NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

COMPUTER CONTROLLED MANUFACTURING SYSTEMS

Graduation Project
COM- 400

Student: Wala Elddin Yousif

Supervisor: Asst. Prof. Dr Rahib Abiyev

Nicosia - 2002

ACKNOWLEDGEMENTS

" First, I would like to thank my supervisor Assist. Prof. Dr Rahib Abiyev for his invaluable advice and belief in my work and myself over the course of this Graduation Project.

Second, I would like to express my gratitude to Near East University for providing me with the required knowledge that made me prepared for my future.

Third, I thank my family for their constant support and encouragement during the preparation of this project and for their belief in me.

Finally, I would also like to thank all my friends for their advice and support."

ABSTRACT

Presently, some discrete industrial processes is characterized with different kinds of output products. For this process development of Automated Flexible Computer Aided Control System is very important. The reorganization process of industrial by using computer technology needs the development corresponding hardware and software.

For this reason the graduation project is devoted to the very actual problem. In the project the Evaluation of Computer Controlled Manufacturing System is considered the structure and function of Flexible Manufacturing System are described as an example control and scheduling problems of manufacturing system are given. Algorithm for single product and N-Product scheduling problems are described. As example the structure and function of principles of computer control manufacturing system of Dylite Expanded Polystyrene Board productions are given.

TABLE OF CONTENTS

INTRODUCTION

A flexible manufacturing system (FMS) is a highly automated GT machine cell consisting of a group of processing stations (usually CNC machine tools) interconnected by an automated material handling and storage system and controlled by an integrated computer system. An FMS is capable of processing a variety of different part styles simultaneously under NC program control at the different workstations.

The FMS relies on the principles of group technology. No manufacturing system can be completely flexible. It cannot produce an infinite range of products. There are limits to the degree of flexibility that can be incorporated in a FMS. Accordingly, a flexible manufacturing system is designed to produce parts (or products) within a range of styles, sizes, and processes. In other words, a FMS is capable of producing a single part family or a limited range of part families. The concept for a FMS originated in the 1960s.

The flexible manufacturing system was first conceptualized for machining, and it required the prior development of numerical control. The credit for the concept is given to David Williamson, a British engineer employed by Mount during the mid 1960s. Molins patented the invention granted in 1965). The concept was called System 24 because it was believed that the group of machine tools comprising the system could operate 24 hr/day, 16 hr of which would be unattended by human workers. The original concept included computer control of the NC machines, a variety of parts being produced, and tool magazines capable of holding various tools for different machining operations.

One of the first, if not the first, FMS installed in the United States was a machining system at Ingersoll-Rand Company (now Ingersoll-Dresser) in Roanoke, Virginia, in 1967 by Sundstrand. By around 1985, the number of FMSs throughout the world had increased to about 300. About 20% to 25% of these were located in the United States. As the importance of flexibility in manufacturing grows, the number of FMSs is expected to increase. In recent times, there has been an emphasis on smaller, less expensive flexible manufacturing cells.

CHAPTER 1

EVOLUTION OF COMPUTER CONTROLLED MANUFACTURING SYSTEM

1.1 The Hierarchy of Computer Control

One of the early applications of a digital computer in an industrial facility was for plant monitoring and supervisory control, as documented by Garrett and McHenry (1981). Figure 1.1 shows an evolution of the use of digital computers in industrial processes.

As illustrated in Figure 1.1, Dupont-Gatelmand (1981) documented that the next step in evolution after the supervisory control is the computer numerical control (CNC) of the machines followed by direct digital control (DDC) for a group of multiple numerical control (NC) machines.

In the DDC, as written by Lukas (1986), a computer reads and directly processes measurements, calculates the proper control outputs, and sends the control commands to the activation devices. In the initial implementations of DDC, backup analog control systems were used to avoid the ill effects of computer failures. In spite of the early computer hardware reliability problems, DDC demonstrated many advantages over analog control systems. They included the use of complex logic to calculate more accurately the control command values, ease of data logging, data trending, alarming, and so on. It also avoided the common problem of set-point drifts associated with analog devices. Several different system architectures evolved for the DDC systems in the late 1970s.

However, a central computer was a dominant feature of all the variations of these architectures. The single largest disadvantage of these architectures, as pointed out before, was the single-point failure of the central computer, which could shut down the process. It necessitated the expense of a second computer as backup. Another disadvantage was that the software for the central computer was very complex and required a team of software/hardware experts to change and maintain the software. It also had limited expansion capability and, when the expansions were made, they were very expensive.

In the mid 1960s the distributed control system architecture was brought forward as a viable option. The technology to implement the DDC in a cost-effective manner was not available until the early 1970s. The price of computers decreased significantly and personal computers could be used economically on the factory floor. A number of production lines with distributed control systems have begun emerge since the early 1970s.

As shown in figure 1.1, every work center has a dedicated control computer.

The work center computer is responsible for making production happen in its

Error!

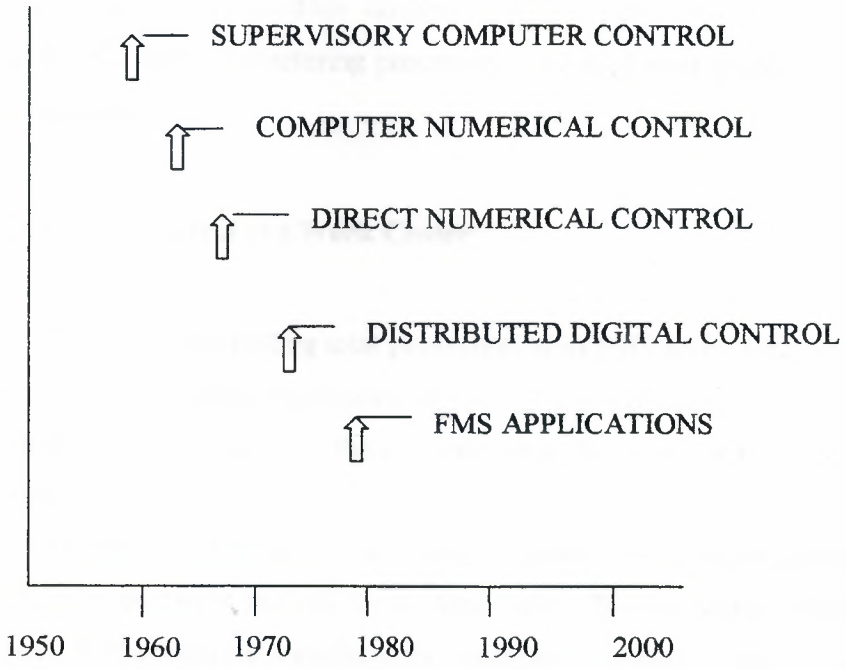


Figure 1.1. Evolution of the use of digital computers

work center. It communicates with each tool and robot and downloads the programs, resulting in appropriate commands for all equipment. It is also responsible for ascertaining the health of all the components in the work center.

The next level in the hierarchy of computer control is the subassembly/main assembly line computer. In general, this computer coordinates and controls the

manufacturing activities within a section of the manufacturing facility. This computer communicates to each work-center computer the type of product to be made and all the appropriate commands. It also serves as a backup to any one of the failed work-center computers in its jurisdiction.

As shown in Figure 1.1, the supervisory computer is at the highest level in the hierarchy of computer control. This computer does overall production planning and scheduling and communicates with the subassembly/main assembly line computers (Koren, 1983). In the event of the failure of any of the subassembly/main assembly computers, the supervisory computer takes over the tasks of the failed computer.
The key for producing economically different products or part numbers from the automated line shown in Figure 1 is the flexibility provided by the computer control. Simple software changes can dictate the automated
Manufacturing line to produce different part numbers by changing the number of subassemblies, the manufacturing process in designated work centers, and the logistics flow of the parts.

1.2 Computer Control in a Work Center

The task of controlling total production in an FMS plant is the management of a complex set of machines interconnected with the automated parts transfer mechanisms. To control a real-time process, the computer must do the following, as documented by Williams (1988):

1. Process Control Commands: The control computer must have the software capability to direct the hardware devices to do their tasks. The hardware includes the actual machines that perform the manufacturing operations and the logistics mechanisms.
2. Process-initiated Interrupts: The control computer must receive and respond to the signals received from the process. Depending upon the importance of the signal, the computer may have to abort its current operation and perform a priority task.
3. Periodic Time-Initiated Events: The control computer must periodically collect management and status information. This is important for creating a history base and performing trend analyses.

4. System and Program-Initiated Events: The work-center computer is connected to subassembly/main assembly control computer. Therefore, it must handle communication and data transfer with the higher level computers.

The lowest level control computer in the hierarchy of FMS control is the work-center control computer. It controls a small collection of hardware, as shown in Figure 1. The type and size of the work-center control computer is a function of the number of machines in the work center, the complexity of the manufacturing process, and the required response time, as pointed out by Sheff (1986). Figure 1.2 shows a set of control functions the work-center control computer must perform. A brief description of each of these control functions is given in the remainder of this section.

1. Schedule Execution: The work-center computer must execute the work-center schedule. It must marshal all the resources in the work center to complete the schedule activities. It should also communicate the execution progress to the subassembly/main assembly computer.

2. Process Device Control: The control computer can exercise true device control functions in the open-loop or closed-loop mode. The preplanned control is of the open-loop mode in which a predetermined standard set of commands or sequences of commands with intermediate logical checks are executed. In a closed-loop model, the control computer attains a desired value of a control variable by comparing its actual value with the desired value and uses the error signal as a control input. All devices which do not have their own control computer are controlled by the work-center control computer in this manner.

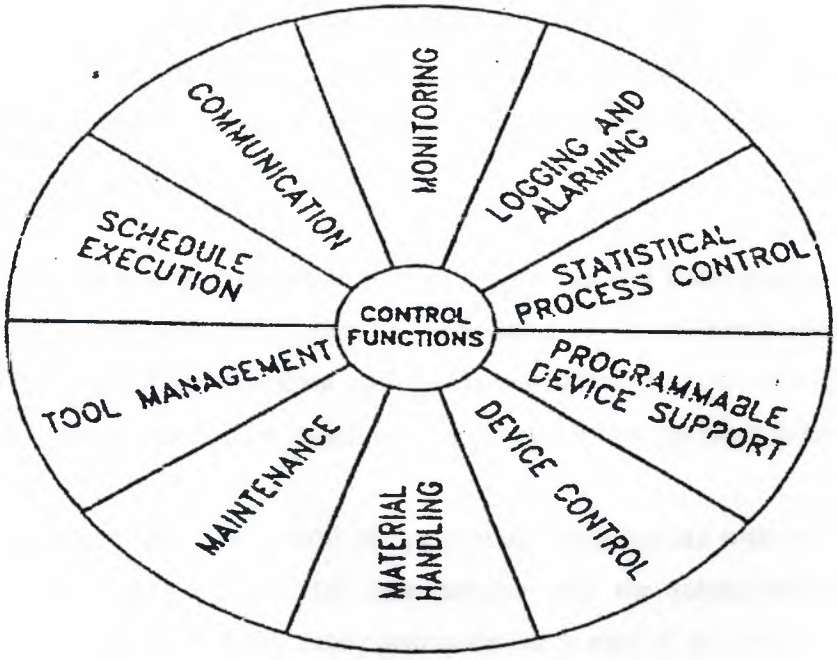


Figure 1.2. Work-center computer control functions

3. Programmable Device Support: the control computer is responsible for downloading the part programs to the devices in the work center that have their own control computers. The downloading is coordinate to the work-center production schedule.

4. Tool Management: This control function includes management of all reusable resources such as drills, bits, gauges, and so on, with the work center. It should keep the inventory of all tools and monitor tool wear.

5. Maintenance: The work-center control computer must keep track of the health of each piece of equipment within the work center. This includes keeping a maintenance

history and creating preventive maintenance schedules. The maintenance function is directly related to the availability of the work center for production.

6. Material Handling: The control computer is responsible for the flow of material into the work center, movement of material within it, and the flow of material exiting it. As shown in Figure 1.2, the devices used for material handling are the conveyors, robots, AGVs, and so on.

7. Statistical Process Control: In a completely automated FMS plant, it is necessary to perform a statistical check within the work center to a predefined statistical norm. The control computer must perform such checks periodically and take appropriate actions to correct any variations in the manufacturing process within the work center.

8. Communication: The control computer must communicate with every device within the work center. It must also communicate with the subassembly/main assembly computer. Orderly and accurate communication is needed to avoid costly waste. The exact product schedule will be communicated to the work-center computer by the subassembly/main assembly computer.

9. Monitoring: The control computer serves as an alert and indefatigable supervisor. The most important monitoring task is that of process monitoring. This usually requires that the control computer establish the status of the instruments and the process variables, the status of the equipment within the work center, and the status of the product itself. The control computer can also monitor indirect measurements, as shown by Savas (1985). These measurements are a function of several directly monitored process attributes. The computer supervision is an important function in producing a defect-free product in a work center.

10. Data Logging and Alarming: The work-center control computer must collect and store the information about all the devices in the work center and the significant events that take place in it. These data will be used to create preventive maintenance schedules and keep the processing capabilities of the devices current. The data can be uploaded to the subassembly/main assembly computer for trend analysis. The data can be used for the alarm management function.

1.3 Computer Control in Subassembly/Main Assembly Lines

The control computer in a manufacturing line communicates with the work-center control computers and ensures scheduled production through its line. The subassembly/main assembly line control computer receives its production schedule from the FMS supervisory computer. The primary functions of the subassembly/main assembly line control computers are as follows.

1. Monitor Production Performance: The subassembly/main assembly line control computer must monitor the production performance of each work-center control computer in its jurisdiction. If the work-center control computer fails, the subassembly/main assembly line control computer has the responsibility to function as a backup control computer.

2. Database Management: The subassembly/main assembly line control computer receives and stores all the process and device- related data in its database, as pointed out by (Meister, 1987). It manipulates these data and uploads them to the EMS supervisory control computer for generating production plans.

3. Production Scheduling: The subassembly/main assembly line control computer receives primary production and two backup production schedules each day from the FMS supervisory control computer. The subassembly/main assembly control computer utilizes the backup production schedule in the anomalous operating conditions. It communicates such operating conditions to work-center computers as well as to the FMS supervisory control computer.

4. Alarm Management Section: The subassembly/main assembly line control computer is responsible for the alarm management system. The control computer must keep log of the reasons for all the alarms in all the work centers in its jurisdiction.

1.4 Flexible Manufacturing System Supervisory Computer Control

As the name implies, the supervisory computer supervises and controls production throughout the entire manufacturing facility. This is the highest and most important level in the hierarchy of computer control. At this level, the demands for various parts are analyzed. Knowing the process involved in the manufacture of each product and the capacity of each of the equipment, a production schedule is created. Simulations are run to understand the capacity pinch-point operations. Akella *et al.* (1984) described a hierarchical production scheduling policy that minimizes the disruptive effects of such disturbances as machine failures. A cost analysis of the schedule is also performed to quantify the unused capacity within each subassembly area. Since an EMS facility is very costly, it is important to utilize the equipment as much as possible, as written by Morton and Smut (1986).

Once the schedule is finalized, material requirements are created. Also, detailed work plans for each subassembly/main assembly work station are developed for each day. Smith et al. (1986) made a survey of the scheduling criteria used by FMS users and their relation to the scheduling techniques published in the literature. In general, a rolling one-week firm schedule with the production outlook for the next four weeks appears to be an adequate scheduling scenario for a complex product. That is, at the end of each day a new day is added to the firm schedule and to the outlook. The detailed work plan for each day is downloaded to each subassembly/main assembly computer at the beginning of each day. An overview of the important control functions of the supervisory computer are presented in the remainder of this section.

1. Production Planning: Production planning necessarily begins with the analysis of Firm demands for each product and the forecast of future sales. The supervisory computer database necessarily contains the description of the production process for each product and the tool capacity in each subassembly/main assembly line. A preliminary production plan for the facility is created for the rolling day that will accommodate the firm and projected demands.

2. Simulation: The demands for each product within the production plan should be simulated to understand the pinch-point resources and the intricate subtleties for the product changeover requirements. "What if" scenarios should also be simulated to

create alternative production plans. They should include tool failures, part shortages, or quality problems. The data used should be based upon the historical data stored in the supervisory computer. The simulation effort may be very complex since a product may be able to take alternative routes made possible by the FMS. Buzacolt (1983) described a basic philosophy to develop models to estimate performance and key technical issues of FMS. If the preliminary production plan does not appear feasible, the production planning step described above may have to be repeated.

3. Master Production Schedule: A detailed production schedule for each day within the rolling week is created for the preliminary production plan, as well as two backup production plans using the "what if" analysis. The details of the master production schedule include the sampling/verification plan for each subassembly and for the final assembly for quality. Fox (1982) gave an overview of the EMS software control functions for variable missions and scheduling procedures.
The master production schedule must be accurate. It will be downloaded each day to the subassembly/main assembly computers and it dictates the total production operations within the FMS.

4. Capacity Analysis: The supervisory computer must determine the unused capacity in each subassembly area. This unused capacity may be utilized to produce spare parts such as field replaceable units (FRUs). Also, the supervisory computer must schedule preventive maintenance consistent with the utilization of the equipment. Kumar and Vannelli (1986) presented a flexible decision process to help balance the capacity of work centers.

5. Communication: A supervisory computer requires constant communication with the subassembly/main assembly computers to be aware of the status of the equipment as well as the production. It can then compare the status with the simulation results to predict the problem and sound appropriate alarms. In a completely automated FMS, the role of the supervisory computer is very important. It can help to fully utilize FMS hardware and therefore help to manufacture cost-competitive products.

CHAPTER 2

FLEXIBLE MANUFACTURING SYSTEMS

2.1 Flexibility and Automated Manufacturing Systems

Flexible manufacturing systems vary in terms of number of machine tools and level of flexibility. When the system has only a few machines, the term flexible manufacturing cell (FMC) is sometimes used. Both cell and system are highly automated and computer controlled. The difference between a FMS and a FMC is not always clear, but is sometimes based on the number of machines (workstations) included. The flexible manufacturing system consists of four or more machines, while a flexible manufacturing cell consists of three or fewer machines [7]. However, this distinction is not universally accepted, and the terminology of this technology is not yet fully sorted out.

Some highly automated manufacturing systems and cells are not flexible, and this leads to confusion in terminology. For example, a transfer line (Section 36.3) is a highly automated manufacturing system, but it is limited to mass production of one part style, so it is not a flexible system. To develop the concept of flexibility in a manufacturing system, consider a cell consisting of two CNC machine tools that are loaded and unloaded by an industrial robot from a parts carousel, perhaps in the arrangement depicted in Figure 2.1. The cell operates unattended for extended periods of time. Periodically, a worker must unload completed parts from the carousel and replace them with new work-parts. This is truly an automated manufacturing cell, but is it a flexible manufacturing cell? One might argue yes. It is flexible since the cell consists of CNC machine tools that can be programmed to machine different part configurations like any other CNC machine. However, if the cell only operates in a batch mode, in which the same part style is produced in lots of several dozen (or several hundred) units, then this does not qualify as flexible manufacturing.

To qualify as being flexible, a manufacturing system should satisfy several criteria. The tests of flexibility in an automated production system are the capability to (I) process different part styles in a nonbatch mode, (2) accept changes in production schedule, (3) respond gracefully to equipment malfunctions and breakdowns in the system, and

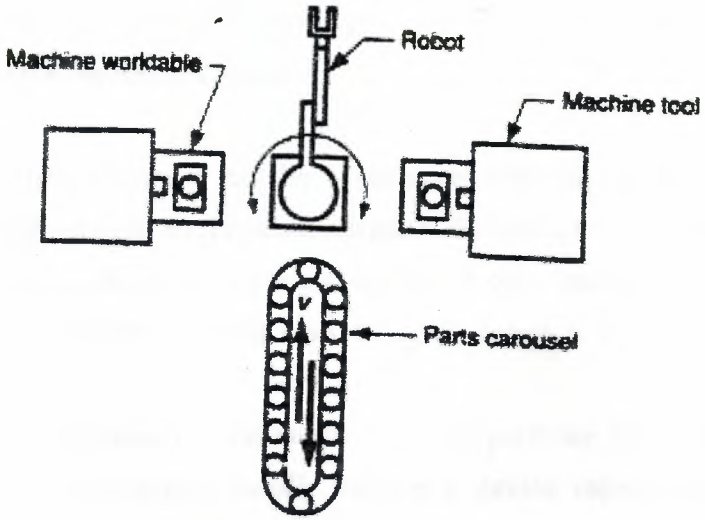


Figure2.1. Automated manufacturing cell with two machine tools and a robot

(4) Accommodate the introduction of new part designs. These capabilities are made possible by the use of a central computer that controls and coordinates the components of the system. The most important criteria are (1) and (2); criteria (3) and (4) are softer and can be implemented at various levels of sophistication.

If the automated system does not meet these four tests, it should not be classified as a flexible manufacturing system or cell. Getting back to our illustration, the robotic work cell would satisfy the criteria if it (1) machined different part configurations in a mix rather than in batches; (2) permitted changes in production schedule and part mix; (3) continued operating even though one machine experienced a breakdown; for example, while repairs are being made on the broken machine, its work is temporarily reassigned to the other machine; and (4) as new part designs are developed, NC part programs are written off-line and then downloaded to the system for execution. This fourth capability also requires that the tooling in the CNC machines as well as the end effector's of the robot be suited to the new part design.

2.2 Integrating the FMS Components

A FMS consists of hardware and software that must be integrated into an efficient and reliable unit. It also includes human personnel In this section we examine these components and how they are integrated.

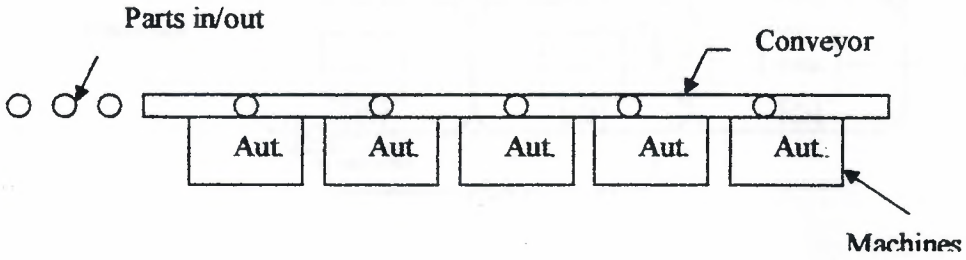
Hardware Components FMS hardware includes workstations, material handling system, and central control computer. The workstations include CNC machines in a machining-type system, plus inspection stations and parts cleaning and other stations, as needed. For a flexible machining system, a central chip conveyor system is usually included below floor level

The material handling system is the means by which parts are moved between stations. The material handling system usually includes a limited capability to store parts. Handling systems suitable for automated manufacturing include roller conveyors, in-floor towline carts, automated guided vehicles, and industrial robots. The most appropriate type depends on part size and geometry, as well as factors relating to economics and compatibility with other FMS components. Nonrotational parts are often moved in a FMS on pallet fixtures, so the pallets are designed for the particular handling system, and the fixtures are designed to accommodate the various part geometries in the family. Rotational parts are often handled by robots if weight is not a limiting factor.

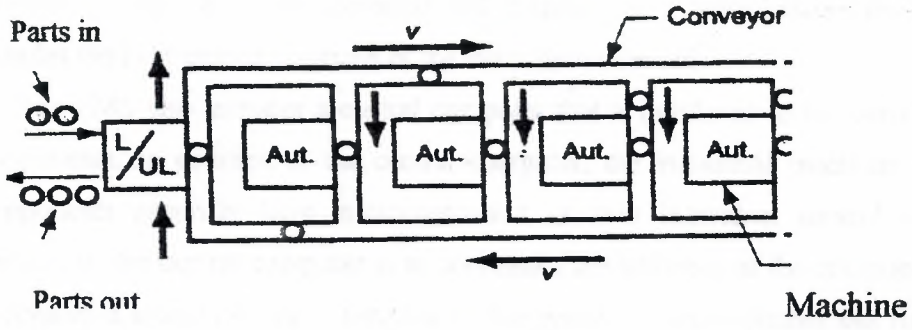
The handling system establishes the basic layout of the FMS. Five layout types can be distinguished: (1) in-line, (2) loop, (3) ladder, (4) open field, and (5) robot centered cell.

These five layout types are shown in Figure 2.2. The in-line layout uses a linear transfer parts between processing stations and load/unload (L/UL) station(s). The system is usually capable of two-directional movement; if not, the FMS op a transfer line, and the different part styles made on the system must follow the same basic processing sequence due to the one-direction flow. The loop layout consists of a conveyor loop with workstations located around its periphery. This configuration permits any processing sequence, because any station is accessible from any other station. This is also true for the ladder layout, in which workstations are located on the

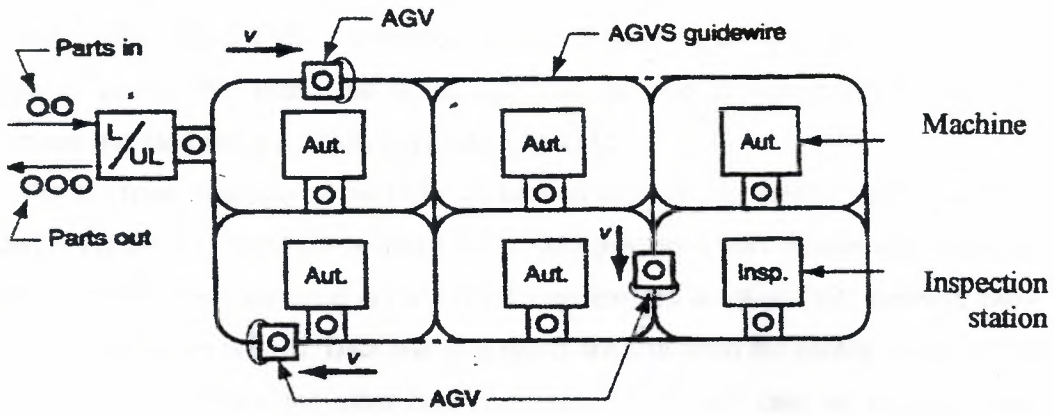
rungs of open field layout is the most complex FMS configuration and consists of several loops tied



(a)



(b)



(c)

Figure2.2. Three of the five FMS layout types: (a) in-line, (b) ladder, and (c) open field. Key: Aut.= automated station; L/UL = load/unload station; Insp. = inspection station; AGV = automated guided vehicle; AGVS = automated guided vehicle system.

together. Finally, the robot-centered cell consists of a robot whose work volume includes the load/unload positions of the machines in the cell.

