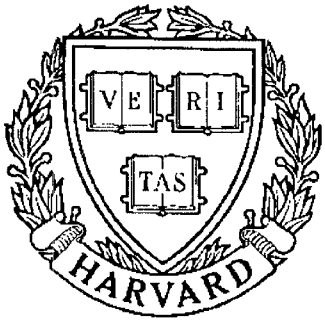
THESIS REPORT
Ph.D.

S Y S T E M S
R E S E A R C H
C E N T E R



Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), the University of Maryland, Harvard University, and Industry

Design and Operation of Hierarchical Production Management Systems

by R. Nagi
Advisor: G. Harhalakis

Ph.D. 91-13

Abstract

Title of Dissertation : Design and Operation of Hierarchical Production Management Systems

Rakesh Nagi, Doctor of Philosophy, 1991

Dissertation directed by : Dr. George Harhalakis, Associate Professor, Department of Mechanical Engineering, and Dr. Jean-Marie Proth, Research Director SAGEP, INRIA-LORRAINE, France

Production Planning, Management and Control of a production system subject to random events are challenging problems. A multi-layer hierarchical approach to tactical and aggregate production planning problems is proposed, wherein the architecture is strongly based on the specific physical system, applicable controls and the complexity of the decision making problem at hand. We address the design and operation of such Hierarchical Production Management Systems (HPMS). Regarding the design aspect, we start by developing schemes for product, machine and temporal aggregation; consistency and controllability issues in the hierarchy have been addressed in the aggregation/disaggregation schemes. These three aggregation schemes for model reduction have been developed and incorporated to the time-scale decomposition of activities, in order to provide a solid theoretical foundation of the architecture. We then proceed to a systematic stepwise design approach for the construction of the hierarchy. It provides the appropriate number of layers and an associated Model as well as Decision Making Problem (DMP) at each level. A model is defined by entities, attributes, links and domains, while a DMP is defined by a set of possible controls (decisions), constraints, and optimality criteria to be optimized over a planning horizon. The operation of the hierarchy consists of a top-down computation of controls, which calls for the resolution of an optimization problem at each level of the hierarchy. The solution of any problem in sequence determines some parameters in the subsequent problem. We detail the mechanism for top-down constraint propagation, which is important in ensuring consistency of criteria and feasibility. The execution then involves the bottom-up feedbacks, and a revision in the plan is carried out if necessary. In particular, the rolling horizon mechanism, and the reaction of the hierarchy to random events has been detailed. A generic job-shop example has been employed to present the design and operation of the HPMS. It is hoped that this methodology can be applied to other types of large-scale complex decision making problems.

Design and Operation of Hierarchical Production Management Systems

by
Rakesh Nagi

Dissertation submitted to the Faculty of the Graduate School
of The University of Maryland in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
1991

Advisory Committee :
Dr. George Harhalakis (Chairman/Advisor)
Dr. Jean-Marie Proth (Advisor)
Dr. D.K. Anand
Dr. S. Azarm
Dr. J.S. Baras
Dr. I. Minis

Acknowledgements

A few words, to place on record ones heart felt feelings are always but necessary. I wish to express my sincere gratitude to my advisors Dr. G. Harhalakis and Dr. J.M. Proth, whose guidance, support, and advice throughout this research were invaluable.

In addition, I wish to thank the members of my thesis committee, Drs. D.K. Anand, S. Azarm, J.S. Baras and I. Minis for their precious time, comments and suggestions for improvement.

I would like to acknowledge the Systems Research Center, the Department of Mechanical Engineering (University of Maryland), and the Institut National de Recherche en Informatique et en Automatique (INRIA-LORRAINE, France) for providing financial support during this project.

Further, I feel especially grateful to my friends, members of the CIM lab : Amy, Howard, Lin, Satish, Sudhanshu, and members of INRIA : Chengbin, Marie-Claude, Thomas, Said, Vanio, Vernadat and Xiaolan, whose help, friendship and encouragement meant a great deal. Lin has left me in awe of his exemplary and unselfish readiness to help and give; I have so often asked unhesitatingly from him. Chengbin and Dahai Chen are acknowledged for their contributions in section 4.9.2. I am also grateful to my apartmentmates Navin, Praveen and Rakesh for their company and encouragement during troubled times encountered during this work.

Lastly, even though words cannot contain, in futility, I attempt to express my deep gratitude and indebtedness to my parents, my family - both in India and in U.S. for their love, affection, moral support and understanding at every step of the way.

Contents

List of Figures

Chapter 1

Introduction

Although recent years have seen significant progress in automating and integrating Design, Process Planning, Machining, and Inspection, Production Planning, Management and Control seem to have received little attention. Decision making and optimal real-time control of a production system, subject to both endogenous (e.g., resource failures), as well as exogenous (e.g., unscheduled orders, delayed receipts of material) random events is a challenging problem. Maxwell *et al* [59] state : *"Billions of dollars are wasted in US each year by manufacturers of discrete parts because of inadequate procedures for controlling inventory and production."* The need for developing and implementing production planning systems for a wide variety of systems is obvious in light of competitiveness. Most currently available software systems, such as Manufacturing Resource Planning or MRP systems, fail to directly address some key aspects of the overall production control problem. Our objective is to define an architecture for efficient production management, which can be well integrated in a CIM environment. A Hierarchical Production Management System (HPMS), organized in several hierarchical levels, is intended to address the complexity of global problems. The number of levels depends on the complexity of the manufacturing system. Progressive decision making centers are located at the strategic, tactical and operational levels. The hierarchy starts at the top-most corporate strategic center and flows through to the bottom-most execution module, where materials are transformed to finished products: the plant floor.

This work presents a methodology for decision making or problem solving in large scale production systems. A framework for building a Hierarchical System to model large scale production systems is proposed. We also present the execution of such a system to effectuate optimal decision and control. It is hoped that the principles of design of such a system will be equally applicable to other types of large scale systems.

1.1 Scope of Production Management

Production planning is a complex decision making process, the end result of which is to balance the demand and supply of part types, quantities, due-dates, and resource levels over a relatively large horizon. Scheduling involves time phasing of the production plan, to generate a Gantt-chart of activities over a relatively short horizon. Monitoring of production activities, and reacting to random disturbances in real-time constitutes the control function.

Two distinct approaches to production planning and scheduling have been adopted in the past. The first is a monolithic approach, wherein the entire problem is formulated as a large mixed-integer linear programming type problem. The second is a hierarchical approach which partitions the global problem into a series of sub-problems that are solved sequentially, such that the solution at each level imposes constraints on the subsequent lower level. The fundamental advantages of the hierarchical approach are: (i) reduction of complexity, and (ii) gradual absorption of random events.

In this work, we primarily address the production planning function of decision making and propose a hierarchical approach to it. In the future, we hope to extend these concepts to lower and higher levels of production management.

1.2 Research Issues

The basic goal of the hierarchical approach in production management is to decompose the global problem into a hierarchy of sub-problems. For example, figure 1.1 represents a possible decomposition of the planning

function. The long term level plans production of an aggregate product entity on the entire facility over a long horizon. The medium term level disaggregates the long term plan over a shorter horizon, leading to the production plan of product families on manufacturing cells. Finally, the short term level disaggregates the medium term plan over an even shorter horizon, to provide the production plan of individual products, on individual machines.

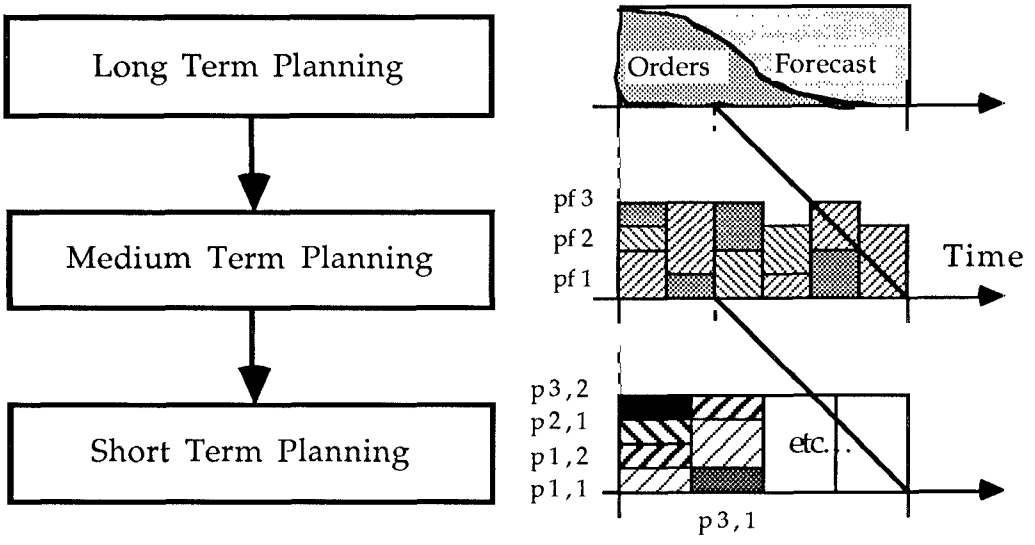


Figure 1.1 : Hierarchical Production Planning

The first step in this direction is to define the manufacturing system/environment to be addressed : (a) definition of the framework, (b) assumptions related to discrete parts manufacture, and (c) consideration of the most important random events. We restrict ourselves to the tactical and operational levels.

Some important issues involved in the **design** of such a hierarchy are :

(1) The appropriate number of levels, their models, entities, definitions, planning horizons, domains etc.

(2) Aggregation/Disaggregation schemes for higher level entities.

(3) Determination of horizon lengths for each decision making problem.

Some important issues involved in the **execution** of such a hierarchy are detailed below.

(1) Solution algorithms to resolve these optimization problems at each level have to be developed.

(2) Mechanism of top-down constraint propagation.

(3) Bottom-up feedback required for a closed loop real time control process.

Finally, the work culminates in recommending integration of this architecture in a CIM environment. Effective management centers around the availability and exchange of a wide variety of reliable information, most of which is already present in the distributed database of a CIM architecture. Integration of this decision making architecture with the CIM environment will eliminate conflicts and reduce redundancy in data input.

1.3 Research Approach

We propose a methodology for the design of hierarchical architectures, along with algorithms for execution and control. The models are derived in a way that facilitates gradual planning decisions, with increased levels of detail. The computation of the control is a top-down process, which calls for the resolution of an optimization problem at each level of the hierarchy. The execution then involves the bottom-up feedbacks, and a revision of the plan is carried out if necessary.

Design Approach:

(1) Architecture of the Hierarchy.

It is observed that the number of levels is proportional to the complexity of the manufacturing system, and the variety of random events affecting it. The appropriate number of levels in the hierarchies, their definitions at each level, planning horizons, domains etc. have to be defined. We approach the problem by modeling each level, including a set of pertinent *entities* to each level, the *links* between these entities, a set of *attributes*

associated with each entity, and the *value set* associated with each attribute. This is followed by the formal definition of the decision making problem at each level along with a set of possible *controls* (decisions), a set of *constraints*, and one or more optimality *criteria* over a given *horizon*.

(2) Aggregation/Disaggregation schemes for higher level *entities*.
Our underlying methodology of aggregation/disaggregation and development of models differs from previous work [40] which is based on a particular cost structure. We propose to develop aggregate models using new and existing clustering techniques so that the methodology finds applicability to more general large-scale systems. Furthermore, we address theoretical issues related to the consistency of such schemes.

(3) Determination of *horizon* lengths for a problem with stochastic events.
Horizons should be derived in such a way that each level is able to absorb most of the random events associated with it, causing minimal feedback-error to the previous level, and infrequent need for re-optimization. This involves some theoretical research relating to the determination of optimal horizon lengths.

Operation :
(1) Solution algorithms.
The computation of the control is a top-down process, which calls for the resolution of an optimization problem at each level of the hierarchy, and thus constraints for the lower level (if any) are created. Thus, solution algorithms to resolve these optimization problems have to be developed. This may even involve development of heuristic or approximation algorithms.

(2) Mechanism for top-down constraint propagation.
Problem solutions in the hierarchies are attempted in a sequential manner; the solution of any problem in the sequence determines and fixes some parameters in the subsequent problem. We propose propagation of constraints in a manner that does not render the subsequent problem infeasible, while ensuring consistency of criteria.

(3) Mechanism for bottom-up feedback.
The execution involves bottom-up feedbacks, and a revision of the plan if necessary. Thus, a closed loop real time control process can be developed.

Regarding integration, our approach consists of the identification of inputs, outputs, and interactions of the HPMS with existing CIM databases.

1.4 Organization

This dissertation is organized as follows. Chapter 2 presents the concepts of hierarchical decomposition, and a literature survey of related work in production management. Chapter 3 contains a performance evaluation of a hierarchical scheduling policy for a simple single machine scheduling problem. This chapter intends to justify the applicability of hierarchies to production management related problems. Chapter 4 is devoted to the theory of aggregation in planning. A production planning problem with a set of criteria is approached by a two level hierarchy. We begin with a restricted aggregation scheme for parts and machines and prove optimality of the hierarchical approach. Then, we present extensions in several directions, and finally present the general aggregation scheme. Chapter 5 addresses the generalized planning problem with general criteria and applicable controls. This chapter employs the aggregation theory developed in chapter 4. An application of this methodology to job-shop example is presented in chapter 6. Finally, in chapter 7 we draw our conclusions and recommend some directions for further work.

Chapter 2

Literature Survey

This chapter is devoted to the basic definitions and concepts involved in the problems of production management. The first section describes the principles of hierarchical decomposition, the major advantages, and classification of hierarchies. A literature survey of the major works in this direction, both from a historical perspective, as well as the current state-of-the-art constitutes the major contents of the second section of this chapter.

2.1 Hierarchical Decomposition: Principles and Types

A hierarchical decomposition can assume different meanings according to its context. However, the essential idea of all hierarchical decomposition schemes is the partition of a global problem into sub-problems. These sub-problems are either solved sequentially, such that the solution of a sub-problem imposes constraints on the subsequent sub-problem, or solved simultaneously in a coordinated fashion (see section 2.1.2).

2.1.1 Advantages of Hierarchical Decomposition

The fundamental advantages of the hierarchical approach are [19] :

(1) Reduction of complexity : Breaking a problem into sub-problems is a standard method of simplifying the solution process. The partitioning should be done in a way that the interaction between sub-problems are acceptably weak, or the global problem is broken down into a series of sub-problems which are sequentially solved, in the sense that the solution of a problem at one level imposes constraints on the problem at the next lower

level one; the global solution is obtained when all problems are solved. Thus, the initially intractable problem can be rendered pliable.

(2) Coping with uncertainty : Another important benefit of hierarchical decision making is obtained when the system is subject to random events and uncertainty. Decision making in uncertainty could lead to frequent recomputations; monolithic models would require the entire problem to be resolved repeatedly, while the hierarchical approach can gradually absorb random events without the need to resolve any higher level problems. This results in large savings in computational burden, apart from added stability to the overall control. Decisions at various levels in the planning process are made at different points in time. Higher level decisions are more aggregate, and need not explicitly consider uncertain data at detailed levels. Random events with a lesser impact on the system can be absorbed at lower levels.

(3) Parallel with hierarchical organization of the physical systems : Hierarchical planning or decision making is often performed parallel to the organizational structure of the physical system at hand. This has important implications from the organizational aspects and personnel hierarchies.

(4) Reduced need for detailed information and better forecasting : Higher levels in hierarchies are more aggregate and do not require detailed information; this not only reduces dimensionality, but also allows a longer "look ahead" capability. Forecasting is usually easier and more accurate for aggregates than for detailed entities.

2.1.2 Basic Types of Hierarchies

Mesarovic *et al* [61], identify three types of hierarchical systems, which in a sense represent a classification of existing hierarchical systems.

2.1.2.1 Descriptive Hierarchies

For the complete and detailed description of complex systems, while retaining simplicity in description and behavioral aspects, descriptive

hierarchies are employed. A system is described by a family of models, each concerned with the behavior of it, as viewed from different levels of abstraction. Models must be independent of the functioning for an effective description of a *"stratified system."* The levels of abstraction are referred to as *strata*. Each stratum in the hierarchy is associated with a different set of relevant variables, which allow the study to be confined to one stratum only. Understanding the system increases by crossing the strata: in moving down the hierarchy, one obtains a more detailed explanation, while in moving up the hierarchy, one obtains a deeper understanding of its significance. Lower strata are concerned with the operation of the subsystems, leaving the study of inter-relationships for the higher strata. Selection of strata, in terms of which system is described, depends upon the observer, his knowledge and interest in the operation of the system, although for many systems there are some strata which appear as natural or inherent [61].

Such hierarchies are primarily employed for system description, and detailing function behavior. They usually do not support decision making aspects or control, hence their applications are limited to system modeling and description. Examples of such hierarchies are illustrated in [61].

2.1.2.2 Multilayer Hierarchies

These hierarchies, also known as decisional hierarchies, appear in the context of a complex decision-making process. Complex decision-making is associated with a fundamental dilemma: on one hand, there is a need to take timely and prompt action, while on the other, there is an equally great need to understand the situation better and retain perspective of the consequences and relationships of actions in a complex situation. This problem may be resolved by multilayer hierarchies, where one defines a family of decision problems whose solution is attempted in a sequential manner, in the sense that the solution of any problem in the sequence determines and fixes some parameters in the subsequent problem. This has to be done in a way that the latter is completely specified and its solution can be attempted. The solution of the original problem is achieved when all sub-problems are solved. This arrangement is

represented diagrammatically in figure 2.1. Each block represents a decision-making unit, which receives information from the system. The output of a unit, say D_1, represents a solution, or the consequence of a solution, of a decision problem which depends upon a parameter fixed by the value of the input X_1, which in turn is the output of a unit on a higher level. Such a hierarchy of decision layers is termed as a multilayer (decisional) hierarchy.

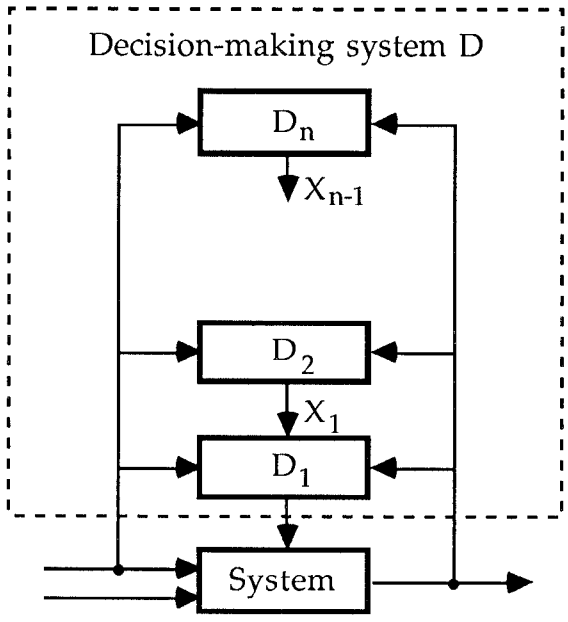


Figure 2.1 : Multilayer hierarchy of a decision-making system [61]

2.1.2.3 Multilevel Hierarchies

These hierarchies are also known as organizational hierarchies. It is necessary that : (i) the system consists of a family of interacting sub-problems which are recognized explicitly, (ii) some of the subsystems be defined as decision (making) units, and (iii) the decision units are arranged hierarchically, in the sense that some of them are influenced or controlled by other decision units.

Designing such a system consists in the assignment of the tasks or roles which various levels or individual units have to perform. Decomposition of the overall systems task is performed, by assigning a sub-problem to

each decision-making unit (infimal unit). Infimal units perform their decision-making independently, though *coordinated* through a supremal unit. Coordination is concerned with the existence of supremal control under which infimal units can solve their local control problems. *Consistency* is an important consideration: i.e., whenever the supremal and infimal units can solve their problems, then an overall solution exists. The supremal-infimal (or master-slave) relationship exists between each consecutive echelon of the hierarchy.

