^k(t) = \frac{E_k \alpha_{ij}(t)}{r_{ij}} \quad (14)$$

and

$$u_j^k(t) \geq 0. \quad (15)$$

The conditioning event of E_k is a subset of that of E_{k-1} . This is because the set of quantities held constant for E_{k-1} is a subset of that for E_k . Consequently,

$$E_{k-1} E_k \alpha_{ij} = E_{k-1} \alpha_{ij}. \quad (16)$$

Taking the level $k-1$ expectation of (14):

$$E_{k-1}u_j^k = E_{k-1} \frac{E_k \alpha_{ij}(t)}{\tau_{ij}}. \quad (17)$$

But this is equal to $\frac{E_{k-1} \alpha_{ij}(t)}{\tau_{ij}}$ according to (16). This implies that

$$E_{k-1}u_j^k = u_j^{k-1}. \quad (18)$$

That is, the level $k-1$ rate of an activity is the level $k-1$ expectation of the level k rate of the activity. This is a very important observation, because it relates quantities at different levels of the hierarchy.

If $L(j) > k$, level k of the hierarchy calculates u_j^k . How that calculation is performed depends on the degree of control of activity j . If activity j can be initiated by the decision-maker rather than by nature, then u_j^k is chosen to satisfy (18).

All activities j appear in three different guises in the hierarchy. At their own level ($k=L(j)$), they appear as pairs of discrete events (the start and the end of the activity). This is, of course, exactly what they are. No approximate representation is possible. At higher levels in the hierarchy ($k < L(j)$), however, their details are ignored, and they are represented by rates (u_j^k). At lower levels ($k > L(j)$), they are treated as constant at their current values.

Controllable activities are chosen from top down. That is, a rate u_j^1 is chosen initially. Then ($k > 1$) is chosen to satisfy (18) and other conditions (according to the *hedging point strategy* of Section 6) for increasing values of k until $k = L(j)$. At that point, α_{ij} is chosen to satisfy (14) according to the *staircase strategy* described in Section 5.

On the other hand, (14) and (18) have different interpretations when activity j is not controllable (for example, machine failures). In that case, the expectations are statistical operations, in which data are collected and sample means are found. The rate $u_j^{L(j)-1}$ is calculated from (14) by observing the values of α_{ij} . If $L(j) < k-1$, (18) is repeated for decreasing values of k .

3.2 Capacity in the hierarchy

For each k , the sum in (1) can be broken into two parts:

$$\sum_{L(j) > k} \alpha_{ij} \leq 1 - \sum_{L(j) \leq k} \alpha_{ij}^k \quad (19)$$

in which (13) is applied to the high-level sum on the right side. If we take a level k expectation of (19), the right side is not affected. From (14),

$$\sum_{L(j) > k} \tau_{ij} u_j^k \leq 1 - \sum_{L(j) \leq k} \alpha_{ij}^k \quad (20)$$

This equation is the basic statement of capacity in the hierarchy. It limits the rates at which low level events can occur as a function of the current states of high level events. If any high level activity is currently at resource i , that resource is not available for any low level events. In that case, the right side of (20) is 0 and all u_j^k that have a positive coefficient must be zero. If none of the high level activities in (20) are currently taking place, this inequality becomes

$$\sum_{L(j) > k} \tau_{ij} u_j^k \leq 1. \quad (21)$$

Capacity is thus a function of hierarchy level and, since it depends on the state of the system, a stochastic function of time. We define the *level k capacity set* as

$$\Omega^k(\alpha^k) = \left\{ u^k \mid \sum_{L(j)>k} \tau_{ij} u_j^k \leq 1 - \sum_{L(j) \leq k} \alpha_{ij}^k \quad \forall i; \quad u_j^k \geq 0 \quad \forall j, L(j) > k \right\}. \quad (22)$$

This set is the constraint on the hedging point strategy. It limits the choice of rates u^k as a function of the current state of the system. Note that the condition

$$u^k \in \Omega^k(\alpha^k) \quad (23)$$

is a necessary but not sufficient condition. That is, $\Omega^k(\alpha^k)$ was constructed so that every sequence of events must satisfy (23). However, we have not demonstrated that for every u^k that satisfies (23) there corresponds a feasible sequence of events. We assume sufficiency in the following, however.

4. CONTROL IN THE HIERARCHY

The goal of the hierarchical scheduler is to select a time to initiate each controllable event. This is performed by solving one or two problems at each level k . We emphasize control -- i.e., scheduling and planning -- here. Data-gathering and processing is also an important function of the hierarchy, but is not discussed in this paper. The hierarchy is illustrated in Figure 4.1.

Problem 1: (The hedging point strategy)

Find u_j^k (for all $j, L(j) > k$) satisfying (18) and (20) (and possibly other conditions).

Problem 2: (The staircase strategy)

Find α_{ij}^k (for all $j, L(j) = k$) satisfying

$$E_{k-1} \alpha_{ij}^k = \tau_{ij} u_j^{k-1} \quad (24)$$

(and possibly other conditions).