The FMS also includes a central computer that is interfaced to the other hardware components. In addition to the central computer, the individual machines and other components generally have microcomputers as their individual control units. The function of the central computer is to coordinate the activities of the components so as to achieve a smooth overall operation of the system. It accomplishes this function by means of application software.

2.2.1 FMS Software and Control Functions

FMS software consists of modules associated with the various functions performed by the manufacturing system. For example, one function involves downloading NC part programs to the individual machine tools, another function is

concerned with controlling the material handling system, another is concerned with tool management, and so on. Table 2.2 presents a listing of the functions included in the operation of a typical FMS. Associated with each function are one or more software modules. Terms other than those in our table may be used in a given installation. The functions and modules are largely application specific.

The modular structure of the FMS application software for system control is illustrated in Figure 2.3. It should be noted that a FMS possesses the characteristic architecture of a distributed numerical control (DNC) system. As in other DNC systems, two-way communication is used. Data and commands are sent from the central computer to the individual machines and other hardware components, and data on execution and performance are transmitted from the components back to the central computer. In addition, an uplink from the EMS to the corporate host computer is provided.

2.2.2 Human Labor

An additional component in the operation of a flexible manufacturing system is human labor. Duties performed by human workers include (1) loading and unloading parts from the system, (2) changing and setting cutting tools, (3) maintenance and repair of equipment, (4) NC part programming, *(5)* programming and operating the computer system, and (6) overall management of the system.

TABLE2.2. Typical Computer Functions Implemented by Application Software Modules in a Flexible Manufacturing System.

Function:	Description
NC part programming	Development of NC programs for new parts introduced into the system. This includes a language package such as APT.
Production control	Product mix, machine scheduling, and other planning functions.
NC program download	Part program commands must he downloaded to individual stations using DNC.
Machine control	Individual workstations require controls, usually CNC.
Work-part control	Monitor status of each work-part in the system, status of pallet fixtures, orders on loading/unloading pallet fixtures.
Tool management	Functions include tool inventory control, tool status relative to expected tool life, tool changing and resharpening, and transport to and from tool grinding.
Transport control	Scheduling and control of handling system.
System management	Compiles management reports on performance (utilization, piece counts, production rates, and the like); FMS simulation sometimes.

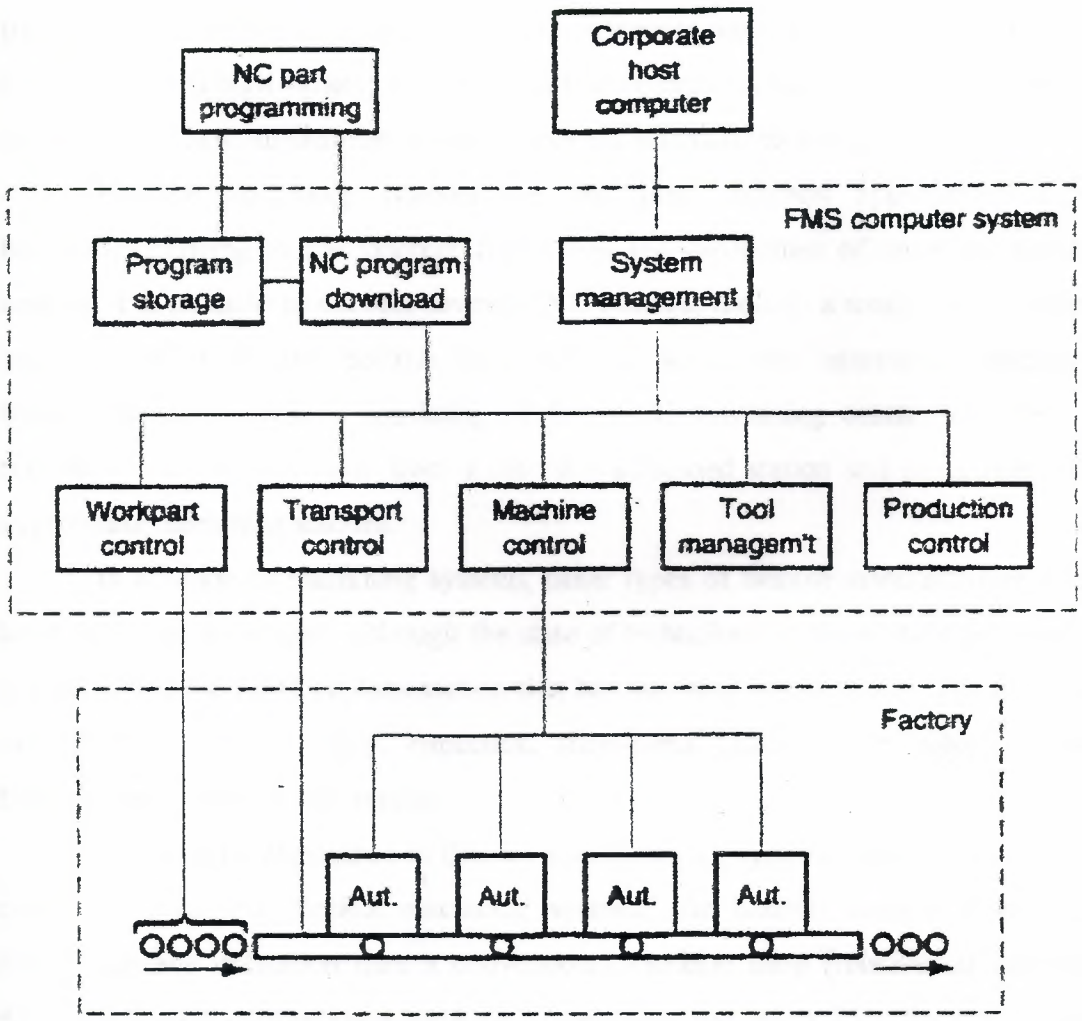


Figure2.3. Structure of FMS application software system.

Key: Aut.= automated station; NC = numerical control

2.3 Applications of Flexible Manufacturing Systems

Flexible manufacturing systems are typically used for mid-volume, mid-variety production. If the part or product is made in high quantifies with no style variations, a transfer line or similar dedicated production system is most appropriate. If the parts are low volume and high variety, numerical control or even manual methods would be more appropriate. These application characteristics are summarized in Figure 2.4.

Flexible machining systems are the most common application of FMS technology. Owing to the inherent flexibilities and capabilities of computer numerical control, it is possible to connect several CNC machine tools to a small central computer and to devise automated methods for transferring work-parts between the machines. A flexible machining system consisting of five CNC machining centers and an in-line transfer system to pick parts from a central load/unload station and move them to the appropriate machining stations.

In addition to machining systems, other types of flexible manufacturing systems have also been developed, although the state of technology in these other processes has not permitted the rapid implementation that has occurred in machining. The other types of systems include assembly, inspection, sheet-metal processing (punching, shearing, bending, and forming), and forging.

Most of the experience in flexible manufacturing systems has been gained in the machining area. For flexible machining systems, the benefits usually given are (I) higher machine utilization than a conventional machine shop (relative utilizations are 40% to 50% for conventional batch-type operations and about 75% for a FMS due to better work handling, off-line setups, and improved scheduling); (2) reduced work in process due to continuous production rather than hatch production; (3) lower manufacturing lead tines; and (4) greater flexibility in production scheduling.

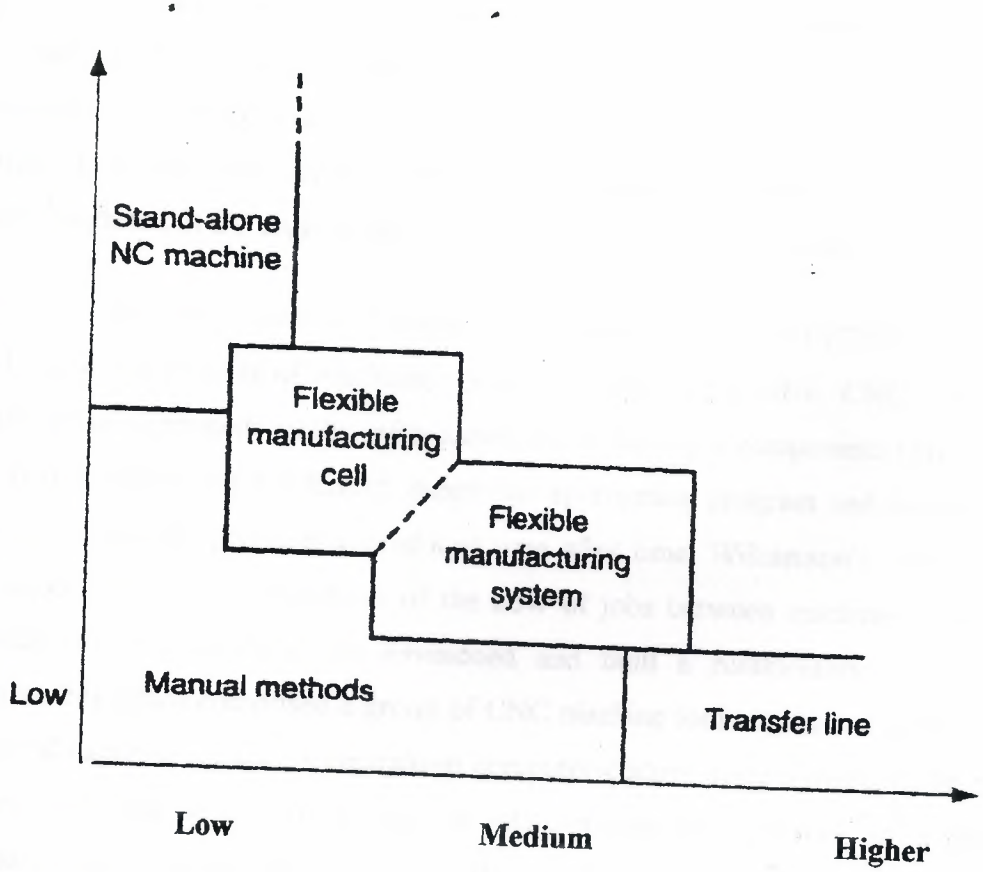
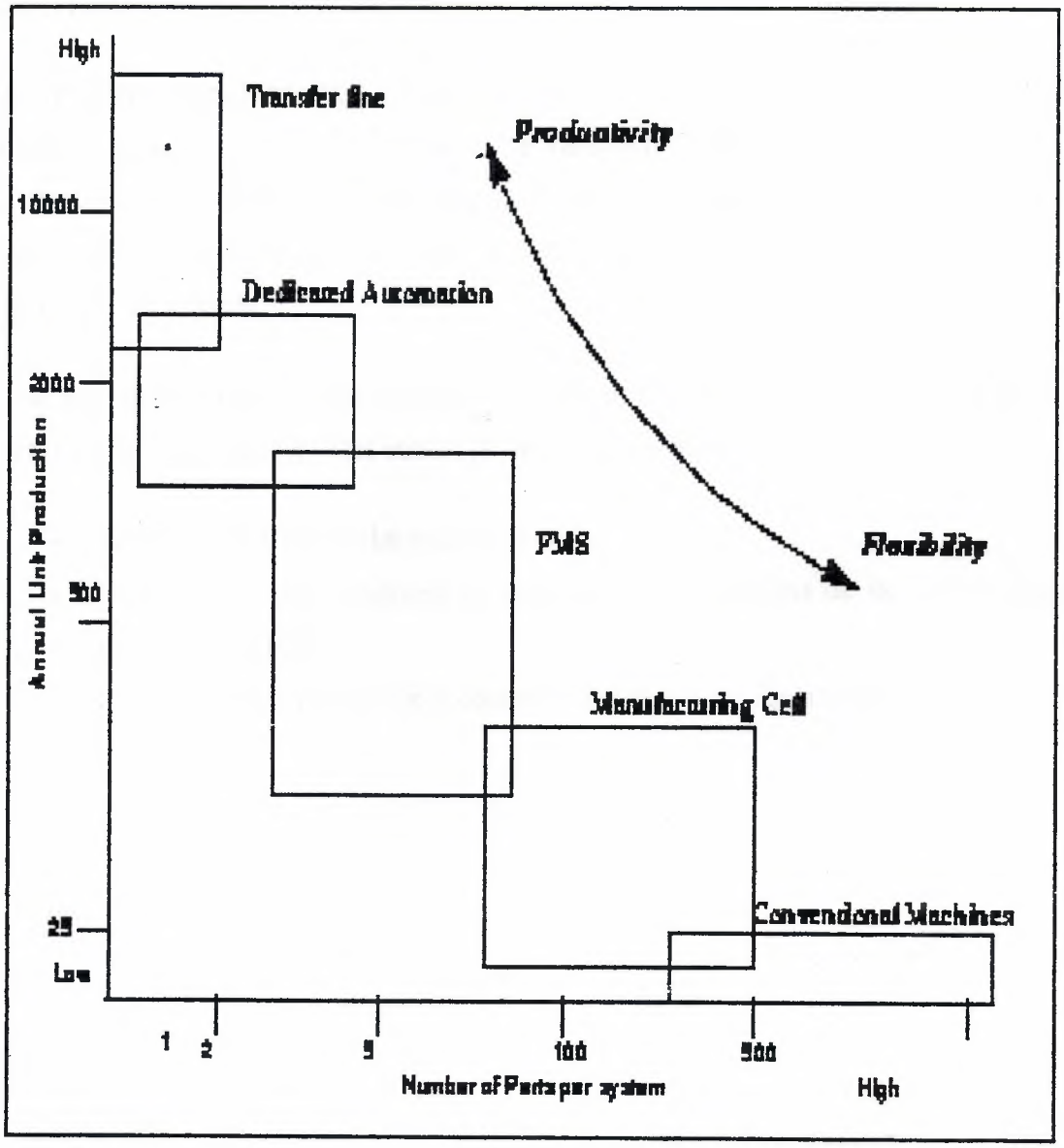


Figure 2.4. Application characteristics of flexible manufacturing systems and cells relative to other types of production systems.

Flexibility is one of the benefits of small-batch manufacturing. While a small-batch shop may produce lower unit output than a shop dedicated to one or two lines, its strength is that it can make a variety of different products in small volumes. The complexity of coordinating manual small-batch production had, until the early 1980s, confined automation of the manufacturing system as a whole to industries producing in large-batches, with a small, slowly-changing range of products. Small-batch production relied on stand-alone processing machines, which were coordinated by human operators and schedulers. The complex nature of producing a wide-range of products brought

what were seen as necessary evils accommodated in the name of flexible manufacturing. If a company was machining earthmover axle components or valve housings in small batches then high inventory, unpredictable, long lead times and quality problems were very common. Manufacturing engineers such as Theo Williamson in the 1960's were inspired by the idea of being able to bring the controllable advantages of the transfer line to the more complicated world of small-batch machining manufacture. This was an important problem: 75% of the value of items produced in US engineering firms was (and is) made in batches of 50 or less.

By the 1970's, the advent of Computer Numerically Controlled (CNC) machine tools had made the process of machining both automatic and flexible. CNC machine tools could be programmed, locally, with a method of making a component. One merely had to load a casting onto a fixture, supply an appropriate program and tooling, and the product would be predictably produced time after time. Williamson's contribution was to suggest that the coordination of the flow of jobs between machines could also be carried out automatically. He envisioned and built a rudimentary system (Molin's System 24) which comprised a group of CNC machine tools connected by an automatic material handling system. A centralized computer-control system oversaw the shop and coordinated and scheduled the flow of jobs between the machines. With the further advances in computer technology, and the stabilization of CNC technology, the early 1980s saw a flurry of installations of systems designed along the lines of Williamson's System 24. Pioneering companies such as Caterpillar and John Deere in the US started to build large systems which went against traditional manufacturing dogma - systems which combined economies of scope and scale. These large computer-controlled systems had a relatively high aggregate output yet were flexible, since they could produce a number of different products.



Flexible Manufacturing Systems (FMS), as they were called, became a great focus of attention in industry and in academic research for a number of years. Although the more skeptical might say that behind the rapid growth of publicity and interest in FMS lay a bubble inflated by a sales-hungry machine-tool industry, it was nevertheless clear that the systems demonstrated a significant technical advance in manufacturing practice. The real strength of these FMS lay in the fact that they brought tremendous benefits in inventory reduction (often 85%), quality improvement and lead time. In many installations, the inventory reduction alone was sufficient to justify the investment in hardware, software and system design effort.

2.5 Traditional FMS

A flexible manufacturing system (FMS) is an arrangement of machines.... interconnected by a transport system. The transporter carries work to the machines on pallets or other interface units so that work-machine registration is accurate, rapid and automatic. A central computer controls both machines and transport system... National Bureau of Standards.

The key idea in FMS is that the co-ordination of the flow of work is carried out by a central control computer. This computer performs functions such as:

- Scheduling jobs onto the machine tools
- Downloading part-programs (giving detailed instructions on how to produce a part) to the machines.
- Sending instructions to the automated vehicle system for transportation

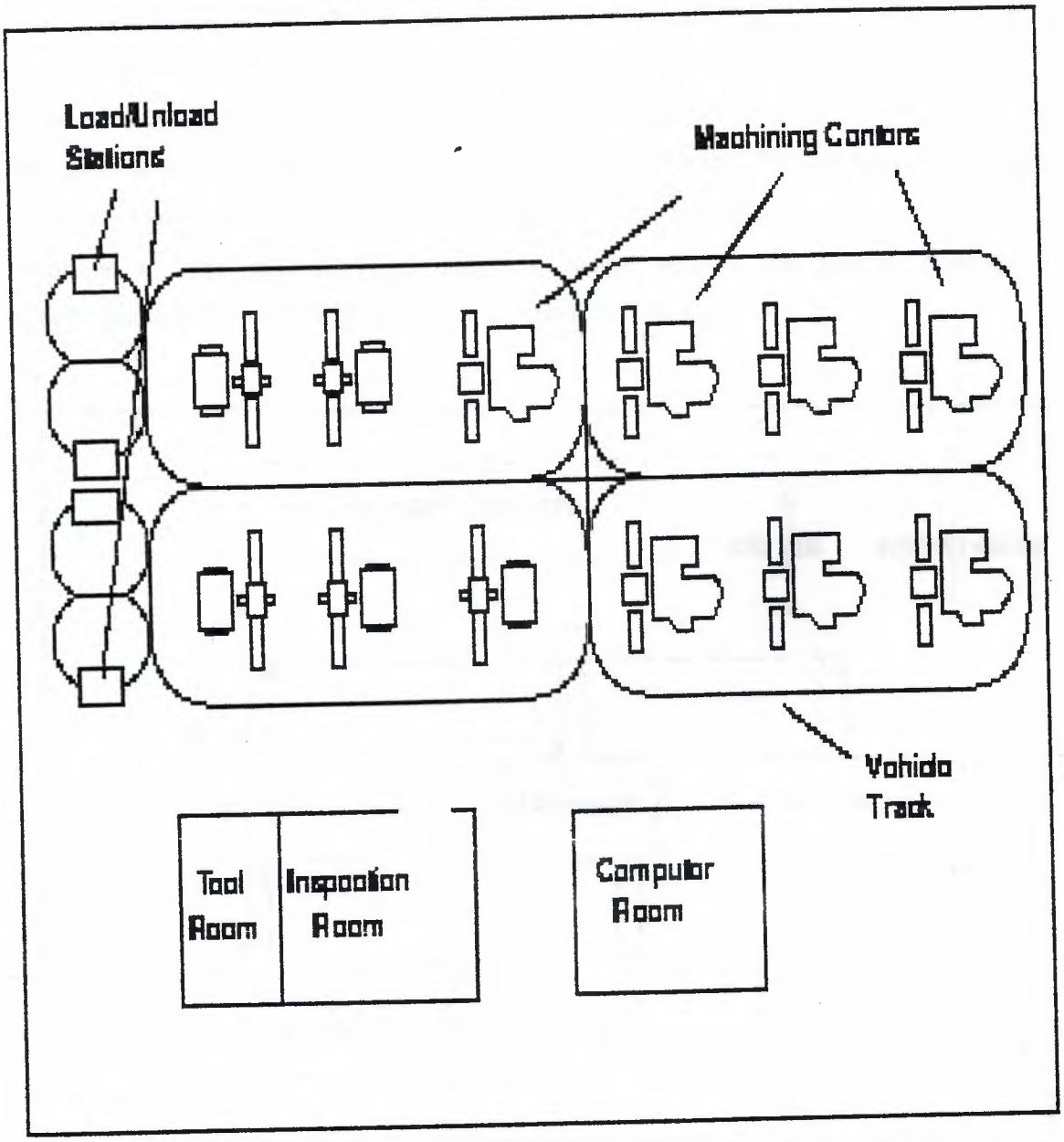


Figure 2.5. A Flexible Manufacturing System

Products to be produced are manually loaded onto pallets at a load station, and the computer system takes over, moving the product to the various processing stations using automatic vehicles, which may be rail-guided, guided by wires embedded in the floor or free-roving. After having visited all necessary stations, usually only two or three, the job is taken back to the load station, where it is removed from the pallet and passed to the next process.

Writing FMS control software is not a trivial matter. The software is often custom-written, and is not straightforward programming task. There are complex, real-time interactions with remote hardware which require great expertise and experience on the part of the programmer, particularly for larger systems. In order to simplify this problem, many systems use a hierarchical approach to real-time control. Each computer controls a team of underlings, collecting status reports and issuing commands. Commands flow down the hierarchy, while status reports flow up.

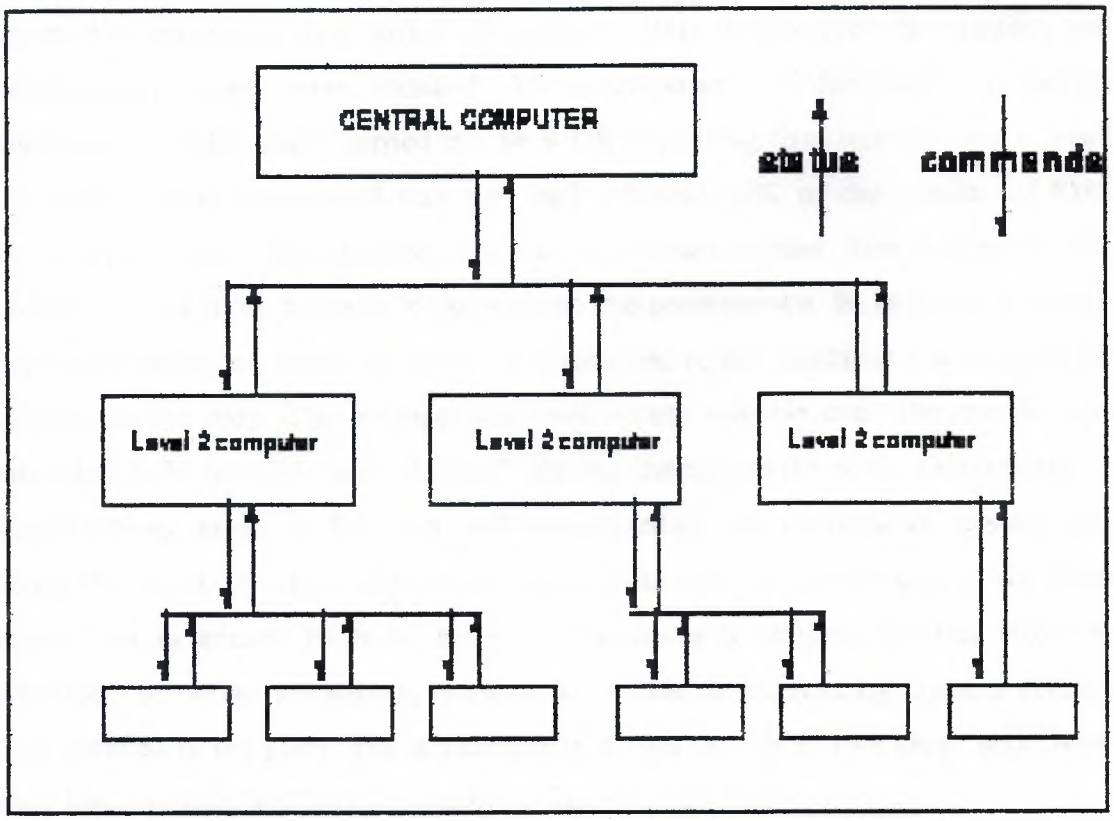


Figure 2.6. Hierarchical Control

Various estimates are that between 20% and 40% of the cost of FMS installations are in computer software and hardware development.