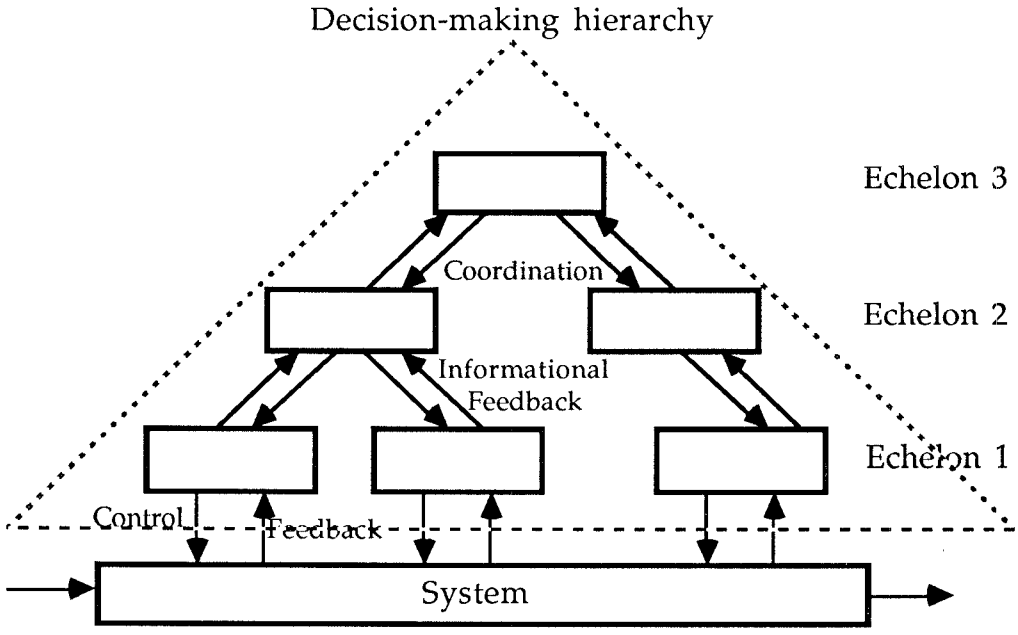


Figure 2.2 : Multilevel organizational hierarchy; multiechelon system [61]

2.2 Hierarchies in Production Management

This section is devoted to the hierarchies proposed in the production management domain. Most of these are multi-layer systems, each with its unique set of assumptions, layering method, number of layers and decision making methodology.

Literature surveys in the field of hierarchical production management can be found in Gelders and VanWassenhove [33], Dempster, Fisher *et al* [19] and Libosvar [57]. Although it is difficult to classify the work surveyed

here in distinct sections, the first sub-section is devoted to the hierarchical production planning related literature. The second sub-section is devoted to hierarchies in FMS, AI approaches, and other conceptual architectures.

2.2.1 Hierarchical Production Planning

The work of Hax and Meal [40], developed at M.I.T. in the seventies has been considered a substantial contribution in the area of hierarchical management. Production planning is the major concern of this work.

2.2.1.1 Framework of HAX and MEAL, and Related Models

Hax and Meal [40], describe a hierarchical model designed for a particular implementation, but also provide a general direction to subsequent work. Four decision levels are considered for a multi-plant firm. The highest level model decomposes the problem into decoupled single plant problems, by determining the products to be produced in each plant. The subsequent hierarchy is intended for the management of a single plant. The highest level involves static (one time) decisions of a decomposing nature. Thus, it differs from the rest of the hierarchy, which is typically multi-layered for dynamic decision making. We restrict our attention to these lower levels. Bitran and Hax (1977) [12] discuss this hierarchical framework for production planning problems.

Based on the analysis of the production process, and a typical cost structure, three levels of aggregation are considered for products:

1. *Items* are final products to be delivered to customers.
2. *Families* are groups of items which share a common manufacturing set-up cost.
3. *Types* are groups of families that have similar costs per unit of production time and similar seasonal patterns.

A production planning level is consistently modeled according to the aggregation scheme. Higher level decisions impose constraints on lower level actions, which in turn provide necessary feedback. In this top-down constrained approach, the assumption is that higher-level decisions have a more significant impact on the objectives.

The highest level of the hierarchical planning system allocates production capacity among product types, by means of a linear programming based aggregate planning model. The horizon is normally longer than a year (15 months), in order to take into account demand fluctuations of products. Production and inventory holding costs are the only costs considered; set-up costs are ignored at this level. This model is intended to determine an optimal trade-off between inventory holding and overtime (production) costs, while leaving set-up costs out because of secondary impact. At this and all subsequent levels, the concept of a rolling horizon is employed, in order to perform repetitive open-loop optimization.

The second step in the planning process is the disaggregation of production quantities for types (computed by the higher level) to determine the production quantities for families. This is performed by a heuristic, based on Economic Order Quantity (EOQ) and safety and overstock computation techniques. Capacity allocated to a product type is split among the families belonging to that type, for which the inventory is below the safety threshold. For these families, production volumes are chosen as close as possible to the EOQ, provided they do not leave an overstock at the end of the period.

Finally, the third decision level consists of an item disaggregation heuristic, based on Equalizing of Run-Out Times (EROT). This decomposition technique maximizes the time between set-ups, by requiring the items of a family to run-out at the same time. Karmarkar [49] provides a proof of optimality of this technique.

Consistency between decisions at different levels is attempted to be ensured by the top-down constraints. However, sometimes the disaggregation is infeasible, i.e. top level constraints may yield an empty feasible set at a lower level. Gabby [28] considers an aggregate plan for which a feasible detailed plan (i.e. one without backlog) exists. However, this requires detailed demands to be known over the consistency horizon, thus, loosing some benefit drawn from the need of less detailed demand

required in hierarchies. He derives necessary and sufficient conditions for a consistent disaggregation; see Gabby [28] for treatment of multi-item, single echelon, capacitated production problems.

Bitran and Hax (1977) [12] propose a computational improvement to Hax and Meal's model by reformulating family and item disaggregations as knapsack problems, for which they provided efficient solution algorithms [13]. Numerical results indicated that for low set-up costs, the solutions are near optimal.

Bitran, Haas and Hax (1981) [10] prove that the EROT method is an optimal disaggregation scheme that minimizes the cost of initial inventory. This work improves the knapsack-based method in that: (i) at the family disaggregation level, the expected number of set-ups are minimized for a shorter horizon, allowing for greater responsiveness to seasonal variations, (ii) a "one step look ahead" procedure is required for disaggregation, and (iii) the families' production volumes are modulated in order to keep them close to their EOQs. The enhanced work was shown to produce superior results through a set of simulations.

Erschler, Fontan and Merce [23] derive necessary and sufficient conditions for consistency, based on mass balance equations. They propose the look ahead procedure to be extended to all periods, for which detailed demand is known. The necessary conditions are then employed to enhance the knapsack method, which was shown to perform better.

Graves [36], adopts a different approach to the problem and introduces feedback between decision layers. Based on the product aggregation scheme of Hax and Meal, the problem is formulated as a monolithic mixed integer program, which is decomposed by lagrangean relaxation. This decomposition yields two subproblems : (i) aggregate planning (linear programming model), and (ii) disaggregation problem (lot sizing problems for each product type). The linking mechanism for these two subproblems is an inventory consistency relationship which is priced out by a set of Lagrange multipliers. The best values of the multipliers are found by an

iterative procedure which may be interpreted as a feedback mechanism in the Hax-Meal framework.

Other extensions of the Hax-Meal model are directed towards incorporating it to multi-stage fabrication and assembly systems. Bitran, Haas and Hax (1982) [11] propose an extension of their previous work to a two-stage fabrication/assembly system. On an industrial test-bed, they successfully compare results obtained by the extended hierarchical planning system to those of an MRP based system. However, under this model product families are subject to a rather restrictive definition.

Advantages and shortcomings of the Hax-Meal framework are summarized as follows:

Advantages :

- reduction in computational size, compared to a monolithic optimization with complete detailed data.
- enabling of managerial interaction.
- reduction of data requirements (due to reduction of part entities).
- explicit consideration of capacity.
- models at each level are consistent with the aggregation scheme, ensuring coherent decisions at each disaggregation level.
- secondary impact of set-up costs is gaining more relevance with the current trends in manufacturing, e.g. Flexible Manufacturing Systems permit quick changeovers between parts.

Shortcomings [57] :

- The product aggregation scheme proposed is only relevant to a particular class of production systems, and models at each level are based on a typical cost structure.

- Detailed data requirement are reduced only if backordering is permitted.
- No randomness is taken into account and forecast errors have to be absorbed by safety stock.
- No spatial decomposition of the system is proposed.

2.2.1.2 MEIER's Hierarchical Control of a Production System

Meier [60] considers planning and control of a general multi-stage assembly/disassembly manufacturing system, to minimize a performance criterion composed of inventory holding costs, backlogging costs, and production costs. A two-layer "master" and "slave" hierarchy is developed.

The master-level determines the amount of operations (per operation type) to be performed on machines, over given planning periods, via a linear programming model. For the treatment of large-scale problems, aggregation-disaggregation procedures are proposed.

The slave controls the activities of the machine in real time, in order to satisfy the plan provided by the master. At this level, a set of priority rules are employed to determine the activities of the machines, as a function of the volumes of operations, and the inventory state.

Discrepancies are possible, if the slave cannot execute the master's decision precisely. This is mitigated by rolling horizons, and repeated optimization. In this way, a close loop planning and control system is emulated, whereby the schedule on the rolling horizon yields to auto-adaptive planning corrections.

Hillion, Meier and Proth [44] present a top level model for a new approach to hierarchical production planning. At this level, the entities of relevance are production sub-systems and part families. Aggregation procedures to determine these entities are detailed, and the problem is formulated as a linear program.

2.2.1.3 Optimality and Aggregation-Disaggregation Issues in Hierarchies

The suboptimality of the hierarchical approach has been addressed by Dempster, Fisher *et al* [19], where the authors review the concepts of hierarchical planning and site some examples. They demonstrate that multi-level decision problems can be modeled as multi-stage stochastic programs. Then, the analytical evaluation of any hierarchical system can be assessed with respect to the optimality of this stochastic program. An analytical evaluation, and a proof of asymptotic optimality of a simplified version of the hierarchical planning system of Armstrong and Hax [6] are presented. The job-shop consists of a set of parallel identical machines; the higher level decision is to determine the optimal number of machines, m. The lower level is concerned with the problem of scheduling n jobs on these m machines, in order to minimize the completion time. There are costs associated with purchasing machines, and costs proportional to the completion time. The performance of this hierarchical system is compared with a stochastic model, in which both costs are treated in a single model, and processing times are random with a known mean value. When the number of jobs tends to infinity, the performances of the two systems become identical. The application of such analytical results is dependent on the criteria and the model chosen. Also, owing to lack of accurate estimation of the quality of solutions provided by hierarchical systems, this evaluation method has found little application to other systems.

Aggregation-Disaggregation is another major issue concerned with the design of hierarchical systems. Infeasibility can be encountered during the disaggregation process. Krajewski and Ritzman [54] provide a survey of the problems and research in this field from a manufacturing, as well as a service organization perspective. These disaggregation problems are encountered between aggregate planning at the top level and more detailed decisions of inventory control and scheduling at the bottom level. They draw attention to the lack of an interface mechanism, which diminishes the utility of the solution procedures of aggregate planning, inventory control and scheduling.

Aggregation schemes are of three types : (i) over time, or temporal, (ii) over products, and (iii) over machines, or spatial. The Hax and Meal framework only considers product and temporal aggregation. Except Meier [60], who includes aggregation of machines, there is no work addressing aggregation in all three dimensions.

Zoller [79], considers product disaggregation with a two-level economic model, in which aggregate production is determined by minimizing a cost function. The product mix and sales price are determined at the lower level in order to maximize profit, assuming that the demand volume depends on sales price and that the function relating the two variables is known. An iterative algorithm to reach optimal solutions of the low-level problem is presented. Gelders and Kleindorfer [31] [32], present a formal model of a one-machine job-shop scheduling problem with variable capacity, considering trade-offs between overtime and detailed scheduling costs. The scheduling problem considers minimizing the sum of weighted tardiness and weighted flow-time costs for a given capacity plan. Various lower-bounding structures for the problem are analyzed, and a branch-and-bound scheme is outlined. Computational experience with the algorithm in presented in [32].

Erschler, Fontan and Merce [23] consider the consistency of the disaggregation process in hierarchical planning, and necessary and sufficient conditions for consistency of decisions in a two-level structure are presented. A sub-set of these conditions improves upon the disaggregation procedure of Bitran and Hax [12], [13].

Axsater [7] addresses a double aggregation over products and machines, which results in an aggregate planning problem in terms of product groups and machine sub-systems at the aggregate level. This aggregate plan may not be possible to disaggregate. He presents perfect aggregation conditions, i.e. necessary and sufficient conditions on the aggregation matrices, where it will be possible to disaggregate any aggregate plan. If perfect aggregation is not possible, a good approximate solution can be

reached through a mathematical formulation of an approximation problem.

2.2.2 Hierarchical Systems for Flexible Manufacturing

Flexible Manufacturing Systems (FMS) offer challenging planning and scheduling problems due to their versatility, flexibility, quick changeover time, and association of automated material handling systems along with the need for efficient resource utilization (brought about by high capital investment). Morton *et al* [63] and Stecke [74] identify some issues that render FMS scheduling different from traditional manufacturing systems. Thus, the need for different principles in FMS design, planning, scheduling and control. Stecke [74], identifies FMS related problems, and suggests mathematical models useful to their analysis.

2.2.2.1 Conceptual Models in Production Management

O'Grady [65] provides an introduction to automated manufacturing systems with their characteristic features. Production Planning and Control structures of traditional and automated manufacturing systems are identified. Three major hierarchical frameworks: AMRF's (Automated Manufacturing Research Facility of the NIST), CAM-i (Computer Aided Manufacturing International Inc.), and one developed by their team have been described. None of these architectures have been implemented in their entirety.

Doumeingts (1985) [20] discusses the necessity of conceptual models in production management. These models help in figuring the whole system through abstract terms, and in designing it with the entities and links between them. The precise objective of such a model is to deal with a production management system, or a physical system, or even the driving of a discrete-continuous system. The GRAI (Graph with Results and Activities Interrelated) method, and especially the GRAI grid, presented in [20] corresponds to the last type of conceptual models. Decentralization methods for decision making through other conceptual models include : (i) ICAM model, (ii) General model of CAM-i, and (iii) Specific models for CAM-i.

The National Institute of Standards and Technology (NIST) within its Automated Manufacturing Research Facility (AMRF), has developed a five level hierarchical structure: Facility, Shop, Cell, Workstation, and Equipment levels of control. This decomposition is functional in nature [45], and is fixed (i.e. the structure is rigid and not system dependent). Their efforts [18], [46], focus on the design of a real-time production scheduler at the shop and cell levels, but not yet to the design of an integrated planning and scheduling hierarchy for the management of the entire factory.

Project 418 of the European Strategic Program for Research and development in Information Technology (ESPRIT) has resulted in a hierarchical structure, with multi-level dynamic planning and scheduling, and a GRAI modeling method for decision making [14]. This structure is conceptual in nature and helps in modeling, and identification of information and decision flow. However, it does not provide solutions to the decision or control problems. Roboam *et al* [67] present an integrated methodology for the analysis and design of manufacturing system.

CIM-OSA (Computer Integrated Manufacturing - Open System Architecture) is a current development carried out by the CEC ESPRIT Consortium AMICE [4], [47], [48], [50] consisting of 21 European organizations. The goal of CIM-OSA is to provide an OSA which covers all the internal and external information needs of all the functions in a manufacturing enterprise; it bears some ideas of hierarchical control and management. First, it presents a functional view in which the enterprise functionality and behavior are modeled, using a functional decomposition principle. Functions of the enterprise are modeled in terms of business process and enterprise activities. Enterprise activities are the leaves of the functional decomposition trees and describe the enterprise functionality. Business processes are nodes of the functional decomposition trees and describe enterprise behavior, i.e. the way enterprise activities and lower-level business processes are procedurally chained to form longer processes. Second, ideas of hierarchical organization can be found in the CIM-OSA

resource view which describes resource organization for the purpose of resource management. Resources (i.e., components, machines, people, applications) are required by enterprise activities for their execution. However, it may happen that resources can be aggregated to form high-level resources (which are called in CIM-OSA resource sets for permanent aggregation and resource cells for temporary aggregation). This recursive principle can be used as much as desired to form larger resource units. Aggregated resources can be linked to enterprise activities and/or to business processes. Finally, CIM-OSA employs hierarchical principles in its organization view, which is concerned with the modeling of the responsibilities and authorities in the enterprise, as well as the modeling of decision levels and decision centers, using the organization unit construct. An organization unit can be a person, a group of persons, a service, a department, a plant, etc. Organization units have control over the functions (i.e., business processes and enterprise activities) of the function view and indicate at which organization level they belong, where an organizational level is defined by its authority, responsibilities, horizon and period.

However, CIM-OSA is essentially an enterprise modeling framework aiming at producing an executable model of the enterprise. It provides no means to decide on the number of organizational levels, which is left to the model designer. We believe that the work presented in this thesis could provide CIM-OSA with a sound basis for the design of its function and organization views. CIM systems to be developed according to CIM-OSA intend to support all levels of management in their strategic, tactical and operational planning, as well as the direct operation of product design, and manufacturing. It is proposed that such systems will support all decision making processes, by both, providing the required information, and by allowing the simulation of alternatives and optimization of solutions before implementation. The emphasis thus far has been enterprise structuring and modeling; a 3-d architecture has been developed [53]. However, the details of decision making as well as optimization processes are not detailed at this time.

2.2.2.2 Artificial Intelligence Based Approaches

Villa *et al* [75] propose a hierarchical framework to model FMS control. They define an FMS as a structure composed of a physical system, an information system and a decision-and-control system. The task performed by the latter can be divided into periodic planning and event-driven control. The authors suggest a decomposition, based on physical insights rather than mathematical techniques, to develop a tree-like structure with decision makers at each node. They present a hierarchical control structure by integrating mathematical tools from Control Theory with relational tools derived from Artificial Intelligence. This matching is suggested to provide an effective implementation of Expert Control System. This framework leading to an "Integrated Control Structure" is based on the same spatial decomposition of the physical system and frequency-band partition of events, whereas the decision-making units use AI tools to solve these problems.

The control units are modeled as a knowledge base and an inference engine; each "layer" of control units is related to a frequency band. Hence the horizon ratios of different control layers must be consistent with the event frequency ratios. The importance of an event is defined as the index of the higher control layer, at which its effects are likely to influence the control. Thus, the optimal control strategy for each decision module consists of solving an open-loop-feedback problem to update its policy, each time an event occurs with an importance greater than its level index. This updating process includes ensuring consistency between the policies determined by the lower level decision modules. The excessive computational capacity required in solving a planning problem is reduced by retrieving information by pattern-matching from the the module's knowledge base. The inference engine: (i) selects the best policy according to the system state and event type, (ii) coordinates the lower level actions, based on rules retrieved from the knowledge base of the control policy, and (iii) feeds the knowledge base with a description of the consequences of the control in terms of the resulting dynamics of the system.

Shaw and Whinston [70] address the application of generic artificial intelligence techniques to the planning (process planning) and scheduling of flexible manufacturing systems. The complex scheduling problems in FMS are made tractable by nonlinear planning. Planning for conjunctive goals is referred to as nonlinear planning because the resulting plans are partially ordered. Nonlinear planning systems seek to break a problem into sub-problems, using the divide-and-conquer strategy, while taking interactions of sub-plans into account. They employ a two-level hierarchy to decompose an n-part-m-machine scheduling problem into n sub-problems, with each sub-problem defined as the routing of one part. The primary interactions between the sub-problems are their sharing of m machines. The objectives are to minimize makespan and avoid conflicts. Finally, the system considers alternative operations and revises existing plans if bottlenecks are detected. To summarize, a planning scheme is composed by the following four step procedure :

1. Generate a linearly-sequenced plan for each task.
2. Identify problematic interactions between the planning steps.
3. Use precedence constraints to avoid conflicts.
4. Identify alternative planning steps to improve performance.