At the top level of the hierarchy ($k=1$), required rates of some of the controllable activities are specified, for example, production rates and maintenance frequencies. Other rates may not be specified, such as setup frequencies. We assume that rates of uncontrollable events are known. The frequency associated with the top level is 0. Consequently, there is no Problem 2 at that level, and Problem 1 reduces to a static optimization. The function of Problem 1 here is to choose all the rates that were not specified. The vector u^1 is the target rate vector for level 2.

If there are any controllable events at level $k > 1$, we solve Problem 2. (An example is the change in setup of a machine.) Controllable events are thereby initiated in such a way that their rates of occurrence are close to the target rates that are determined at level $k-1$.

Then we solve Problem 1 to determine the level k rates of occurrence u_j^k of all activities j whose frequencies are much higher than f_k . These rates are refinements of the target rates determined at level $k-1$: u_j^{k-1} . They differ from the higher level rates in that they are affected by the level k discrete events. These events, if they are controllable, were chosen by Problem 2 at this level. However, even if the level k events are not controllable, the level k rates differ from the higher level rates. These rates are then the targets for level $k+1$.

For example, if at level k we choose setup times, the production rates must be calculated so that they are appropriate for the current setup. If we are making Type 1 parts

at the rate of 4 per day, but the necessary machine is only set up for that part on Tuesdays, then we must work at a rate of 20 per day on Tuesday and 0 Type 1 parts per day during the rest of the week.

Similarly, the activities associated with level k may not be controllable, such as machine failures. It is still necessary to refine the production rates. If the overall requirements for Type 1 parts are 20 per day, and the machine is down 10% of the time, *and failures occur several times per day*, then the appropriate strategy is to operate the machine at a rate of 22.2 parts per day while it is up. Note that this only makes sense if failures are much more frequent than setups and much less frequent than operations. If not, related but different calculations must be performed in a different order. That is, a different hierarchy is appropriate.

An important feature of this hierarchy is that rates u_j^k are always chosen to be within the current capacity of the system. When a level m event occurs ($m \leq k$), the capacity set (22) changes. Problem 2 is then recalculated so that the new rates remain feasible. As mentioned earlier, this is necessary for feasibility. In all the simulation experiments that we have performed, it appeared to be sufficient as well.

5. THE STAIRCASE STRATEGY

The staircase strategy was introduced by Gershwin, Akella, and Choong (1985) and Akella, Choong, and Gershwin (1984), although stated somewhat differently from here. It was used to load parts in a simulation of a flexible manufacturing system.

Instead of treating the statement of Problem 2 in Section 4 directly, we choose starting times for events α_{ij}^k to satisfy (2), or rather

$$N_{ij}^k(t) \sim \int_0^t u_j^k(s) ds \quad (25)$$

where $N_{ij}^k(t)$ is the number of times activity j occurs at resource i during $[0, t]$. This expression is only approximate because the left side is an integer and the right side is a real number. The objective is to develop an algorithm which keeps the error in (25) less than 1. This is because, approximately,

$$E_{k-1} \alpha_{ij}^k(t) = \frac{1}{t-t_1} \int_{t_1}^t \alpha_{ij}(s) ds = \frac{1}{t-t_1} (N_{ij}^k(t) - N_{ij}^k(t_1)) \tau_{ij} \quad (26)$$

in steady state. If the times to start activities are chosen to satisfy (25), then

$$E_{k-1} \alpha_{ij}^k(t) = \tau_{ij} u_j^k(t_1) \quad (27)$$

The difference between (25) and (24) is that a simple algorithm can be devised to implement (25). It is called the *staircase strategy* because of the graph of $N_{ij}^k(T)$.

Staircase strategy: For all activities j such that $L(j) = k$, perform activity j at resource i as early as possible after the *eligibility rule* is satisfied.

$$\text{Eligibility rule: } N_{ij}^k(T) \leq \int_0^T u_j^k dt \quad (28)$$

If there were only one activity in the system, it would be initiated as soon as (28) were satisfied with equality. Immediately afterward, the left side of (25) would exceed the right side by exactly 1. The difference would then start to diminish until, again, (28) is satisfied with equality. Thus, the error in (25) would never grow larger than 1. Figure 5.1 represents this strategy, and illustrates the term "staircase." The solid line represents the right side of (28), and the dashed line represents the left side. Note that the change in slope of the solid line poses no difficulties for this strategy.

Example: If activity j is an operation on Type A parts at Machine 6, attempt to load a Type A part into the machine whenever (28) is satisfied.

In reality, there are two complications. First, because there are other activities, activity j may not be the only one to satisfy (28) at any instant. Therefore, there must be a mechanism or an additional eligibility rule for selecting one. Consequently, we can no longer assert that (25) is satisfied with an error no larger than 1.

Second, there are relationships among activities other than non-simultaneity. For example, some manufacturing operations may not be performed unless the system is set up in a certain way. That is, in order to perform an operation on Type 1 parts, the system must be set up for them. The most recent setup activity must have been one that is appropriate for Type 1 parts. This leads to additional eligibility rules.

Example: If activity j is an operation on Type A parts at Machine 6, attempt to load a Type A part into the machine whenever (28) is satisfied and Machine 6 is set up for Type A and the part that has been waiting longest for Machine 6 that can be produced in its current configuration is Type A.

Methods for implementing this strategy can be developed based on the methods of Ramadge and Wonham (1985), Maimon and Tadmor (1986).