2.6 FMS as a panacea for small-batch machining

While the interest in FMS as a solution to the problem of automating the job-shop was growing rapidly in the early 1980s, some researchers began to urge caution. Jaikumar's work showed a marked difference between the US and Japan in the advantage taken of the flexibility possible with FMS. Far from exploiting the possibilities of the technology, US manufacturers were using FMS as fixed lines which happened to produce a small group of products, rather than to provide versatility and mutability after they were installed. The management of flexibility was poorly understood. Another study carried out by a UK consulting firm reported that a small group of companies estimated that they had achieved 40% of the benefits of FMS before any hardware was installed or a line of software written. The reasons for this became clear as more companies experienced the phenomenon. In order to automate shop coordination, the company needed to understand it, and formalize a solution to the problems in the shop. This process was itself a very valuable one. The coordination system had to be brought "under control". Having understood the needs, rationalized the products being made in the shop, and understanding the coordination system well enough to avoid obvious inefficiencies, the advantages of computerizing the shop became less significant. Even so, there is no evidence to suggest that this 40% was achievable, let alone sustainable, without a Flexible Manufacturing System actually being installed in the plant. The advantages of a well-run FMS were clear: short lead-times, low inventory and a step towards the factory of the future.

2.7 The significance of FMS in the 1990s

The installed worldwide FMS base in 1989 was estimated to be around be between 500 and 1200 systems, the higher figure arising when a system is defined as having 2 or more CNC machine tools connected by a materials handling system, and controlled by a central computer. Ranta and Tchijov suggest that this number will rise to around 2500-3500 by the year 2000. This led them to suggest that "the strategic majority of production of the metal-working industries in the industrialized countries will be produced by FMS or similar systems [by the year 2000]."

Kelley's empirical research in 1987 strongly contradicts this. In a large (>1000 firms) survey of US metal-working firms, she found that less than 5% of those plants ***with*** computerized automation have an FMS and that FMS constituted only 1.5% of the total number of installations of computerized automation. Why are there still so few FMS in the world given that small-batch engineering production is a significant proportion of manufacturing output? There are significant practical reasons for the disparity between the promise of FMS in the 1980s and its narrowness and scarcity of application in the early 1990s. These reasons are outlined below separately, though they are very much interdependent.

2.7.1 Narrow Process Focus

The types of manufacturing processes suitable for integration into traditional FMS remain limited: turning, milling and sheet metal work dominate FMS processes while many other, less well automated, processes remain unintegrated. This is mainly because they are not computerized at the machine level and are hence not yet ready for computer integration at the system level. Nevertheless, even in metal *cutting*, with much wider application of Computer Numerical Control, comparatively little output is due to FMS.

2.7.2 Technological uncertainty

When FMS were first introduced, the novelty of the integration technology naturally made many firms "wait-and-see" until the technology had settled. This was particularly true in the smaller companies. The technology of FMS has, at least in the West, not become mature and well understood and many companies would still consider FMS technology to be "high-risk".

2.7.3 All-or-Nothing

The monolithic all-or-nothing nature of FMS increases the risk of projects, causing companies to shy away. This is particularly true of those companies whose products are a little different from those for which FMS has already proven itself the scale of the effort required, in conjunction with their less standard processes is sufficient to dissuade them from undertaking the project.

2.7.4 Productivity

In many applications, the productivity of the prospective system --- in terms of its output with respect to its capital input --- is insufficient. Practical experience has also shown that the utilization of the systems may be much lower than predicted when they were designed, further reducing productivity. While productivity may not be the manufacturing performance criterion most closely associated with the competitive focus of the system, there are bare minima to be exceed in any industry. Without a reasonable level of practical productivity (and hence return) from capital, the project will founder, perhaps rightly, in the capital investment procedures of the firm.

2.7.5 Shallow learning curve

It takes a long time for an organization to learn about FMS technology. Much of the technology is embodied in software integration, and software engineering is not a skill which many manufacturing companies acquire quickly.

Second, the highly interdependent and specialized nature of the technology means that integration is best handled by a very tight nucleus of people . While this might get the job done at the outset (once these scarce people have been found), it often means that just a few people hold the key competencies. This concentration of knowledge inhibits learning in the organization as a whole.

The nature of the skills required means that these skilled people have often been imported from outside the firm and owe it only fleeting allegiance. When they leave, they take their skills with them, which further flatten the learning curve of the company.

2.7.6 Level of Investment

The investment in FMS (as characterized by Ingersoll Engineers) is often in the range of $10 to 15 million. The amount of money needed to finance an FMS is thus a significant barrier to its introduction, particularly in smaller companies. Smaller firms currently perform most of the small batch work, so it is here where FMS would be most appropriate. However, for most small firms, an investment in FMS would mean "betting

the farm". Quite reasonably, given the plethora of other difficulties, they choose not to. These six reasons, in concert, marshal against the diffusion of current FMS technology. This is not to say however, that these are sound reasons why FMS should not be embraced. Many argue that the difficulties described above are the price one has to pay, and that technologies such as FMS must be seen as a strategic investment --- the short-term hurdles must be compared against the long-term strategic and intangible cost of being ignorant of the technology. If this argument were truly compelling, one might expect many more of the forward-thinking companies, whose competitiveness is tightly linked to their small-batch effectiveness to have grasped the nettle, and to have adopted FMS technology as a stepping-stone towards the future factory and as a strategic investment in the flexible technology of the 21st-century plant. This is not the case. There are foreboding reasons (inherent in the existing technology) why current FMS, even when justified on strategic grounds, simply do not make sense.

2.8 FMS as a strategic cul de sac

Despite many claims that FMS investment should be viewed as a strategic investment in flexibility, I will argue that FMS as they are currently structured are often characterized by a common feature: their inflexibility.

2.8.1 Inflexible Flexible Manufacturing Systems

The main disadvantage with FMS technology lies, paradoxically, in its inflexibility. FMS are flexible in that they can, in the short-term, produce a range of known products. However, the complexity necessary to automatically achieve short-term flexibility makes it difficult to introduce new families of products into the system, and certainly much more difficult than in a manual shop. Similarly, when new machines are to be added (or old ones updated) it can be very costly. Changes in system configuration require time-consuming, expensive alteration to software, particularly in complex, Western systems.

IIASA's FMS database shows that successful systems with payback in less than 5 years are either

- very small (< 4 CNC machines) and simple

or are

- very large (> 15 machines) and complex.

Simple systems work because they may be committed to a focussed group of components, and be relatively unsophisticated in their use of software. Large systems work because the cost of the control software may be distributed over more output. Even the large systems, however, show unimpressive returns given the risks involved, and only moderate ability to introduce new products. Only 18 of the 800 FMS in IIASA's sample belong to this category.

The software for controlling medium/large FMS has to handle the tremendous complexity of scheduling and dispatching multiple products through a variety of processing routes, transporting them around the system and recovering from any failures in system components. This cleverness means very complex software. The complexity of the software necessary to provide short-term flexibility has frequently become the millstone which constrains long-term flexibility.

While companies may have an idea of the functions of products they might be producing in five years' time, they may be unable to guarantee that the components in those products conform to a specific engineering family, and this is what FMS often currently demands. If new products need to be introduced, and the manufacturing system equipped with new machines here and there, FMS will exact a high fee, particularly for firms without sophisticated expertise. Research has found that the considerable expertise assembled to install the original system has often dwindled by the time it is necessary to re-configure it.

For new families of products and the addition of new machines, FMS simply costs a lot to change: surely a definition of inflexibility rather than flexibility. This means that companies are less nimble in chasing changing markets with new product ranges.

2.8.2 Failure and recovery as sources of complexity

In a complex, automatic system like an FMS, many failures must be anticipated and catered for. For example, tool breakage, and machine and vehicle failure are fairly common. In order to deal with such circumstances, methods for recovering from the failure must be built into the central control software. This has a number of effects.

- The control software becomes more complex, further limiting long-term flexibility.
- Any changes to the software also demand similar recovery procedures built into them, making the changes even more expensive.
- In order to run automatically, the system often has to use many sensors so that the control computer can deduce the failure state of the system and can recover from it.

Despite all the effort put into the software, the sheer complexity of running a small-batch shop means that the system programs cannot anticipate all possible failure states, which means that manual intervention is inevitable. It is clear that a centralized, all-seeing, all-knowing control computer is not possible --and the closer systems come to having one, the more complex (and less long-term flexible) they become. The software becomes hardware.

2.9 The demise of monolithic FMS?

The evidence paints a bleak, inflexible picture of FMS. In spite of the occasional well-publicized exception, the promise of the 1980s has not been fulfilled. Many companies have seen their inflexible white elephants incur tremendous costs, as markets shifted and their "flexible" manufacturing system was found to be unable to accommodate the change. This is reflected in the market. Sales of FMS, particularly large, complex systems with many machine tools, have slowed markedly, even when one takes into account the recession. The reasons for this are slowly becoming clear: as a strategic investment in automated flexibility, FMS as they are currently conceived are anything but.

2.10 An Alternative Approach

One solution to this problem (and one which many companies have adopted) is to accept the need for long-term flexibility in their manufacturing system, and provide it with manual, skilled labor and CNC machines. This is a very flexible system. Just-in-time methods, Group Technology (GT) and manufacturing cells all help the manual small-batch shop combat the tendency to disorganized chaos which is often seen in the industry. These techniques are all effective methods of managing the flow of work around the shop, especially for smaller parts. However, this solution essentially abandons automation, and demands that people continue to attend to the processes required to make the part, though they may attend to multiple machines. How might one free people to do more meaningful work, than attending to machines which are, nowadays, largely automatic? How might one go about controlling or *orchestrating* a large (say 50) congregation of CNC machine tools without the long-term inflexibility of the traditional FMS, and without the need for human intervention?

The following section describes one alternative approach to a hierarchical FMS for a machining system.

2.10.1 The Computerized Product in a Society of Machines

Consider a machine shop with fifty CNC machine tools and automatic vehicles for transportation of materials. This section outlines a new type of control structure for such a system. It is best outlined by describing an example of the production procedure which is followed in order to produce an individual item. A production-control computer requests that a casting which has arrived at the shop be machined. The casting is manually bolted in a flexible fixture on a pallet, and a small computer is fitted to the pallet. This computer contains a processor, some memory and a radio. This assembly will be referred to as "the part". The production control system loads the memory of the part's computer with the processing requirements of the product. In other words, the control system tells the part what it needs to look like. The product enters a short system entry buffer, and while it is waiting, its computer broadcasts its description throughout the system to the flexible CNC machine tools. Some machines examine the job's

description and decide that they are unable to machine the product because it is too big for their bed, say. Others simply have the wrong type of geometry for the job. Still others decide that they can do the processing work on the product. The machines' computers plan a process for the job, and decide how long it would take them (and how much it might cost in terms of tool-wear) to do the work.

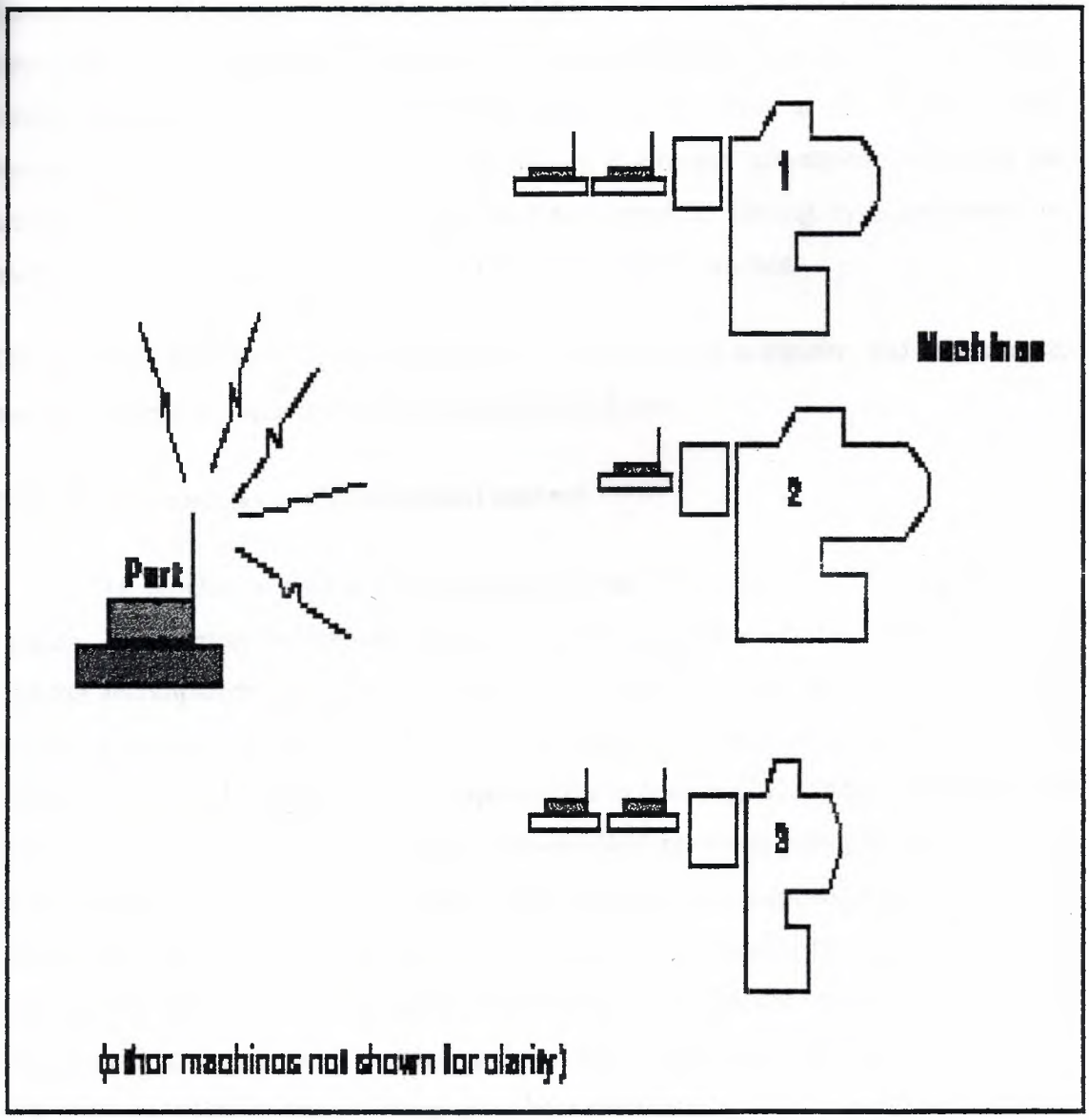


Figure 2.7. Part transmits processing request and data to the system.

Having determined how much processing time is needed, the machine checks its local buffer, determines how many jobs it has waiting for it, and forms a "bid" on the job. It transmits this bid across the network to the waiting part. The part waits until a system-set deadline to receive bids and having collected them, selects a *winner* from the bidding machines. It sends a message to that machine that it has selected it and expects to arrive for processing. The next arrangement the part needs to make is for transportation to the machine. Here there are a number of possibilities but let us say, for now, that the part essentially "calls a cab": It requests transportation from an automated vehicle dispatcher which sends a vehicle to it, and it arrives at the machine. After waiting in line it is machined. While it is waiting it arranges subsequent machining and ultimately leaves the system, once all tasks are complete. Having been processed the part moves out of the system to be assembled into its final product.

The part has thus been processed *without* a central control computer, and with simple, modular, physically decentralized hardware and software.

2.11 The Advantages of Heterarchical control

The benefits of such a Heterarchical approach to system control are quite wide-ranging. The control method obviates the need for an on-line central control computer and the accompanying software. A number of authors describe variations of this basic approach to a variety of control problems. Pioneering manufacturing work was carried out by Duffie and Parunak. Shaw explores the scheduling advantages of distributed control over its centralized counterpart. An excellent review is provided by Dilts et al.. In the system described in this paper, each machine is equipped with an on-board process planner, which determines how *that machine* will process the job. This allows a wide variety of machines to contribute to the capacity of the system. The machines also have a local (cheap) bidding microcomputer which is a interfaced to the process planner and with its communicator. These may be retrofitted to CNC machines which are already in the company.

A shop run like this reflects the nature of manual job-shops much more closely than a rigid hierarchy of computers. While the original attempts at hierarchical control attempted to mirror the hierarchical function of a manual shop (foreman ë\xdf chargehand \xdf operator), they ignored the fact that the most decisions in a job-shop

are made by interaction and negotiation within a hierarchical level. The shop described here more closely resembles a manual job-shop run in a Just-In-Time fashion.

2.11.1 Short-term flexibility

In order to make FMS control feasible, many companies split their manufacturing systems into manageable "chunks" of 7-12 CNC machines and integrate these into an FMS. In manual production, machine shops of 1000 machines are not unusual. An often-quoted figure is that 1 CNC machine tool can match the capacity of 5-10 manually controlled, stand-alone machine tools. Where then are the FMS equivalents of the large shops, with 100 or 200 CNC machines integrated together? The answer is that such a shop would not be economically controllable with traditional, centralized FMS. Few companies would have the skills available to put one together as a monolith. In any case, the poor long-term flexibility caused by the complexity needed for the system would make it financially and strategically unattractive. For this reason, FMS are limited in size by their controllability and poor flexibility. However, in a distributed system like this, there need be no such size constraints and the part may avail itself of a much larger group of machines, not just the one which it has access to in its traditionally controllable chunk. The fact that the part has access to more machines, and that no group of components is specified in advance for those machines to make, means that many different parts can be serviced appropriately by the system.

The relative advantage here depends to a large extent on the degree of multiple redundancies in the system as a whole. Such redundancy is becoming more prevalent as a result of technological changes in machine tools. Today's CNC machines are considerably more versatile than they were 15 years ago. The move from 3-axis to 5 and 6 axis machines, the use of modular tooling systems and pallet-changers along with the improvement of tool-management systems have dramatically increased the scope of jobs which an individual machine can undertake. This trend is likely to continue as machine tool manufacturers accommodate more and more operations into one work piece setting and one machine tool. An allied trend is the increase in information processing ability at the machine level. The continuing technological advances in physical versatility and in the distribution of computing in machine tools imply increased multiple redundancy in systems, as well as local, rather than central, information processing.

2.11.2 Long-term Flexibility

Consider the software changes necessary in order to add a machine to this system. An important question is how long the manufacturing system has to be down in order to implement the software changes. In this system, there is no need to take the system down. A new machine may simply be told the "rules of the game", and be installed in the plant with power and access to tooling. It becomes part of the system with no lost production. Removing a machine from the system is also straightforward. The machine simply stops bidding, and jobs stop coming to it.

Each addition and removal has very limited system-wide ramifications. The machines rise and fall in utilization as they become more or less appropriate for the current product range. This greatly facilitates new product and process introductions, since any peculiar requirements for processing of a new product may be introduced without systemic disruption.

2.11.3 Recovery from Failure

The system is particularly graceful in recovering from failures. This is because it does not rely on a centralized computer, which may need to scramble to find out the failure state, or be told the failure state by a human operator. Failures are limited to the locale where they occur and system-wide consequences are avoided.

2.11.4 Machine Failure

If a machine fails, a number of local actions may be taken which avoid the need for complex centralized control software and hardware. Machines are given the rule:

If you are defunct, do not bid.
This is fortunately the default for machines whose computers are too defunct to use the rule and avoids the problem of parts being assigned to machines, which have failed.

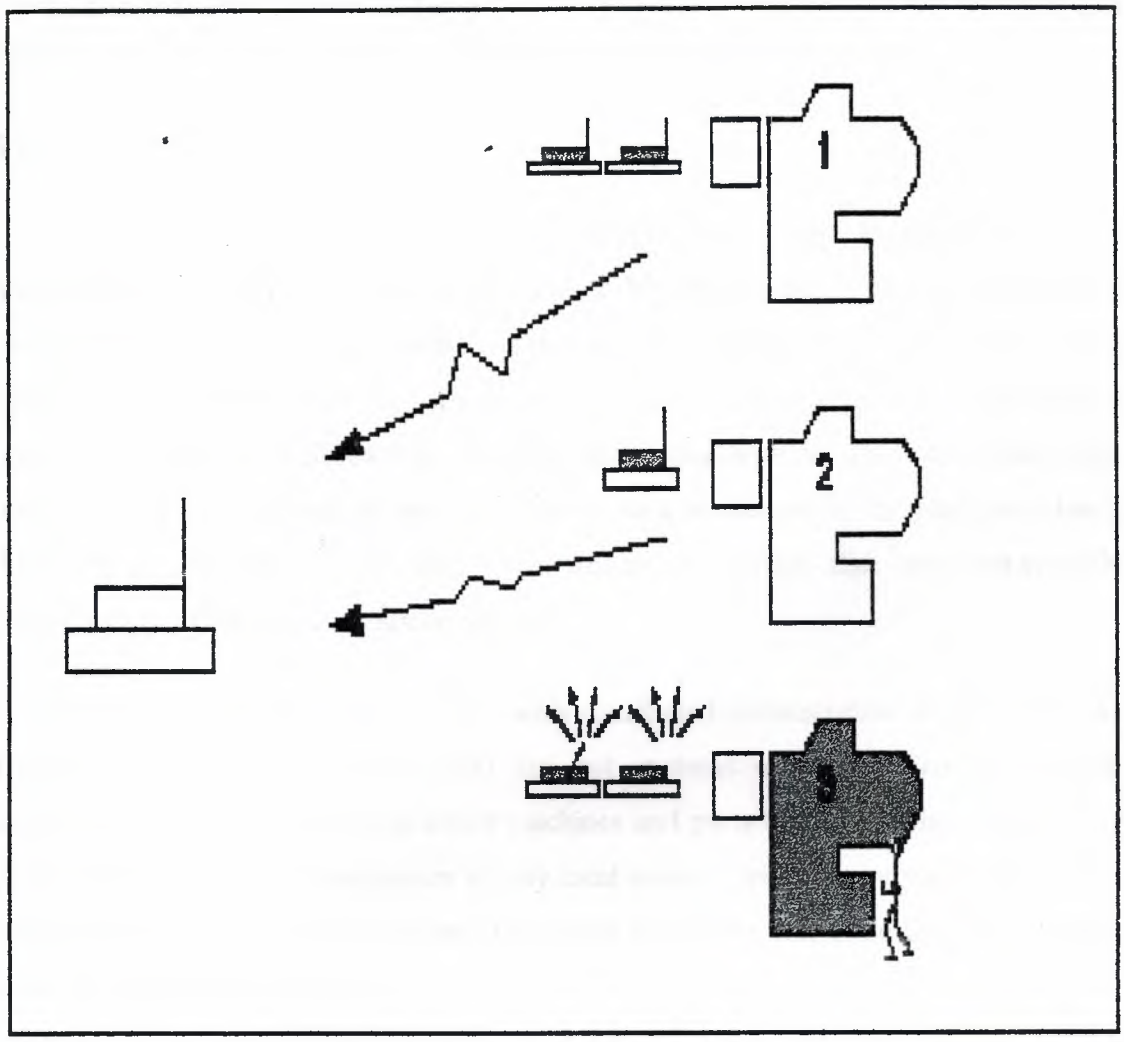


Figure 2.8 Machines transmit an estimate to the job. In this case, machine 3 is down, and fails to respond to the request.

Maintenance staff can repair the machines, and then simply allow them to start bidding again. Temporary removal and re-introduction of the machine does not require system-wide shut down procedures. Parts which are "stranded" (that is, are in the input buffer of a failed machine), have a rule which says:

If you are waiting for a machine for more than twice what you expected to wait, re-instigate the bidding procedure and find another machine.
The part will find another taker and be transported from the input buffer without human intervention. It can reinstigate the auction process and arrange to have itself removed from the carousel. The system may thus recover without more far-reaching

consequences or the need for central software to be written to deal with the situation. The part, with its simple, generic, software can arrange its own recovery.

2.11.5 Human Intervention

In a monolithic control system, humans are easily excluded from the manufacturing system for a number of reasons: If a human being changes the state of the system, then that person needs to either tell the central control computer or that computer needs to have sensors to detect the change. Such a change might include taking down an individual machine for preventative maintenance. This often means that only people with an understanding of the system as a whole can be involved, or at least, that systems engineering "experts" are around to ensure that the system-wide consequences of local actions are anticipated.

A "society" of machines such as this, with distributed orchestration ensures that the system is accessible to people who are not systems engineers. Provided people understand the basic, local rules which machines and parts obey, then they will quickly be able to foresee the consequences of any local action. Since recovery is built into the system through its distributed nature, they need not worry that these local actions will bring the whole system down.

2.12 The Learning Product

Once a part has been machined, its computer will retain a record of its processing experience. This record will maintain such information as:

Machine identifying itself as "Number 4" made the following features <feature list>. Machine bid 25 minutes, actually took 45 minutes. Suspect machine failure. or

Machine 17 failed to complete promised operations. Found successful alternative in machine 42.

The part will thus keep a record of its history. The part may relinquish its memory /computer so that other products coming through the system may make use of it, and draw inferences from the experiences of previous parts processed by the system. For example,

Machine 7 appears to underestimate its processing time. Make correction of +25% to any bids.

or

Machine 14 has a probability of failure of .07 for any job it undertakes. While the details would depend on the actual application, the idea is simple. The parts should have a collective memory, which grows and allows the system to learn about itself, and improve its own performance. The fact that the parts' memory is attached to a succession of multiple bodies (parts) does not affect this idea of progressive learning.

2.12.1 The learning product in a dynamic system

An important issue here is the provision of a mechanism for forgetting. There need to be a method by which the information which the parts learn may progressively forgotten to take into account system changes, and the repair of chronic unreliability problems. There are two mechanisms by which this make take place.

First, we may build "forgetfulness" into the system functionality. For example, if a part has just finished processing, and needs to update the current predictor of some system parameter which it has experienced, say the fractional error in processing time estimates of machine 24, τ_{24} . If this part's experience of this parameter is τ, then the new estimate of the parameter, τ^1_{24} may be generated by setting

$$\tau^1_{24} = (\alpha - 1)\,\tau_{24} + \alpha\,\tau$$

where α $(0 < \alpha < 1)$ is a parameter reflecting the "forgetfulness" of the system.

Second, we may intervene as managers of the collective memory. When a persistently faulty machine is corrected, we may wish to step in and deliberately erase all memory of the miscreant device . Parts may then begin to learn afresh from new samples, rather than slowly forgetting the now unrepresentative experience; we are thus forcing the parts to forgive quicker than they otherwise might.