In step 1, the inference engine calls upon a backward chaining procedure to search for the best planning steps, based on "means-ends analysis." In each intermediate planning step, the inference engine searches the rule-base to select the best operation to apply; node expansion is carried out with heuristic and constraint information. Steps 2 and 3, for conflict detection and resolution, dynamically decide the precedence relationships between conflicting actions. Parallelism among subplans is maximized by the "least commitment strategy". Step 4, called Plan-Revision, reassigns waiting jobs to alternative resources, so as to achieve better utilization and performance.

Doumeingts (1986) [21] identifies the significance and potential application of AI techniques in the field of CIM. The GARI system for process

planning (University of Grenoble) and the ISIS (Intelligent Scheduling and Information System) system in scheduling (Carnegie Mellon University - Robotics Institute) are detailed below, and referenced as major contributions in this field.

PATRIARCH is a hierarchical planning and scheduling system developed at Carnegie-Mellon University. In [63], Morton and Smunt highlight issues relating to FMS scheduling, which should integrate: (1) hierarchical structure, (2) decision support capability, (3) advanced knowledge representation, and (4) accurate practical large-scale heuristics.

The four levels of the PATRIARCH system include: Level (1) strategic planning, Level (2) capacity planning, Level (3) scheduling, and Level (4) dispatching. The first level considers ill-structured, long range problems found in strategic planning. Due dates, penalties, resource constrains, activity design, and other costs are all quite fuzzy. Hence, human expertise and manual decision support are essential. In the second level, semi-ill-structured medium range problems are considered. Capital is fixed in this time-frame, but employment levels, number of shifts, and active machines can be chosen. The planner can test and select among several scenarios; hence, it is termed as a semi-automatic mode. Level (3) represents the largest level of detail in terms of activity and resource structure, at which it is feasible to solve a well-structured problem by operations research heuristics. Level (4) is fully detailed, and too large to be solved by operations research heuristics. Instead, fast dispatch routines guided by priorities and assignments passed from level (3), and a large number of simple historical rules-of-thumb, make dispatch decisions. Some manual intervention may be required.

Further, the Intelligent Systems Laboratory of the Carnegie-Mellon Robotics Institute has developed large scale scheduling systems with sophisticated knowledge representations (Fox [24] [25]). These systems, namely ISIS - a job shop scheduling system, and CALLISTO - a project scheduling system, attempt to integrate all levels of hierarchical production planning. The original work of ISIS utilized a reservation

system and backward scheduling with beam search, which were only moderately successful for loading of machines. CALLISTO uses a forward dispatch approach, improved in Morton, *et al* [63]. The FMS version incorporates many of the concepts of CALLISTO and ISIS, along with the consideration of multiple machines and dispatching priorities from detailed planned schedules.

The ISIS was followed by the development of a successor system called OPIS (Opportunity Intelligent Scheduler [73]). OPIS employs constraint propagation techniques to update schedule descriptions and detect inconsistencies introduced. These systems were tested using simulated data from actual manufacturing environments.

2.2.2.3 An Approach by Hildebrant and Suri

Hildebrant [41], Hildebrant and Suri [42], present a methodology and a multi-level algorithm structure for scheduling in real-time control of Flexible Manufacturing Systems (FMS) with unreliable machines connected by automatic transportation means. The control consists of satisfying a given demand of part types in order to minimize the makespan (production cycle time). The hierarchical decomposition of the problem leads to a three level hierarchy.

At the high level, a multi-class queueing model is used to compute the completion time of tasks as a function of the utilization of machines and fixtures, which accounts for the possibility of failures. For each failure state of the system, a formulation (nonlinear programming with mean value analysis) helps seeking optimal allocation of tasks to resources. The solution results in the allocation of resources to parts and routes.

The intermediate level, Even-Flow Algorithm, consists of a dynamic programming algorithm which sequences parts into the system so as to minimize deviations of these discrete inputs from the assumed homogeneous stream (of the high level).

The lower level, consists of dispatching rules in order to determine the sequence of parts on all machines, given the loading sequence from the previous level. Owing to the complexity of this problem, simple decision rules that operate in real-time are employed.

The level (1) and (2) algorithms operate "off-line". However, level (3), requiring detailed information, is run "on-line" on the actual system, or on a simulator. The method is therefore essentially open-loop. An enhancement to the procedure allows level (1) and (2) algorithms to be periodically re-run in order to modify their decisions, based on the actual system performance. This leads to a feedback control strategy. Simulations indicate the superior performance of the suggested hierarchical procedure over other heuristics.

2.2.2.4 Framework of KIMEMIA and GERSHWIN

Kimemia [51] and Kimemia and Gershwin [52], present a Multi-layer hierarchical control algorithm for the optimal control of a stochastic FMS. The manufacturing system consists of machines capable of processing a variety of parts, and are subject to failure. Changeover times are assumed negligible. The problem consists in meeting production requirements while the machines fail and are repaired at random times. The failure/repair process is assumed to be memoryless, hence the machine state is modelled as a Markov chain. Part production requirements are stated in terms of a steady rate. This, along with the assumption that the mean time between changes in machine state is longer than the operation processing times, enables a continuous model for part flows. Under these assumptions, a three-level controller is devised. Additionally, there exists a highest level of the control scheme for off-line calculation of the control policies or decision tables (to be used at subsequent levels), which completes a long-term feedback loop when the system parameters change. This resembles adaptive control.

At the highest level resides the Flow Control Level that determines the short-term part production rates. The demand, level of downstream buffers, reliability of work stations, and sharing of resource capacity are

taken into account in the process. Statistical estimates of failure/repair, anticipating downtimes, are incorporated in this model.

The flow control problem penalizes deviations between actual and target production rates. The optimal control policy is derived as a hedging point strategy, which consists of loading parts at one of the maximal feasible rates, in order to drive the system surplus state towards "hedging points" and to maintain it once it is reached. These accumulated inventories allow one to hedge against future failure at minimal cost. In the derivation of this control policy, optimal cost-to-go functions have to be derived from a system of coupled partial differential equations, which is only possible for small dimension problems of academic interest. However, "estimate-based" cost-to-go functions are derivable for sub-optimal control policies which also perform well.

The intermediate level is the Routing Control Level that calculates the route splits, i.e. the proportion that entering parts must follow among the alternative paths possible for processing. The objective is to meet the production rates dictated by the high level, while minimizing congestion and delay within the system. The system is modeled as a network of queues, where flow rate on each path can be calculated by a mathematical programming technique.

The lowest level of the control, the Scheduling Controller, is composed of scheduling algorithms that dispatch parts into the system. The objective is to maintain the flow rates specified by the route controller, which is achieved by a simple method.

Maimon and Gershwin [58] modify the flow control problem to allow for the consideration of alternative routings at this level. The consideration of loading and routing together eliminate the need for the intermediate level. This also eliminates the problem associated with the occasional infeasibility encountered at the intermediate level, during the routing decision calculation. The basic methodology was extended to improve upon : (i) simplification in computation of the cost-to-go function by a

quadratic approximation, (ii) reduction in the "chattering behavior" by keeping the surplus state trajectory on the optimal path to the hedging point, and (iii) improvement in part sequencing to achieve conditional future trajectory. Akella, Choong and Gershwin [3] test the modified controller on a simulation of a printed circuit card assembly line at IBM, and report superior performance.

Gershwin (1986) [34] proposed an additional layer to the hierarchy in order to determine the set-up frequencies. The generalization of the entire work to address general events relating to production system, with a time-scale decomposition (or frequency separation) is presented by Gershwin (1988) [35]. A hierarchical structure based on the characteristics of the production system is suggested. The levels of the hierarchy correspond to classes of events that occur with distinct frequencies. At each level, feedback laws select : (i) times for the controllable events whose frequency class is addressed at that level, and (ii) frequency targets for much higher frequency controllable events. Calculations of expected rates of high frequency activities (under resource capacity constraints), conditioned on current state of low frequency activities, are determined by a dynamic programming model with good approximate solutions. Once again, the hedging point strategy is used to translate rates, and the staircase strategy (basically, a loading policy) is used to determine the start of an activity while complying with the specified rate.

Xie [76] extends the work of Kimemia and Gershwin to address non-identical parallel machines. Quadratic approximation of the cost functions are used. A new technique to compute parameters of these quadratic functions is also proposed. Extensions to incorporate machine failures of a wide spectrum band are presented in Xie [77]. The failures are clustered near some discrete points on the failure spectrum, in order to define the hierarchical model. Each level in the hierarchy corresponds to a discrete point on the failure spectrum. At each level, faster varying failures are modeled by their mean behavior, and slower varying ones are treated as static. Thus, a hierarchical controller of a multiple time scale type is proposed.

Further, the concept of system configuration is introduced in [77]. In a particular configuration (or set-up of the entire set of machines), the system can produce a sub-set of part types to be produced. For the management of such systems, a three level hierarchy is proposed : (1) Sequencing of Configurations, (2) Flow Control, and (3) Real-time Sequencing of parts.

The first level determines the optimal sequence of configurations, the time the system should remain configured in each, and the production volumes of the relevant part types. At the second level, for each configuration, flow control is performed to incorporate machine failures, and production volumes are constantly adjusted as a function of the repair state of machines and the inventory level. At the third level, real time sequencing of part types to be dispatched is performed, while respecting the flow rates computed by the higher level. For the first two levels, formal mathematical models are established to determine optimal control policies. Properties of these model and solution techniques are developed. For the third level, simple production (dispatching) rules are employed to reach good feasible solutions.

This class of work primarily addresses the scheduling and control aspects of manufacturing systems. The downstream demand is assumed known and continuous. Hence, the production planning aspects are not addressed. It has a sound theoretical foundation, but seems difficult to be extended for larger and more complex manufacturing systems that can be found in practice.

2.3 Conclusions

The progress in the field has been promising. The work done thus far has the following general deficiencies : (i) aggregation schemes are relevant to a particular class of production systems, (ii) consistency issues remain unresolved for the most part, (iii) randomness is generally not addressed, (iv) the architecture, relating to the number of levels and models at each

level are too rigid, or (v) the models may be developed for very small problems of academic interest. The literature has failed to address the issues relating to a systematic hierarchical design as relevant to the specific production management problem at hand. However, there exists a fair scope of extension to the ideas proposed and development of new ideas. The room for enhancement is plenty, and further work in this direction is not without its gratifications.

Chapter 2

Literature Survey

This chapter is devoted to the basic definitions and concepts involved in the problems of production management. The first section describes the principles of hierarchical decomposition, the major advantages, and classification of hierarchies. A literature survey of the major works in this direction, both from a historical perspective, as well as the current state-of-the-art constitutes the major contents of the second section of this chapter.

2.1 Hierarchical Decomposition: Principles and Types

A hierarchical decomposition can assume different meanings according to its context. However, the essential idea of all hierarchical decomposition schemes is the partition of a global problem into sub-problems. These sub-problems are either solved sequentially, such that the solution of a sub-problem imposes constraints on the subsequent sub-problem, or solved simultaneously in a coordinated fashion (see section 2.1.2).

2.1.1 Advantages of Hierarchical Decomposition

The fundamental advantages of the hierarchical approach are [19] :

(1) Reduction of complexity : Breaking a problem into sub-problems is a standard method of simplifying the solution process. The partitioning should be done in a way that the interaction between sub-problems are acceptably weak, or the global problem is broken down into a series of sub-problems which are sequentially solved, in the sense that the solution of a problem at one level imposes constraints on the problem at the next lower

level one; the global solution is obtained when all problems are solved. Thus, the initially intractable problem can be rendered pliable.

(2) Coping with uncertainty : Another important benefit of hierarchical decision making is obtained when the system is subject to random events and uncertainty. Decision making in uncertainty could lead to frequent recomputations; monolithic models would require the entire problem to be resolved repeatedly, while the hierarchical approach can gradually absorb random events without the need to resolve any higher level problems. This results in large savings in computational burden, apart from added stability to the overall control. Decisions at various levels in the planning process are made at different points in time. Higher level decisions are more aggregate, and need not explicitly consider uncertain data at detailed levels. Random events with a lesser impact on the system can be absorbed at lower levels.

(3) Parallel with hierarchical organization of the physical systems : Hierarchical planning or decision making is often performed parallel to the organizational structure of the physical system at hand. This has important implications from the organizational aspects and personnel hierarchies.

(4) Reduced need for detailed information and better forecasting : Higher levels in hierarchies are more aggregate and do not require detailed information; this not only reduces dimensionality, but also allows a longer "look ahead" capability. Forecasting is usually easier and more accurate for aggregates than for detailed entities.

2.1.2 Basic Types of Hierarchies

Mesarovic *et al* [61], identify three types of hierarchical systems, which in a sense represent a classification of existing hierarchical systems.

2.1.2.1 Descriptive Hierarchies

For the complete and detailed description of complex systems, while retaining simplicity in description and behavioral aspects, descriptive

hierarchies are employed. A system is described by a family of models, each concerned with the behavior of it, as viewed from different levels of abstraction. Models must be independent of the functioning for an effective description of a *"stratified system."* The levels of abstraction are referred to as *strata*. Each stratum in the hierarchy is associated with a different set of relevant variables, which allow the study to be confined to one stratum only. Understanding the system increases by crossing the strata: in moving down the hierarchy, one obtains a more detailed explanation, while in moving up the hierarchy, one obtains a deeper understanding of its significance. Lower strata are concerned with the operation of the subsystems, leaving the study of inter-relationships for the higher strata. Selection of strata, in terms of which system is described, depends upon the observer, his knowledge and interest in the operation of the system, although for many systems there are some strata which appear as natural or inherent [61].

Such hierarchies are primarily employed for system description, and detailing function behavior. They usually do not support decision making aspects or control, hence their applications are limited to system modeling and description. Examples of such hierarchies are illustrated in [61].

2.1.2.2 Multilayer Hierarchies

These hierarchies, also known as decisional hierarchies, appear in the context of a complex decision-making process. Complex decision-making is associated with a fundamental dilemma: on one hand, there is a need to take timely and prompt action, while on the other, there is an equally great need to understand the situation better and retain perspective of the consequences and relationships of actions in a complex situation. This problem may be resolved by multilayer hierarchies, where one defines a family of decision problems whose solution is attempted in a sequential manner, in the sense that the solution of any problem in the sequence determines and fixes some parameters in the subsequent problem. This has to be done in a way that the latter is completely specified and its solution can be attempted. The solution of the original problem is achieved when all sub-problems are solved. This arrangement is

represented diagrammatically in figure 2.1. Each block represents a decision-making unit, which receives information from the system. The output of a unit, say D_1, represents a solution, or the consequence of a solution, of a decision problem which depends upon a parameter fixed by the value of the input X_1, which in turn is the output of a unit on a higher level. Such a hierarchy of decision layers is termed as a multilayer (decisional) hierarchy.

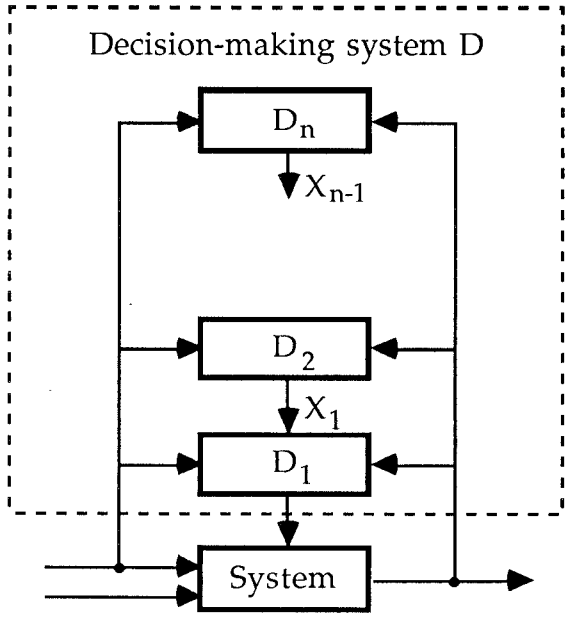


Figure 2.1 : Multilayer hierarchy of a decision-making system [61]

2.1.2.3 Multilevel Hierarchies

These hierarchies are also known as organizational hierarchies. It is necessary that : (i) the system consists of a family of interacting sub-problems which are recognized explicitly, (ii) some of the subsystems be defined as decision (making) units, and (iii) the decision units are arranged hierarchically, in the sense that some of them are influenced or controlled by other decision units.

Designing such a system consists in the assignment of the tasks or roles which various levels or individual units have to perform. Decomposition of the overall systems task is performed, by assigning a sub-problem to

each decision-making unit (infimal unit). Infimal units perform their decision-making independently, though *coordinated* through a supremal unit. Coordination is concerned with the existence of supremal control under which infimal units can solve their local control problems. *Consistency* is an important consideration: i.e., whenever the supremal and infimal units can solve their problems, then an overall solution exists. The supremal-infimal (or master-slave) relationship exists between each consecutive echelon of the hierarchy.

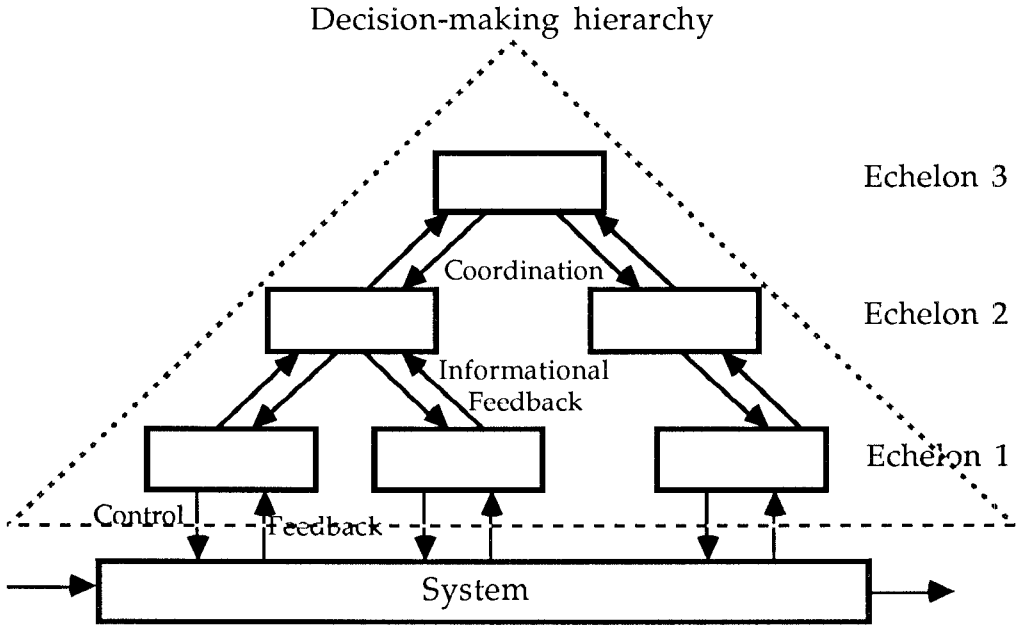


Figure 2.2 : Multilevel organizational hierarchy; multiechelon system [61]

2.2 Hierarchies in Production Management

This section is devoted to the hierarchies proposed in the production management domain. Most of these are multi-layer systems, each with its unique set of assumptions, layering method, number of layers and decision making methodology.

Literature surveys in the field of hierarchical production management can be found in Gelders and VanWassenhove [33], Dempster, Fisher *et al* [19] and Libosvar [57]. Although it is difficult to classify the work surveyed

here in distinct sections, the first sub-section is devoted to the hierarchical production planning related literature. The second sub-section is devoted to hierarchies in FMS, AI approaches, and other conceptual architectures.

2.2.1 Hierarchical Production Planning

The work of Hax and Meal [40], developed at M.I.T. in the seventies has been considered a substantial contribution in the area of hierarchical management. Production planning is the major concern of this work.

2.2.1.1 Framework of HAX and MEAL, and Related Models

Hax and Meal [40], describe a hierarchical model designed for a particular implementation, but also provide a general direction to subsequent work. Four decision levels are considered for a multi-plant firm. The highest level model decomposes the problem into decoupled single plant problems, by determining the products to be produced in each plant. The subsequent hierarchy is intended for the management of a single plant. The highest level involves static (one time) decisions of a decomposing nature. Thus, it differs from the rest of the hierarchy, which is typically multi-layered for dynamic decision making. We restrict our attention to these lower levels. Bitran and Hax (1977) [12] discuss this hierarchical framework for production planning problems.