6. THE HEDGING POINT STRATEGY

The hedging point strategy was introduced by Kimemia and Gershwin (1983) and refined by Gershwin et al. (1985) for a restricted version of the scheduling problem discussed here. In that problem, there were only two activities: operations and failures. The hedging point strategy was used to calculate the production rates of parts in response to repairs and failures of machines.

In the present context, the purpose of the problem is to find u_j^k (for all j such that $L(j) > k$) to satisfy (18) and (23) (and possibly other conditions). That is, we find the optimum frequencies of controllable events whose frequencies are much higher than f_k . These frequencies are chosen in response to changes in low frequency activities: those whose values change at a frequency roughly f_k or slower.

6.1 Surplus

We introduce x_j^k , the *activity j surplus*. This quantity represents the excess of occurrences of activity j as determined by u_j^k over the number of occurrences required by u_j^{k-1} . The surplus is illustrated in Figure 6.1. It satisfies

$$x_j^k(t) = \int_0^t u_j^k(s) ds - \int_0^t u_j^{k-1}(s) ds \quad (29)$$

or

$$\frac{dx_j^k}{dt} = u_j^k - u_j^{k-1}. \quad (30)$$

To keep u_j^k near u_j^{k-1} , we must keep x_j^k near 0. We therefore define a strictly convex function g such that $g(0) = 0$; $g(x) \geq 0 \forall x$; and $\lim_{|x| \rightarrow \infty} g(x) = \infty$, and we seek u_j^k to minimize

$$E_{k-1} \int_0^T g(x_j^k(t)) dt \quad (31)$$

in which T is long enough so that the dynamic programming problem has a time-invariant solution $u_j^k(x^k, \alpha^k)$. Thus T is much greater than $1/f_k$. If (31) is small, then $x_j^k(t)$ must be small for all t . Equation (29) then implies that $u_j^k(t)$ is near u_j^{k-1} .

6.2 Capacity constraints

The activity rate vector $u^k(t)$ must satisfy the stochastic capacity constraints

$$u^k(t) \in \Omega^k(\alpha^k(t)) \quad (32)$$

where $\Omega^k(\alpha^k(t))$ is given by (22). This means that the activity rates of all high frequency activities are restricted in a way that depends on the current states of activities whose frequencies are roughly f_k or less. Those whose frequencies are much less than f_k can be treated as constant at their present values, but the variations of those that change at a frequency comparable to f_k must be considered.

Because Kimemia and Gershwin were dealing with machine failures and repairs, they could treat $\alpha^k(t)$ as the state of a Markov process. Here, however, some components of $\alpha^k(t)$ are chosen by the scheduler according to the staircase strategy of Section 5. For the purpose of determining the frequencies of high-frequency activities, we treat $\alpha^k(t)$ as though it is generated by some exogenous stochastic process with transition rate matrix λ^k :

$$\lambda_{\alpha\beta}^k \delta t = \text{prob} \{ \alpha^k(t+\delta t) = \beta \mid \alpha^k(t) = \alpha \}, \quad \alpha \neq \beta; \quad \lambda_{\alpha\alpha}^k = - \sum_{\beta \neq \alpha} \lambda_{\alpha\beta}^k. \quad (33)$$

By treating all level k events this way, we are ignoring information that could be used, in principal, to improve the performance function (31). Since the time for the next event is known and not random, the optimal trajectory should be different. This requires further study.

6.3 Other constraints

Some activities are non-controllable, such as machine failures. Their frequencies cannot be chosen; they are given quantities. Thus, if \mathcal{N} is the set of uncontrollable activities,

$$u_j^k \text{ specified, } j \in \mathcal{N}. \quad (34)$$

Other activities require special constraints because of their special nature. For example, when a resource may have more than one configuration, and setups require significant time, setup frequencies are constrained to satisfy a set of equality constraints. Assume resource i has configurations $1, \dots, C(i)$. Denote $j=(iab)$ as the activity of changing the configuration of resource i from a to b . Then u_{iab}^k is the level k frequency of changing the configuration of resource i from a to b . These frequencies must satisfy

$$\sum_a u_{iab}^k = \sum_a u_{iba}^k. \quad (35)$$

since the frequency of changing into setup b must be the same as the frequency of switching out of setup b . Related formulations appear in Gershwin (1986) and Choong (1987).

We summarize all such miscellaneous constraints as

$$m(u^k(t)) = 0. \quad (36)$$

6.4 Problem statement

Here we present a compact statement of the problem. It is a dynamic programming problem whose states are $x^k(t)$ and $\alpha^k(t)$ and whose control is $u^k(t)$. (The rates u^{k-1} are treated as exogenous constants.)

Find the feedback control law $u^k(x^k(t), \alpha^k(t), t)$ to minimize (31) subject to (30), (32), and (36) in which α^k is the state of an exogenous Markov process, with parameters λ^k . The initial conditions at $t=0$ are $x^k(0)$, $\alpha^k(0)$. T is very large.