2.12.2 Difference between this and conventional FMS "learning".

In a conventional FMS, the complexity of the control software act as an inhibitor which resists change in the system, and hence learning. In traditional FMS, which are

architecturally static, managers must rely on humans to notice systemic problems and to risk opening a Pandora's box of software to remedy them. The distributed method described above learns experientially and acts on that knowledge autonomously. Learning is an inherent feature of the architecture.There is much work to be done in exploring this idea; the conditions for effective and efficient learning, along with the development of criteria to determine which factors are best learned and which are best "told" are important avenues of research. Work is currently being carried out in these areas in particular, in the application of automata theory to the problem (viewing the parts as learning stochastic automata).

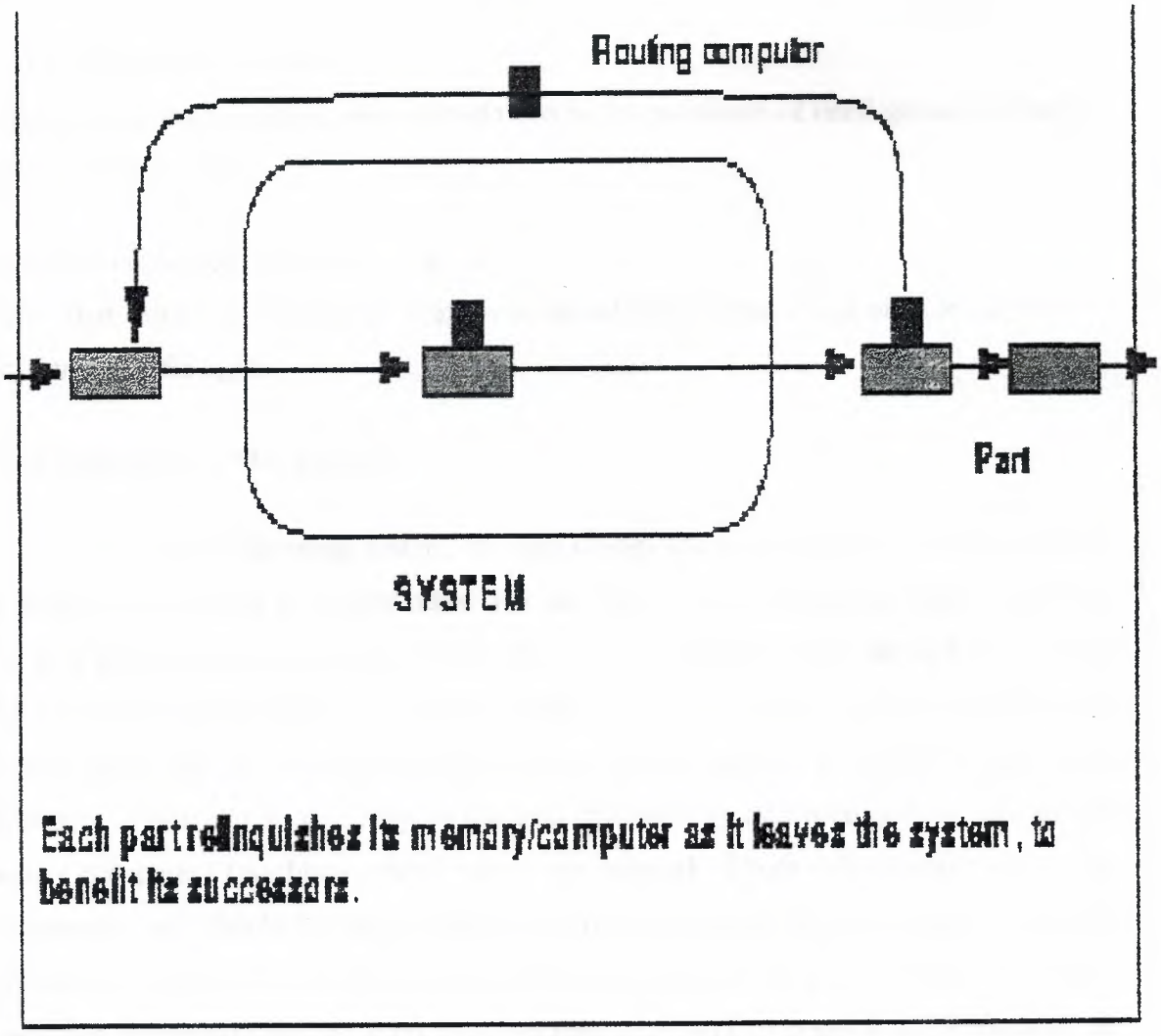


Figure 2.9. The Collective Memory

2.13 Changing the Rules in the Control Room

While the idea of allowing entities to negotiate in order to orchestrate the system is attractive, we still need some method of changing the behavior of the system in order to effect "control" at a higher level. Control in the system described above is effected by using *procedural* rather than *substantive* rules.

The system as it is described attracts a variety of metaphors. A useful one is that of the system which gets commuters to and from work by automobile in a large city every day. There is no central controller or hierarchy of computers governing the operation of the system at a micro-level with commands such as:

Car 43 turn left at 6.17 am

Instead, each entity in the system is endowed with a modicum of intelligence and has a goal with some rules.

get to work but obey the rules of the road

After that, control is effected by using *procedural* rules. These might include one-way streets or traffic lights.

2.14 Learning by Management

In the manufacturing system, we may change the rules which the various entities use when negotiating with each other. For example, we may decide to allow a machine to pick priority parts out of its buffer if their priority number is high enough (see . This, of course, will mean that commitments made to previous parts will have been broken. Nevertheless, this is one control choice which can be made in a straightforward way. Changes in procedural rules may be put into operation by transmitting new rules to the entity computers (machines, parts) across the network. These rule changes should be infrequent, and should be made after very careful consideration and simulation by the system managers. It is straightforward for the company to progressively learn the best methods for controlling the shop by experimentation and to embody these methods in the distributed rule-base.

While the computers on the floor do not operate hierarchically, the heterarchy is nevertheless embedded in a broader corporate/plant hierarchy. The shop simply behaves

as a different kind of subordinate: as a group of processors using robust, distributed rules to perform rather than a group obeying direct commands passed down a bureaucracy from a dictatorial central computer.

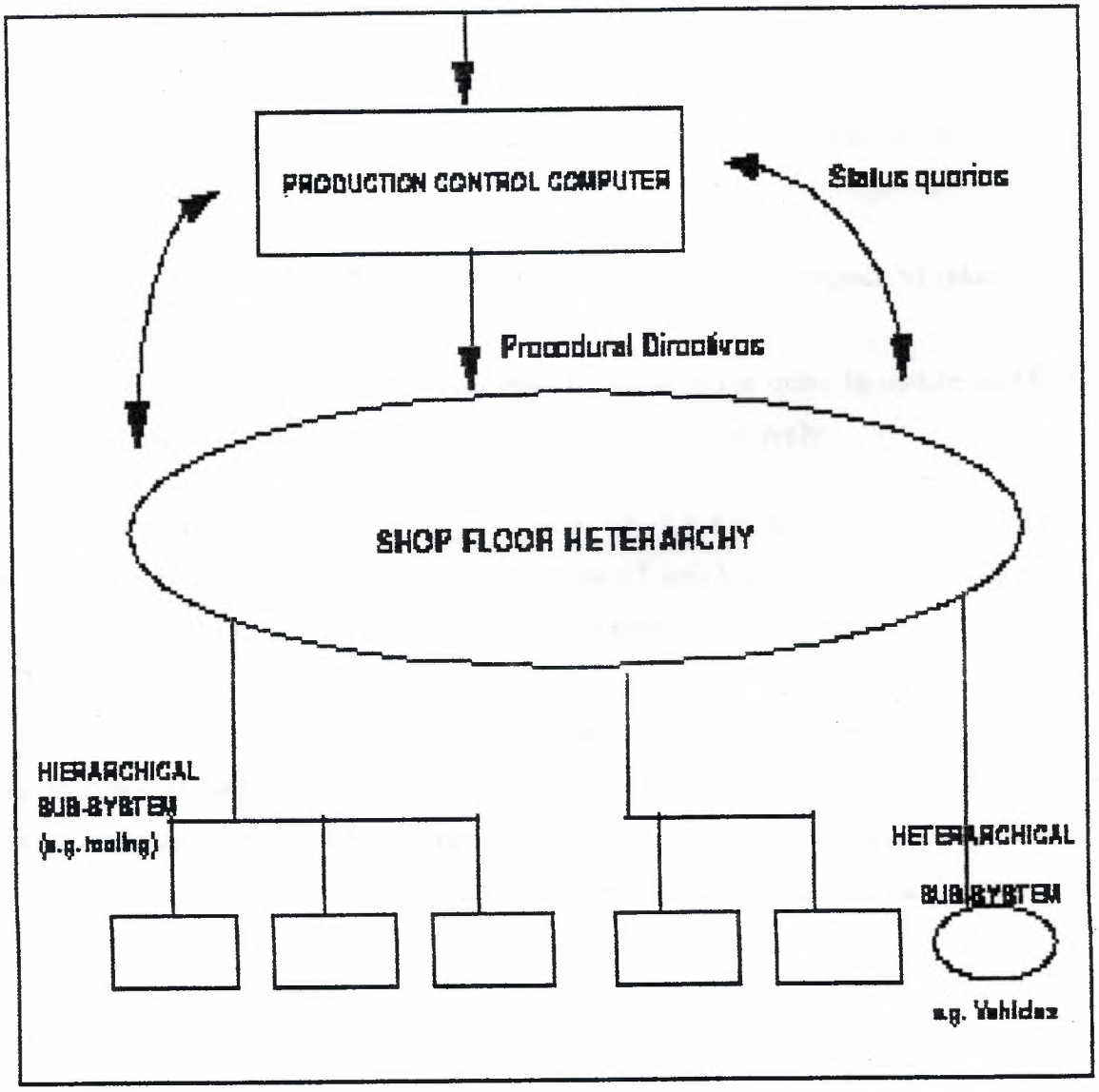


Figure 2.10. Hybrid Control (heterarchy within a hierarchy)

2.15 Data Collection for Management Control

In a traditional FMS, data about parts is collected by the central control computer whether it is needed or not. Hopefully, when it is needed it is up-to-date and accurate - unfortunately, manufacturing is not a deterministic activity.

This system does not, in general, collect data as a matter of course. If information about a job is required, the system is interrogated in real-time across the network. An example of such a question is:

Part 17843, Order 2873. Where are you? How long do you expect to take to be completed?
The production control computer might need to know this in order to update an MRP system running in the plant (as an example only). The part may reply:

At least 38 minutes. I am currently at Machine 4, which bid 17 minutes on the current processes. I then need to be processed by Machine 17 which took 21 minutes the last time it processed a part of my type. It does have a history of taking a mean of 3 minutes longer than it estimates.
(Natural language is used for communication where possible to increase the transparency of the system for its operators). The principle is that data is not collected and stored only to go out of date or not be required. A query system is provided, and the information collected in real-time, from the entities closest to the action (as shown in Figure 2.10.).

2.15.1 Managing Bottlenecks

Most manufacturing systems are characterized by having some processes, which can be handled by only one or two machines. Despite the redundant nature of the machines in this system, there will still be processes, which may only be effected by a small group of machines, or even just one machine. In a highly dynamic system, with multiple redundancy, such capacity bottlenecks will not be constant in either severity or location, since the processing requirements and mix of jobs changes over time. In addition, such bottlenecks may not be on the ultimate processing path for all jobs. While any manufacturing system must have a bottleneck, the managerial problem concerns identifying those bottlenecks which cause a high degree of imbalance of workload

between machines and ensuring that jobs with a unique requirement for the bottlenecks are served in preference to other jobs which have alternatives available. For this reason, we may divide bottlenecks into two classes for the system described here: global and specific.

Global bottlenecks are those processors which currently limit the capacity of the system as a whole. These bottlenecks may be identified by asking machines (by broadcast message) to report their current utilization. It is then up to the system managers to decide whether they wish to relieve the bottleneck by supplying another suitably equipped machine.

Specific bottlenecks apply more severely to a particular group of jobs, and grow more troublesome as the demand for those jobs increases. In order that these jobs are not further delayed at their specific bottlenecks (due to jobs which could have gone to some other machine), machines may have a rule which says (for example):

If your utilization is more than 90%, confine your bidding to jobs which have been rejected by the system more than 15 times in the last 5 minutes. In this way, machines may intelligently pick up orphan parts, and ignore jobs which can find alternatives processors, regardless of how effective they are in producing those parts.

2.15.2 Managing Priority Jobs

It is also necessary to provide a mechanism for priority jobs, and allow them to jump ahead of jobs which have already arranged processing. Methods for providing this capability as an integral part of the structure described above are the subject of an ongoing research project. Some of the specific techniques are outlined in general terms below:

2.15.2.1 Grand Auction Reversal

When the system becomes very busy, and there are a large number of priority "grades" among the parts, we may allow the direction of auction to change throughout system. That is, machines no longer bid on jobs, rather, jobs bid on machines. Thus, the customers become the servers, and jobs bid for machine time with some function of

their priority as currency. In these circumstances, all entities must recognize conditions which "flip" the system into this new state. The key issues here are:

- the development of criteria for reversing auction direction: for example, number of late jobs, global average machine utilization, utilization of bottlenecks.
- the distributed management of the transition state
- the exploration of auction protocols for this "reversed" state - this reversed state may be indeed be the "normal" state for some systems.

2.15.2.2. Local Auction Reversal

Similarly, there are methods by which an individual job may pre-empt machines involved in the "normal" bidding. Jobs may be tagged with a *trump card*, giving machines permission to bid on them, while ignoring existing pending jobs, and allowing them to bring the job to the front of their processing queue if they are successful in winning the auction. This is one of the simpler solutions to the priority problem, particularly in cases when there are only two classes of job (urgent and regular).

2.15.2.3. Renegade machines

We may allow only a small group of the machines in the community to renege on contracts they have previously arranged. The provision of a group of renegade machines permits the swift processing of urgent jobs, but allows the system as a whole to avoid continually evaluating the prospect of breaking existing contracts.

As the above research continues, there will emerge many opportunities for operations researchers. A very small part of the vast existing operations research literature on scheduling and dispatch is useful here. The reason that most of the work is not useful, is that operations research model of scheduling and dispatch usually assume the existence of a centralized being (computer/human) whose commands the system obeys. We have removed this object, for the sake of producing a long-term competitive manufacturing system. This means however, that there are a number of practical challenges for the OR community working in manufacturing: the practicable integration of effective heuristics with the control architecture which must affect them.

2.15.3 Enabling Technologies

There are some infrastructural elements which are essential in order to allow a system like this to work. Much of this infra-structure is a result of on-going research in a variety of technologies.

Various associated technological advances are also needed, in addition to those required for infrastructure.

2.15.3.1 Intelligent Parts

The idea of putting memory into parts is not new, and it is certainly not very difficult to also endow them with some processing capability. A small computer, with the ability to carry part-descriptions in static RAM is the minimum requirement. Otto Bilz in Germany already have a system for cutting tools which allows the tool to carry its offset information electronically and communicate by UHF radio to the machine tool on which it is being used.

2.15.3.2 Radios

Cheap radio communications systems for entities in the system are essential. These must have access to a network which allows both broadcast and point-to-point messages, and allows seamless addition/removal of nodes without central control. A number of network protocols now have this capability.

2.15.3.3 Cheap computers

In order for each job in the system to have its own computer, computers need to be relatively cheap. For some products, it may be worth leaving its manufacturing computer embedded in it for the rest of its life so that manufacturing information may be retrieved at some time in the future for quality control or replacement reasons. A copy of the parts memory will, of course, have been made in order to perform the learning functions.

2.15.4 Advances in process planning

The automatic generation of process plans for parts is currently feasible only in manufacturing research laboratories for general products. Headway has been made in research and industry in order to make this a possibility.

The important thing is that a part's description of itself and its processing requirements should not be so specific as to pre-select the machine which will ultimately make part of it, but also not be so general (a full 3D description for example) that exceptional local machine process planners are required and the data carried by the part is too expansive. The dramatic increase in the flexibility of CNC machine tools is, however, making more "generic" part programs increasingly practicable.

Research is currently underway exploring such issues as learning, auction-reversal (where, in a very busy system, parts begin to bid on machines rather than vice-versa), the control of tooling and the effect of various communications protocols. Research on the system described here has been carried out in both discrete-event and object-oriented simulation. Performance predictions are promising. A number of less obvious advantages show themselves in a more detailed investigation. For example, it appears that these systems organically form and decompose virtual manufacturing cells without the need to pre-specify them. (A virtual manufacturing cell is a small pre-specified sub-system of machines which are temporarily aggregated as a cell without physical collocation. The virtual cell is described by McLean et al. in)

There are many shortcomings in the system described here and it will not work in all circumstances. We believe, however, that advancing technology in CNC machine tools, in process planning and in computers is progressively making the "monolithic FMS" solution to flexible manufacturing system control unacceptably ineffective. Not only are these systems not flexible, but they do not take advantage of the technologies which are becoming available.

The system described is dynamic and flexible in the long-term and straightforward to reconfigure; new machines may be added and old ones removed without affecting the system as a whole. A less sophisticated degree of expertise is required of the operators in the system, and they are able to interact with it without

having to have an in-depth understanding of any broad systemic implications of their local actions. While the shop takes advantage of those tasks which humans are good at, it is much less chaotic and controlled than a manual shop. For example, as the plant learns about successful shop-floor procedures, they become embodied in the controllers of the entities in the system, rather than being forgotten as they might be in a manual shop. Failure and recovery are handled by virtue of the structure of the system rather than by anticipating actions for every possible failure state in a central computer which won't know what the failure state is anyway.

There are some useful analogies from economics (this is like a market) or from organizational behavior (this is like cooperation/negotiation). While these things may be true, and may provide some insights, it is important to emphasize that this is a straightforward engineering solution to the problem of distributed orchestration, and came from application of engineering and computer-science techniques rather than from a forced economic or anthropomorphic analogy. Clearly, societal manufacturing systems like this are a few years away. However, we are convinced that such structures provide a clear direction for advance in manufacturing systems, particularly in machining. While they may be "sub-optimal" in the short-term Operations Research sense, in the long-term, they will provide long-lasting, robust flexibility and avoid the static dead end which FMS have progressively become.

7. Experimental Results

The system described above was, in its various aspects, simulated in SLAM II and G2.This model was developed over the period 1986-1991 and has now been adapted and enlarged by researchers at Purdue's Center for Intelligent Manufacturing Systems, Loughborough, Cambridge and Harvard. The system modelled comprised between 10 and 50 different machines in a heterarchical structure. The *maximal* system is shown diagrammatically in Figure 13. The communications systems were modeled as a limited resource, along with the automated vehicle system. Monte Carlo experiments were carried out after validation, using an exponential demand stream for products which had been generated from expert estimates of processing time distribution parameters. Various failures and operating conditions were imposed on the system. In order to reproduce the experiments, readers are invited to email the author for the code.

7.1 Utilization

As one might expect, system utilization increases as jobs arrive to be processed more frequently. When the system is very lightly loaded, the faster machines tend to process all of the jobs arriving. As the system becomes more heavily loaded, less effective machines gradually take on more work, until a saturation point is reached, when machines receive jobs with probability in proportion to mean processing rate across all jobs. At saturation, each machine had statistically identical utilization of around 85%. Each machine had local rules concerning the amount of work-in-progress it was allowed to attract to itself. In general, this was limited to two parts. This, of course, means that WIP is limited by a kanban type system. For this reason, WIP did not exceed 200 jobs, except in the case of system-wide communications failure. Lead times (throughput times) were between 2 and 3 times the total machine processing times for the product, depending on the loading of the system.

2.15.5 Performance and Routing Flexibility

A key issue which the experiment was designed to explore was the following: How does the flexibility of the machines affect the applicability of a heterarchical structure. Routing flexibility here is the ability of the various machines to process multiple jobs. An entropic measure of machine flexibility was developed for the purpose of this experiment and is described in Section 9. As flexibility increases, the performance of the system improves, as one might expect. However, the system performance deteriorates as machines become very flexible, when they are able to work on almost every job. This is a result of the fact that the number of bids made for each job tends to jam the communications system and generally adds to the complexity of a job's decision about which machine to assign itself. Surprisingly, it became clear that there were some advantage for the heterarchy in specializing some machines slightly. Figure 8 shows some results from these experiments. Processing times are short in this experiment in order to stress the control system.

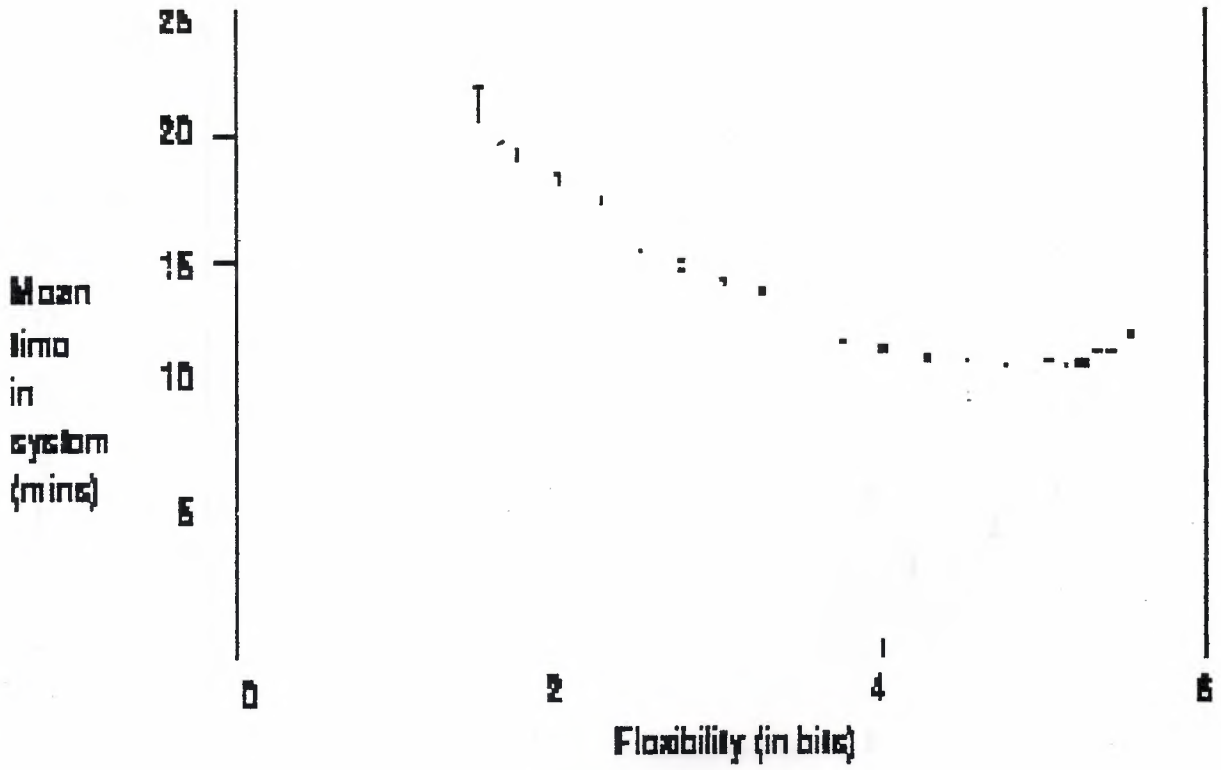


Figure 2.11. Mean Time in System versus Flexibility (200 jobs)

This phenomenon can be better understood by considering the rejection results in Figure 9. When a job can find no takers (i.e. no machine bids because buffers are full) it waits for a short period of time before re-submitting itself to auction. If it still finds no takers, it extends its waiting period. This "exponential back-off" avoids communications saturation when the system becomes very busy.

When machines bid, they do so by taking into account bids they have already made, and keep a "pending list" of jobs they might win. This conservative approach avoids the possibility that they might win too many jobs and have their buffer overflow. If a machine does not hear of the outcome of the auction, the bid expires and it forgets it. When nearly all machines can process all part types, most machines tend to have long pending lists, which means they are less likely to bid. At the same time, the probability of any particular machine winning is reduced, so pending lists guard against increasingly improbable events. This causes the problem of very flexible systems rejecting jobs. Of course, in general, increases in flexibility improve the performance of the system.