Based on the analysis of the production process, and a typical cost structure, three levels of aggregation are considered for products:

1. *Items* are final products to be delivered to customers.
2. *Families* are groups of items which share a common manufacturing set-up cost.
3. *Types* are groups of families that have similar costs per unit of production time and similar seasonal patterns.

A production planning level is consistently modeled according to the aggregation scheme. Higher level decisions impose constraints on lower level actions, which in turn provide necessary feedback. In this top-down constrained approach, the assumption is that higher-level decisions have a more significant impact on the objectives.

The highest level of the hierarchical planning system allocates production capacity among product types, by means of a linear programming based aggregate planning model. The horizon is normally longer than a year (15 months), in order to take into account demand fluctuations of products. Production and inventory holding costs are the only costs considered; set-up costs are ignored at this level. This model is intended to determine an optimal trade-off between inventory holding and overtime (production) costs, while leaving set-up costs out because of secondary impact. At this and all subsequent levels, the concept of a rolling horizon is employed, in order to perform repetitive open-loop optimization.

The second step in the planning process is the disaggregation of production quantities for types (computed by the higher level) to determine the production quantities for families. This is performed by a heuristic, based on Economic Order Quantity (EOQ) and safety and overstock computation techniques. Capacity allocated to a product type is split among the families belonging to that type, for which the inventory is below the safety threshold. For these families, production volumes are chosen as close as possible to the EOQ, provided they do not leave an overstock at the end of the period.

Finally, the third decision level consists of an item disaggregation heuristic, based on Equalizing of Run-Out Times (EROT). This decomposition technique maximizes the time between set-ups, by requiring the items of a family to run-out at the same time. Karmarkar [49] provides a proof of optimality of this technique.

Consistency between decisions at different levels is attempted to be ensured by the top-down constraints. However, sometimes the disaggregation is infeasible, i.e. top level constraints may yield an empty feasible set at a lower level. Gabby [28] considers an aggregate plan for which a feasible detailed plan (i.e. one without backlog) exists. However, this requires detailed demands to be known over the consistency horizon, thus, loosing some benefit drawn from the need of less detailed demand

required in hierarchies. He derives necessary and sufficient conditions for a consistent disaggregation; see Gabby [28] for treatment of multi-item, single echelon, capacitated production problems.

Bitran and Hax (1977) [12] propose a computational improvement to Hax and Meal's model by reformulating family and item disaggregations as knapsack problems, for which they provided efficient solution algorithms [13]. Numerical results indicated that for low set-up costs, the solutions are near optimal.

Bitran, Haas and Hax (1981) [10] prove that the EROT method is an optimal disaggregation scheme that minimizes the cost of initial inventory. This work improves the knapsack-based method in that: (i) at the family disaggregation level, the expected number of set-ups are minimized for a shorter horizon, allowing for greater responsiveness to seasonal variations, (ii) a "one step look ahead" procedure is required for disaggregation, and (iii) the families' production volumes are modulated in order to keep them close to their EOQs. The enhanced work was shown to produce superior results through a set of simulations.

Erschler, Fontan and Merce [23] derive necessary and sufficient conditions for consistency, based on mass balance equations. They propose the look ahead procedure to be extended to all periods, for which detailed demand is known. The necessary conditions are then employed to enhance the knapsack method, which was shown to perform better.

Graves [36], adopts a different approach to the problem and introduces feedback between decision layers. Based on the product aggregation scheme of Hax and Meal, the problem is formulated as a monolithic mixed integer program, which is decomposed by lagrangean relaxation. This decomposition yields two subproblems : (i) aggregate planning (linear programming model), and (ii) disaggregation problem (lot sizing problems for each product type). The linking mechanism for these two subproblems is an inventory consistency relationship which is priced out by a set of Lagrange multipliers. The best values of the multipliers are found by an

iterative procedure which may be interpreted as a feedback mechanism in the Hax-Meal framework.

Other extensions of the Hax-Meal model are directed towards incorporating it to multi-stage fabrication and assembly systems. Bitran, Haas and Hax (1982) [11] propose an extension of their previous work to a two-stage fabrication/assembly system. On an industrial test-bed, they successfully compare results obtained by the extended hierarchical planning system to those of an MRP based system. However, under this model product families are subject to a rather restrictive definition.

Advantages and shortcomings of the Hax-Meal framework are summarized as follows:

Advantages :

- reduction in computational size, compared to a monolithic optimization with complete detailed data.
- enabling of managerial interaction.
- reduction of data requirements (due to reduction of part entities).
- explicit consideration of capacity.
- models at each level are consistent with the aggregation scheme, ensuring coherent decisions at each disaggregation level.
- secondary impact of set-up costs is gaining more relevance with the current trends in manufacturing, e.g. Flexible Manufacturing Systems permit quick changeovers between parts.

Shortcomings [57] :

- The product aggregation scheme proposed is only relevant to a particular class of production systems, and models at each level are based on a typical cost structure.

- Detailed data requirement are reduced only if backordering is permitted.
- No randomness is taken into account and forecast errors have to be absorbed by safety stock.
- No spatial decomposition of the system is proposed.

2.2.1.2 MEIER's Hierarchical Control of a Production System

Meier [60] considers planning and control of a general multi-stage assembly/disassembly manufacturing system, to minimize a performance criterion composed of inventory holding costs, backlogging costs, and production costs. A two-layer "master" and "slave" hierarchy is developed.

The master-level determines the amount of operations (per operation type) to be performed on machines, over given planning periods, via a linear programming model. For the treatment of large-scale problems, aggregation-disaggregation procedures are proposed.

The slave controls the activities of the machine in real time, in order to satisfy the plan provided by the master. At this level, a set of priority rules are employed to determine the activities of the machines, as a function of the volumes of operations, and the inventory state.

Discrepancies are possible, if the slave cannot execute the master's decision precisely. This is mitigated by rolling horizons, and repeated optimization. In this way, a close loop planning and control system is emulated, whereby the schedule on the rolling horizon yields to auto-adaptive planning corrections.

Hillion, Meier and Proth [44] present a top level model for a new approach to hierarchical production planning. At this level, the entities of relevance are production sub-systems and part families. Aggregation procedures to determine these entities are detailed, and the problem is formulated as a linear program.

2.2.1.3 Optimality and Aggregation-Disaggregation Issues in Hierarchies

The suboptimality of the hierarchical approach has been addressed by Dempster, Fisher *et al* [19], where the authors review the concepts of hierarchical planning and site some examples. They demonstrate that multi-level decision problems can be modeled as multi-stage stochastic programs. Then, the analytical evaluation of any hierarchical system can be assessed with respect to the optimality of this stochastic program. An analytical evaluation, and a proof of asymptotic optimality of a simplified version of the hierarchical planning system of Armstrong and Hax [6] are presented. The job-shop consists of a set of parallel identical machines; the higher level decision is to determine the optimal number of machines, m. The lower level is concerned with the problem of scheduling n jobs on these m machines, in order to minimize the completion time. There are costs associated with purchasing machines, and costs proportional to the completion time. The performance of this hierarchical system is compared with a stochastic model, in which both costs are treated in a single model, and processing times are random with a known mean value. When the number of jobs tends to infinity, the performances of the two systems become identical. The application of such analytical results is dependent on the criteria and the model chosen. Also, owing to lack of accurate estimation of the quality of solutions provided by hierarchical systems, this evaluation method has found little application to other systems.

Aggregation-Disaggregation is another major issue concerned with the design of hierarchical systems. Infeasibility can be encountered during the disaggregation process. Krajewski and Ritzman [54] provide a survey of the problems and research in this field from a manufacturing, as well as a service organization perspective. These disaggregation problems are encountered between aggregate planning at the top level and more detailed decisions of inventory control and scheduling at the bottom level. They draw attention to the lack of an interface mechanism, which diminishes the utility of the solution procedures of aggregate planning, inventory control and scheduling.

Aggregation schemes are of three types : (i) over time, or temporal, (ii) over products, and (iii) over machines, or spatial. The Hax and Meal framework only considers product and temporal aggregation. Except Meier [60], who includes aggregation of machines, there is no work addressing aggregation in all three dimensions.

Zoller [79], considers product disaggregation with a two-level economic model, in which aggregate production is determined by minimizing a cost function. The product mix and sales price are determined at the lower level in order to maximize profit, assuming that the demand volume depends on sales price and that the function relating the two variables is known. An iterative algorithm to reach optimal solutions of the low-level problem is presented. Gelders and Kleindorfer [31] [32], present a formal model of a one-machine job-shop scheduling problem with variable capacity, considering trade-offs between overtime and detailed scheduling costs. The scheduling problem considers minimizing the sum of weighted tardiness and weighted flow-time costs for a given capacity plan. Various lower-bounding structures for the problem are analyzed, and a branch-and-bound scheme is outlined. Computational experience with the algorithm in presented in [32].

Erschler, Fontan and Merce [23] consider the consistency of the disaggregation process in hierarchical planning, and necessary and sufficient conditions for consistency of decisions in a two-level structure are presented. A sub-set of these conditions improves upon the disaggregation procedure of Bitran and Hax [12], [13].

Axsater [7] addresses a double aggregation over products and machines, which results in an aggregate planning problem in terms of product groups and machine sub-systems at the aggregate level. This aggregate plan may not be possible to disaggregate. He presents perfect aggregation conditions, i.e. necessary and sufficient conditions on the aggregation matrices, where it will be possible to disaggregate any aggregate plan. If perfect aggregation is not possible, a good approximate solution can be

reached through a mathematical formulation of an approximation problem.

2.2.2 Hierarchical Systems for Flexible Manufacturing

Flexible Manufacturing Systems (FMS) offer challenging planning and scheduling problems due to their versatility, flexibility, quick changeover time, and association of automated material handling systems along with the need for efficient resource utilization (brought about by high capital investment). Morton *et al* [63] and Stecke [74] identify some issues that render FMS scheduling different from traditional manufacturing systems. Thus, the need for different principles in FMS design, planning, scheduling and control. Stecke [74], identifies FMS related problems, and suggests mathematical models useful to their analysis.

2.2.2.1 Conceptual Models in Production Management

O'Grady [65] provides an introduction to automated manufacturing systems with their characteristic features. Production Planning and Control structures of traditional and automated manufacturing systems are identified. Three major hierarchical frameworks: AMRF's (Automated Manufacturing Research Facility of the NIST), CAM-i (Computer Aided Manufacturing International Inc.), and one developed by their team have been described. None of these architectures have been implemented in their entirety.

Doumeingts (1985) [20] discusses the necessity of conceptual models in production management. These models help in figuring the whole system through abstract terms, and in designing it with the entities and links between them. The precise objective of such a model is to deal with a production management system, or a physical system, or even the driving of a discrete-continuous system. The GRAI (Graph with Results and Activities Interrelated) method, and especially the GRAI grid, presented in [20] corresponds to the last type of conceptual models. Decentralization methods for decision making through other conceptual models include : (i) ICAM model, (ii) General model of CAM-i, and (iii) Specific models for CAM-i.

The National Institute of Standards and Technology (NIST) within its Automated Manufacturing Research Facility (AMRF), has developed a five level hierarchical structure: Facility, Shop, Cell, Workstation, and Equipment levels of control. This decomposition is functional in nature [45], and is fixed (i.e. the structure is rigid and not system dependent). Their efforts [18], [46], focus on the design of a real-time production scheduler at the shop and cell levels, but not yet to the design of an integrated planning and scheduling hierarchy for the management of the entire factory.

Project 418 of the European Strategic Program for Research and development in Information Technology (ESPRIT) has resulted in a hierarchical structure, with multi-level dynamic planning and scheduling, and a GRAI modeling method for decision making [14]. This structure is conceptual in nature and helps in modeling, and identification of information and decision flow. However, it does not provide solutions to the decision or control problems. Roboam *et al* [67] present an integrated methodology for the analysis and design of manufacturing system.

CIM-OSA (Computer Integrated Manufacturing - Open System Architecture) is a current development carried out by the CEC ESPRIT Consortium AMICE [4], [47], [48], [50] consisting of 21 European organizations. The goal of CIM-OSA is to provide an OSA which covers all the internal and external information needs of all the functions in a manufacturing enterprise; it bears some ideas of hierarchical control and management. First, it presents a functional view in which the enterprise functionality and behavior are modeled, using a functional decomposition principle. Functions of the enterprise are modeled in terms of business process and enterprise activities. Enterprise activities are the leaves of the functional decomposition trees and describe the enterprise functionality. Business processes are nodes of the functional decomposition trees and describe enterprise behavior, i.e. the way enterprise activities and lower-level business processes are procedurally chained to form longer processes. Second, ideas of hierarchical organization can be found in the CIM-OSA

resource view which describes resource organization for the purpose of resource management. Resources (i.e., components, machines, people, applications) are required by enterprise activities for their execution. However, it may happen that resources can be aggregated to form high-level resources (which are called in CIM-OSA resource sets for permanent aggregation and resource cells for temporary aggregation). This recursive principle can be used as much as desired to form larger resource units. Aggregated resources can be linked to enterprise activities and/or to business processes. Finally, CIM-OSA employs hierarchical principles in its organization view, which is concerned with the modeling of the responsibilities and authorities in the enterprise, as well as the modeling of decision levels and decision centers, using the organization unit construct. An organization unit can be a person, a group of persons, a service, a department, a plant, etc. Organization units have control over the functions (i.e., business processes and enterprise activities) of the function view and indicate at which organization level they belong, where an organizational level is defined by its authority, responsibilities, horizon and period.

However, CIM-OSA is essentially an enterprise modeling framework aiming at producing an executable model of the enterprise. It provides no means to decide on the number of organizational levels, which is left to the model designer. We believe that the work presented in this thesis could provide CIM-OSA with a sound basis for the design of its function and organization views. CIM systems to be developed according to CIM-OSA intend to support all levels of management in their strategic, tactical and operational planning, as well as the direct operation of product design, and manufacturing. It is proposed that such systems will support all decision making processes, by both, providing the required information, and by allowing the simulation of alternatives and optimization of solutions before implementation. The emphasis thus far has been enterprise structuring and modeling; a 3-d architecture has been developed [53]. However, the details of decision making as well as optimization processes are not detailed at this time.

2.2.2.2 Artificial Intelligence Based Approaches

Villa *et al* [75] propose a hierarchical framework to model FMS control. They define an FMS as a structure composed of a physical system, an information system and a decision-and-control system. The task performed by the latter can be divided into periodic planning and event-driven control. The authors suggest a decomposition, based on physical insights rather than mathematical techniques, to develop a tree-like structure with decision makers at each node. They present a hierarchical control structure by integrating mathematical tools from Control Theory with relational tools derived from Artificial Intelligence. This matching is suggested to provide an effective implementation of Expert Control System. This framework leading to an "Integrated Control Structure" is based on the same spatial decomposition of the physical system and frequency-band partition of events, whereas the decision-making units use AI tools to solve these problems.

The control units are modeled as a knowledge base and an inference engine; each "layer" of control units is related to a frequency band. Hence the horizon ratios of different control layers must be consistent with the event frequency ratios. The importance of an event is defined as the index of the higher control layer, at which its effects are likely to influence the control. Thus, the optimal control strategy for each decision module consists of solving an open-loop-feedback problem to update its policy, each time an event occurs with an importance greater than its level index. This updating process includes ensuring consistency between the policies determined by the lower level decision modules. The excessive computational capacity required in solving a planning problem is reduced by retrieving information by pattern-matching from the the module's knowledge base. The inference engine: (i) selects the best policy according to the system state and event type, (ii) coordinates the lower level actions, based on rules retrieved from the knowledge base of the control policy, and (iii) feeds the knowledge base with a description of the consequences of the control in terms of the resulting dynamics of the system.

Shaw and Whinston [70] address the application of generic artificial intelligence techniques to the planning (process planning) and scheduling of flexible manufacturing systems. The complex scheduling problems in FMS are made tractable by nonlinear planning. Planning for conjunctive goals is referred to as nonlinear planning because the resulting plans are partially ordered. Nonlinear planning systems seek to break a problem into sub-problems, using the divide-and-conquer strategy, while taking interactions of sub-plans into account. They employ a two-level hierarchy to decompose an n-part-m-machine scheduling problem into n sub-problems, with each sub-problem defined as the routing of one part. The primary interactions between the sub-problems are their sharing of m machines. The objectives are to minimize makespan and avoid conflicts. Finally, the system considers alternative operations and revises existing plans if bottlenecks are detected. To summarize, a planning scheme is composed by the following four step procedure :

1. Generate a linearly-sequenced plan for each task.
2. Identify problematic interactions between the planning steps.
3. Use precedence constraints to avoid conflicts.
4. Identify alternative planning steps to improve performance.

In step 1, the inference engine calls upon a backward chaining procedure to search for the best planning steps, based on "means-ends analysis." In each intermediate planning step, the inference engine searches the rule-base to select the best operation to apply; node expansion is carried out with heuristic and constraint information. Steps 2 and 3, for conflict detection and resolution, dynamically decide the precedence relationships between conflicting actions. Parallelism among subplans is maximized by the "least commitment strategy". Step 4, called Plan-Revision, reassigns waiting jobs to alternative resources, so as to achieve better utilization and performance.

Doumeingts (1986) [21] identifies the significance and potential application of AI techniques in the field of CIM. The GARI system for process

planning (University of Grenoble) and the ISIS (Intelligent Scheduling and Information System) system in scheduling (Carnegie Mellon University - Robotics Institute) are detailed below, and referenced as major contributions in this field.

PATRIARCH is a hierarchical planning and scheduling system developed at Carnegie-Mellon University. In [63], Morton and Smunt highlight issues relating to FMS scheduling, which should integrate: (1) hierarchical structure, (2) decision support capability, (3) advanced knowledge representation, and (4) accurate practical large-scale heuristics.

The four levels of the PATRIARCH system include: Level (1) strategic planning, Level (2) capacity planning, Level (3) scheduling, and Level (4) dispatching. The first level considers ill-structured, long range problems found in strategic planning. Due dates, penalties, resource constrains, activity design, and other costs are all quite fuzzy. Hence, human expertise and manual decision support are essential. In the second level, semi-ill-structured medium range problems are considered. Capital is fixed in this time-frame, but employment levels, number of shifts, and active machines can be chosen. The planner can test and select among several scenarios; hence, it is termed as a semi-automatic mode. Level (3) represents the largest level of detail in terms of activity and resource structure, at which it is feasible to solve a well-structured problem by operations research heuristics. Level (4) is fully detailed, and too large to be solved by operations research heuristics. Instead, fast dispatch routines guided by priorities and assignments passed from level (3), and a large number of simple historical rules-of-thumb, make dispatch decisions. Some manual intervention may be required.

Further, the Intelligent Systems Laboratory of the Carnegie-Mellon Robotics Institute has developed large scale scheduling systems with sophisticated knowledge representations (Fox [24] [25]). These systems, namely ISIS - a job shop scheduling system, and CALLISTO - a project scheduling system, attempt to integrate all levels of hierarchical production planning. The original work of ISIS utilized a reservation

system and backward scheduling with beam search, which were only moderately successful for loading of machines. CALLISTO uses a forward dispatch approach, improved in Morton, *et al* [63]. The FMS version incorporates many of the concepts of CALLISTO and ISIS, along with the consideration of multiple machines and dispatching priorities from detailed planned schedules.

The ISIS was followed by the development of a successor system called OPIS (Opportunity Intelligent Scheduler [73]). OPIS employs constraint propagation techniques to update schedule descriptions and detect inconsistencies introduced. These systems were tested using simulated data from actual manufacturing environments.

2.2.2.3 An Approach by Hildebrant and Suri

Hildebrant [41], Hildebrant and Suri [42], present a methodology and a multi-level algorithm structure for scheduling in real-time control of Flexible Manufacturing Systems (FMS) with unreliable machines connected by automatic transportation means. The control consists of satisfying a given demand of part types in order to minimize the makespan (production cycle time). The hierarchical decomposition of the problem leads to a three level hierarchy.