6.5 Solution

Kimemia and Gershwin (1983) derived a Bellman's equation for this problem:

$$0 = \min \left\{ g(x^k) + \frac{\partial J}{\partial x}(u^k - u^{k-1}) + \frac{\partial J}{\partial t} + \sum_{\beta} \lambda_{\alpha\beta} J[x^k, \beta, t] \right\} \quad (37)$$

in which $J[x^k(t), \alpha^k(t), t]$ is the cost-to-go function, the cost incurred during (t, T) if the initial conditions are $x^k(t)$ and $\alpha^k(t)$ at time t . The minimization in (37) is performed at every t subject to (32) and (36). If such a J function could be found to satisfy this nonlinear partial differential equation, the optimal control u^k could be determined from the indicated minimization.

If J were known, determining u^k would reduce to solving

$$\left. \begin{array}{l} \min \frac{\partial J}{\partial x} u^k \\ \text{subject to (32) and (36).} \end{array} \right\} \quad (38)$$

If $m(u^k)$ is a linear function, this is a linear programming problem.

Akella and Kumar (1986), Bielecki and Kumar (1987), and Sharifnia (1987) have obtained analytic solutions for versions of this problem in which x^k and u^k are scalars. In no other cases are exact solutions to this problem known. Numerical solutions are equally unavailable because of the "curse of dimensionality." To overcome this difficulty, Akella et al. (1984) show that a quadratic approximation of J can produce excellent performance.

Kimemia and Gershwin ran several simulations to test a simple hierarchical policy: solve (38) at every time instant to determine u^k , and then load parts (in a manner somewhat more complex and less effective than the staircase strategy of Section 5) so that the rate of loading parts was close to u^k . This worked well until the solution of (38) changed abruptly. (This is an important possibility since (38) is a linear program.) Very often, it changed abruptly again at the next time instant, and this led to reduced performance.

Gershwin et al. (1985) avoided this chattering by observing a behavior similar to that of a closely related problem of Rishel (1975). The continuous part of the state, x^k , is restricted to reduced dimensional surfaces whenever u^k would otherwise chatter. In the present problem, chattering is avoided by adding linear equality constraints to (38) whenever x^k reaches certain planes.

This step has the additional benefit of reducing computational effort. It is no longer necessary to solve (38) at every time instant. Instead, a series of computations is performed at every time t_c when there is a change in α^k . At those instants (38) is solved, and then solved repeatedly with additional constraints, as described above. The outcome of these calculations is a piecewise constant function of t , $u_j^k(t; \alpha^k(t_c^+))$, defined for $t > t_c$. This function is the set of target rates for level $k+1$. When α^k changes, the function is recalculated.

There are two kinds of states α^k : feasible and infeasible. *Feasible* states are those for which $u^{k-1} \in \Omega^k(\alpha^k(t))$. All other states are infeasible. If α^k is feasible and constant for a long enough period, the strategy drives x^k to the value that minimizes $J(x^k, \alpha^k, t)$. In steady state, this is a constant which we call the *hedging point*. We have assumed that T is large

enough so that the system can be assumed to be in steady state.

The hedging point represents a safety level of the surplus. Infeasible states are certain to occur eventually. While α^k is infeasible, x^k must decrease, and possibly become negative. The hedging point represents a compromise between a cost for positive x^k and a cost for negative x^k . When the activities considered are production operations on parts, for example, the tradeoff is between production that is ahead of demand (and therefore can lead to inventory) and production that is behind demand (and therefore leads to starved downstream resources or unhappy customers). The hedging point need not be positive. Bielecki and Kumar show that it can sometimes be 0.

7. SIMPLE EXAMPLE

In this section, we illustrate the ideas developed in this paper with a two-part, two-machine system. There are only two phenomena in this system: failures and operations. The former are much less frequent, but of much greater duration, than the latter. This is an example of the methods of Kimemia and Gershwin (1983) and Gershwin, Akella, and Choong (1985). An extension of this system, in which setup plays a role, is described in Gershwin (1987).

7.1 Description of System

Figure 7.1 illustrates the two-machine system. In this system, Machine 1 is perfectly flexible. That is, it can do operations on either part type, without time lost for changeover. It is unreliable, however: it fails at random times and stays down for random lengths of time. Machine 2 is perfectly reliable, but totally inflexible. It can only make Type 1 parts. Thus Machine 1 is shared among the two part types and Machine 2 is devoted entirely to Type 1.

The data that are specified are the demand rates for the parts, the failure (p) and repair (r) rates, and the durations of the operations (τ_{11} , τ_{12} , and τ_{21} , where τ_{ij} is the duration of an operation on a Type j part at Machine i). To simplify the problem, we assume that the demand rate for Type 1 parts is broken down by the machine at which the operation is performed, so that the specified demand rates are d_{11} , $d_{12}=d_2$, and d_{21} .

For this problem, Assumption 1 becomes:

$$\tau_{11}, \tau_{12}, \tau_{21}, 1/d_{11}, 1/d_2, 1/d_{21} \text{ are the same order of magnitude.}$$

These quantities are all smaller than $1/r$, $1/p$, which are the same order of magnitude.

7.2 Level 1: Hedging point strategy

The states of the system are α , the repair state of Machine 1, an exogenous random variable; and x_{11} , x_{12} , and x_{21} , the surpluses. The control variables are u_{ij} , the level 1 flow rate of Type j parts to Machine i ($ij = 11, 12, 21$).