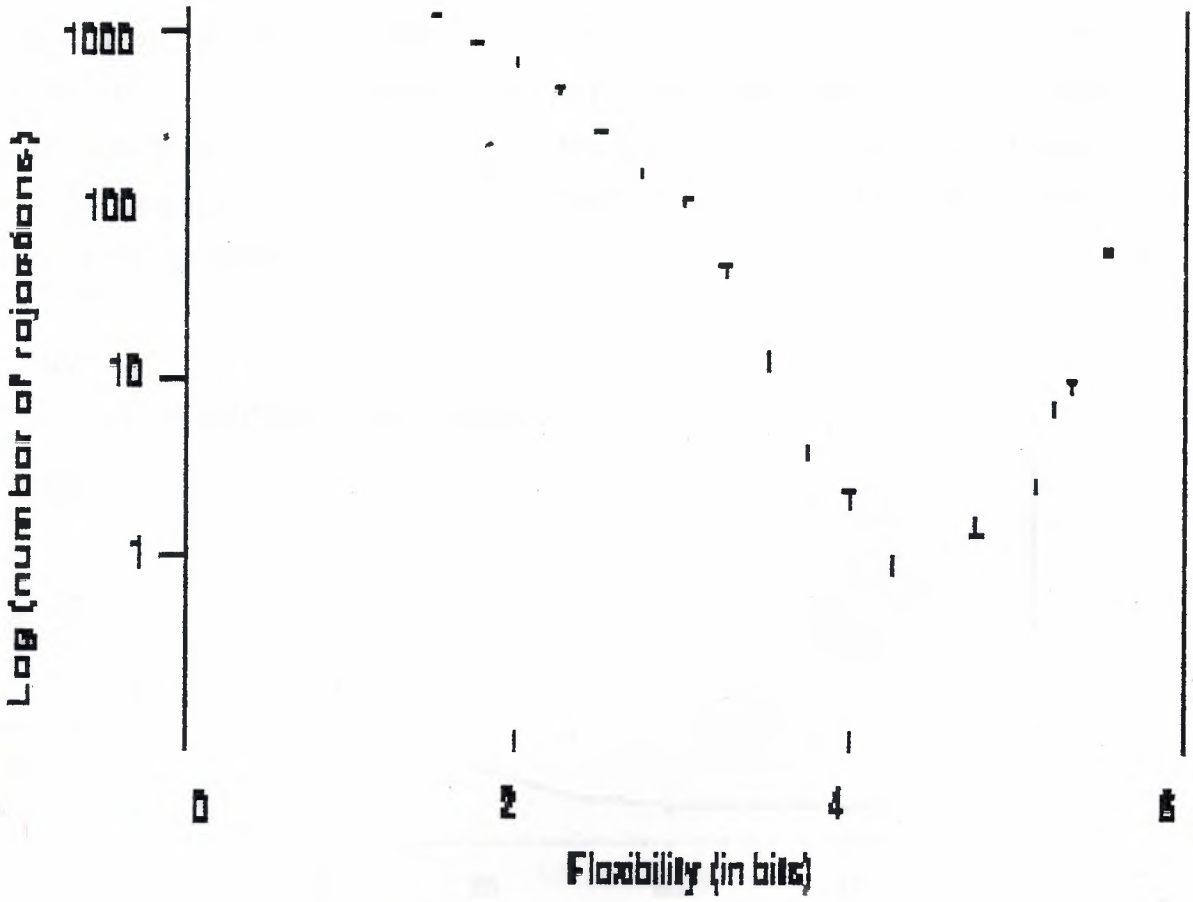


Figure 2.12. Number of Rejections versus Flexibility (200 jobs)

2.15.6 Tolerance of Failures

Machine failures had no effect on the system apart from the loss of capacity of the individual machine, and a tendency for longer throughput times by those parts in the failed machines' buffers at the time of failure. The system was made tolerant of communications failures by ensuring that all of the entities had sensible rules, which would avoid their "hanging" while waiting for a message. All entities used time-outs or similar alternatives to avoid this. Local communications failures and short, system wide communications failures had little effect on the system, but long, system-wide failures (more than a few minutes) caused failures which would demand some human intervention, parts arriving at a machine unannounced, for example, causing a buffer

overflow. It should be emphasized that these failures occurred under fairly severe conditions, and also serve to underscore the fidelity of the simulation. Part failures were not explored and certainly should be. A part failure would manifest itself in some indirect way by the physical obstruction of other jobs or by failing to respond to queries from the supervisory computer. Vehicle failures did not cause system wide problems, except when unreasonably severe .

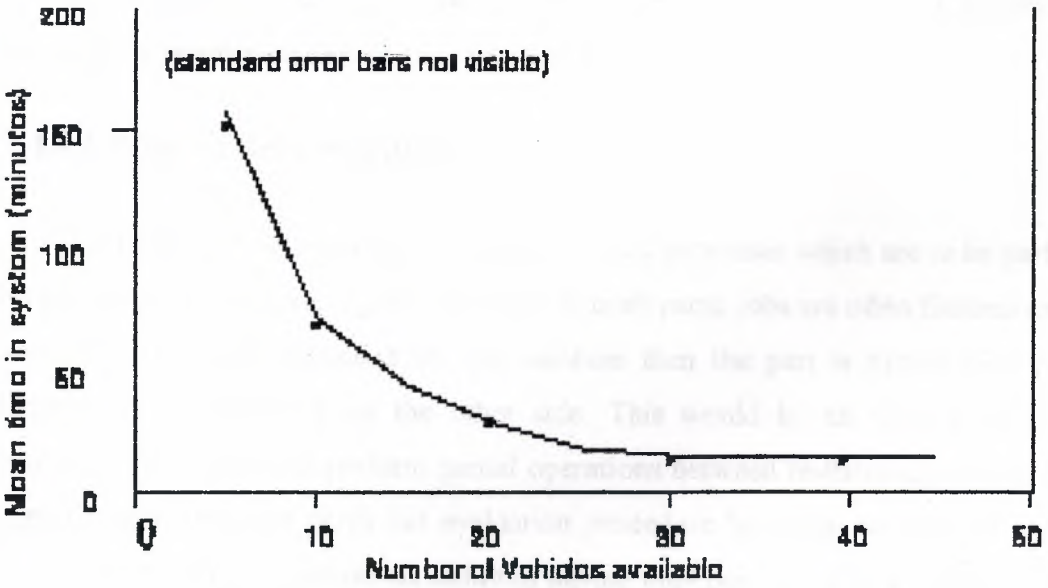


Figure 2.13. Mean time in system versus Vehicles available

2.15.7 Resource requirements

The system was clearly constrained by lack of vehicles if there were fewer than 30 vehicles in the system. The communications resource is critical. The exact requirements depend heavily on the amount of data passed in the bid request messages). It is clear, however, that any problems in the ability to put packets on the network cause fairly severe problems, consistent with those caused by communications failure.

2.16 Further considerations

2.16.1 Product limitations

This system is designed to manufacture the types of products made by current FMS. While the ideas may be extendable to other products and manufacturing systems in general it is suitable for machined metal products produced in very small batches. The processing machines which make up the system must, in general, be autonomously computer controlled as they usually are in an FMS.

2.16.2 Process decomposition

There must be a clear decomposition of the processes which are to be performed on a product. In the case of, say, machined aircraft parts, jobs are often fixtured once, all accessible faces are machined by one machine then the part is turned over and re-fixtured to be machined on the other side. This would be an ideal application. If machines are allowed to perform partial operations between re-fixturing, and to bid for a partial job, then the part's bid evaluation procedure becomes much more complex. How should a part decide on the utility of having only part of a job performed? It could negotiate ahead in its process until it has formed a complete path for itself, but the process becomes messy. This is definitely an avenue for further research.

2.16.3 Sequentiality

If a job requires processing by a large number of machines in sequence, the parts will make decisions which are too myopic. They may opt for a machine which is the most appropriate in the short-term, but find themselves a long way from their optimal total processing path. This is the justification often given for centralized control - that there must be some entity with a good temporal overview. Again, the system could be extended to cover such systems with more advanced negotiation schemes. Each move away from simplicity however, makes the argument for this type of system weaker. The strength of the method relies on the appropriate application of the method.

CHAPTER3

CONTROL AND SCHEDULING OF MANUFACTURING SYSTEM

Consider a flexible manufacturing system that consists of a machining Subsystem and an assembly subsystem. The two subsystems are linked by a material handling carrier, for example, an automated guided vehicle (AGV), as shown in Figure 1.

Consider an example product C with parts to be machined and then assembled (Figure 2). It consists of subassembly A_1, final assembly A_2, and three parts, P_1, P_2, $P3$.

Parts P_1 and P_2 are to be machined before the subassembly A_1 is obtained. Assembling P_3 and A_1 results in the product C (final assembly A_2 in Figure 2).

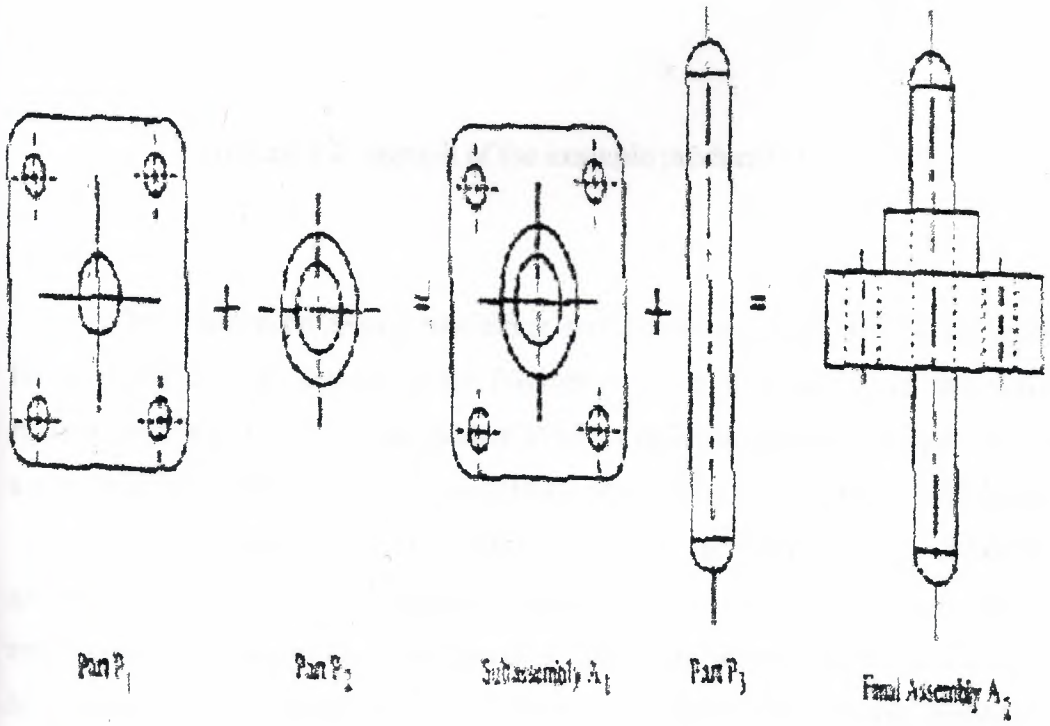


Figure 3.1. An example product C.

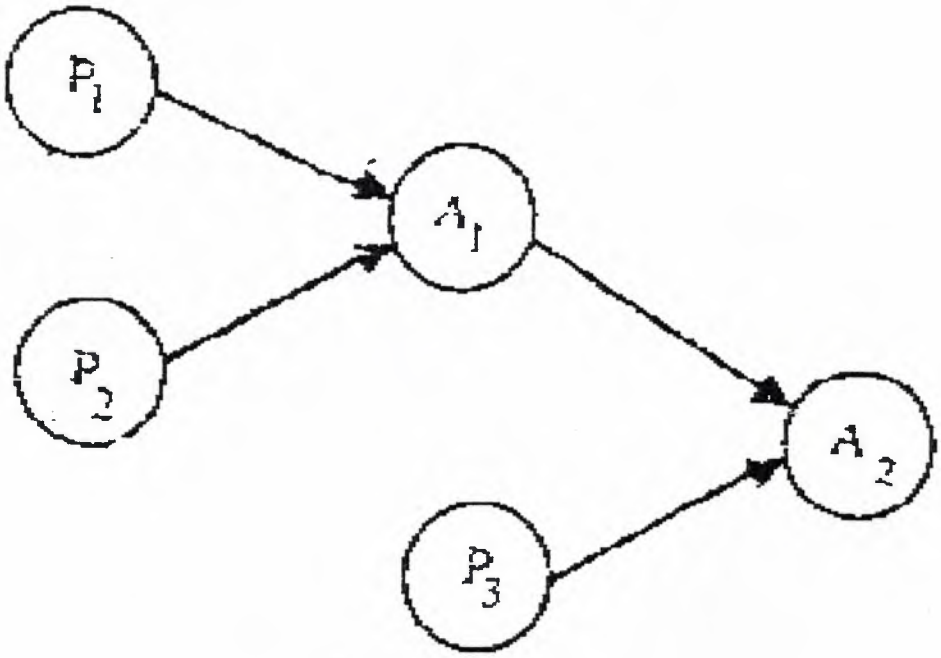


Figure 3.2. digraph of the example product C in Figure 2.

The precedence among machining and assembly operations for the product can be represented by a directed graph (digraph) shown in Figure 3. In this digraph any node of degree 1, i.e., with the number of edges incident to the node equal to 1, denotes a part; and any node of degree greater than I denotes a subassembly or a final product.

Another example of a digraph is shown in Figure 4(a). Without loss of generality, in this chapter, rather than representation of the digraph in Figure 4(a), the representation shown in Figure 4(b) is used. The latter representation does not allow one to assemble at a particular node more than one subassembly with any number of parts. At node A_3 in Figure 4(a), subassemblies A_1, A_2 and parts P_5. P_6 are assembled. The same subassembly A_3 has been obtained using the representation in Figure 4(b), where an additional subassembly A_{12} was inserted.

Four different aggregate scheduling problems are considered:

1. The single-product scheduling problem concerned with scheduling parts and subassemblies belonging to a single product.
2. The N-product scheduling problem concerned with scheduling parts and subassemblies for *N* distinct products.
3. The single-batch scheduling problem concerned with scheduling parts and subassemblies for a batch of *n* identical products.
4. The N-batch scheduling problem concerned with scheduling of parts and subassemblies for N batches of products.

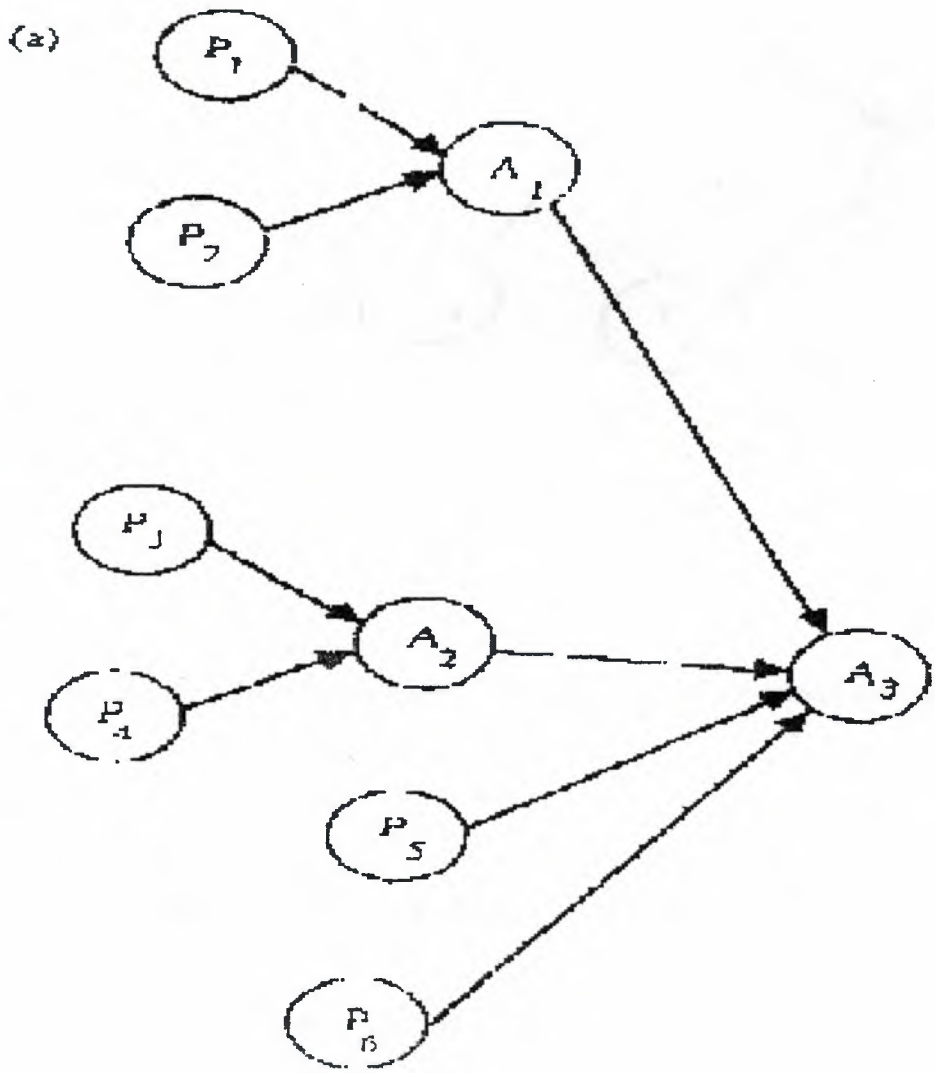


Figure 3.3(a).Two different representations of the same product.

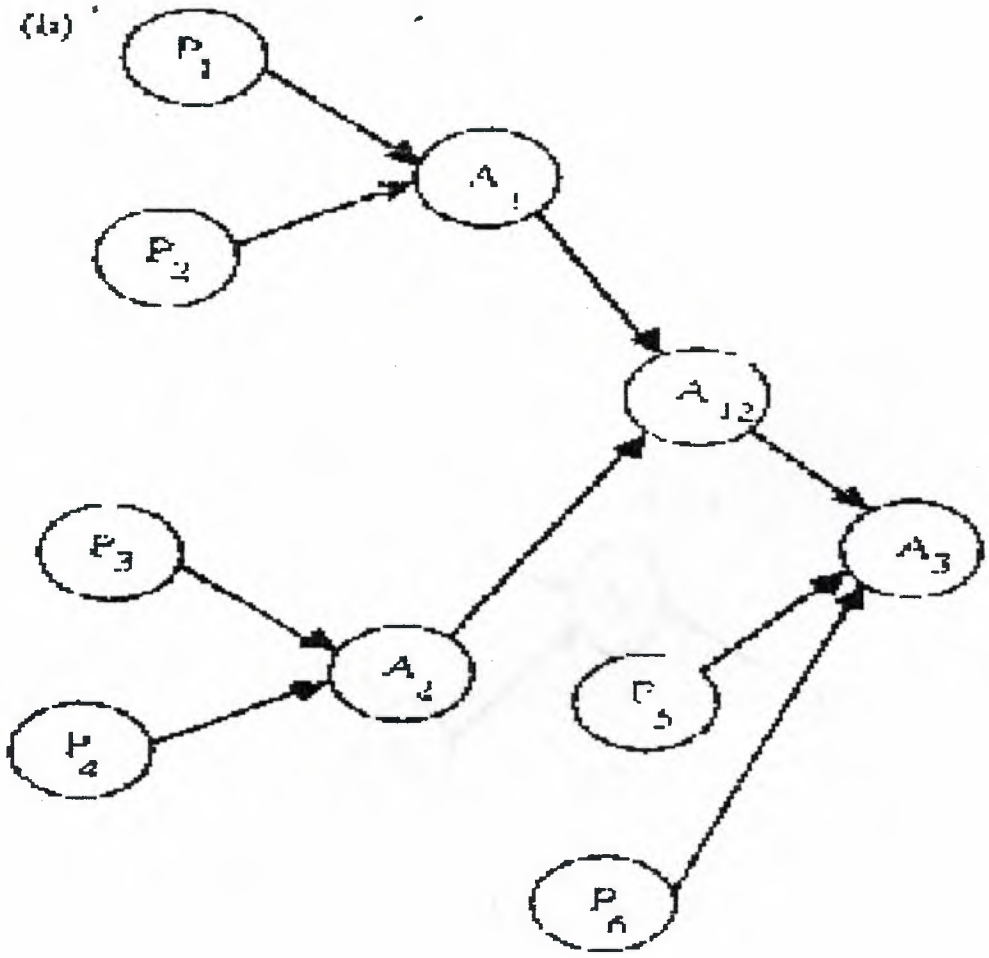


Figure 3.3(b) Two different representations of the same product.

3.2 The Single-Product Scheduling Problem

Consider a digraph representation of the product which consists of a number of parts and subassemblies. In the digraph each node is labeled *(a,b,c)* where a is the machining time, *b* is the subassembly time, and *c* is the level of depth of the node considered. The level of depth is assigned as follows: value 0 is assigned to the root node (for example, node A_2 in Figure 3) and, working backward from the root node to

the initial nodes (i.e., nodes F_1, F_2, and F_3), values of increment 1 are assigned. Digraph G from Figure 3 with labeled nodes is illustrated in Figure 5.

Before an algorithm for solving the single-product scheduling problem will be developed, a definition and two theorems are presented.

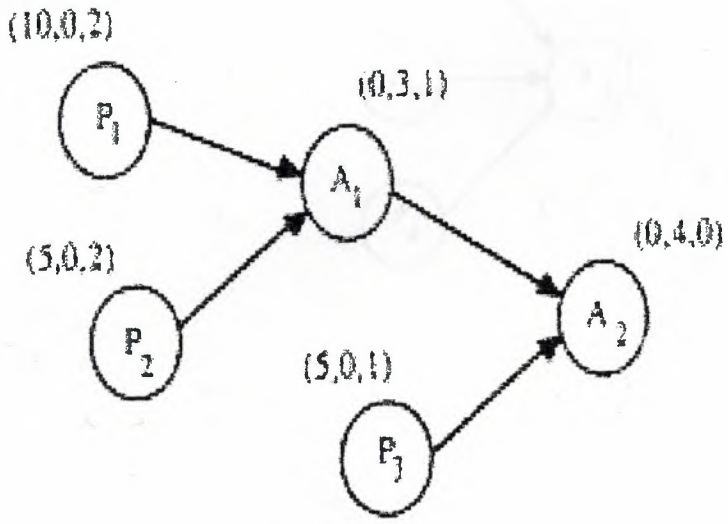


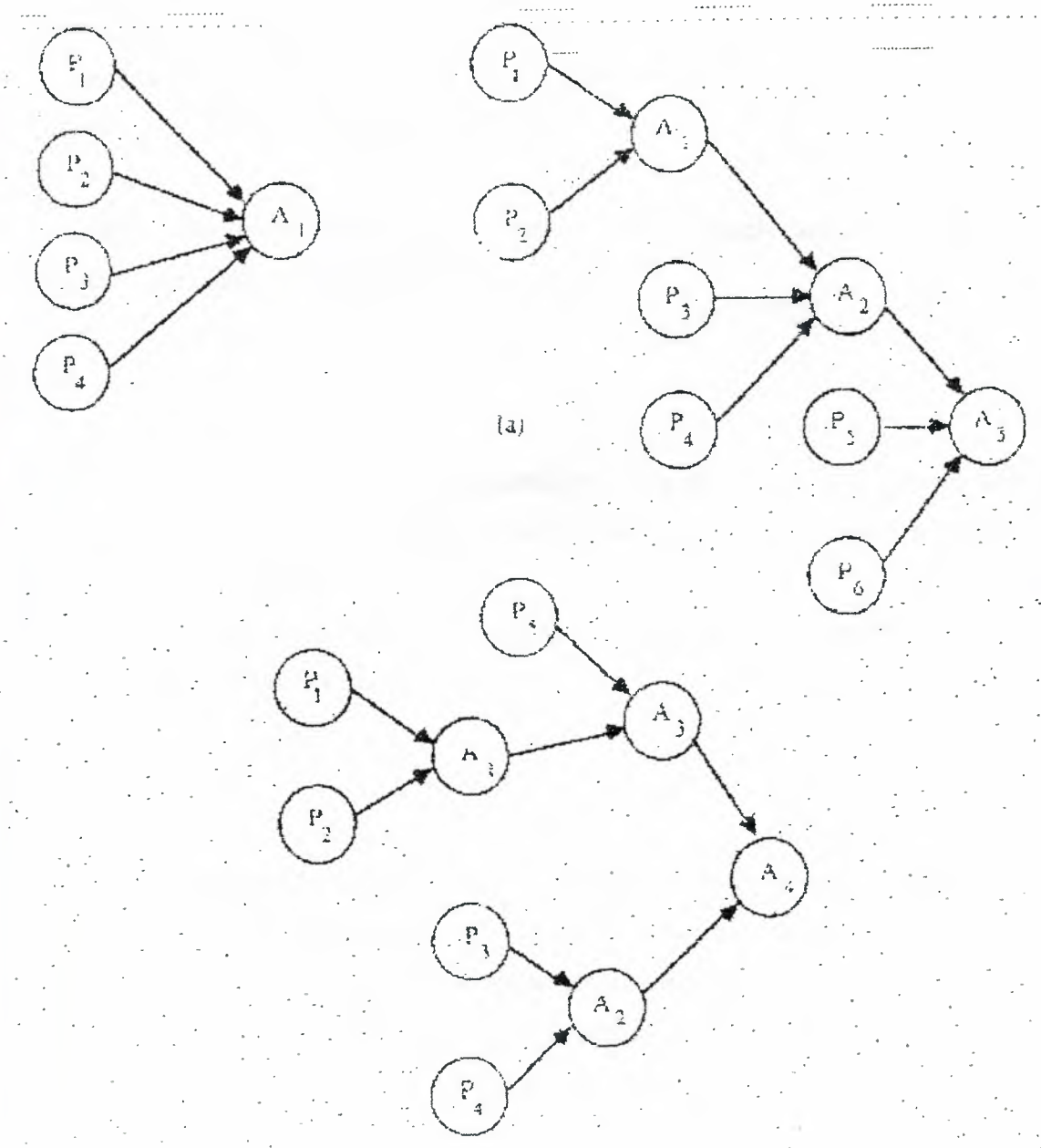
Figure 3.4. A digraph with labeled nodes.

Definition:

A simple digraph Gs is a digraph in which each node of a degree greater than 1 has at most one preceding node of a degree greater than 1 [see Figure 3.4(a)].

Consequently, a complex digraph C is a digraph which is not a simple digraph [see Figure 3.4(b)].

Based on the preceding definition, it is obvious that any complex digraph can be decomposed into simple sub digraphs by removing a number of nodes corresponding to the final assembly or subassemblies.



(b)

Figure 3.4. Example of two types of digraph: (a) two simple digraphs G_s and (b) complex digraph G.

Theorem 1 (see appendix for the proof)

Scheduling nodes (parts or subassemblies) of a simple digraph G. with the maximum level of depth first (MLDF) provides the minimum make-span schedule. [ ] Theorem I is illustrated in Figure 3.5.