At the high level, a multi-class queueing model is used to compute the completion time of tasks as a function of the utilization of machines and fixtures, which accounts for the possibility of failures. For each failure state of the system, a formulation (nonlinear programming with mean value analysis) helps seeking optimal allocation of tasks to resources. The solution results in the allocation of resources to parts and routes.

The intermediate level, Even-Flow Algorithm, consists of a dynamic programming algorithm which sequences parts into the system so as to minimize deviations of these discrete inputs from the assumed homogeneous stream (of the high level).

The lower level, consists of dispatching rules in order to determine the sequence of parts on all machines, given the loading sequence from the previous level. Owing to the complexity of this problem, simple decision rules that operate in real-time are employed.

The level (1) and (2) algorithms operate "off-line". However, level (3), requiring detailed information, is run "on-line" on the actual system, or on a simulator. The method is therefore essentially open-loop. An enhancement to the procedure allows level (1) and (2) algorithms to be periodically re-run in order to modify their decisions, based on the actual system performance. This leads to a feedback control strategy. Simulations indicate the superior performance of the suggested hierarchical procedure over other heuristics.

2.2.2.4 Framework of KIMEMIA and GERSHWIN

Kimemia [51] and Kimemia and Gershwin [52], present a Multi-layer hierarchical control algorithm for the optimal control of a stochastic FMS. The manufacturing system consists of machines capable of processing a variety of parts, and are subject to failure. Changeover times are assumed negligible. The problem consists in meeting production requirements while the machines fail and are repaired at random times. The failure/repair process is assumed to be memoryless, hence the machine state is modelled as a Markov chain. Part production requirements are stated in terms of a steady rate. This, along with the assumption that the mean time between changes in machine state is longer than the operation processing times, enables a continuous model for part flows. Under these assumptions, a three-level controller is devised. Additionally, there exists a highest level of the control scheme for off-line calculation of the control policies or decision tables (to be used at subsequent levels), which completes a long-term feedback loop when the system parameters change. This resembles adaptive control.

At the highest level resides the Flow Control Level that determines the short-term part production rates. The demand, level of downstream buffers, reliability of work stations, and sharing of resource capacity are

taken into account in the process. Statistical estimates of failure/repair, anticipating downtimes, are incorporated in this model.

The flow control problem penalizes deviations between actual and target production rates. The optimal control policy is derived as a hedging point strategy, which consists of loading parts at one of the maximal feasible rates, in order to drive the system surplus state towards "hedging points" and to maintain it once it is reached. These accumulated inventories allow one to hedge against future failure at minimal cost. In the derivation of this control policy, optimal cost-to-go functions have to be derived from a system of coupled partial differential equations, which is only possible for small dimension problems of academic interest. However, "estimate-based" cost-to-go functions are derivable for sub-optimal control policies which also perform well.

The intermediate level is the Routing Control Level that calculates the route splits, i.e. the proportion that entering parts must follow among the alternative paths possible for processing. The objective is to meet the production rates dictated by the high level, while minimizing congestion and delay within the system. The system is modeled as a network of queues, where flow rate on each path can be calculated by a mathematical programming technique.

The lowest level of the control, the Scheduling Controller, is composed of scheduling algorithms that dispatch parts into the system. The objective is to maintain the flow rates specified by the route controller, which is achieved by a simple method.

Maimon and Gershwin [58] modify the flow control problem to allow for the consideration of alternative routings at this level. The consideration of loading and routing together eliminate the need for the intermediate level. This also eliminates the problem associated with the occasional infeasibility encountered at the intermediate level, during the routing decision calculation. The basic methodology was extended to improve upon : (i) simplification in computation of the cost-to-go function by a

quadratic approximation, (ii) reduction in the "chattering behavior" by keeping the surplus state trajectory on the optimal path to the hedging point, and (iii) improvement in part sequencing to achieve conditional future trajectory. Akella, Choong and Gershwin [3] test the modified controller on a simulation of a printed circuit card assembly line at IBM, and report superior performance.

Gershwin (1986) [34] proposed an additional layer to the hierarchy in order to determine the set-up frequencies. The generalization of the entire work to address general events relating to production system, with a time-scale decomposition (or frequency separation) is presented by Gershwin (1988) [35]. A hierarchical structure based on the characteristics of the production system is suggested. The levels of the hierarchy correspond to classes of events that occur with distinct frequencies. At each level, feedback laws select : (i) times for the controllable events whose frequency class is addressed at that level, and (ii) frequency targets for much higher frequency controllable events. Calculations of expected rates of high frequency activities (under resource capacity constraints), conditioned on current state of low frequency activities, are determined by a dynamic programming model with good approximate solutions. Once again, the hedging point strategy is used to translate rates, and the staircase strategy (basically, a loading policy) is used to determine the start of an activity while complying with the specified rate.

Xie [76] extends the work of Kimemia and Gershwin to address non-identical parallel machines. Quadratic approximation of the cost functions are used. A new technique to compute parameters of these quadratic functions is also proposed. Extensions to incorporate machine failures of a wide spectrum band are presented in Xie [77]. The failures are clustered near some discrete points on the failure spectrum, in order to define the hierarchical model. Each level in the hierarchy corresponds to a discrete point on the failure spectrum. At each level, faster varying failures are modeled by their mean behavior, and slower varying ones are treated as static. Thus, a hierarchical controller of a multiple time scale type is proposed.

Further, the concept of system configuration is introduced in [77]. In a particular configuration (or set-up of the entire set of machines), the system can produce a sub-set of part types to be produced. For the management of such systems, a three level hierarchy is proposed : (1) Sequencing of Configurations, (2) Flow Control, and (3) Real-time Sequencing of parts.

The first level determines the optimal sequence of configurations, the time the system should remain configured in each, and the production volumes of the relevant part types. At the second level, for each configuration, flow control is performed to incorporate machine failures, and production volumes are constantly adjusted as a function of the repair state of machines and the inventory level. At the third level, real time sequencing of part types to be dispatched is performed, while respecting the flow rates computed by the higher level. For the first two levels, formal mathematical models are established to determine optimal control policies. Properties of these model and solution techniques are developed. For the third level, simple production (dispatching) rules are employed to reach good feasible solutions.

This class of work primarily addresses the scheduling and control aspects of manufacturing systems. The downstream demand is assumed known and continuous. Hence, the production planning aspects are not addressed. It has a sound theoretical foundation, but seems difficult to be extended for larger and more complex manufacturing systems that can be found in practice.

2.3 Conclusions

The progress in the field has been promising. The work done thus far has the following general deficiencies : (i) aggregation schemes are relevant to a particular class of production systems, (ii) consistency issues remain unresolved for the most part, (iii) randomness is generally not addressed, (iv) the architecture, relating to the number of levels and models at each

level are too rigid, or (v) the models may be developed for very small problems of academic interest. The literature has failed to address the issues relating to a systematic hierarchical design as relevant to the specific production management problem at hand. However, there exists a fair scope of extension to the ideas proposed and development of new ideas. The room for enhancement is plenty, and further work in this direction is not without its gratifications.

Chapter 3

A Hierarchical Scheduling Policy and Performance Evaluation

Production scheduling problems have been recognized for their complexity. This chapter considers the problem of scheduling a given set of jobs on a single machine in order to minimize the total earliness and tardiness costs. Although this scheduling problem is rather simple for most practical applications, we decided to address it because it becomes increasingly difficult to develop exact algorithms to solve more complex problems. With the help of this problem, we demonstrate numerically the efficacy of a hierarchical scheduling policy. Fundamental principles of hierarchical design are employed in developing the policy. Favorable numerical results relating to the performance of this hierarchical policy against an exact algorithm are presented. This suggests that hierarchical approach is a "good" approach to an otherwise "hard" problem. Problems of more practical significance can be treated by the theory and methodology presented in chapters 4 and 5.

A brief description of the scheduling problem is as follows. The horizon is divided into elementary periods; jobs have due-dates at the end of these periods. All jobs are assumed initially available. Jobs have unique (weighted) early and tardy penalty functions that are discrete. No preemption of jobs is permitted, and idle time may be inserted. This problem is NP-complete. A branch-and-bound scheme that solves the above mentioned problem optimally is presented. We then propose the hierarchical policy under the assumption that tardiness costs are much

greater than earliness costs for all jobs. We also assume that the system is able to meet the production requirements on the average.

3.1 Introduction

Many scheduling problems with varying criteria, assumptions relating to release and due dates, and number of production stages have been treated in the literature. Among these, the single stage (machine) scheduling has been extensively studied. French (1982) [26], and Conway *et al* (1967) [17] present some criteria of interest, while Gupta and Kyparisis (1987) [37] review the literature related to this problem. Typical criteria include : mean tardiness, mean lateness, total weighted tardiness, maximum tardiness, number of tardy jobs, weighted number of tardy jobs, etc. Previously, the research focus was on single performance measures, referred to as *regular* measures. Regular measures are nondecreasing in job completion time, i.e., if we have two schedules where completion times of jobs for the first schedule are no later than the ones of the second schedule, then under the regular measure, the first schedule is at least as good as the second one. More recently, with the advent of the Just-In-Time (JIT) concept, research attention has been drawn towards combined penalization of earliness as well as tardiness. This gives rise to a nonregular performance measure.

Furthermore, the majority of the scheduling literature considers that due-dates and delivery times of jobs can assume continuous values over a time horizon. Consequently, the earliness and tardiness also assume continuous values. Models with such characteristics, although applicable to a good variety of practical scheduling problems, fail to capture an important class of problems encountered in business practice. Consider for instance a business with periodic shipment, e.g. the delivery truck leaves once a day. Failure of a job to be completed by the delivery time entails it to wait for the next shipment, regardless of the actual tardiness (expressed in a continuous manner). On the other hand, the inventory book-keeping function can also be periodic, e.g. interest on capital is accrued at the end of a business day. Similarly, a part taken into stock is penalized (by a fixed

amount) at the end of the day, regarless of the actual completion time during the day. Similar examples can be found in a make-to-forecast environment, where due-dates are assigned at the end of a reasonable "time-step" rather than in a continuous fashion.

Thus, in this direction, we consider the problem of scheduling a set of initially available jobs, to be produced within a given scheduling horizon (e.g. week), on a single machine, to minimize the total earliness and tardiness penalties, subject to no preemption constraint. The scheduling horizon is discretized into elementary periods; *jobs are due, and can be delivered only, at the end of these periods* (e.g. day). The penalty functions are discrete and weighted (staircase). This problem differs from previous work in this area in that the earliness and tardiness penalty functions are discontinuous. Completion of a job in a period prior to that of its due date entails it to be held in inventory until shipment, while failure to complete a job by its due date causes backlogging, with shipment possible only at the end of the period in which it is completed. As the basic idea of the earliness penalty stems from discouraging the holding of finished parts in inventory, we believe in penalizing the job by the number of periods (an integer) it spends in inventory, weighted appropriately by its holding cost per period. Analogously, given that the tardiness penalty discourages violation of due dates, we believe in penalizing the job by the number of periods (an integer) by which it is delayed, weighted appropriately by its backlogging cost per period. We do not penalize a job completing within the period at the end of which it is due to be shipped, as it can be held on the shop floor for less than one period, until transported to the shipping area, without incurring any inventory carrying costs. Hence our decision to employ staircase penalty functions for earliness and tardiness of each job, as opposed to linear or quadratic functions considered in the literature.

It is important to indicate that this does not emerge as a minor variation of the continuous-time version of single machine Earliness and Tardiness (E/T) scheduling problem. The solution to the continuous-time problem may provide very poor results for our version of the problem. This is

because the continuous-time problem penalizes a due-date violation by the weight of the part multiplied by the duration of violation E/T; if the E/T tends to zero, the penalty tends to zero. However, in our version, penalization presents jumps, and even a small tardiness violation may cause a significant cost. Thus, one can appreciate that the characteristics of optimal (or good) solutions to the two versions of the problem may differ significantly.

This variation does not seem to have drawn much attention in the literature, while the continuous-time version has been extensively studied. The majority of the previous work in single-stage scheduling dealt with single performance measures (see Baker 1974 [8]), especially tardiness. Gupta and Kyparisis (1987) [37], and Sen and Gupta (1984) [69] review literature related to variations of this problem. Sidney (1977) [71], and Lakshimarayan *et al* (1978) [55] present algorithms for minimizing the maximum penalty of early or late jobs. A comprehensive survey relating to E/T models can be found in Baker and Scudder (1990) [9]. They site several publications with different assumptions like: common due-dates, unweighted or equal E/T weights, no insertion of idle time, etc. We restrict the references to the continuous-time, distinct due-date, weighted total E/T model, as more relevant to our proposed approach. Garey *et al* (1988) [30] prove that this problem is NP-complete. Abdul-Razaq and Potts (1988) [1] consider this problem without inserted idle time. They employ a branch-and-bound scheme, using dynamic programming state-space relaxation techniques for obtaining lower bounds. Their computation results suggest that problems with more than 20 jobs may lead to excessive solution times. Ow and Morton (1989) [66] also consider the problem without inserted idle time. They suggest dispatch heuristics, and propose a Filtered Beam Search method. Fry *et al* (1987) [27], and Garey *et al* (1988) [30] develop efficient procedures of optimally inserting idle time for a given job sequence. Fry *et al* (1987) [27] decompose the problem by determining a good sequence, followed by the optimal insertion of idle time. We are not aware of any literature related to periodic distinct due-date weighted total E/T problem with staircase penalty functions.

Owing to the nonregular nature of our objective function (i.e., that can decrease with increasing completion times), important theorems (Emmon 1969 [22], Lawler 1977 [56], Smith 1956 [72], etc.) on job ordering cannot be used as pruning devices for Branch and Bound or Dynamic Programming algorithms. Furthermore, the insertion of idle time makes the problem even more complex. The discrete penalty functions, however, render the ordering of jobs within a period unimportant. Thus, the problem reduces to the determination of the period in which each job is completed.

We develop a Branch and Bound scheme to obtain optimal results for the above stated problem. We then demonstrate the design and operation of a hierarchical production scheduling policy and its main advantages over traditional methods in solving similar problems.

In section 3.2, the problem formulation is presented, detailing first the manufacturing system, the demand, and the criteria of interest. The informal proof of NP-completeness is presented in section 3.3. In section 3.4, we describe the Branch and Bound scheme that solves the problem optimally, followed by our proposed hierarchical model in section 3.5. The problem formulations at each level of the hierarchy are presented in section 3.6, and the algorithms for them are detailed in section 3.7. Section 3.8 is devoted to the comparison between the hierarchical approach and the optimal algorithm. Finally, in section 3.9 we draw our conclusions.

3.2 Problem Formulation

In this section, we formally describe the details and notations of the problem at hand, and present its formulation.

We consider a single-stage manufacturing system consisting of a single machine M, capable of producing a set of n part types represented by $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$. Let τ_j denote the processing time required on M to produce one unit of part type p_j; j=1,2,...,n. We also assume that data regarding average part demand is available, either from historical information or

sales forecasting. Let n_j represent the average number of products of type p_j required per unit time; $j = 1,2,\ldots,n$.

3.2.1 Demand

We assume that orders are accepted in a particular planning horizon. Let this horizon be denoted by H, which is composed of z elementary periods. An elementary period is any convenient period, e.g. a day; this will be qualified later. A set of production orders are assumed to appear in every elementary period. Each order contains the following information :

- Part type, i.e., an index $j \in \{1,2,\ldots,n\}$.
- Quantity, q_j, $q_j \in$ IN.
- Due date, s_j, i.e., $s_j \in \{1,2,\ldots,z\}$.

An elementary period has a duration of Δ time units. It is assumed that $\tau_j << \Delta$. Furthermore, for simplicity, we assume that the delivery of orders takes place at the end of an elementary period.

Thus, for each elementary period, we know the part types and their respective quantities to deliver. Let d(j,k) denote the number of units of part type p_j to be delivered at the end of the k-th period.

3.2.2 Criteria

We consider the total weighted earliness and tardiness penalties in the scheduling model as the criteria to be minimized. These criteria combined result in a nonregular performance measure. Let c_j^+ represent the cost associated with holding one unit of part type p_j in stock for one elementary period. Equivalently, let c_j^- represent the cost associated with one unit of part type p_j delayed by one elementary period. In our approach, we are not restricted by assumptions such as *"agreeable"* weighting of jobs (Lawler 1977 [56]). In our formulation, backlogging costs, inventory costs, and processing times need not hold any specific relationship.

We now present the problem formulation as follows. Let there be z elementary periods in the scheduling horizon. Although it is not

restrictive that the duration of all elementary periods be identical (equal to Δ), we assume this for simplicity. We consider the problem in terms of scheduling jobs as opposed to parts introduced earlier; a job is a specific instance of a part with a due date. This is to differentiate each part required by a unique identifier. Let the total number of jobs to be scheduled be N, (Note : $N = \Sigma_k \Sigma_j d(j,k)$). Job i is described by : (1) t_i, denoting the processing time, and (2) d_i, denoting the due date ($d_i \in \{1,2,\ldots,z\}$). Let C_i denote the completion time of job i. Then, x_i, denoting the elementary period in which job i is completed, can be computed by $\lceil C_i/\Delta \rceil$, where $\lceil \bullet \rceil$ denotes the smallest integer greater than or equal to $\bullet$. Let E_i and T_i represent the earliness and tardiness, respectively, of job i; these are detailed as : $E_i = \text{Max}(d_i - x_i, 0) = (d_i - x_i)^+$ and $T_i = \text{Max}(x_i - d_i, 0) = (x_i - d_i)^+$, respectively. Note : $E_i, T_i \in$ IN. We associate with each job i, an earliness weight w_i^+, and a tardiness weight w_i^-, when i is completed one period before or after its due date, respectively. For instance, if job i represents a part type j that is due at the end of period k, we have : $t_i = \tau_j$, $d_i = k$, $w_i^+ = c_j^+$, $w_i^- = c_j^-$. Let S represent a schedule.

The scheduling problem can be formally stated as follows :

$$\text{Minimize :} \quad \text{Cost}(S) = \sum_{i=1}^{N} (w_i^+ \times E_i + w_i^- \times T_i) \tag{2.1}$$

Subject to :

$$[C_r - t_r, C_r) \cap [C_s - t_s, C_s) = \varnothing ; \qquad r, s = 1,2,\ldots,N, \text{ and } r \neq s. \tag{2.2}$$

$$C_i - t_i \geq 0 ; \qquad i = 1,2,\ldots,N \tag{2.3}$$

Constraint (2.2) signifies that two jobs cannot be processed at the same time, and (2.3) implies that jobs cannot have non-negative starting times.

3.3 NP-completeness

In this section, we informally demonstrate that the scheduling problem detailed in section 3.2 is NP-complete, hence it is unlikely to be solved by a polynomial time algorithm. More precisely, we show that the decision problem, $\mathcal{D}$, "Is there a schedule with total cost no more than F ?," is NP-complete; hence the optimization problem is at least as hard.

(1) The scheduling problem is in NP.

Proof : It is easily seen to be in NP as a nondeterministic algorithm for it need only determine the value of x_i (nondeterministic polynomial time), and check in polynomial time if the schedule costs no more than F.

(2) The scheduling problem is in NP-complete.

Proof by Restriction : We restrict $\mathcal{D}$ to solve the KNAPSACK problem (Garey and Johnson 1979 [29]), which is a well known NP-complete problem.

KNAPSACK problem instance : A finite set U, a "size" $s(u) \in Z^+$ and a "value" $v(u) \in Z^+$ for each $u \in U$, a size constraint $B \in Z^+$, and a value goal $K \in Z^+$.

Decision problem : Is there a subset U' of U such that :

$$\sum_{u \in U'} s(u) \leq B \quad \text{and} \quad \sum_{u \in U'} v(u) \geq K \tag{3.1}$$

Restrictions on $\mathcal{D}$:

(1) Let number of periods, z = 2.

(2) Let duration of first period, $\Delta = B$, letting duration for second period = ∞.

(3) $\forall u_i \in U$, let $t_i = s(u_i)$, $d_i = 1$, $w_i^- = v(u_i)$ and $w_i^+ = 0$.