Here, (30) becomes

$$\dot{x}_{ij} = u_{ij} - d_{ij} \text{ for } ij = 11, 12, \text{ and } 21. \quad (39)$$

The linear programming problem of Section 6.5, which determines u_{ij} , becomes

$$\min_{u_{ij}} \sum c_{ij}(x, \alpha) u_{ij} \quad (40)$$

subject to:

$$\tau_{11}u_{11} + \tau_{12}u_{12} \leq \alpha \quad (41)$$

$$\tau_{21}u_{21} \leq 1 \quad (42)$$

$$u_{ij} \geq 0 \quad (43)$$

where for $ij = 11, 12,$ and 21 and for $mn = 11, 12,$ and $21,$

$$c_{ij}(x, \alpha) = \sum_{mn} A_{ijmn}(\alpha)x_{mn} + b_{ij}(\alpha). \quad (44)$$

Here, $c(x, \alpha)$ is the approximation of $\partial J / \partial x$. Satisfactory results have been obtained with diagonal A matrices, so we choose $A_{ijmn} = 0$ if $(mn) \neq (ij)$. The hedging point is then

$$z_{ij}(\alpha) = \frac{b_{ij}(\alpha)}{A_{ijij}(\alpha)}.$$

The outcome of this calculation is a piecewise constant function of time $u_{ij}(t)$, as described by Gershwin et al. (1985). This function is used in the staircase strategy, below, until the repair state α changes. When that happens, a new function is calculated at this level.

7.3 Level 2: Staircase strategy

Loading a Type j part into Machine 1 is eligible if:

1. The number of Type j parts made on Machine 1 is less than

$$\int_0^t u_{1j}(s) ds, \text{ and} \quad (45)$$

2. Machine 1 is now idle.

Loading a Type 1 part into Machine 2 is eligible if:

1. The number of Type 1 parts made on Machine 2 is less than

$$\int_0^t u_{21}(s) ds, \text{ and} \quad (46)$$

2. Machine 2 is now idle.

7.4 Simulation results

Figure 7.2 demonstrates how the cumulative output follows the cumulative requirements when the system is run with this strategy.

8. CONCLUSIONS

A hierarchical scheduling and planning strategy has been described for manufacturing system. It is based on two major propositions:

1. Capacity. No resource can function more than 100% of the time.
2. Frequency separation. We assume that the spectrum of events is discrete. The frequencies of important events are grouped into distinct clusters.

This work is in its early stages. Among the important outstanding research problems are proving the conjecture that hierarchical decomposition is asymptotically optimal as times scales separate; determining how to deal with systems in which time scales are not

widely separated; formulating and solving the hedging point problem with non-Markov events (such as those generated by a staircase strategy); developing sufficiency conditions for capacity. To improve on the staircase policy, new formulations of combinatorial optimization problems are required in which the objective is to load material as close as possible to a given rate.

We have not discussed at all the collection and processing of data in the hierarchy. This will require the solution of statistics problems. Some extensions include the reduction of the problem size at higher levels. This requires aggregation of activities (so that one considers, for example, large classes of part types, rather than individual types) and of resources (so that the smallest unit can be a cell or workshop or even factory, rather than a machine).

The last issue is related to the long time that parts spend in some kinds of manufacturing, particularly semiconductor fabrication. Preliminary work in extending the Kimemia-Gershwin formulation to systems with both operation and queuing delay and is described in Lou et al. (1987) and Van Ryzin (1987).

ACKNOWLEDGMENTS

This work was supported by the Defense Advanced Research Projects Agency and monitored by ONR under contract N00014-85-K-0213.

REFERENCES

- P. Afentakis, B. Gavish, U. Karmarkar (1984), "Computationally Efficient Optimal Solutions to the Lot-Sizing Problem in Multi-stage Assembly Systems," *Management Science*, Vol. 30, No. 2, February 1984, pp. 222-239.
- R. Akella, Y. F. Choong, and S. B. Gershwin (1984), "Performance of Hierarchical Production Scheduling Policy," *IEEE Transactions on Components, Hybrids, and Manufacturing Technology*, Vol. CHMT-7, No. 3, September, 1984.
- R. Akella and Kumar (1986), "Optimal Control of Production Rate in a Failure Prone Manufacturing System," *IEEE Transactions on Automatic Control*, Vol. AC-31, No. 2, pp. 116-126, February, 1986.
- T. Bielecki, and P. R. Kumar (1986), "Optimality of Zero-Inventory Policies for Unreliable Manufacturing Systems", Coordinated Science Laboratory, University of Illinois Working Paper.
- G. R. Bitran, E. A. Haas, and A. C. Hax (1981), "Hierarchical Production Planning: A Single-Stage System," *Operations Research*, Vol. 29, No. 4, July-August, 1981, pp. 717-743.
- Y. F. Choong (1987), MIT Ph.D. Thesis in preparation.
- F. Delebecque, J. P. Quadrat, and P. V. Kokotovic (1984), "A Unified View of Aggregation and Coherency in Networks and Markov Chains," *International Journal of Control*, Vol. 40, No. 5, November, 1984.
- M. A. H. Dempster, M. L. Fisher, L. Jansen, B. J. Lageweg, J. K. Lenstra, and A. H. G. and Rinnooy Kan, (1981), "Analytical Evaluation of Hierarchical Planning Systems," *Operations Research*, Vol. 29, No. 4, July-August, 1981, pp. 707-716.
- S. B. Gershwin (1986), "Stochastic Scheduling and Setups in a Flexible Manufacturing System," In *Proceedings of the Second ORSA/TIMS Conference on Flexible Manufacturing Systems*, Ann Arbor, Michigan, August, 1986, pp. 431-442.