In Figure 7(b) the in-process idle time refers to the assembly subsystem, whereas the terminal time refers to the machining subsystem.

Theorem 1

Scheduling nodes (parts and subassemblies) of a simple digraph G_s with the maximum level of depth first (MLDF) provides the minimum make-span schedule. [ ] Theorem 1 is illustrated in Figure7.

In Figure 7(b) the in-process idle time refers to the assembly subsystem, whereas the terminal time refers to the machining subsystem.

Theorem 2

Consider a subassembly or final product C represented by a complex digraph C and decompose it into sub digraphs g_1, g_2,..., g, by removing the root node v_0 of C. Let $S(g_i)$ be the minimum make-span partial schedule associated with g, $i = 1,\ldots,t$. if parts and subassemblies corresponding to g_i, and g_j, $I \neq j$ are preempted, then the minimum make-span schedule of product C is as follows:

$S(C) = \{S_1(G), S_2(C), v_0\}$

Where

$$S_1(G) = [S(g_{[1]}), S(g_{[2]}, \ldots, S(g_{[k]})], \text{ for } I_{[1]} \leq T_{[1]}$$

i = 1, . . . , k is a schedule obtained using the longest in-process idle time last (LITL) rule and

i= k+ 1, k + 2, . . . , t is a schedule obtained using the longest terminal time first (LTTF) rule and

$S_2(G) = [S(g_{[k+1]}), S(g_{[k+2]}, \ldots, S(g_{[k]}), \ldots, S(g_{[t]})$, for $I_{[I]} > T_{[I]}$

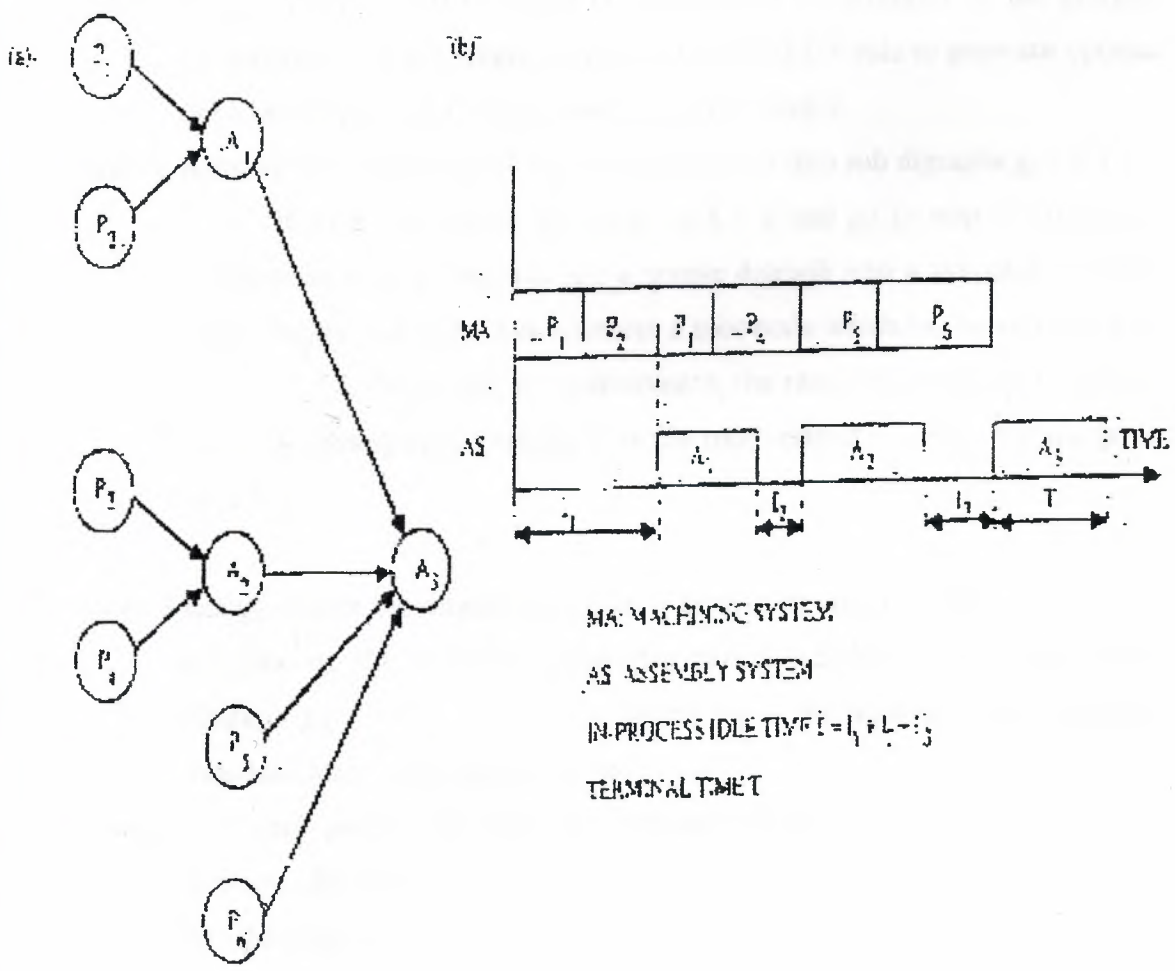


Figure3.5. Application of the MDLF scheduling rule: (a) simple digraph illustrated Theorem 1 and (b) corresponding minimum make-span schedule.

$i = k + 1, k + 2, \ldots, t$ is a schedule obtained using the longest terminal time first (LTTF) rule

The proof of this theorem follows from the considerations presented in Kurisu (1976). Based on Theorems 1 and 2, Algorithm I is developed.

Algorithm 1 (The Single-Product Scheduling Problem)

Step 1. Label all nodes of the digraph G representing the structure of the product considered. If G is a simple digraph, use the MLDF rule to generate optimal schedule of product C, stop; otherwise, go to step 2.

Step 2. Remove root node v_0 from G and decompose it into sub digraphs g_l, $l = 1, \ldots, L$. If all g_l are simple digraphs, set $k = 0$ and go to step 3; otherwise, decompose each g_l which is not a simple digraph into a simple digraph by removing its root node. Let v_j denote a root node which has been removed, $j = 1, \ldots, J$. (Note that, for convenience, the removed nodes are numbered in the increasing order starting from the root node of C.) Set $k = J$ and go to step 3.

Step 3. Let g_{ik} denote the simple sub digraphs associated with v_k Use the MLDF rule to generate the minimum make-span partial schedule $S(g_{ik})$ for each sub digraph g_{ik}, $i = 1, \ldots, N_k$, where N_k is the number of sub digraphs obtained after v_k has been removed.

Step 4. For each partial schedule $S(g_{ik})$ obtained in step 3, determine (i) the in-process idle time I_{ik}

(ii) the terminal time T_{ik}, $i = I, \ldots, N_k$

Step 5. Separate the $S(g_{ik})$ into two lists:

List 1 schedules $S(g_{ik})$ such that $I_{ik} \leq T_{ik}$

List 2 schedules $S(g_{ik})$ such that $I_{ik} > T_{ik}$,

$$i = 1, \ldots, N_k$$

Step 6. Use the LITL rule to generate

$S_1(g_k) = [S(g_{[1]k}), S(g_{[2]k}), \ldots, S(g_{[r]k})]$,
for $S(g_{ik})$ in list 1, $i = 1, \ldots, r$

and use the LITF rule to generate

$S_2(g_k) = [S(g_{[r+1]k}), S(g)_{[r+2]k}), \ldots, S(g_{[r]k})]$,
for $S(g_{ik})$ in list 2, $i = r + 1, \ldots, t$; $t = N_k$

Then generate the partial schedule $S(g_k)\ [S_1(g_k), S_2(g_k), v_k]$

Step 7. If $v_k = v_0$, then $S(C) = S(g_k)$ is the optimal schedule, stop; otherwise, go to step 8.

Step 8. Consider S(Ck) as a simple sub digraph schedule and calculate I_k and T_k. Set $k = k - 1$ and go to step 3.

Algorithm 1 is illustrated in Example 1.

Let's consider example, that

Example 1

Find the minimum make-span for a product *C*, with the structure represented by the digraph G shown in Figure 3.6.

Solution Procedure

Step 1. Since the labeled digraph G (product *C)* in Figure 8 is not a simple digraph, go to step 2.

Step 2. Remove the root node v_0 and nodes v_1, v_2, v_3 so that the simple sub digraphs g_{11}, g_{21}, g_{13}, g_{23}, g_{33} are obtained (Figure 3.7).

Step 3.Use the MLDF rule to generate the partial schedules for the sub digraphs g_{13}, g_{23}, and g_{33} associated with v_3. The Gantt chart for each sub digraph is shown in Figure 10.

Step 4.For each simple sub digraph, the in-process idle time and terminal time are

$$I_{13} = 5,\ T_{13} = 3;\ I_{23} = 3,\ T_{23} = 4;\ I_{33} = 3,\ T_{33} = 2$$

Step 5. Since

$I_{13} > T_{13}$, $I_{23} < T_{23}$, $I_{33} > T_{33}$

Sub digraph g_{23} is placed in list 1; g_{13} and g_{33} are placed in list 2.

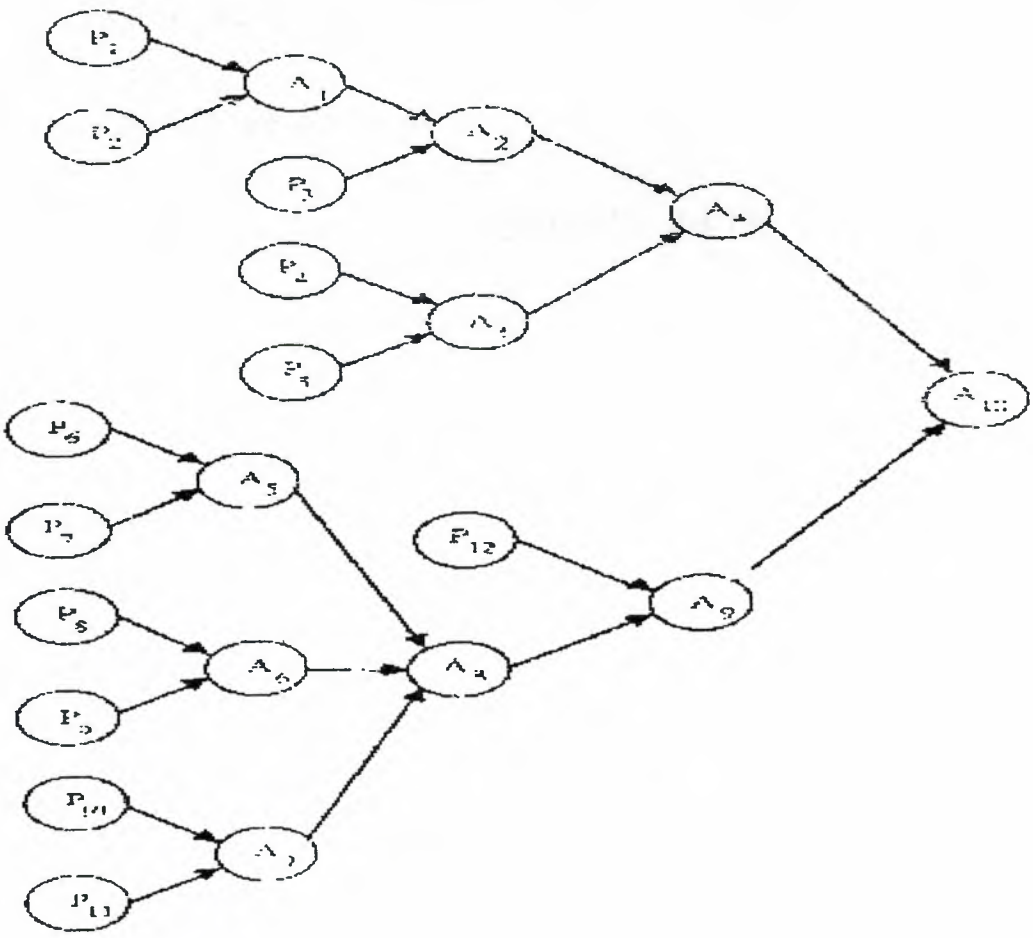


Figure3.6. Product C structure

Step 6.Use the LITL rule to generate

$S_1(g_3) = [(P_8, P_9, P_6)]$

and use the LTTF rule to generate

$$S_2(g_3) = [(P_6, .P_7, A_5), (P_{10}, P_{11}, A_7)]$$

Then the partial schedule associated with v_3 is

$$S(g_3) = [S_1(g_3), S_2(g_3), v_3]$$

$$= [(P_8, P_9, A_6), (P_6, P_7, A_5), (P_{10}, P_{11}, A_7), P_8]$$

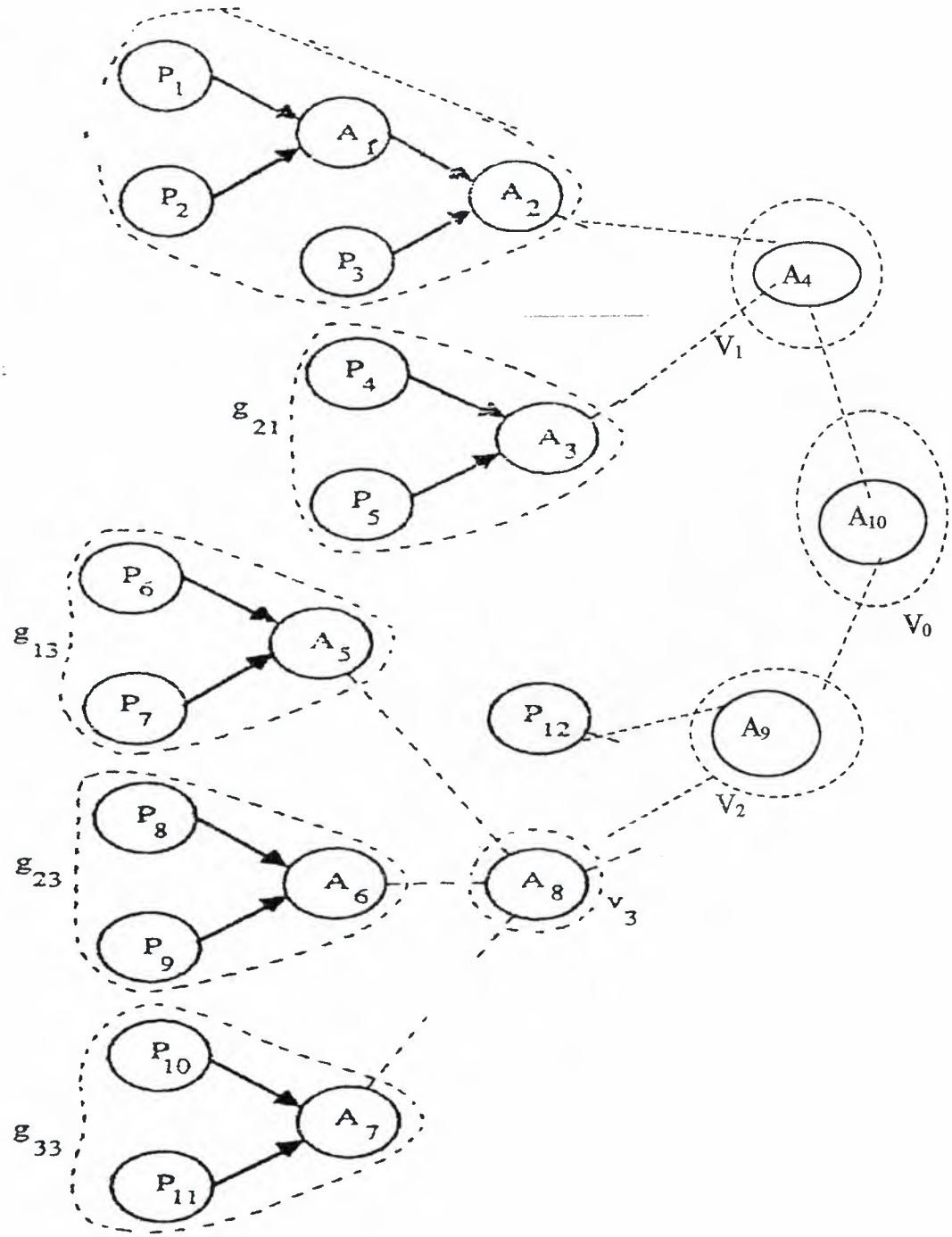


Figure 3.7. The decomposed digraph G of Figure 3.6.

The Gantt chart of the preceding partial schedule is shown in Figure 3.19.

Step 7.Since $v_3 \neq v_0$, go to step 8.

Step 8.For partial schedule S (g_3) calculate

$$I_3 = 4,\ T_3 = 4$$

Set $k = 3 - 1 = 2$, go to step 3.

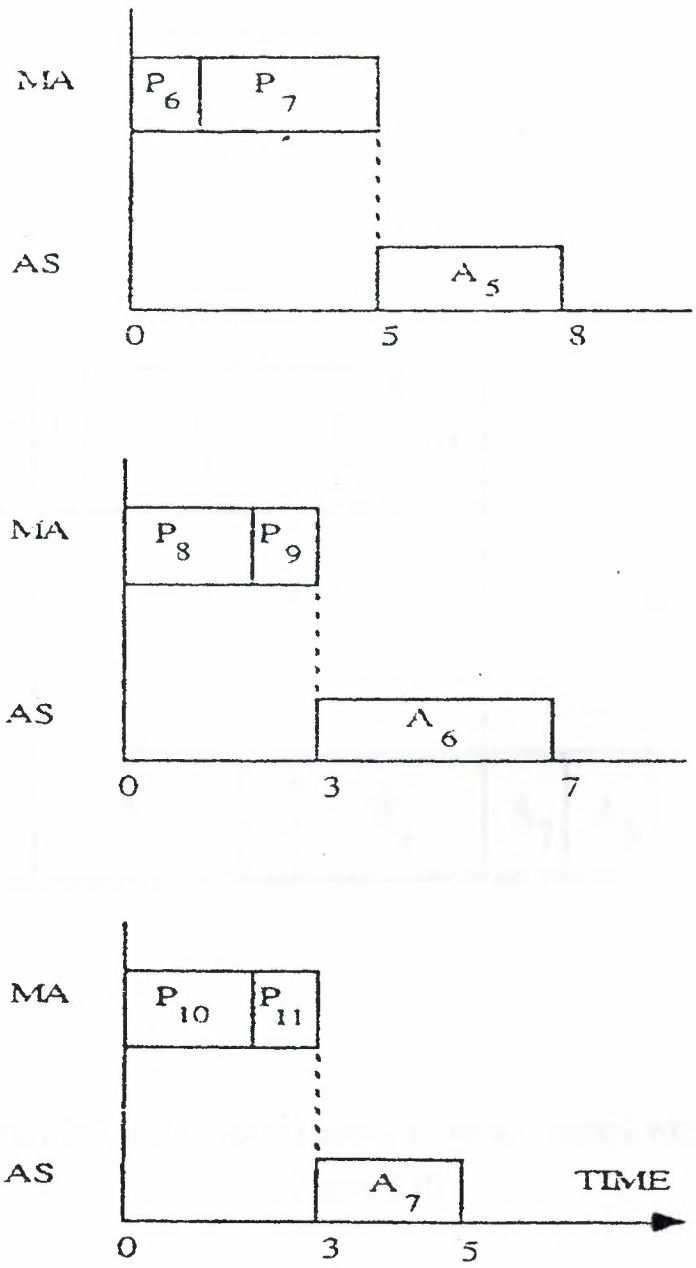


Figure 3.8. Gantt charts for the partial schedules of g_{13}, g_{23}, and g_{33} in Figure 3.9.

In step 3 (the second iteration), a simple sub digraph associated with v_2 is obtained. Using the MLDF rule, a partial schedule illustrated with the Gantt chart is shown in Figure 3.10.

The following values for in-process idle time and terminal time are calculated

$I_2 = 4,\ T_2 = 5$

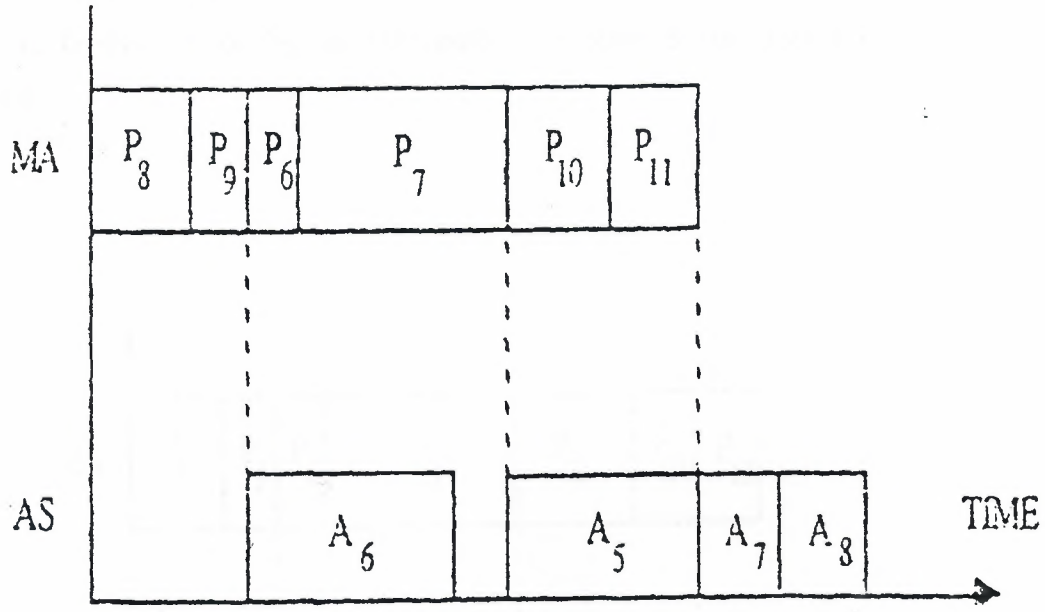


Figure 3.9.Partiail schedule corresponding to the sub digraph with the node v_3 in Figure 3.10.

The same process is applied to v_1, and the Gantt chart of each partial schedule is shown in Figure 13. The corresponding values of in-process idle time and terminal time are calculated

$I_1 = 9,\ T = 5$

In the last iteration, the above sequencing process is applied to v_0 and since $(12 \text{ K } T_2)$, g2 is placed in list 1, $(1 \sim > T_1)$, g_1 is placed in list 2.

The minimum make-span schedule for product C (Figure 3.6.) is

$S(C) = (S_1(go), S_2(go), i'_0)$

$= \{[(P_8, P_9, A_6), (P_6, P_7, A_5), (P_{10}, P_{11}, A_7), A_8, P_{12}, A_9] [(P_1, P_2, A_1), P_3, A_2, (P_4, P_5, A_3), A_4], A_{10}\}$

The Gantt chart of the minimum make-span schedule for product C is shown in Figure 14.

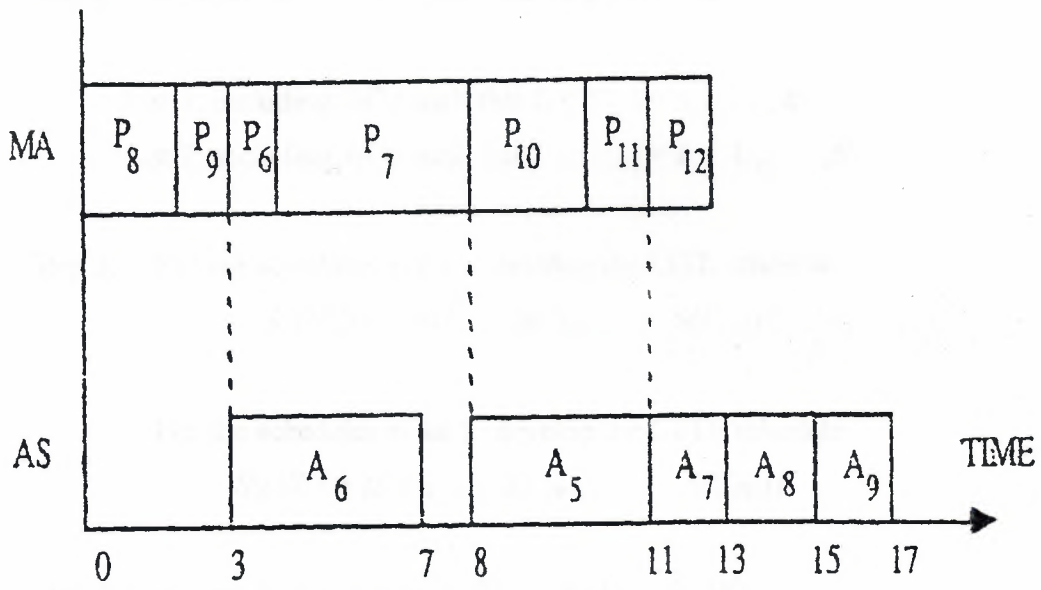


Figure 3.10.Partial schedule corresponding to the subdigraph with the node v_2 in Figure 3.7.

3.3 The N-Product Scheduling Problem

In this section, the scheduling problem for N distinct products, each in quantity of one, is considered. To solve this problem, a "product-by-product" policy is used. The product-by-product policy assumes that the N product scheduling problem can be decomposed into N single-product scheduling problems. The algorithm for scheduling of N products C_1, C_2 C_N is presented below.

Algorithm 2 (The N-Product Scheduling Problem)

Step I. Using Algorithm 1, determine the optimal schedule $S(Q)$ for each product $C_i = 1, \ldots, N$.