(4) If $F = \sum_{u \in U} v(u) - K$

It can now be seen that the answer to the decision problem $\mathcal{D}$ is in fact the answer to the Knapsack problem (3.1). Since the Knapsack problem is NP-complete, it implies that the decision problem $\mathcal{D}$ is also NP-complete.

Q.E.D.

3.4 The Branch and Bound Algorithm

Owing to the discrete nature of the penalty functions, the nonregular nature of our criterion, as well as the introduction of idle time, it is difficult to apply or develop job ordering theorems. At this time we are unaware of any applicable job ordering theorems for this problem, hence the present version of the Branch and Bound algorithm does not employ

any pruning devices. Except, of course, in a very specific case, where for two jobs i and j, $t_i = t_j$, $d_i \geq d_j$, $w_i^+ \geq w_j^+$, and $w_i^- \leq w_j^-$, it is implied that j preceeds i in an optimal schedule. Although we can introduce precedence relationships for jobs corresponding to the same part type (i.e. jobs with equal processing times) to reduce the search space, this was not employed, as the algorithm provided one among the multiple optima in a reasonable time for most cases.

We employ four heuristic rules to obtain feasible solutions of the scheduling problem; the lowest cost obtained serves as the initial upper bound. These heuristic rules, and their performances are detailed in section 3.4.1.

We order jobs according to earliest due dates, breaking ties with lower job number first. The algorithm begins with the scheduling of the last job (i.e., the one with the latest due date), and procedes in a backward fashion until the first job is scheduled. At any step of the algorithm, the deepest node in the tree (a partial schedule with the maximum number of scheduled jobs) is expanded, thus breaking ties by lowest value first, provided its value is strictly less than the current upper bound. The value of a node is defined as the cost of the partial schedule so far, plus a lower bound estimate of the cost of scheduling the remaining jobs. We detail the calculation of the lower bound in section 3.4.2. For the node being expanded, we have a set of tentatively scheduled jobs (i.e., x_i are known), and we explore the possibility of scheduling the completion of the subsequent job in each of the periods. For each admissible assignment we can compute the value of the new node and add it to the list of unmarked nodes, provided its value is strictly lower than the current upper bound. The parent node is then marked. The upper bound is updated when a node corresponding to the first job is marked. The algorithm stops when there is no unmarked node with a value strictly lower than the current upper bound. The current upper bound is now the optimal cost, and the corresponding schedule is the optimal schedule. Note that the "depth first" strategy is preferred over

the "best first" one, because it helps in restricting the search and updates the upper bound faster.

3.4.1 Initial Upper Bound : Heuristic Rules

In this section, we describe four heuristic scheduling rules in order to obtain a feasible schedule with a low cost. These rules essentially prioritize the jobs; jobs with higher priority are scheduled in the most appropriate available time slot first.

R1 : Jobs are arranged in nonincreasing order of the ratio $(w_i^- + w_i^+)/t_i$.
R2 : Jobs are arranged in nonincreasing order of $(w_i^- + w_i^+)$.
R3 : Jobs are arranged in nonincreasing order of the ratio $Max(w_i^- , w_i^+)/t_i$.
R4 : Jobs are arranged in nonincreasing order of $Max(w_i^- , w_i^+)$.

At each step, the heuristic selects the first unscheduled job and schedules it in the most appropriate time slot. The most appropriate time slot is defined as the one that contributes the least to the cost function. In other words, if feasible, schedule the completion of a job in the period corresponding to its due date, else try scheduling its completion in the period before (after) its due date, is the earliness penalty is less than (greater than) the tardiness penalty. If the job remains unscheduled, schedule its completion in the first feasible period before or after its due date that imposes the least penalty.

Several trials were performed with varied number of jobs, penalties, processing times and due-dates. In 50% of the cases, one of the heuristics was able to provide a bound within 10% of the optimal. In table 4.1, we present the number of times (in percentage) that a rule provided the lowest cost. Note that the summation is greater than 100 because for some cases more than one rule provided the lowest cost.

Rule 1	Rule 2	Rule 3	Rule 4
75%	43%	56%	12%

Table 4.1 : Percentage of cases the lower bound was reached by a rule

3.4.2 Lower Bound Estimate of Scheduling Remaining Jobs

We consider an elementary period k, and determine the optimal cost to schedule the remaining jobs due at the end of this period $\{i \mid d_i \in k$, and is unscheduled$\}$ with unconstrained boundary conditions from the neighboring period(s). Boundary conditions of the scheduling horizon (external) are still respected by the first elementary period. The optimal cost for one period is once again obtained using the Branch and Bound algorithm. Owing to the small size of this problem, it is usually computed quite fast. Furthermore, the unconstrained version can easily be converted into a problem with a single criterion, by replacing the unit earliness and unit tardiness weights by a single weight w_i as follows :

$$w_i = \begin{cases} w_i^- & \text{if } d_i = 1 \text{ (Job due in first period)} \\ \text{Min}\{w_i^-, w_i^+\} & \text{otherwise} \end{cases} \qquad (4.1)$$

The computational efficiency of this bound is further improved by using a "quick bound procedure," detailed in section 3.4.3.

The above process is repeated for each of the periods (k=1,2,...z). The lower bound estimate for all remaining jobs is the summation of the optimal costs for each of the periods. Lower bounds computed in this manner were usually obtained quickly. The closeness of the bound to the actual costs was found to be "good" in most cases, especially when overloaded (i.e., total duration of jobs due at the end of a period exceed, Δ) and underloaded elementary periods bordered each other. However, when the load profile was triangular in nature (i.e., when most jobs were due in the intial (or ending) periods), the estimates obtained were fairly below the actual costs. For 25 problem instances, the initial lower bound was about 65% of the optimal on an average.

3.4.3 Quick Bound Procedure

Once again, we consider an elementary period k, and try to schedule the remaining jobs due at the end of this period $\{i \mid d_i \in k$, and is unscheduled$\}$ with unconstrained boundary conditions from the neighboring period(s);

the difference here is that instead of determining the optimal scheduling cost, we estimate it by a lower bound that permits partial job penalization. In other words, if a job begins being processed in period k, but is completed in k+1, we prorate the penalization by the duration the job spent in period k+1. This lower bound can be computed efficiently by considering jobs, ordered according to the nonincreasing (w_i/t_i) ratio, to be included in period k. The above process is repeated for each period in the horizon under consideration. The lower bound estimate for all remaining jobs is the summation of the "quick bounds" for each of the periods. We refer to this as the "quick bound procedure." This procedure can be employed for (i) the computation of the lower bound in the Branch and Bound algorithm, and/or (ii) the lower bound of section 3.4.2. Note : in case of (i) the horizon under consideration will have z periods, while in case of (ii) it will have 2 periods.

Although it is difficult to comment in greater detail about the quality of the bound, and the reduction in search space gained thereof, in table 4.2 we present the influence of the use of this bound on the c.p.u. times required by random problem instances (see Table 8.1 and 8.2 for instance details).

Problem instance (see tables 8.1 & 8.2)	LB = 0	Quick bound procedure	Lower bound procedure
1(a)	10.38 sec	5.75 sec	3.73 sec
8(a)	30.41 sec	21.15 sec	6.58 sec
7(a)	unsolved	2166.58 sec	295.80 sec

Table 4.2 : Influence of bound procedure on c.p.u. time for 3 problem instances

3.5 Description of the Hierarchical Model

The hierarchical approach is a heuristic that is intended to overcome the intractability associated with the problem described in section 3.2. The original problem is decomposed into a set of simpler sub-problems that are solved sequentially, in order to obtain the solution of the overall problem.

Such models can provide results very fast. This benefit is, not however, obtained at no cost; the compromise is in the quality of the solution.

Our objective is to develop a general hierarchical model that is not only capable of addressing the current problem, but the following issues associated with planning/scheduling problems as well :

(1) Unreliable machine : Machines are prone to failures or breakdowns. While a machine is under repair it cannot process a part, and this leads to a reduction in capacity. We assume that the working period and repair periods are exponentially distributed random variables with means of MTBF and MTTR respectively, along with MTBF and MTTR$<<\Delta$. The assumptions of nonpreemption still hold.

(2) Random demand : Although scheduling horizons are usually small with demand known deterministically, random events like order cancellations, revisions, expeditions, etc. are not unusual. Especially in a sales forecast environment, the demand is subject to change due to uncertainty. In such cases, frequent recomputations of the already time consuming scheduling algorithm may not be possible. The hierarchical model can accommodate the uncertainty to a certain degree, by aggregating similar parts.

In this section, we develop a hierarchical model under a set of assumptions that is applicable to a particular class of problems. We will then specialize the general hierarchical model to address the deterministic scheduling problem at hand and evaluate the perfromance of this model by comparison to the Branch and Bound scheme.

This hierarchical model is intended to address a class of problems in which we assume that: (i) $c_i^- >> c_i^+ \ \forall i$, or the backlogging costs are much greater than inventory costs in general, and (ii) within the horizon H we are able to meet production requirements and absorb cyclicity (seasonality) of the demand. Assumption (i) is in fact consistent with the trend

followed in industry, where backorders are penalized severely. This results in JIT with excess production "pulled" to prior periods, enabling delivery on the due date. On the other hand, $c_i^- << c_i^+$ will result in a JIT with excess production "pushed" to periods after the due date, resulting in undesirable backlog. Refer to section 3.8 for further justification. Assumption (ii) can be justified by a choice of H such that for the most typical demand patterns backward smoothing is possible (see section 3.6.1). These are also known as "pull systems."

The proposed hierarchical model consists of two levels : (i) High level, and (ii) Low level. The high level model is more aggregate, and the entities of relevance are part families. The low level model is detailed, and the entities of relevance are parts. Part families are constructed by grouping parts having similar processing times as well as similar holding costs. The part types are aggregated into part families by the K-mean algorithm in cluster analysis (Hartigan 1975 [39]). The advantages of aggregation in this manner allow for some uncertainty in the demand to be absorbed. The aggregate part family can be substituted by any part belonging to this family; this reduces the variance while retaining samilar processing durations and holding costs in the schedule. The set of f part families is represented by $\mathcal{F} = \{f_1, f_2, \ldots, f_r\}$. Let θ_j denote the processing time required on M to produce one unit of family type f_j; j=1,2,...,r. Let v_j represent the cost associated with holding in stock one unit of family type f_j for one elementary period. See figure 3.1 for details.

Now, for each elementary period, we can aggregate the demand of the parts to generate the gross demand of families. Let q(j,k) denote the number of units of family type f_j to be delivered at the end of the k-th period.

3.6 Problem Formulation

In this section, we describe the decomposed problem as two sub-problems at the two levels of the hierarchy, and formulate them as optimization problems.

3.6.1 High Level Optimization Model

At the high level of the hierarchy, we are interested in planning for the "flow" of part families : given an aggregate production requirement plan for the planning horizon H, the objective is to determine a production flow plan for part families, such that the system capacity, as well as backlogging constraints, are respected, and the total holding cost is minimized. Note that we treat backlogging cost as a constraint here because of the two assumptions made in section 3.5, and the following additional reasons. It is assumed that in steady-state (infinite horizon), we are able to meet production requirements, i.e., the production requirement is less than or equal to the average system capacity. We propose that a rolling horizon is capable of emulating steady-state behavior, and the horizon chosen is large enough to absorb cyclicity (seasonality) of the demand. Thus, under these propositions, the backlogging constraint is not over-restricting at this level. This process is also called backward smoothing.

Let u(j,k), u(j,k) $\in$ IR^+, denote the number of units of family type f_j to be produced during the k-th period, and u(j,0) denotes the initial stock level of family type f_j. We define x(i,k) to be the excess production for part type p_i at the end of the k-th period, and is detailed in (6.1).

$$x(j,k) = \left\{ \sum_{t=0}^{k} u(j,t) - \sum_{t=1}^{k} q(j,t) \right\} \tag{6.1}$$

The high level optimization problem can then be formally stated as follows :

$$\text{Minimize : } \sum_{j=1}^{r} \left(v_j \times \sum_{k=1}^{z} x(j,k) \right) \tag{6.2}$$

Subject to :

$$\sum_{j=1}^{r} \theta_j \times u(j,k) \leq \frac{MTBF}{MTBF+MTTR} \times \Delta; \qquad k = 1,2,\ldots,z \tag{6.3}$$

$$\sum_{t=0}^{k} u(j,t) \geq \sum_{t=1}^{k} q(j,t); \qquad j = 1,2,\ldots,r;\ k = 1,2,\ldots,z \qquad (6.4)$$

The objective (6.2) refers to the minimization of holding costs for part families. Contraint (6.3) implies that production time of part families is no higher than the average available capacity on the machine for each period, and (6.4) is the constraint for not backlogging. By substituting for x(j,k) from equation (6.1) in objective (6.2), and realizing that the latter summation is a constant, the objective (6.2) can be reformulated as follows:

$$\text{Minimize : } \sum_{j=1}^{r} \left(v_j \times \sum_{k=1}^{z} \sum_{t=0}^{k} u(j,k) \right) \qquad (6.5)$$

If we replace $\theta_j \times u(j,k)$ by u'(j,k), the above linear programming problem can be transformed into the following :

$$\text{Minimize : } \sum_{j=1}^{r} \left(v'_j \times \sum_{k=1}^{z} \sum_{t=0}^{k} u'(j,k) \right) \qquad (6.6)$$

Subject to :

$$\sum_{j=1}^{r} u'(j,k) \leq \frac{MTBF}{MTBF+MTTR} \times \Delta; \qquad k = 1,2,\ldots,z \qquad (6.7)$$

$$\sum_{t=0}^{k} u'(j,t) \geq \sum_{t=1}^{k} q'(j,t); \qquad j = 1,2,\ldots,r;\ k = 1,2,\ldots,z \qquad (6.8)$$

where : $q'(j,k) = \theta_j \times q(j,k)$, and $v'_j = v_j/\theta_j$.

It can be shown that these problems are equivalent. Interestingly, this problem can be optimally solved without having to formally solve the linear programming problem. In section 3.7, we propose the algorithm that optimally solves (6.6) subject to (6.7) and (6.8).

3.6.2 Low Level Optimization Model

At the low level of the hierarchy, we are interested in scheduling parts on the machine under constraints of machine state, as mentioned in section

3.5. When the machine is in repair state it cannot produce parts; therefore, there is no available capacity. However, when the machine is in working state, it can be utilized to full capacity. The problem at the low-level is then to determine the sequence of parts to load in the low-level horizon, such that orders are satisfied with minimal backlogging, and the high-level flow plan is respected as closely as possible. The low-level horizon is composed of w elementary periods[1]. Note that it may not always be possible to satisfy the orders (without backlog) because of machine failures and the aggregate nature of the high-level flow plan. In fact, the feasibility of the high-level flow plan depends on : (i) the "closeness" of the parts belonging to the same family (see section 4.11 for details), and (ii) the ratio (mix) of parts within families. We hope that "good" part families (aggregation) can ensure consistency of the high-level plan. Generally, we can satisfy short term production requirements on the average.

Let y(i,k), y(i,k) $\in$ IN, denote the number of units of part type p_i produced during the k-th period, and y(i,0) denote the initial stock level of part type p_i. We define s(i,k) to be the deficit production for part type p_i at the end of the k-th period, and is detailed in (6.9).

$$s(i,k) = \text{Max}\left\{\sum_{t=1}^{k} d(i,t) - \sum_{t=0}^{k} y(i,t) , 0\right\} \tag{6.9}$$

We also define the occupation of machine M by a part type p_i by a binary variable $e_i(t)$:

$$e_i(t) = \begin{cases} 1 & \text{if part type } p_i \text{ is being operated by machine M at time t} \\ 0 & \text{otherwise} \end{cases} \tag{6.10}$$

The low level problem can then be formally stated as follows.

$$\text{Minimize : } \sum_{i=1}^{n} \left(c_i^- \sum_{t=1}^{w} s(i,t) \right) \tag{6.11}$$

---

[1] If the low level horizon is composed of w elementary periods, and the high level horizon is composed of h low level horizons, it follows that : $w \times h = z$.

Subject to :

$$\sum_{i=1}^{n} e_i(t) \leq \begin{cases} 1 & \text{if machine M is operational} \\ 0 & \text{otherwise} \end{cases} \qquad t \in IR^+ \qquad (6.12)$$

$$\sum_{i \mid p_i \in f_j} y(i,k) = u(j,k); \qquad j = 1,2,\ldots,r;\ k = 1,2,\ldots,z \qquad (6.13)$$

Constraint (6.12) implies that a maximum of one part can occupy the machine when it is operational, while none can occupy it when it is under repair. Constraint (6.13) satisfies the high level requirement for part families. Note that the variables y(i,k) are integers while flow variables u(j,k) are continuous. Thus, we can schedule the beginning time of $p_i \in f_j$, such that only a portion of it be produced in the relevant period, while it is completed in the subsequent period. In other words, a fractional part requirement means that the part is completed in the subsequent period. This, however, requires that there be no more than two partial u(j,k) for every period k; k = 2,...,z-1 (corresponding to the start and to the end of the period). This way we can still respect the upper level constraints.

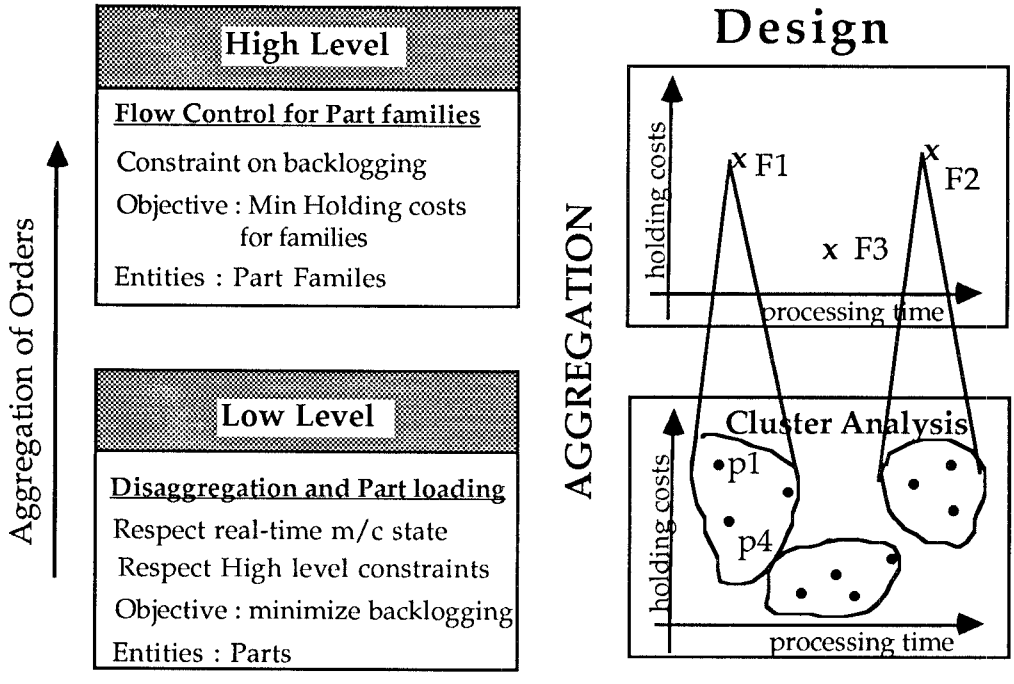


Figure 3.1 : Structure and Design of the Hierarchical Scheduling Policy

The problem detailed in this section can be solved using a heuristic rule which yields acceptable results. Thus, we do not formally solve the linear programming type problem for which the sequence of failures are not known *a priori*. In section 3.7, we propose this heuristic algorithm.

3.7 Algorithms

3.7.1 High level Algorithm

In this section, we detail the backward smoothing algorithm that solves (6.6) subject to (6.7) and (6.8). Note that before we use this algorithm, we transform the gross demand, q(j,k), into an effective demand by netting against on hand stock, u(j,0). Thus, for this algorithm, q(j,k) represents the net demand of family type f_j required at the end of the k-th period.