- S. B. Gershwin (1987), "A Hierarchical Framework for Manufacturing System Scheduling," *Proceedings of the 26th IEEE Conference on Decision and Control*, Los Angeles, California, December, 1987
- S. B. Gershwin, R. Akella, and Y. F. Choong (1985), "Short-Term Production Scheduling of an Automated Manufacturing Facility," *IBM Journal of Research and Development*, Vol. 29, No. 4, pp 392-400, July, 1985.
- S. C. Graves (1981), "A Review of Production Scheduling," *Operations Research*, Vol. 29, No. 4, July-August, 1981, pp. 646-675.
- S. C. Graves (1982), "Using Lagrangean Relaxation Techniques to Solve Hierarchical Production Planning Problems," *Management Science*, Vol. 28, No. 3, March 1982, pp. 260-275.
- A. C. Hax and H. C. Meal (1975), "Hierarchical Integration of Production Planning and Scheduling," North Holland/TIMS, Studies in Management Sciences, Vol. 1, *Logistics*.
- J. Kimemia and S. B. Gershwin (1983), "An Algorithm for the Computer Control of a Flexible Manufacturing System," *IIE Transactions* Vol. 15, No. 4, pp 353-362, December, 1983.
- B. J. Lageweg, J. K. Lenstra, and A. H. G. and Rinnooy Kan (1977), "Job-Shop Scheduling by Implicit Enumeration," *Management Science*, Vol. 24, No. 4, December 1977, pp. 441-450.
- B. J. Lageweg, J. K. Lenstra, and A. H. G. and Rinnooy Kan (1978), "A General Bounding Scheme for the Permutation Flow-Shop Problem," *Operations Research*, Vol. 26, No. 1, January-February 1978, pp. 53-67.
- X.-C. Lou, J. G. Van Ryzin and S. B. Gershwin (1987), "Scheduling Job Shops with Delays." In *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, Raleigh, North Carolina, March-April 1987.
- O. Z. Maimon and S. B. Gershwin (1987), "Dynamic Scheduling and Routing For Flexible Manufacturing Systems that have Unreliable Machines," In *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, Raleigh, North Carolina, March-April 1987. See revised version: MIT LIDS Report No. LIDS-P-1610, revised July, 1987.
- O. Maimon and G. Tadmor (1986), "Efficient Low-Level Control of Flexible Manufacturing Systems," MIT LIDS Report No. LIDS-P-1571.
- E. F. P. Newhouse (1975a), "Multi-Item Lot Size Scheduling by Heuristic, Part I: With Fixed Resources," *Management Science*, Vol. 21, No. 10, June 1975, pp. 1186-1193.
- E. F. P. Newhouse (1975b), "Multi-Item Lot Size Scheduling by Heuristic, Part I: With Variable Resources," *Management Science*, Vol. 21, No. 10, June 1975, pp. 1194-1203.
- C. H. Papadimitriou and P. C. Kannelakis (1980), "Flowshop Scheduling with Limited Temporary Storage," *Journal of the ACM*, Vol. 27, No. 3, July, 1980.
- P. J. Ramadge and W. M. Wonham (1985), "Supervisory Control of a Class of Discrete Event Processes," Systems Control Group Report No. 8515, University of Toronto.
- Rishel, R. "Dynamic Programming and Minimum Principles for Systems with Jump Markov Disturbances", *SIAM Journal on Control*, Vol.13, No.2 (February 1975).
- V. R. Saksena, J. O'Reilly, and P. V. Kokotovic (1984), "Singular Perturbations and Time-Scale Methods in Control Theory: Survey 1976-1983", *Automatica*, Vol. 20, No. 3, May, 1984.

A. Sharifnia (1987), "Optimal Production Control of a Manufacturing System with Machine Failures," Department of Manufacturing Engineering, Boston University.

H. M. Wagner and T. M. Whitin (1958), "Dynamic Version of the Economic Lot Size Model," *Management Science*, Vol. 5, No. 1, October, 1958, pp. 89-96.

J. G. Van Ryzin (1987), "Control of Manufacturing Systems with Delay," MIT EECS Master of Science Thesis, MIT LIDS Report LIDS-TH-1676.