Step 2. Separate all $S(C_i)$ into the following two lists:

List 1: including $S(C_i)$ such that $I_i \leq T_i$, $i = 1, \ldots, k$

List 2: including $S(C_i)$ such that $I_i > T_i$, $i = k + 1, \ldots, N$

Step 3. For the schedules in list 1, develop the LITL schedule:

$$S_1(NC) = \{S(C_{[1]}), S(C_{[2]}), \ldots, S(C_{[k]})\}$$

For the schedules in list 2, develop the LTTF schedule;

$$S_2(NC) = \{S(C_{[k+1]}), S(C_{[k+2]}), \ldots, S(C_{[N]})\}$$

Step 4. Generate final schedule $S(NC) = \{S_1(NC), S_2(NC)\}$

Theorem 3

The schedule $S(NC) = \{S_1(NC), S_2(NC)\}$ generated by Algorithm 2 is the minimum make-span schedule of the N-product scheduling problem.

The proof of Theorem 3 follows the results presented in Kurisu (1976), except that the sub digraph g, is replaced with the digraph representing product C_i.

Algorithm 2 is illustrated in Example 2.

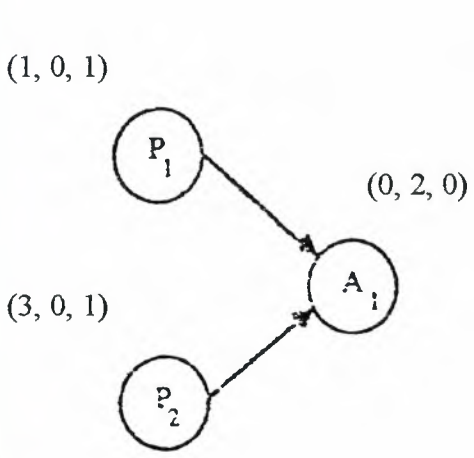
Example 2

Consider an $N = 4$ product-scheduling problem with each product structure shown in Figure *15*. For simplicity, assume that the structure of each product is represented by a simple digraph.

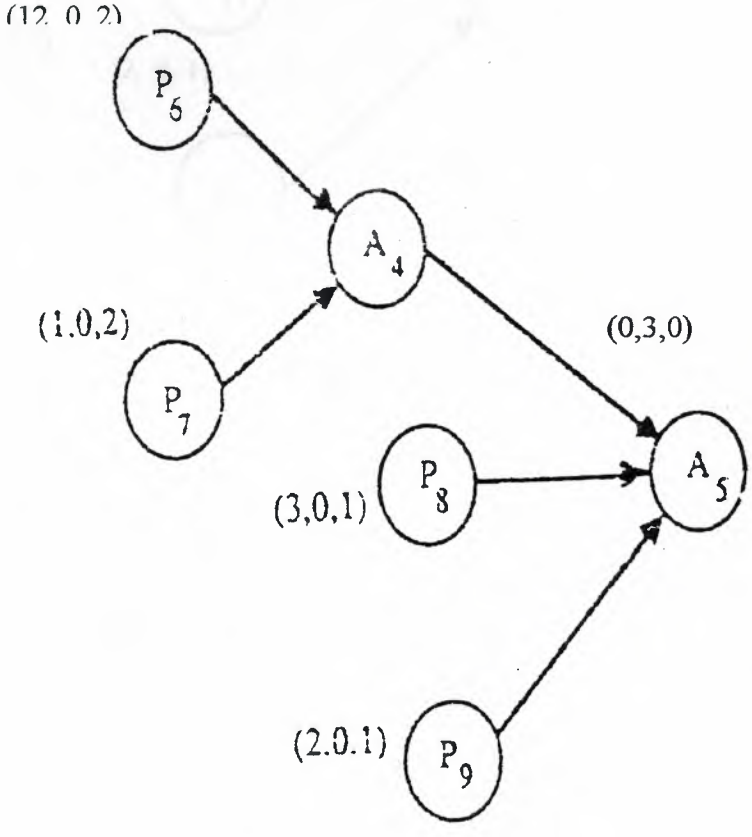
Step 1.Using Algorithm 1, the following schedules (illustrated in Figure 16) are obtained:

Product C_1: $S(C_1) = \{P_1, P_2, A_1\}$

Product C_2: $S(C_2) = \{(P_3, P_4, A_2), P_5, A_3\}$



Product C_1



Product C_3

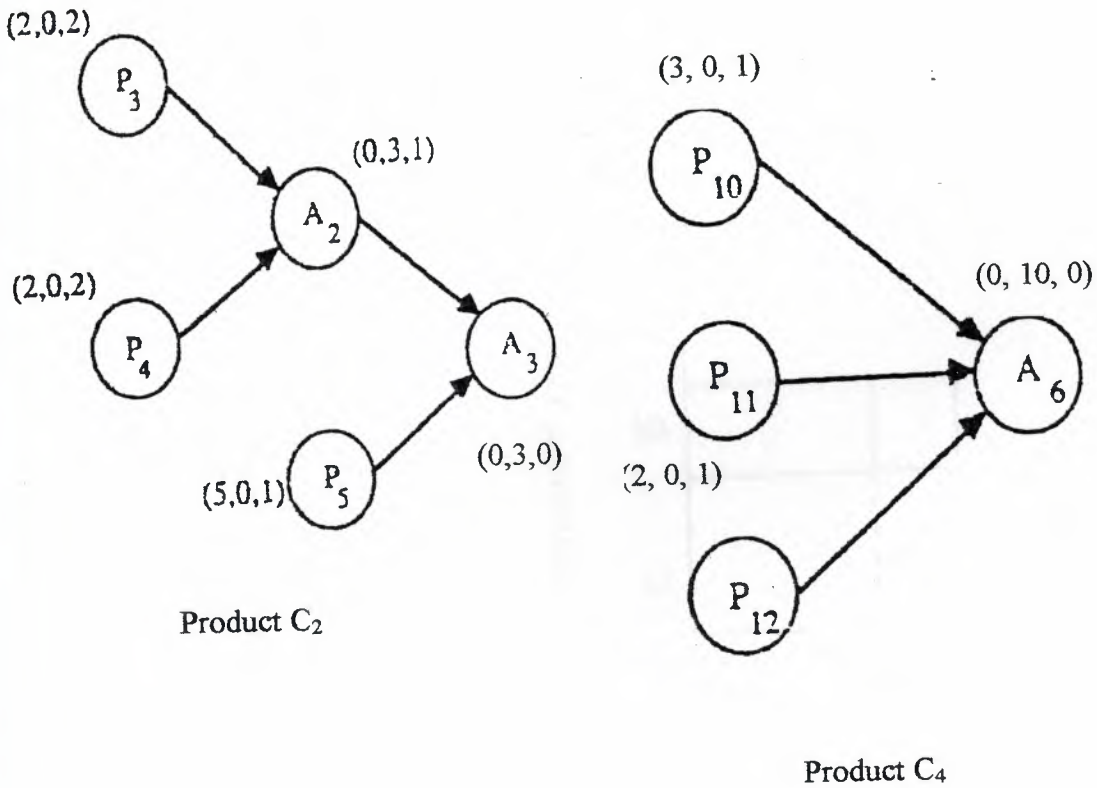


Figure 3.11.Structure of products C_1, C_2, C_3, and C_4.

Product C_3: $S(C_3) = \{(P_6, P_7, A4, P_8, P_9, A_5\}$

Product C_4: $5(C_4) = \{P_{10}, P_{11}, P_{12}, A_6\}$

Step 2.From the Gantt charts in Figure 16, the following data is obtained:

Table 3.1

Product	I_i	T_i	List number
C_1	4	2	2
C_2	6	3	2
C_3	18	3	2
C_4	9	10	1

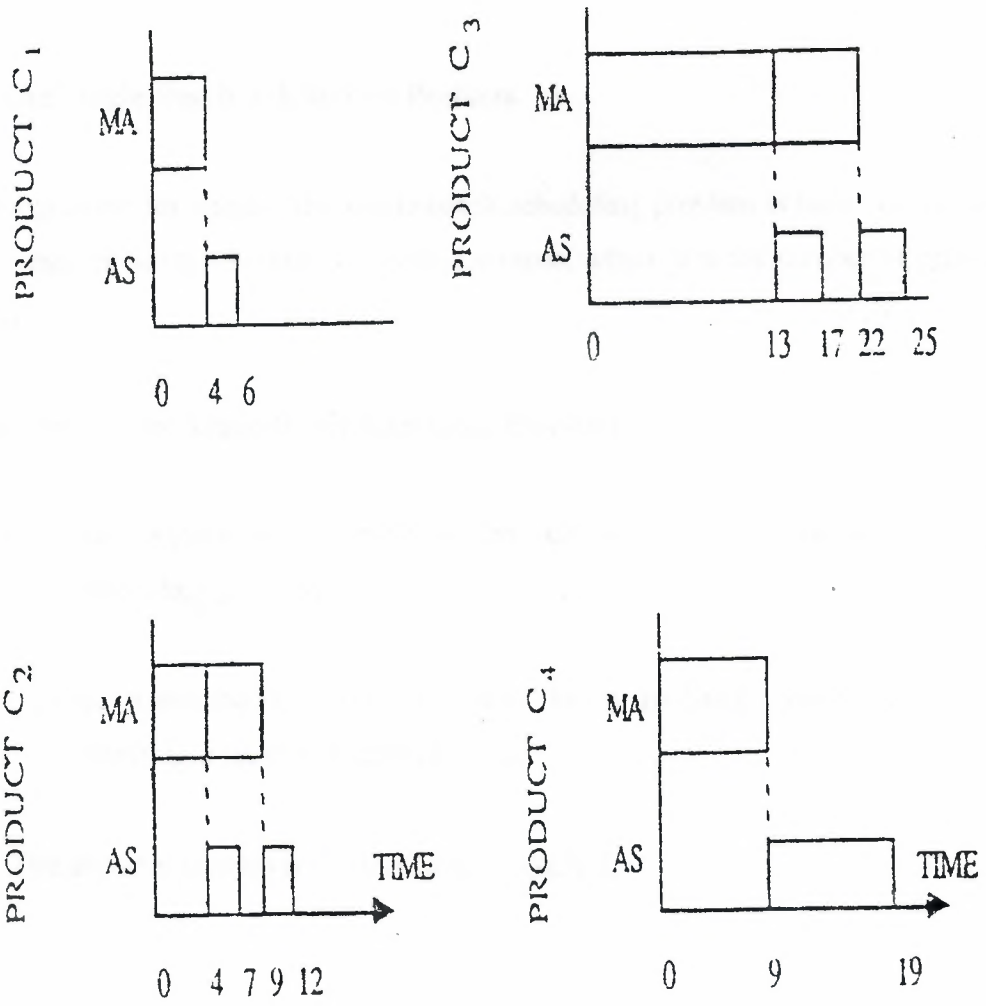


Figure 3.12.Schedules for the products in Figure3.11.

Step 3.Using the LITL and LTTF rules results in the following schedules:

$$S_1(NC) = \{C_4\}$$

$$S_2(NC) = (C_3, C_2, C_4\}$$

Step 4.The optimal schedule is as follows;

$S(NC) = \{[(P_{10}, P_1, P_{12}, A_6)], [(P_6, P_7, A_4), P_8, P_9, A_5],$

$[(P_3, P_4, A_2), P5, A_3], [P_1, P_2, A_1]\}$

3.4 The Single-Batch Scheduling Problem

The algorithm for solving the single-batch scheduling problem is based on Algorithm 1. The single-product schedule is repeated n times, where n is the number of products in a batch.

Algorithm 3 (The Single-Batch Scheduling Problem)

Step 1.Using Algorithm 1; determine the schedule $S(C)$ for the single- product scheduling problem.

Step 2.Determine the schedule $S(B)$ for the single-batch scheduling problem by repeating a times the schedule $S(C)$

The above algorithm is illustrat2d in Example 3.

Example 3

Develop the optimal schedule *S(B)* for a = 5 identical products based on the following data;

Table 3.2.

Part or Subassembly	P_1	P_2	P_3	A_1	A_2
Machining time	10	5	6	-	-
Assembly time	-	-	-	4	3

and precedence constraints:

$$P_1, P_2 \longrightarrow A_1$$

$$P_3, A_1 \longrightarrow A_2$$

Step 1. Solving the single-product scheduling problem, the following schedule is obtained:

$S(C) = \{(P_1, P_2, A_1), P_3, A_2\}$

Step 2. The optimal schedule *S(B)* is generated as follows:

$$S(B) = \{[P_1, P_2, A_1), P_3, A_2], [(P_1, P_2, A_1), P_3, A_2],$$
$$[(P_1, P_2, A_1), P_3, A_2], [(P_1, P_2, A_1), P_3, A_2],$$
$$[(P_1, P_2, A_1), P_3, A_2]\}$$

In [ ] the n-batch scheduling problem is described

CHAPTER 4

COMPUTER-CONTROLLED MANUFACTURING SYSTEM FOR DYLITE EXPANDED POLYSTYRENE BOARD PRODUCTION

4.1. Main components of the cutting plant of Dylite expanded polystyrene boards

The Kurtz oscillating hot-wire cutting plant type VBR S-3 was developed in order to efficiently and flexibly produce high-quality expanded polystyrene boards (EPS sheets). Compared with the vibration higher cutting speeds as well as a higher surface quality of the cut EPS sheet are achieved, due to the oscillation of the cutting wires. The picture frame effect occurring with irregularly dried blocks is avoided to a great extend.

Main components of the cutting plant

- Automatic block transport unit for the automatic block transport through the cutting line

Components:

- Block magazine in front of the cutting plant
  - Turning table
- Block turning and aligning unit
  - Transport chain conveyor with lifting table
  - Parallel block guiding in front and after the sheet cutting station
- Alignment unit in front of the trimming station (option)
  - Acceleration conveyor for transport out of trimming station
- Roller conveyor
  - Oscillating sheet cutting machine with integrated remote wire adjustment and short stroke or long stroke device for the production of the required sheet thicknesses. With the long stroke device, the capacity of the cutting plant can be increased by up to 30 percent.
  - Trimming station with short stroke oscillation for oscillating face trimming as well as for longitudinal cutting of the sheet stack

- Cross cutting station, optionally equipped with short stroke oscillation for face trimming as well as for cutting the trimmed sheet stack into sheets of the required size
- Waste removal for automatic removal and pre-breaking of all trimmings

Table 4.1.

Length	Width	Height
3.0 – 10.0	1.00 – 1.40	1.00 – 1.40

Block sizes in m

Electrical components

Clearly arranged control elements in connection with state of the art PLC control components for easy operation. The cutting plant is equipped with an automatic, freely programmable control. The driving elements are electronically controlled and harmonized by a frequency converter.

Special advantages

- The cutting technologies "hot wire" and "oscillation" are connected to a fully automatic system for highest throughput capacities,
  - Best surface qualities and dimensional accuracy.
  - Wear-resisting and shock-proof oscillation by:
    - two opposite vibrating cutting frames (long stroke device)
    - one vibrating cutting frame (short stroke device)
  - Electronically controlled hot wire cut with thyristor voltage stabilizer
- High cutting speeds, due to oscillation and electronic control of the wire temperature
- Dimensional accuracy, due to precise wires at the cutting frame
  - Precise transport of the block through the cutting plant
  - Easy operation, due to control at the touch of a button
- Trouble-free operation, due to solid construction and high-quality design
  - Highly-efficient cutting wire cooling

- Trimming recycling integrated into the process with parallel guiding, only two prebreakers and automatic conveying into the silo as an option.

Already existing cutting plants (machines of other brands as well) can be retrofitted with

The several components and optional equipment versions of the cutting plant.

Options

- Oscillation for cross cutting
- Automatic remote wire adjustment to prevent adjustment and measurement errors
- Different sheet thicknesses can be retrieved directly from the control
- Higher cutting accuracy, as adjustments can be made in the range of a tenths of a millimetre
- High production safety, due to electronic wire breakage monitoring
- Drag control to limit the bending of the cutting wire in cutting direction

Table 4.2.

Density	Long stroke device	Short stroke device
PS 15	2.0 – 2.5	1.4 – 2.2
PS 20	1.6 – 2.2	1.0 – 1.8
PS 30	0.6 – 1.6	0.6 – 1.2

Cutting speed in m / min

4.2.Structure of FMS for EPS sheets production

For the productions, producing different kind of products, the developer flexi able computer-aided control system is in primary concern. The main flexibility in these productions is the reconstruction of the program for control of (machinery) technological process. In this subchapter the structure of FMS for stray per productions is considered.

As described above the stray per productions is characterized by different kind of output products. Each output product is characterized by its own special parameters. To this parameters concern width, height, length, radius, density, etc. to produce different kind of products for the customers; it is necessary to set these parameters to certain values, in on-line regime. The choosing these parameters for different kind of products manually needs certain time which decrease the affectivity of the productions. Taking into account the special characteristics of the operators: such as tiredness and intensive working regime the accurate setting those parameters in some case.

To increase the quality output products and affectivity of the productions in the work. The development FMS is considered in Figure 1. The structure of flexi able manufacturing system for expanded polystyrene board productions is given. The primary material in column 1 and 2 in 180 °C is formed to (secondary material) stray per; this process takes 20 minutes. These expanded polystyrene board products by manipulators (Crain) 4 and 3 are loaded to the conveyer 5. These primary products by conveyor are transported to the drying-room. Manipulator 6 reload expanded polystyrene board and carry it to the room for drying. After another dried product by the same manipulator is loaded to the conveyor and is fringing for horizontal, vertical and circular cutting in 7, 8 and 9. It is needed to note that the all cutting processes are not necessary. In more case only the horizontal and vertical cutting depends on form of output product. The quality of the processed output expanded polystyrene board product is

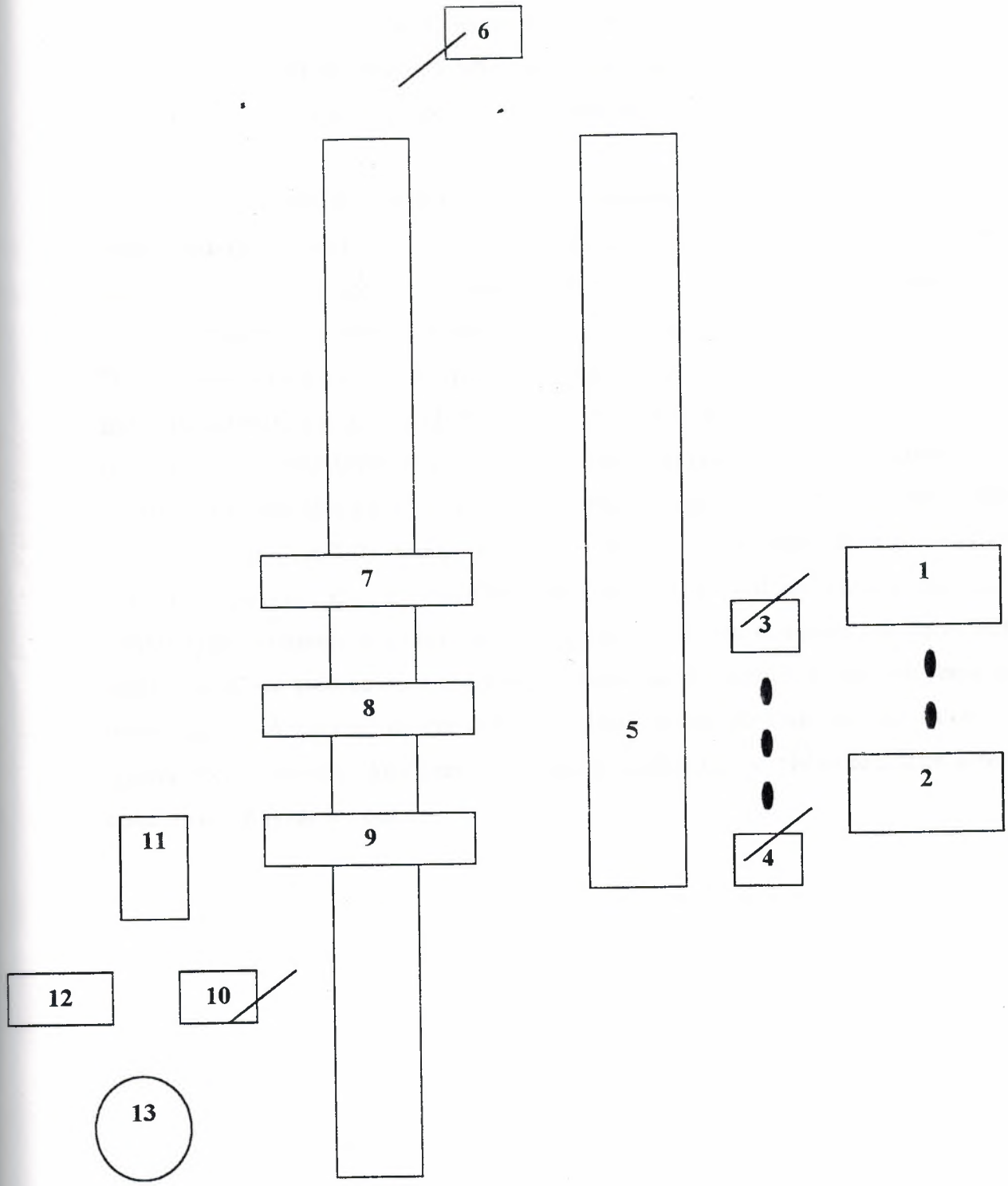


Figure 4.1. The structure of the technological processes of stray per production

controlled by manipulator 10. Input parameters for defining quality are: density and curvature of the output products. As an output we may have three types of products. Normal products are the products which need secondary processing and defects; they are selected to the tanks 11, 12 and 13 correspondingly.

For atomization, the described processes as controlled parameters are: the height, length, radius, curvature and density of expanded polystyrene board products are selected. The processing time of primary products in columns 1 and 2, the transporting time of products by conveyor 5 also are controlled parameters.

The value of the parameters is controlled by the control signal given by the computer and transferred to the inputs of the actuator (motors). In Figure 2 the structure of computer-aided control system for expanded polystyrene board production is given.

The value of controlled parameters: height of product, length of product, curvature and density of output products through converter are given to the analog-digital-converter (ADC); where their values are transformed to digital signals and given to the computer. Using program passage and input data of output products the computer calculates and control signal for each actuator and through digital-analog-converter transmits them to the actuators. According to the coming control signals actuator set the value of parameters on devices. And then the computer simulation of given computer aided control system is carried out.

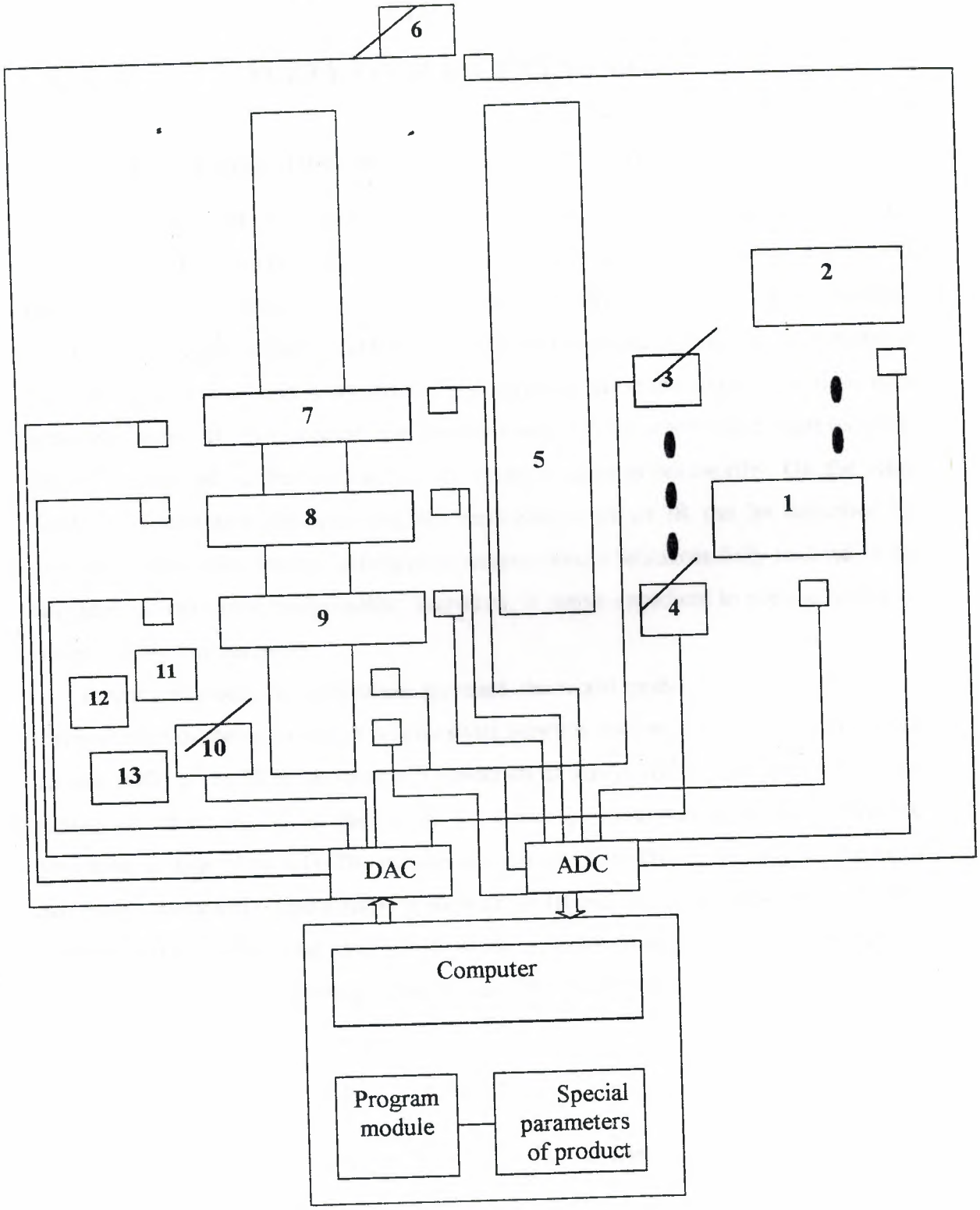


Figure 4.2. The structure of computer-aided control system for stray per productions.