If NR = {1,2,...,r}, we define an ordering function O1, O1 : NR $\rightarrow$ NR, with $v'_{o1(i)} \geq v'_{o1(j)}$ $\forall i < j$, $i,j \in$ NR.

```
for k = z down to 1 do
    Initialize :
    u(j,k) = 0 for j = 1,2,...,r;
    Available_capacity := MTBF/(MTBF+MTTR) × Δ;
    for i = 1 to r do
        j = O1(i);
        load = θ_j × q(j,k);
        if(load ≤ Available_capacity) then
            Available_capacity = Available_capacity - load;
            u(j,k) = q(j,k)
        else
            u(j,k) = (Available_capacity/load) × q(j,k);
            if(k = 1) then
                "Infeasible Plan"; exit;
            q(j,k-1) = q(j,k-1) + (q(j,k) - u(j,k))
            for s = i+1 to r do
                j = O1(s);
```

q(j,k-1) = q(j,k-1) + q(j,k);
exit loop of i;

Although this algorithm is optimal, it could result in more than two partial jobs in an intermediate period, thus making it difficult for the low level to respect the constraint (6.13). Therefore, we modify the above algorithm to account for the constraint of no more than two partial jobs per period; this version is presented below.

Modified algorithm :
Initialize :
u(j,k) = 0 for j = 1,2,...,r; k = 1,2,...,z.
used_capacity = 0;
for k = z down to 1 do
Available_capacity := $\frac{\text{MTBF}}{\text{MTBF+MTTR}} \times \Delta$ - used_capacity;
used_capacity = 0;
for i = 1 to r do
j = O1(i);
load = $\theta_j \times$ q(j,k);
if(load ≤ Available_capacity) then
Available_capacity = Available_capacity - load;
u(j,k) = u(j,k) + q(j,k)
else
load = (Available_capacity/load) × q(j,k);
u(j,k) = u(j,k) + load;
if(k = 1) then
"Infeasible Plan"; exit;
u(j,k-1) = ⌈load⌉ - load;
used_capacity = (⌈load⌉ - load)× θ_j;
q(j,k-1) = q(j,k-1) + (q(j,k) - ⌈load⌉)
for s = i+1 to r do
j = O1(s);
q(j,k-1) = q(j,k-1) + q(j,k);
exit loop of i;

3.7.2 Low Level Algorithm

In this section, we detail the scheduling algorithm that solves (6.11) subject to (6.12) and (6.13). This algorithm schedules parts in real-time and can be used along with consideration of failure and repair events.

If NN = {1,2,...,n}, we define an ordering function O2, O2 : NN $\rightarrow$ NN, with $a_{o2(i)} \geq a_{o2(j)}$ $\forall i < j$, $i,j \in$ NN. Where : $a_i = c_i^- / \tau_i$. Ordering of parts by the ratio a_i is an equivalent to the shortest processing time ordering, weighted appropriately by the backlogging cost.

```
for k = 1 to w do
    repeat
     if M is operational and idle :
            i = find_next_part(k,delay);
            if(i ≠ none & delay = 0)
                  schedule p_i immediately; wait until part is completed;
            if(i ≠ none & delay ≠ 0)
                  schedule p_i after delay; wait until part is completed;
            if(i = none)
                   system idle; wait until end of the period;
     else
            wait till machine repair is complete;
    until(i = none);

find_next_part(k,delay)
    delay = 0;
    /*  Satisfy immediate demand    */
    for i = 1 to n do
        j = O2(i);
        if(s(j,k) > 0)  /*  see equation 6.9 for definition of s(j,k)    */
            return j;
    /*  Verify high level flow plan    */
```

```
if  Σ_{i | p_i ∈ f_j} y(i,k)  = u(j,k)   for i = 1 to n
        return none;
/*  Try to satisfy high level flow plan   */
for r = k+1 to w do
    for i = 1 to n do
        if u(j,k) - Σ_{i | p_i ∈ f_j} y(i,k)  ≥ 1
            if d(i,r) > 0       return j;
/*  Only one partial part left - schedule with(out) delay     */
for r = k+1 to w do
    for i = 1 to n do
        if u(j,k) - Σ_{i | p_i ∈ f_j} y(i,k)  ≥ 0
            if d(i,r) > 0
                if τ_i < remaining time in period k
                    delay = (remaining time in period k - τ_i - epsilon)^+
                return i;
/*  Default     */
return none;
```

3.8 Comparisons

This section is devoted to comparisons between the performance of the hierarchical model and the branch and bound algorithm. At first, test examples were created; data related to 10 parts were constructed. For each example, a set of random demands was generated using different starting seeds. The generation of this demand is detailed as follows.

As explained in section 3.2, n_i, represents the average production per unit time for part p_i. We introduce an integer Q_i indicating the maximal size of demand that can arise for part p_i in an elementary period. Now for each elementary period, we decide to generate a demand for p_i with a probability μ_i, i = 1,2,...,n. Then, the size of the demand is chosen, with

equal probability, among the set $\{1,2,\ldots,Q_i\}$ of integer values. In other words, once we decide to generate a demand, we determine the size of the demand by the discrete uniform distribution, $U(1,Q_i)$. The process is repeated for each part type and then for each elementary period of the horizon H, i.e., for z periods. The probabilities of generating a demand with a probability μ_i, can be calculated as follows:

$$\mu_i = \frac{2\, n_i \times \Delta}{(Q_i + 1)}; \qquad i = 1,2,\ldots,n \tag{8.1}$$

Thus, using different seeds, we can generate different sets of demand for period H. Generation of demand in this fashion will ensure the close conformance of the simulated mean demand to the given mean demand (n_i) in the long run.

Since the number of parts considered is very small, reduction of dimensionality by aggregating parts at the high level of the hierarchical model was not considered as necessary. Moreover, the demand is deterministic, so the other benifits of aggregation cannot be exploited. Thus, at the high level each family represents a single part. Note that the function of the low level is merely to prioritize parts within an elementary period, and cannot change the value of the objective function.

All comparisons were performed in a deterministic environment. Furthermore, for the first set, demand was considered for only 16 elementary periods, and the horizon was not rolled. This was done with the intent to demonstrate the performance of the hierarchical approach in random isolated cases. It is not difficult to appreciate that the performance of the hierarchical model with a rolling horizon will be better due to its continual "look ahead" nature, while the optimal algorithm is intended to solve the problem in a horizon-after-horizon fashion. This constitutes the second set of comparisons. Also, in an unreliable environment, the performance of the hierarchical model is expected to be relatively good, due to the incorporation of averages at the high level, as opposed to the need for repeated computations of the optimal solution after every repair

event in the monolithic case. However, this is not part of the present work. In this study we wish to study a simple set of problems in a deterministic environment that may not necessarily be the most favorable to the hierarchical model.

Table 8.1 presents the characteristics of the part related data. Examples 1 through 4 are the ones for which our assumption related to $c_i^- >> c_i^+ \forall i$, is reflected. In the examples 5 through 9, $c_i^- > c_i^+$ in general, but not consistently. Finally, in example 10, $c_i^- << c_i^+ \forall i$, which is absolutely contrary to our assumption. Through these examples, we attempt to study the performance under the conditions that reflect our assumption, and then study the decay when the assumptions cease to hold true.

Example	n	Earliness Penalties	Tardiness Penalties	Average Demand	Maximal Demand	Processing time
1	10	U(1,10)	U(100,500)	U(.15,.25)	2	U(0.2,0.5)
2	10	U(1,10)	U(100,500)	U(2,3)	5	U(0.02,0.05)
3	10	U(1,10)	U(100,500)	U(.15,.25)	4	U(0.05,0.1)
4	10	U(1,10)	U(100,500)	U(.3,.5)	3	U(0.1,0.3)
5	10	U(10,60)	U(50,100)	U(.1,.2)	2	U(0.2,0.5)
6	10	U(10,60)	U(50,100)	U(.1,.2)	2	U(0.2,0.5)
7	10	U(10,60)	U(50,100)	U(.3,.5)	3	U(0.1,0.3)
8	10	U(10,60)	U(50,100)	U(.15,.25)	2	U(0.2,0.5)
9	10	U(10,60)	U(50,100)	U(2,3)	5	U(0.02,0.05)
10	10	U(100,500)	U(1,10)	U(.3,.5)	3	U(0.1,0.3)

Table 8.1 : Part related data for 10 examples

The Branch and Bound algorithm, as well as the algorithms for the Hierarchical Model presented in section 3.7 were coded in C and executed on the SUN SPARC/Unix station 330 platform. Table 8.2 presents a summary of the results relating to the value of the objective function and the processing times reported to the nearest second. The Branch and Bound algorithm was of course not able to determine (or verify) the optimal in some cases where the program was terminated after about 1

hour, or because of memory constraints. We still report the best cost obtained in these cases and indicate it with an asterisk (*), but processing times are not reported.

Example	case	Number of Jobs	Seed	Branch and Bound		Hierarchical Model	
				Cost	Time	Cost	Time
1	a	35	1234567	17.28	3.73	17.28	0.06
	b	36	999999	102.87*	--	102.57	0.08
	c	21	11111	7.28	0.05	9.85	0.05
2	a	400	1234567	19.35	588.74	19.35	0.15
	b	416	999999	139.83*	--	139.83	0.15
	c	383	11111	47.32*	--	46.03	0.12
3	a	166	1234567	9.44	2.33	10.88	0.10
	b	171	999999	38.85*	--	54.12	0.10
	c	186	11111	50.29*	--	54.59	0.10
4	a	60	1234567	39.13*	--	39.13	0.07
	b	52	11111	9.84	0.33	9.84	0.08
	c	54	55555	109.93*	--	112.78	0.07
5	a	27	1234567	98.60	0.38	133.86	0.07
	b	27	999999	201.29	2.21	250.21	0.08
6	a	32	1234567	31.51	0.12	46.47	0.07
	b	30	99999	115.79	0.18	115.79	0.07
7	a	60	1234567	152.02	295.80	179.09	0.07
	b	52	11111	79.60	0.25	86.52	0.07
8	a	35	1234567	100.85	6.58	100.85	0.08
	b	36	999999	536.40*	--	590.79	0.08
9	a	400	1234567	183.05*	--	212.98	0.13
	b	416	999999	819.53*	--	1308.09	0.13
10	a	60	1234567	45.82*	--	2195.14	0.07
	b	52	11111	9.84	0.23	871.77	0.08
	c	54	55555	103.84*	--	2943.31	0.07

Table 8.2 : Summary of results for 25 demand samples

To summarize the comparison, we divide the results into three categories depending on the part penalties. The first category belongs to examples 1-4 ($c_i^- >> c_i^+ \forall i$), which are in line with our assumptions. For this category, the Hierarchical Model was able to obtain the optimal (or the best solution obtained by Branch and Bound) in about 60% of the cases. For 75% of the cases the Hierarchical Model obtained a result within 5% of the optimal value. Solutions were 8% worse than the optimum on the average.

Furthermore, it is important to indicate that the Hierarchical Model always found a solution better (if not the same) than the best upper bound heuristic (section 3.2.1).

The second category belongs to examples 5-9 ($c_i^- > c_i^+$ in general, but not always), which reflect a fair deviation from our assumptions. For this category, the Hierarchical Model was able to obtain the optimal (or the best solution obtained by Branch and Bound) in about 20% of the cases. For 40% of the cases the Hierarchical Model obtained a result within 10% of the optimal value. Solutions were 20% worse than the optimum on the average. This decay in performance can be expected.

Finally, the last category is that of example 10 ($c_i^- << c_i^+ \forall i$), which is absolutely contrary to our assumptions. For this category, the Hierarchical Model obtained results very far from the optimum. Although Branch and Bound provides low cost schedules, it is important to indicate in such schedules 13% to 21% of the jobs were tardy, and seldom were jobs early. In fact some jobs were upto 7 to 10 periods late. These figures indicate signs of an unhealthy system with overdue orders for significant durations. These results reenforce our assumptions relating to penalties presented in section 3.5. However, if our objective is "no inventory," for this it would be better to design a complementary hierarchy with a constraint of no holding at the high-level, along with an objective of minimizing the backlogging cost. Such a hierarchy is expected to yield results close to the optimal solution under these set of conditions.

The second set of comparisons is performed on relatively longer time horizons. We consider a total of 160 elementary periods. The Branch and Bound had a horizon of 16 periods (z=16), so 10 (or 5) problems were solved on a period-by-period basis. The Hierarchical Model had a similar high level horizon (z=16), with a low level horizon of 4 periods. Thus, the horizon at the high level was rolled every 4 elementary periods. This required the high level problem to be solved 40 (or 20) times. The results pertaining to these runs are presented in table 8.3.

Example	case	Number of Jobs	Seed	Branch and Bound		Hierarchical Model	
				Cost	Time	Cost	Time
1	a	10	1234567	585.22	182.16	143.51	1.85
	b	5	999999	279.32*	738	126.18	0.93
	c	10	11111	107.62	153.63	115.88	1.91

Table 8.3 : Summary of results for 25 demand samples

These results indicate that the Hierarchical Model takes advantage of rolling horizon, and can even perform better than Branch and Bound solved period-by-period. Furthermore, the c.p.u. times are more than 100 times longer for the Branch and Bound scheme. We attempted to enhance the Branch and Bound scheme with a rolling horizon and compare this enhanced scheme with the Hierarchical Model. This was done with the intention of eliminating the bias of rolling. This posed several difficulties. Firstly, as the Branch and Bound only suggests the completion periods of the jobs, selection of a job that crosses the boundary of two periods during which we perform rolling is arbitrary. This implies that we cannot guarantee optimality of this scheme. Secondly, and more importantly, it was very difficult to obtain 40 successful runs of Branch and Bound in sequence, besides the extremely high computation times. Thus, we were unable to perform this comparison.

3.9 Conclusions

We consider a one-machine scheduling problem with discrete, weighted earliness and tardiness. The problem considered is somewhat new, and finds place is some real-world applications. A branch-and-bound scheme is developed to solve this problem optimally. We have developed a hierarchical model that addresses a class of these scheduling problems. The major contribution of the chapter lies in the methodology for the design of a hierarchical model, intended to overcome the intractability of associated problems. Decomposition of the problem (for tractability), aggregation of entities (for variance reduction), problem formulations and algorithms at each level (for speed), and the rolling horizon concept (to emulate an infinite horizon), have been strung to yield a consistent architecture and efficient solutions for such problems. The hierarchical

approach, by the nature of its design, is extendable to cater for (1) an unreliable machine, and (2) demand uncertainty (by aggregating similar parts), in the domain of stochastic scheduling. Performance of the hierachical model has been studied.

Although the number of examples tried were limited, we can draw the following conclusions. When our assumptions related to the tardiness penalty (being much greater than the earliness penalty) hold, the hierarchical model seems to result in acceptable solutions. For 75% of the cases the Hierarchical Model obtained a result within 5% of the optimal value. Solutions were 8% worse than the optimum on the average. Furthermore, when employed for large scheduling durations, the hierarchical model takes advantage of rolling horizons and can outperform Branch and Bound solved on a period-by-period basis. It requires extremely low cpu time, making it amenable to real-time applications, such as more complex job-shop problems. It can be demonstrated with similar numerical tests that in an unreliable environment the hierarchical approach continues to perform well.

Chapter 4

Aggregation Theory in Planning

Aggregate production planning is a technique which has not only been practiced in manufacturing industries, but in service industries as well. It is essentially performed to decide resource/work force levels, the fashion in which the company reacts to demand forecasts, and planning required to allow for changes in the product mix. Translating demand forecasts for a wide range of products into resource requirements is a difficult task, which is further complicated by the uncertainty of demand forecasts. Owing to these complexities, production planning is better performed hierarchically. An aggregate production planning methodology can be applied to any level of the hierarchy, where it is designed to address product families (groups of products) with different degrees of detail or level of aggregation. Higher levels in a hierarchy are more aggregate, in fact sometimes in practice the highest level considers a single aggregate measure, like dollar value at cost, product surface, volume, weight, to represent the entire product line. As we progressively go down the hierarchy, the level of aggregation of the product groups decreases, and more detailed planning decisions are performed. Very often, spatial aggregation, i.e. aggregation of production facilities is also performed along with the aggregation of products.

What the most appropriate aggregating schemes should be is not always obvious in practice. It depends on the context of the particular planning problem, the range of products, and the level of aggregation required. The scheme is very often chosen based on a typical cost structure (see Hax and Meal [40]), which may not be applicable to the entire gamut of production

systems. From the point of view of resource level requirements, in the first level of aggregation it is natural to consider products having similar processing requirements, so that they can be well represented by a common entity. Such approaches have been applied in practice as well as adopted in the literature (see Meier [60]). Most of these approaches tend to ignore other attributes of products, such as holding cost and backlogging cost, which are essential for the optimal allocation of resources among competing products. Very often in practice, optimality is not addressed explicitly; the production planning process is performed based on experience and/or with the help of support systems like MRP II. However, with industry having to face stiffer competition, the need for better production planning and allocation of resources is becoming increasingly important.

4.1 Introduction

In this chapter, we examine a generic manufacturing system consisting of a set of work-centers, and a set of product types to produce. The objectives are to design a hierarchy for planning, such that a set of criteria are optimized. We present an aggregation scheme under a set of simple but demonstrative conditions, which we refer to as the "perfect case," in order to develop the underlying aggregation theory. We develop a two-level hierarchy under this aggregation scheme and demonstrate the optimality of this approach. That is, the production planning problem solved using a hierarchy (based on this aggregation) leads to a solution, with a value of the criterion equal to that of the optimal solution of the problem solved non-hierarchically (monolithically).

In section 4.2, we detail the issues relating to the manufacturing system under consideration, the demand and the horizon over which it is considered, and the criterion of relevance. The problem formulation for a continuous production case is presented in section 4.3. In section 4.4, we detail the design of a two-level hierarchy to solve the problem presented in the section 4.3. Section 4.5 is devoted to the proof of optimality of the hierarchical approach. The proof of asymptotic optimality of the

continuous production approximation to the discrete part production case is detailed in section 4.6. The subsequent four sections (4.7 through 4.10) are devoted to the extensions of the perfect case, where we relax some restrictions to obtain more general aggregation schemes. The asymptotic proof of optimality of these extensions have also been included in these sections. We present the general aggregation scheme we propose for such planning problems in section 4.11. Finally, section 4.12 is devoted to the temporal aggregation issues.

4.2 Generalities

4.2.1 The Manufacturing System

We consider a set $M = \{m_1, m_2, \ldots, m_m\}$ of *m* unique work-center types in a given manufacturing system. We may have multiple copies of a work-center type.

The set $P = \{p_1, p_2, \ldots, p_n\}$ represents *n* part types to be manufactured. Each part type has associated with it a manufacturing process or routing. A routing is defined by a sequence of operations, each performed on one of the work-centers. Let $T = [t_{ij}]$ represent the matrix of processing times ($i = 1,2,\ldots,n$; $j = 1,2,\ldots,m$), where t_{ij} represents the processing time of one unit of part type p_i on machine m_j (zero if machine m_j is not required in the routing of p_i). Set-up times are not considered. Further, in the case of products with multi-level Bills-Of-Material (BOM), the processing time, t_{ij}, represents the cumulative processing time (i.e. over all the components) required on machine m_j to produce one unit of of the end item p_i. Thus, P respresents the set of end items, or items for which independent demand is present.

4.2.2 Demand

We consider a planning horizon H composed of z elementary periods of duration Δ each. It is assumed that the duration of Δ is much larger than the lead times of parts, along which production of parts (initiation to completion) is performed. The demand is represented as follows. Parts are due at the end of each elementary period; let d_{ik} denote the number of

units of part type p_i required at the end of the k-th elementary period; i = 1,2,...,n; k = 1,2,...,z. Let x_{ik} denote the production planned for part p_i during the k-th elementary period; i = 1,2,...,n; k = 1,2,...,z. We consider two cases : (A) $x_{ik} \in IR^+$, i.e. the production is continuous, and (B) $x_{ik} \in IN$, i.e. the production is discrete. Case (A) corresponds typically to process industry, and it can be used also for production planning of discrete parts over a relatively long-term horizon, while case (B) corresponds typically to the production planning of discrete parts over a relatively short horizon.