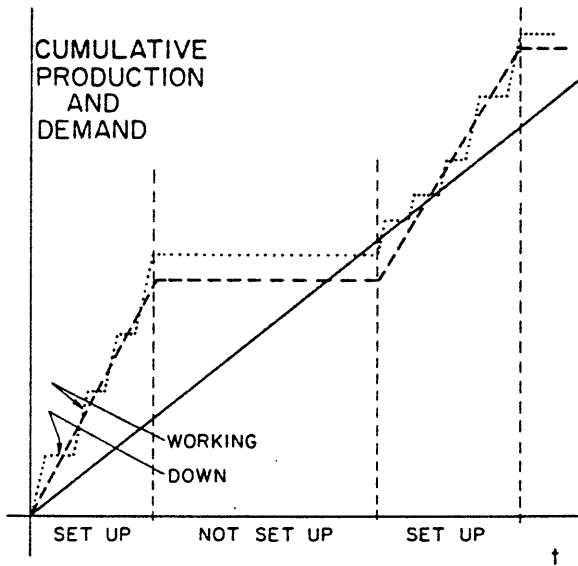


Figure 1.1 Production and Other Events

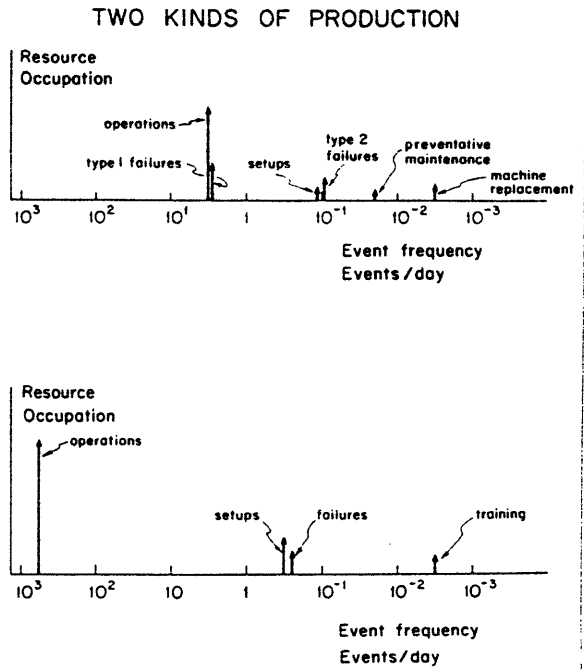
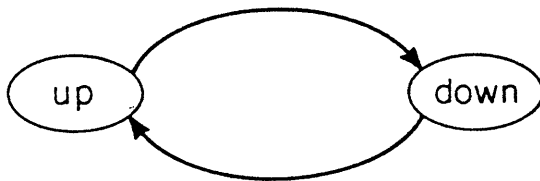