CHAPTER5

FUZZY INTELLIGENT ROBOTS

5.1 Knowledge Representation in Fuzzy Intelligent Robots

Recent research has shown the necessity of using the methodology and mathematics of Artificial Intelligence to represent knowledge in intelligent robots (IR's). Further, the construction of knowledge bases (KB's) for such robots has been implemented mainly in the framework of declarative representations, i.e., in systems of inference rules, reductions, and proofs of theorems based on the language of first-order predicate logic. In this context the awkwardness of the knowledge representation models developed on the basis of the above approaches is noteworthy. On the other hand, the uncertainty characterizing the environment of an IR can be described by qualitative categories whose formalization cannot always be successfully realized in the language of traditional mathematics. Therefore, it seems expedient to use the theory of fuzzy sets for this purpose.

In our knowledge representation approach the world model of an IR is based on a formalization of the cause-effect relationships between entities and events of the object domain (OD) of the IR in the form of a collection of fuzzy production rules. The idea of composite inference [1] is then used for the implementation of logical inference according to data of the OD. The entities and events of the OD of the IR (i.e., the input parameters measured by the sensory system of the IR and the output parameters, i.e., the control actions on the actuators of the IR or the information display for the operator) are interpreted as fuzzy sets forming linguistic variables describable by triplets of the form:

$$\overset{o}{x}_1 = \{< X_1^{\lfloor} , U_{x1} , \tilde{X}_1 >\}, \qquad X_1^{\lfloor} \in T_1^{\bullet}(u), \qquad i = \overline{1,n}$$

$$\overset{o}{Y} = \{ < Y_K^{\lfloor} , V_{yk} , \tilde{Y}_K >\}, \qquad Y_K^{\lfloor} \in T_K^{\bullet}(V), \quad K=\overline{1,m} , \quad j=\overline{0,N}$$

where $T_1^{\bullet}$ (u) and $T_K^{\bullet}$ (V) are extended term-sets of linguistic variables- input and output parameters, respectively - determined by the collection of linguistic terms (extremely little, litle,..., average, much,...), while the $\tilde{X}_1$ and $\tilde{Y}_K$ are the corresponding normal

fuzzy sets described by the membership functions $\mu_{X1} : U_{X1} \rightarrow [0, 1]$ and $\mu_{Yk} : V_{Yk} \rightarrow [0, 1]$ of the form:

$$\tilde{X}_1 = \int_{U_{X1}} \mu_{X1}(u)/u, \quad i=\overline{1,n} \qquad \tilde{Y}_K = \int_{V_{Yk}} \mu_{Yk}(v)/v, \quad k=\overline{1,m} \qquad (1)$$

$$\tilde{X}_1 \subseteq U_{X1}, \qquad \tilde{Y}_K \subseteq V_{Yk},$$

Where U_{X1} and V_{Yk} are universes.

The structure of the KB of an IR is a collection of fuzzy production rules of the form $< \tilde{X}_1 \rightarrow \tilde{Y}_K >$, $i=\overline{1,n}$, $k=\overline{1,m}$ formalized as conditional propositions in a language close to the natural one:

IF X_1 is $\tilde{X}_1$ AND . . . AND X_n is $\tilde{X}_n$, THEN Y_1 is $\tilde{Y}_1$

OR

.

.

.

(

2)

OR

1F X_1 is $\tilde{X}_1$ AND . . . AND X_n is $\tilde{X}_n$ THEN Y_k is $\tilde{Y}_k$.

As already mentioned, we use a fuzzy conditional inference rule to plan the behavior of an IR according to current values of the input parameters. This rule can be written in the following form:

Premise 1. IF X_1 is $\tilde{X}_1$ AND X_n is $\tilde{X}_n$, THEN Y_k is $\tilde{Y}_k$.

Premise 2. X_1 is $\tilde{X}_1^*$ AND . . . . X_n is $\tilde{X}_n^*$. (3)

Consequence. y_k is $\tilde{Y}_k^*$, $k=\overline{1,m}$, where $\tilde{X}_1^* \subseteq U_{X1}$, $i=\overline{1,n}$ are fuzzy sets corresponding to the current measured values of the input parameters.

Thus, to get the logical consequence $\tilde{Y}_k^*$ by using (3), Premise 1 and Premise 2 must be carried over to a fuzzy binary relation of the form $R_k(A_1(x),A_2(y))$, $k=\overline{1,m}$ and to a fuzzy unary relation of the form $R(A_1(x))$. Here $A_1(x)$ and $A_2(y)$ correspond to the attributes X_1 and y_k, while $R(A_1(x))$ is defined as:

$$R(A(x)) = \cap_1 \tilde{X}_1^*.$$

Then, according to [1], the logical consequence $R(A_2(y))$, which is $\tilde{Y}_k^*$, $k=\overline{1,m}$ is determined as follows in (3): $\forall k = 1, \ldots, m$, $\tilde{Y}_k^* = R(A_2(y)) = R(A_1(x)) \circ R(A_1(x),A_2(y))$, (4) where $\circ$ is the operation of maximin composition on fuzzy sets.

Two important circumstances in the process of developing industrial IR's should be mentioned. First, it is necessary to ensure that an IR functions in the ON-LINE and REAL-TIME mode, which places increased requirements on the high speed performance of the microcomputers used in IR's. Second, such microcomputers may have a limited memory capacity, and this leads to definite difficulties in storing fuzzy productions as bases for the KB of the IR. Therefore, the proposed method for representing and storing fuzzy productions is based on the following considerations. It is known that elementary fuzzy binary relations characterizing cause-effect pairs are defined in the form [2]:

$$R_{1k}(A_1(x), A_2(y)) = \tilde{X}_1 \times V_{yk} \rightarrow U_{x1}; \tilde{Y}_k, \quad i=\overline{1,n}, \quad k=\overline{1,m} \qquad (5)$$

where $\rightarrow$ is the implication operation in some fuzzy logic. And to form the matrix of fuzzy binary relations characterizing the k-th row (production) in (2) one should take the union of the matrices of elementary fuzzy binary relations, i.e.,

$$\forall\ k=\overline{1,m}, \quad R_k(A_1(x), A_2(y)) = \bigcup_1 R_{1k}(\bullet) \qquad (6)$$

It is natural that for sufficiently complicated OD, i.e., in the case of a significant number of input parameters, such a procedure hardly acceptable due to the limit on the memory capacity of the microcomputer. Therefore, the following path is proposed for resolving this difficulty. We prove the

Theorem. If the fuzzy sets of premises and consequences $\tilde{X}_i, i=\overline{1,n}$, and $\tilde{Y}_n$, $k=\overline{1,m}$, in the conditional propositions P_k "IF X_1 is $\tilde{X}_1$ AND . . . AND X_n is $\tilde{X}_n$, THEN y_k is $\tilde{Y}_n$ " are defined in the form (1) and the corresponding fuzzy binary relations have the form $D = R(\bullet) = ([\bigcap_1 \tilde{x}_1] \times V_{yk} \xrightarrow[2]{} U_{x1} \times \tilde{Y}_k)$, and if the elementary fuzzy binary relations D_{1k} are defined as $D_{1k} = R_{1k}(A_1(x), A_2(y)) = \tilde{X}_1 \times V_{yk} \xrightarrow[2]{} U_{x1} \times \tilde{Y}_k$, $i=\overline{1,n}$, $k=\overline{1,m}$ then for the proposed fuzzy logic with implication operation defined in the form:

$$\mu_{x1}(u) \xrightarrow[2]{} \mu_{yk}(v) = \begin{cases} 1, & \mu_{x1}(u) \le \mu_{yk}(v) \\ (1-\mu_{x1}(u)) \wedge \mu_{yk}(v), & \mu_{x1}(u) > \mu_{yk}(v) \end{cases} \qquad (7)$$

Either the relation

$\forall\ k=\overline{1,m}$, $D_k = D_{1k} \cup \ldots \cup D_{nk}$,

or, as a consequence, the equality

$$(\,[\bigcap_{1}\tilde{x}_1\,]\,x\,V_{Yk} \xrightarrow[2]{} U_{X1}\ x\ \tilde{Y}_k\,) = \bigcup_{1}\,(\,\tilde{X}_{\,1}\,x\,V_{Yk} \xrightarrow[2]{} U_{X1}\ x\ \tilde{Y}_k\,) \qquad (8)$$

$i=\overline{1,n}$, $k=\overline{1,m}$

holds.

Thus, each row (production) in (2) is determined by only a single matrix of fuzzy binary relations, with dimension determined by the cardinality of the universes. For example, the dimension of the matrix is Card U_{X1} x Card V_{Yk} = 25 in the case when Card U_{X1} = Card V_{Yk} = 5. Obviously, the memory capacity of the microcomputers used in industrial IR's permits the storage of dozens of such matrices.

By the foregoing, according to (I) and (7), the logical inference procedure in the language of the membership functions from (4) is realized as follows:

$$\int_{V_{Yk}} \mu_{Yk}^{*}(v)/(v) = \int_{U_{X1}} \underset{1}{\wedge}\ \mu_{X1}^{*}(u)/(u) \circ \int_{U_{X1}xV_{Yk}} (\,[\underset{1}{\wedge}\ \mu_{X1}(u)\,] \xrightarrow[2]{} \qquad (9)$$

$$\mu_{Yk}(v)\,)\,/(u,v) = \int_{V_Y} \underset{u\in U_{X1}}{\vee}\,[\underset{1}{\wedge}\,\mu_{X1}^{*}(u)\,]\,\wedge\,(\,[\underset{1}{\wedge}\,\mu_{X1}(u)\,] \xrightarrow[2]{}$$

$$\mu_{Yk}(v)\,)\,/\,v.$$

REMARK. Other features of the fuzzy binary relation D_k of the subnormal fuzzy set $\tilde{X} = \bigcap_{1}\tilde{x}_1$ used to form the matrices are discussed in [3].

An analysis of the use of this approach for representation knowledge of industrial IR's under the conditions of a specific industrial application enabled us to single out the following main characteristic features:

1) The formation of a collection of fuzzy production rules in a restricted natural language makes it easy for us to act as knowledge englers-fo-r4wulatingwas_conditional propositions the knowledge of experts (industry workers) who fulfilled in the past the same functions that are assigned to the IR's, independently of their level of qualification. This enables us to take into account their experience and intuition, i.e., their knowledge and ideas about the connection between entities and events of a given OD.

2) The translation of the conditional propositions in the language of the membership functions is implemented by passing from linguistic to physical scales of technological parameters, i.e., the entities and events of the OD must be scaled in advance.

3)The given approach ensures the simplicity of the program realization of the logical inference procedure and enables us, in the final analysis, to realize controlling actions, for example, on the servomechanism or on the pneumatics of the manipulator —the actuator of the IR in the direct digital control mode.

5.2 Decision Making in Fuzzy Intelligent Robots

In designing a flexible manufacturing control system for discrete process the authors had to face an important problem of synchronizing the functioning of a group of robots on different bays. This problem has been solved by singling out a robot-coordinator as a master among the group of robots. In this connection some theoretical aspects of decision making are considered here by the sorter robot which is installed in a heat exchanger production shop for the task of on-line rejection of parts.

The IR's sensoring system secures the picking off the tactile (input) information on the single part in the form of a set $x = \{ x_1 \}$, $i=\overline{1,n}$. Here $x_1 \in [x_{1\min} , x_{1\max}]$ where $x_{1\min}$, $x_{1\max}$ are minimal and maximal values of i-th parameter. The output information on the part quality is defined by some y. The parameters of the set $\{x_1\}$, $i=\overline{1,n}$, and y are considered to be fuzzy sets, forming the linguistic variables "Parameter i" and "Quality 1" in the form of triplets:

$$\overset{\circ}{X}_1 = \{ < X_1^j , U_{X1} , \tilde{X}_1 >\}, \qquad X_1^j \in T_1^* (u) , \qquad i=\overline{1,n}$$

$$\overset{\circ}{Y} = \{ < Y_J , V_Y , \tilde{Y} >\}, \qquad Y_J \in T^* (v), \qquad J=\overline{1,10}$$

where T_1^* (u), T^* (v) are term-sets of linguistic variables "Parameter i" and "Quality 1", respectively; $\tilde{X}_1$, $\tilde{Y}$ are normal fuzzy sets of the kind :

$$\tilde{X}_1 = \int_{U_{X1}} \mu_{X1} (u) / u , \quad i=\overline{1,n} \qquad \tilde{Y} = \int_{V_Y} \mu_Y (v) / v \qquad (10)$$

$U_{X1} = V_Y = \{0 , 1 , . . . , 10\}$ are universes. The values of the linguistic variable "Parameter i" are presented in the form of a set of linguistic terms: (negligible, small, . . . , average, extreme}, and those of the linguistic variable "Quality 1" as: {very low, low,... ,average, . . . , highest}.

Define the mapping $q_1 : \tilde{X}_1 \rightarrow U_{X1}$, $i=\overline{1,n}$ in the form:

$$u_j = \text{Ent} \{ (\text{card } U_{X1} - 1). [\frac{x_1^c - x_{1\min}}{x_{1\max} - x_{1\min}}]^{\alpha} \}, \; j = \overline{1,10} \qquad (11)$$

$$i = \overline{1,n}$$

where x_1^c is a current measured value of i-th parameter; and a is a coefficient ($\alpha \geq 1$).

In order to define membership functions in (10) the next procedure is used:

$$\mu (u_j) = 1 - \frac{1}{card U_{X1} - 1} \; | u_j - \qquad (12)$$

$$- \text{Ent} \{ (\text{card } U_{X1} - 1). [\frac{x_1^c - x_{1\min}}{x_{1\max} - x_{1\min}}]^{\alpha} \} |$$

The executive organ of the IR is a standard rotational manipulator. The task of making a decision by the IR consists in determining the manipulator's turning angle depending on the part's quality. For this purpose fuzzy sets $\tilde{Y}_1$ and $\tilde{Z}$ are introduced, forming respectively the linguistic variables "Quality 2" and "Angle" in the form of triplets:

$$\overset{\circ}{Y}_1 = \{ <Y_{11}, W_{Y1}, \tilde{Y}_1>\}, \qquad Y_{11} \in T^*(W),$$

$$\overset{\circ}{Z} = \{<Z_1, W_Z, \tilde{Z}>\}, \qquad Z_1 \in T^*(W_Z) \qquad i=\overline{1,10}$$

where $T^*(W)$, $T^*(W_Z)$ are term-sets of linguistic variables "Quality 2" and "Angle" respectively; $\tilde{Y}_1$, $\tilde{Z}$ are normal fuzzy sets being described in the form:

$$\tilde{Y}_1 = \int_{W_{Y1}} \mu_{Y1}(W)/W, \qquad \tilde{Z} = \int_{W_Z} \mu_Z (Z)/Z \qquad (13)$$

$W_{Y1} = W_Z = \{0, 1, 2\}$ are universes. The mapping $M_1 : V_Y \rightarrow W_{Y1}$ is accomplished as follows: let $= V_Y = V_{1Y} \vee V_{2Y} \vee V_{3Y}$ with $\bigcap_1 V_{1Y} = \phi$, $i = \overline{1,3}$,

i.e.:

$V_{1Y} = \{0, 1, \ldots, 4\}$, $V_{2Y} = (5, 6, 7)$, $V_{3Y} = \{8, 9, 10\}$

In this case there exists $V_1^* \in V_Y$ such that

$$\mu(V_1^*) = \max_{V_Y} \dot{\mu}(v_1), \qquad i=\overline{1,10}$$

where $\mu(v_1)$ is the estimation of the membership function for the i-th singleton $\mu(v_1)/v_1$, i=1, 10 from (10). Then

$$w^* \in W_{Y1} = \begin{cases} 0, & \text{if } V_1^* \in V_{1Y} \\ 1, & \text{if } V_1^* \in V_{2Y} \\ 2, & \text{if } V_1^* \in V_{3Y} \end{cases}$$

Membership functions in singletons from (13) may be estimated as follows:

$$\mu_{Y1}(w_1) = 1 - \frac{1}{cardW_Z - 1} | w_1 - w^* |, \; i = \overline{1,3} \qquad (14)$$

$$\mu_Z(w_{Zj}) = 1 - \frac{1}{cardW_Z - 1} | w_{Zj} - \text{Ent}\{(\text{card } W_Z - 1), [\frac{\psi_1 - 90}{180}]\} |, \; j = \overline{1,3} \qquad (15)$$

In (15) we suppose that the turning angle is $\psi \in [\,90^\circ, 180^\circ\,]$.

The values of the linguistic variable "Quality 2" are given in the form of a set of terms: {hard defect, repeated processing is possible, ready for the next operation), and those for the variable "Angle" in the form of {small, average, big).

5.3 Fuzzy Intelligent Robot for the detection and withdrawal of Defect Parts in the Computer Aided Manufacturing of Evaporators

Here we consider the process of decision making by an industrial sorter-robot (SR) for on-line detection and rejection of defect parts, which is installed in the heat exchanger production shop. Practically, the decision, making process is being executed by an on-board computer and two stages of this process should be remarked:

- the qualitative estimation of parts on the base of input in formation;

- the definition of the manipulator's turning angle using the results of the previous stage.

The input information of the part is picked off in the form of the set $x=\{x_1\}, i=\overline{1,n}$. Here $X_1 \in [\,x_{1min}, x_{1max}\,])$, $i=\overline{1,n}$ where x_{1min} and x_{1max} are minimal and maximal values

of the i-th parameter. The parameters of the set $\{x_1\}$, $i = \overline{1,n}$ are considered to be fuzzy sets, forming the linguistic variables in the form of triplets:

$$\overset{\circ}{X}_1 = \{ < X_1^j , U_{X1} , \tilde{X}_1 >\}, \qquad X_1^j \in T_1^* (u) , \qquad i=\overline{1,n} \qquad J=\overline{1,10}$$

where T_1^* (u) is the extended term-set of the linguistic variable "Parameter i" ; $\tilde{X}_1$ is a normal fuzzy set of the kind:

$$\tilde{X}_1 = \int_{U_{X1}} \mu_{X1} (u) / u \qquad (1)$$

U_{X1} ={0 , 1 , 2 , . . . , 10} is the universe. The values of the linguistic variable "Parameter i" are .

As a mapping: $q_1 : \tilde{X}_1 \rightarrow U_{X1}$, $i=\overline{1,n}$ we use the expression and the membership functions (1). Calculated estimations of membership functions in singletons permit to apply the L-R representation of the membership function and the equation,

$$\mu (u_j) = 1$$

takes place in the point $u_j^* \in U$.

Assume the output parameter (the quality of the part) in the form of a linguistic variable $\overset{\circ}{Y}$:

$$\overset{\circ}{Y} = \{ < Y_J , V_Y , \tilde{Y} >\}, \qquad Y_J \in T^* (v)$$

where T^* (v) is the extended term-set of the linguistic variable "Quality 1"; $\tilde{Y}$ is the normal fuzzy set of the kind:

$$\tilde{Y} = \int_{V_Y} \mu_{\tilde{Y}} (v) / v \qquad (2)$$

$V_Y = \{0, 1, \ldots, 10\}$ is the universe. The values of the linguistic variable "Quality 1" are given .

For the definition of the output parameter the rule of fuzzy conditional inference is used. For the correspondent membership functions the rule of fuzzy conditional inference is taken and calculated using the conditional proposition:

Pl="IF ($\tilde{X}_1$ is A_1) AND ($\tilde{X}_2$ is A_2) AND . . . AND $\tilde{X}_n$ is A_n) THEN $\tilde{Y}$ is B or $\tilde{Y}$ is non B"

where $A_1 \varsigma U_{X1}$, $i = \overline{1,n}$ $B \varsigma V_Y$ are fuzzy sets, corresponding to the terms of linguistic variables. At the same time, the current output fuzzy set may be defined using the rule of Compositional inference of Zadeh [1]:

$$\tilde{\tilde{Y}}^{\circ} = R(A_2(y)) = R(A_1(x)) \circ R_1(A_1(x), A_2(y)) =$$

$$= \left[\bigcap_1 \tilde{\tilde{X}}_1^{\circ} \right] \circ R_1(A_1(x), A_2(y)) \qquad (3)$$

where = R (A_1(x)) and R (A_2(y)) are unary relations, corresponding to the current values of input and output parameters, respectively.

In order to support the proposition P1 we prove the

Theorem If $A_1 \subsetneq U_{X1}$ are logical antecedents in the form of normal fuzzy sets (1) and if the proposition P1 takes place and if $\bigcap_1 A_1 = \varphi$, $\forall i = \overline{1,n}$ then the logical consequent $R(A_2(y))$ in the compositional rule is unknown.

LEMMA. If in proposition P1 the inequality,

$$A_1 \neq A_k \quad (1 \neq k)$$

holds, then the logical consequent of the fuzzy set $\overset{\circ}{B}$ has a discontinuity point.

Thus, taking into consideration the results of Lemma, the subnormal fuzzy set $\overset{\sim c}{X} = \bigcap_1 \overset{\sim c}{X_1}$ from (3) ought to be normalized:

$$\overset{\sim c}{X}_n = \text{NORM}(\overset{\sim c}{X}) = \int_{U_{X1}} \left[\frac{\wedge_1 \mu_{X1}(u)}{\sup_{U_{X1}}(\wedge_1 \mu_{X1}(u))} \right] / u \qquad (4)$$

The input information about the turning angle of the manipulator is represented in form of the linguistic variable "Quality 2":

$$\overset{\circ}{Y}_1 = \{ <Y_{11}, W_{Y1}, \tilde{Y}_1>\}, \qquad Y_{11} \in T^*(W)$$

where $T^*(W)$ is the extended term-set of the linguistic variable "Quality 2"; $\tilde{Y}_1$ is the normal fuzzy set:

$$\tilde{Y}_1 = \int_{W_{Y1}} \mu_{Y1}(W) / W$$

$W_{Y1} = \{ 0, 1, 2 \}$ is the universe. Estimations of membership functions in singletons $\mu_{X1}(w_1) / w_1$ are calculated and have the L-R form. Note that,

$\mu_{X1}(w_1) = 1$ if $w_1 = w^*$

The turning angle of the manipulator is represented by the linguistic variable "Angle":

and $\overset{\circ}{Z} = \{< Z_1, W_Z, \tilde{Z} >\}$ correspondent estimations are calculated.

In order to construct the binary relation $R_2(A_2(y_1)\ A_2(z))$ the following conditional proposition P2 is used:

P2 = "IF $\tilde{Y}_1$ is B_1 THEN $\tilde{Z}$ is C OR $\tilde{Z}$ is non C"

Here, as a binary relation we use the rule of fuzzy conditional inference for the corresponding membership functions,

$$R_2(A_2(y_1), A_2(z)) = \int_{W_{Y1} x W_Z} (\mu(w_{Y1}) \xrightarrow[s]{} \mu(w_Z)) \wedge ((1 - \mu(w_{Y1})) \tag{5}$$

$$\xrightarrow[s]{} (1 - \mu(w_Z))/(w_{Y1}, w_Z) = \begin{cases} 1, \mu(w_{Y1}) = \mu(w_Z) \\ 0, \mu(w_{Y1}) \neq \mu(w_Z) \end{cases}$$

The current value of the fuzzy set $\overset{\sim c}{Z}$ may be defined, as in previous case, using the Zadeh compositional rule of inference:

$$\overset{\sim c}{Z} = R(A_2(z)) = R(A_1(y)) \circ R(A_1(y_1)\ A_2(z)) = \overset{\sim c}{Y_1} \circ R(A_1(y_1), A_2(z))$$

Further, we define $w_z^* \in W_z$ and taking into account that Card W_z =3, we obtain an expression for the value of the turning angle:

$$\psi_1 = \frac{\pi}{2} \cdot w_z^* + \frac{\pi}{2} \tag{7}$$

Thus, if, for example, $w_z^* = 0$, then $\psi_1 = \pi/2$; and if $w_z^* = 1$ then $\psi_1 = \pi$ and so on (not here there that in practice the value of the turning angle vary in the range $\psi_1 \in [\pi/2, 3\pi/2]$).

CONCLUSION

The project is devoted one of important problems Computer Aided Manufacturing Systems. The scheduling and control problems of Computer Aided Manufacturing Systems are described. As an example, the development of Computer Aided Manufacturing Systems for Dylite Expanded polystyrene sheet is carried out. The structure and operation principle of this system are described.

The Computer simulation of Computer Aided Manufacturing System is performed. A program has been written and designed using Macromedia Flash and is realized in IBM PC.

Computer Aided manufacturing system provides the factories and other laces with the required equipments to perform several tasks in a short time using Intelligent machines like Robots.

However, now a days Computer Aided Manufacturing System is very necessary for students to learn because of its multiple approaches.

REFERENCES

[1] Rafik Aliev, Fuad Aliev and Mamedgasan Babae

Fuzzy Process Control knowledge Engineering in Petrochemical and Robotic Manufacturing.

© by Verlag TUV RheinLand GmbH, Koln 1991

[2] Nand K.Jha

Handbook of Flexible Manufacturing Systems

© by ACADEMIC PRESS, INC.

[3] Warnecke, H.J. , and Claussen, C.(1988). Planning of Flexible sheet metal manufacturing systems.

Int.Conf. Flexible Manuf.Syst. 7th

Stuttgart, 1988

[4] Mikell P. Groover

Professor of Industrial and Manufacturing Systems Engineering

Lehigh University

Fundamentals of Modern Manufacturing

© 1996 by Prentice – Hall, Inc.

[5] Kusiak, A.(1990). " Intelligent manufacturing system." Prentice – Hall, Englewood cliffs, New jersy