Figure 2.1 Two Spectra



REPAIR STATE OF MACHINE i -
NOT CONTROLLABLE

Figure 2.2 Repair States

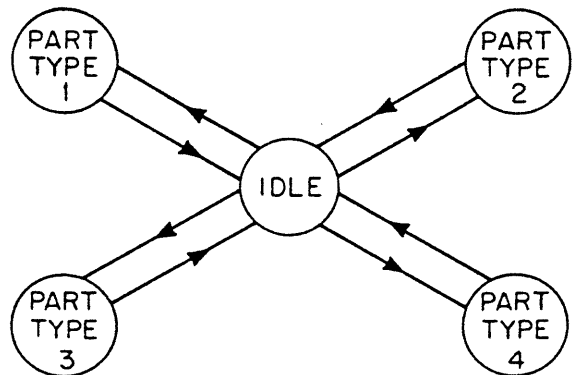


Figure 2.3 Operation States

File page count type title list

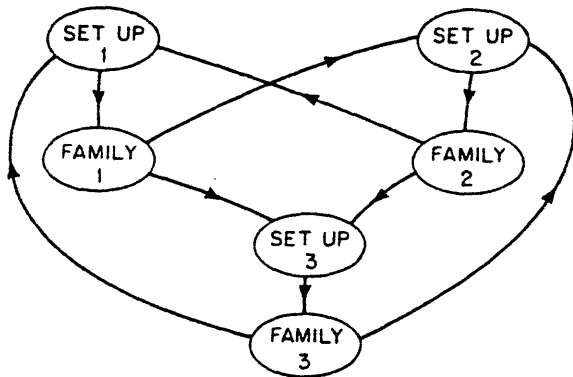


Figure 2.4 Configuration States

ARCHITECTURE OF THE HIERARCHY

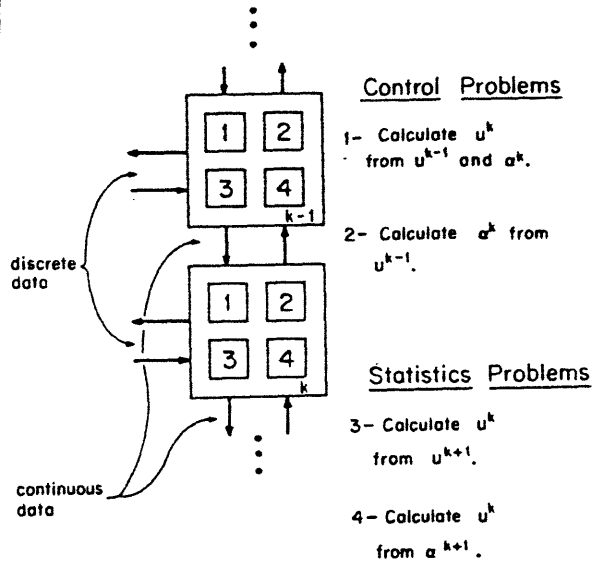


Figure 4.1 Hierarchy

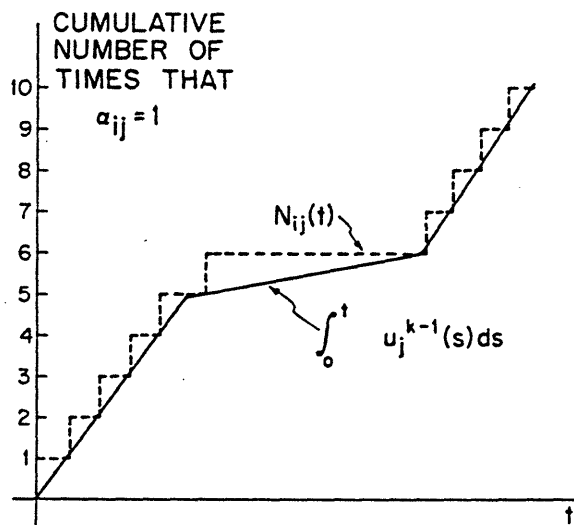


Figure 5.1 Staircase Strategy

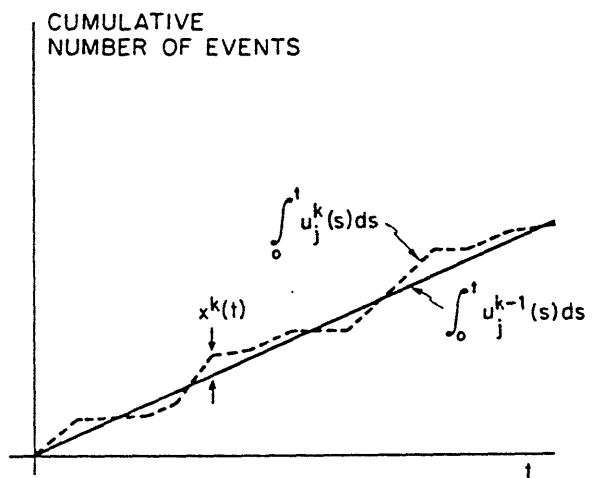


Figure 6.1 Surplus

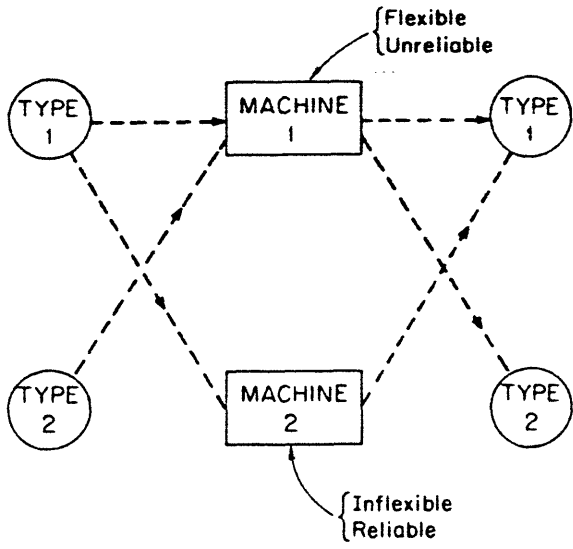


Figure 7.1 Simple System

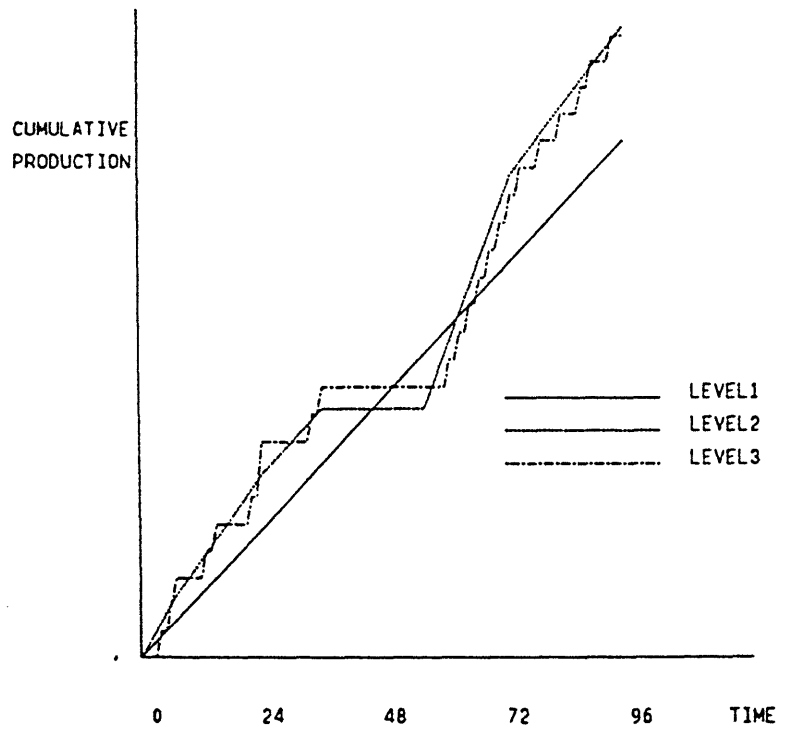


Figure 7.2 Behavior of Strategy