4.2.3 Criteria

We consider the total weighted earliness and tardiness penalties to constitute the criteria to be minimized. In other words, we minimize the total inventory holding and backlogging costs. These criteria have been considered because they are almost always of relevance to a typical production planning problem. The penalty functions are defined at the end of a period, and are piece-wise linear functions of the inventory state. The functions are increasing on IR^+, and decreasing on IR^-.

Let w_i^+ represent the cost associated with holding one unit of part type p_i in stock for one elementary period. Similarly, let w_i^- represent the cost associated with one unit of part type p_i being delayed by one elementary period; i = 1,2,...,n.

4.3 Problem Formulation

The production planning problem in the continuous case (A), consists of determining the production, x_{ik}, in order to minimize the total inventory holding and backlogging cost.

The problem can be formally stated as the following linear programming problem (*P4.1*) :

$$\text{Minimize: } Cost(P) = \sum_{k=1}^{z} \sum_{i=1}^{n} (w_i^+ \times [Q_{ik} - D_{ik}]^+ + w_i^- \times [D_{ik} - Q_{ik}]^+) \quad (3.1)$$

where : $[\bullet]^+ = Max\{0, \bullet\}$.

D_{ik} denotes the cumulative demand of part p_i at the end of the k-th period, and is represented as :

$$D_{ik} = \sum_{a=1}^{k} d_{ia} ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad (3.2)$$

$$\text{Subject to : } Q_{ik} = Q_{i0} + \sum_{a=1}^{k} x_{ia} ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad (3.3)$$

Where Q_{ik} denotes the cumulative production plus initial inventory (denoted by Q_{i0}) of part p_i at the end of the k-th period.

$$\sum_{i=1}^{n} t_{ij} x_{ik} \leq \Delta; \qquad j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \qquad (3.4)$$

$$x_{ik} \geq 0 ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad (3.5)$$

Constraint (3.4) represents the capacity constraints on each machine. Note that, in this formulation, we assume that for simplicity only one machine of each type is available. However, in the general case, if b_j ($b_j \in \{1,2,3\ldots\}$) represents the number of machines of type m_j present in the system, the constraint can be replaced by the following :

$$\sum_{i=1}^{n} t_{ij} x_{ik} \leq \Delta \times b_j; \qquad j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \qquad (3.4')$$

Constraint (3.5) implies the non-negativity of production. We can equivalently formulate the problem ($\mathcal{P}4.0$) with zero initial inventory and a corresponding effective demand. This result is demonstrated in Appendix A.1, where we also present the computation of the effective demand. Thus, from here on, without loss of generality, we shall assume that the initial inventory is zero for all part types.

4.4 Hierarchical Approach : Perfect Case

The hierarchical approach is based on grouping parts into part families, and machines into cells. This reduces the dimension of the planning problem, and allows for faster computation of results.

In this section, we confine ourselves to a "perfect" case which is detailed below. We then prove optimality of the hierarchical approach under the "perfect" conditions. Later, we will demonstrate how these ideas can be extended to more general cases.

Aggregation of parts into part families is performed in a way that parts are grouped in a family if they have exactly the same processing and stocking characteristics, i.e., they use identical resources with equal processing times, as well as equal holding costs, and equal backlogging costs. The set of N part families is represented by $\mathcal{F} = \{f_1, f_2, \ldots, f_N\}$; note $N \leq n$. Two parts p_a and $p_b \in f_r$, $r = 1,2,\ldots,N$, iff :

(i) $t_{aj} = t_{bj}$; $j = 1,2,\ldots,m$
(ii) $w_a^+ = w_b^+$
(iii) $w_a^- = w_b^-$

Aggregation of machines is performed in a manner that groups machines in the same cell if they process the same set of parts with exactly the same processing times. The set of M cells is represented by $C = \{c_1, c_2, \ldots, c_M\}$; note $M \leq m$. Two machines m_a and $m_b \in c_r$, $r = 1,2,\ldots,M$, iff :

$t_{ia} = t_{ib}$; $i = 1,2,\ldots,n$.

Under these aggregation schemes, the processing time of one unit of family f_i on cell c_j is denoted by θ_{ij} ; i=1,2,...N, j=1,2,...,M. Let v_i^+ represent the cost associated with holding one unit of family type f_i in stock for one elementary period. Equivalently, let v_i^- represent the cost associated with one unit of family type f_i being delayed by one elementary period; i = 1,2,...,N. We can relate these new parameters as follows :

(i) $\theta_{ij} = t_{ab}$; where : $p_a \in f_i$, and $m_b \in c_j$; i=1,2,...N, j=1,2,...,M.
(ii) $v_i^+ = w_a^+$; where : $p_a \in f_i$; i=1,2,...N.
(ii) $v_i^- = w_a^-$; where : $p_a \in f_i$; i=1,2,...N.

The hierarchy consists of two levels, (i) the high level that plans production for part families on manufacturing cells, and (ii) the low level

that computes the disaggregation of the high level solution to determine the production plan for parts on individual machines.

4.4.1 High Level Problem

The high level problem consists of production planning for part families in manufacturing systems. Let y_{ik}, $y_{ik} \in IR^+$, denote the production planned for part family f_i during the k-th elementary period; i = 1,2,...,N; k = 1,2,...,z. The production planning problem consists of determining the production, y_{ik}, in order to minimize the total weighted earliness and tardiness costs for families.

The problem can be formally stated as the following linear programming problem (*P4.1*) :

$$\text{Minimize: Cost(F)} = \sum_{k=1}^{z} \sum_{i=1}^{N} (v_i^+ \times [U_{ik} - R_{ik}]^+ + v_i^- \times [R_{ik} - U_{ik}]^+) \quad (4.1)$$

R_{ik} denotes the cumulative demand of part family f_i at the end of the k-th period, and is represented as :

$$R_{ik} = \sum_{a=1}^{k} \sum_{b \mid p_b \in f_i} d_{ba} ; \qquad i = 1,2,...,N, \text{ and } k = 1,2,...,z \quad (4.2)$$

$$\text{Subject to: } U_{ik} = \sum_{a=1}^{k} y_{ia} ; \qquad i = 1,2,...,N, \text{ and } k = 1,2,...,z \quad (4.3)$$

Where U_{ik} denotes the cumulative production of part family f_i at the end of the k-th period.

$$\sum_{i=1}^{N} \theta_{ij} y_{ik} \leq \Delta; \qquad j = 1,2,...,M, \text{ and } k = 1,2,...,z \quad (4.4)$$

$$y_{ik} \geq 0 ; \qquad i = 1,2,...,N, \text{ and } k = 1,2,...,z \quad (4.5)$$

Constraint (4.4) represents the capacity constraints on each cell, and (4.5) implies the non-negativity of production of part families. This problem is of smaller dimension than that of the one presented in the previous section, and can be solved by any of the existing linear programming tools.

4.4.2 Low Level Problem

The low level problem consists of production planning for parts, x_{ik}, from the aggregate solution, y_{jk}, of the high level. The disaggregation is performed over a short term horizon, L (L < H), in a manner that the high level constraints are respected and the value of the objective function remains unchanged over L. Production planning is usually performed on a rolling horizon basis, i.e. altough the high level solution is computed over H, only a part of it (L) is implemented, followed by a recomputation of the high level solution after L. This is in order to incorporate for future demands/forecasts progressively, and thus emulate an infinite production horizon. However, in this formulation, in order to avoid disaggregation inconsistencies inherent in most aggregate planning schemes, we assume L = H. In other words, the disaggregation is performed over the entire horizon H in order to ensure consistency over the entire horizon H. This is represented as the following problem (*P4.2*) :

$$y_{ik} = \sum_{b \mid p_b \in f_i} x_{bk} \; ; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \tag{4.6}$$

$$\text{If } U_{ik} - R_{ik} \begin{cases} \geq 0, \text{ then, } Q_{bk} - D_{bk} \geq 0 \;\forall\; b \mid p_b \in f_i \\ \leq 0, \text{ then, } Q_{bk} - D_{bk} \leq 0 \;\forall\; b \mid p_b \in f_i \end{cases} ; \quad \forall i, \text{ and } \forall k \tag{4.7}$$

$$x_{ik} \geq 0 \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \tag{4.8}$$

Constraint (4.6) ensures that the high level production plan for families is respected. Constraint (4.7) ensures that when the inventory level of a family is non-negative (resp. negative), then the inventory levels of each of the parts belonging to this family are also non-negative (resp. non-positive). This constraint ensures that the low level does not change the value of the criterion determined by the high level, thus, ensuring optimality. For justification of this constraint refer to Appendix A.3.

It is important to indicate that several solutions of equal cost may exist for the disaggregation problem. Thus, it will suffice to find one of these. In the next section, we present an algorithm to solve this problem. Alternatively,

we could take advantage of the property presented in Appendix A.3, to develop an iterative algorithm.

4.4.3 Disaggregation Algorithm

We only consider families which are composed of more than one part, because otherwise no disaggregation is required. At the beginning of the horizon, the inventory of all families is zero. Further it is claimed that there exists at least one optimal solution for which the ending inventory (final inventory at the end of the horizon) is not positive; it will either be zero or negative. This can be demonstrated by the fact that if we reduce the most recent production of a part p_a, for which the ending inventory is $Q_{az} - D_{az} > 0$, by one unit, we can then reduce the value of the objective function by at least w_a^+. Note that, if the most recent production was made in a period before the last one, we can reduce the objective function even more. On the other hand, if the ending inventory is negative, it implies that some demand remains unsatisfied.

The basic idea of this algorithm consists of considering one family at a time. We begin from the first period onwards. For each period, we examine the production planned versus the unfulfilled demand at this period as: (a) production is greater than or equal to the unfulfilled demand, or (b) production is less than the unfulfilled demand. By unfulfilled demand, we imply a demand for which there doesn't exist any, corresponding previously planned, production.

In case (a),we first decide to produce the unfulfilled demand of the current period, followed by production to satisfy the demand of the next period. If the demand in the next period is exhausted, we continue to produce against demand of the period thereafter until the production requirements for the family are complete. The demand in future periods is then updated to provide the new unfulfilled demand.

In case (b),we first decide to produce a portion of the unfulfilled demand of the current period in order to satisfy the production planned. Then, we

add the remaining unsatisfied demand to that of the next period. This will provide the updated demand for subsequent periods. Additionally, in the case where the ending inventory is negative in the last period, the demand remains unsatisfied. The process is repeated for each period and for each family.

Let r_{ik} represent the demand of family f_i at the end of the k-th period:

$$r_{ik} = \sum_{b \mid p_b \in f_i} d_{bk}; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z$$

For i = 1 to N do
 For k = 1 to z do
  Case $y_{ik} \geq r_{ik}$
   Produce_Forward(i,k);
  Case $y_{ik} < r_{ik}$
   $\forall\, b \mid p_b \in f_i$, select x_{bk} such that $0 \leq x_{bk} \leq d_{bk}$, and $\sum_b x_{bk} = y_{ik}$;
   If k = z
    /* Abandon unsatisfied demand */
   else
    $\forall\, b \mid p_b \in f_i$, $d_{bk+1} = d_{bk+1} + d_{bk} - x_{bk}$;
End.

Procedure Produce_Forward(i,k)
$\forall\, b \mid p_b \in f_i$, $x_{bk} = d_{bk}$; /* Produce immediate demand */
remainder = $y_{ik} - \sum_{b \mid p_b \in f_i} x_{bk}$
For j = k+1 to z do
 if remainder = 0
  return;
 $\forall\, b \mid p_b \in f_i$, select e_{bj} such that $0 \leq e_{bj} \leq d_{bj}$, and $\sum_b e_{bj} \leq$ remainder;
 /* Produce some later demand */
 $\forall\, b \mid p_b \in f_i$, $x_{bk} = x_{bk} + e_{bj}$, $d_{bj} = d_{bj} - e_{bj}$;
 remainder = remainder - $\sum_b e_{bj}$;
End.

Example 1

We present an example to illustrate the disaggregation algorithm. Consider a horizon composed of 15 elementary periods. We consider a single family composed of three part types. In table 4.1, we detail the demand for parts, production of the family computed by the high level, and the inventory levels. In this example, we assume that the high level solution is given. We take a part with a lower index first, when trying to satisfy demand versus planned production, i.e., we take part p_1 first and exhaust its current demand if possible, then p_2, and finally p_3. Producing against future demand also follows the same ordering. The results are presented in tables 4.2 and 4.3.

Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Total
Demand p_1	1	3	5	6	0	0	2	3	4	4	0	0	5	3	5	41
Demand p_2	1	0	5	2	6	5	2	3	4	4	5	5	5	2	5	54
Demand p_3	3	2	3	3	4	5	2	3	2	2	5	5	0	5	0	44
Demand F	5	5	13	11	10	10	6	9	10	10	10	10	10	10	10	139
Production	10	10	10	7	5	6	9	12	10	12	12	8	6	6	12	135
Inventory	5	10	7	3	-2	-6	-3	0	0	2	4	2	-2	-6	-4	

Table 4.1 : Demand, and production plan for the family F (Given)

Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Total
Part p_1	4	5	6	0	0	0	2	3	4	4	0	2	3	3	5	41
Part p_2	1	5	1	4	3	5	2	3	4	6	7	1	3	3	6	54
Part p_3	5	0	3	3	2	1	5	6	2	2	5	5	0	0	1	40

Table 4.2 : Disaggregated production plan for parts (Results)

Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Part p_1	3	5	6	0	0	0	0	0	0	0	0	2	0	0	0
Part p_2	0	5	1	3	0	0	0	0	0	2	4	0	-2	-1	0
Part p_3	2	0	0	0	-2	-6	-3	0	0	0	0	0	0	-5	-4

Table 4.3 : Inventory of parts (Results)

These results indicate that : (i) the summation of the entries of each column of table 4.2 equals the corresponding entry in the 5th row of table

4.1, i.e. the sum production of parts equals the production of the family, and (ii) the summation of the entries of each column of table 4.3 equals the corresponding entry in the last row of table 4.1, i.e. the sum inventories of parts equals the inventory of the family. Further, it can be seen from table 4.3, that for each period, the inventory sense of parts is consistent with that of the family. This demonstrates that the algorithm is able to provide a disaggregation that solves the low level problem.

4.5 Optimality of the Hierarchical Approach

In this section, we demonstrate the optimality of the hierarchical approach to the problem (*P4.1*). A rigorous demonstration of the equivalence of the hierarchical and monolithic approaches is presented in Appendix A.4.

Result 5.1 :
The solution of the problem (*P4.1*), followed by (*P4.2*), leads to a value of the criterion which is equal to the value of the optimal solution of problem (*P4.1*).
Proof :
First, we consider the hierarchical approach. Let the optimal solution to problem (*P4.1*), SH, have the value of the corresponding criterion (4.1) equal to C(SH). The solution of problem (*P4.2*), SL, i.e. the disaggregated solution of (*P4.1*), cannot change the value of the criterion. Thus, the hierarchical approach provides a solution, SL, having a value for the criterion (3.1) equal to C(SL), along with C(SL) = C(SH).

Now, let the optimal solution to problem (*P4.1*), S*, have the value of the corresponding criterion (3.1) equal to C(S*). Let us assume C(SL) > C(S*), i.e. the hierarchical approach is sub-optimal. In this case, C(SL) = C(SH) > C(S*). However, S*, can be aggregated (according to the procedure mentioned in the introduction of section 4.4) to result in solution SH*, such that C(SH*) ≤ C(S*) and C(SH) > C(SH*). This implies that SH is not the optimal solution of problem (*P4.1*), which contradicts our assumption. Therefore C(SL) = C(SH) = C(S*).

Q.E.D.

4.6 Continuous Approximation to Planning of Discrete Parts

In section 4.2.2, we introduced two types of production planning : case (A) x_{ik} (production planned for part p_i during the k-th elementary period) $\in$ IR^+, i.e. the production is continuous, and case (B) $x_{ik} \in$ IN, i.e. the production is discrete. Case (A) leads to a real-linear programming problem, which is tractable. Case (B) leads to a mixed integer linear programming problem, which belongs to the class of NP-Complete problems (chapter 3). Unfortunately, production usually relates to discrete parts. Therefore, it would be interesting to approximate it as a continuous problem, i.e., solve case (B) as case (A) followed by a procedure to discretize the real variables. Undoubtedly, the solution time required would be considerably reduced. The question that naturally follows is that what is the sacrifice in optimality? Further, if the solution is not highly sub-optimal, this may be a "good" approximation. In this section, although we do not intend to address the question of sub-optimality of the continuous approximation, we do present an asymptotic proof of optimality of it. In other words, under a given set of conditions, we can demonstrate that the continuous approximation is near optimal.

We define asymptotic conditions as follows :

The number of parts due in a period tend to infinity, $\sum_{i=1}^{n} d_{ik} \rightarrow \infty, \forall k$.

From which we have : (i) $t_{ij} \rightarrow 0, \forall i, \forall j$, which follows from the fact that Δ is finite, and (ii) $w_i^+, w_i^- \rightarrow 0, \forall i$, which is based on the argument that the inventory monetary value is finite.

Result 6.1 :

The continuous approximation to planning of discrete parts is near optimal under asymptotic conditions.

Proof :

First, we consider the continuous formulation. Let the optimal solution, S_c^*, have a corresponding value of the criterion (3.1) equal to $C(S_c^*)$. The

solution obtained by discretizing S_c^* is represented by S_d with the corresponding value of the criterion equal to $C(S_d)$. For discretization, we can adopt (i) truncation, (ii) rounding-off, (iii) higher integer, or a scheme composed of a combination of these; of course this is performed under the machine capacity constraints. We now access the maximal difference between $C(S_c^*)$ and $C(S_d)$ introduced due to this discretization process.

Let $w_{max} = \underset{i}{\text{Max}}\{w_i^+, w_i^-\}$

For each period, the maximal difference introduced due to discretization is: $n \times w_{max}$. Starting from the first period it is carried on for z periods, while from the second it is carried on for z-1 periods, and so on. Thus, for the entire horizon, the maximal difference is : $n \times w_{max} \times z(z+1)/2$. Or, the difference is of the order of $O(n \times w_{max} \times z^2)$; let this be defined as E.

We have the following inequality :

$$C(S_c^*) \leq C(S_d) \leq C(S_c^*) + E \tag{6.1}$$

Under asymptotic conditions, i.e. as $\sum_{i=1}^{n} d_{ik} \to \infty$, $\forall k$, and $w_{max} \to 0$, $E \to 0$ because $n \times z^2$ is a constant. So we can say that $E = \varepsilon$, where $\varepsilon > 0$ and sufficiently small. Alternatively, we imply the following : $\forall\, \varepsilon > 0, \exists\, A_\varepsilon > 0$ such that $\sum_{i=1}^{n} d_{ik} > A_\varepsilon, \forall k, \Rightarrow E \leq \varepsilon$.

In the asymptotic case, we can write (6.1) as :

$$C(S_c^*) \leq C(S_d) \leq C(S_c^*) + \varepsilon \tag{6.2}$$

Now, consider the discrete formulation, and let the optimal solution, S_d^*, have a corresponding value of the criterion equal to $C(S_d^*)$. Since S_d^* is a feasible solution of the continuous problem formulation, we have :

$$C(S_c^*) \leq C(S_d^*) \tag{6.3}$$

Further, we have $C(S_d^*) \leq C(S_d)$, because S_d^* is an optimal solution of the discrete formulation. Thus, the combination of (6.2) and (6.3) leads to :

$$C(S^*_c) \leq C(S_d^*) \leq C(S_d) \leq C(S_c^*) + \varepsilon \tag{6.4}$$

Q.E.D.

4.7 Extensions

In the previous sections, we examined a generic manufacturing system consisting of a set of work-centers, and a set of product types to produce. A two-level hierarchy for planning was designed, to optimize a set of criteria. Under conditions of perfect aggregation, the optimality of this approach was also demonstrated. It is reminded that by "perfect" aggregation, we imply the following :

(I) Parts are aggregated into the same part family if they have identical processing requirements, i.e., they require the same resources with equal processing times, and have equal holding costs and equal backlogging costs.
(II) Machines are aggregated into the same cell if they process the same set of parts with equal processing times.

Such an aggregation scheme, although academically appealing, appears to be over-restricting in a manufacturing environment, and may lead to very little reduction in dimensionality of the planning problem. However, this is not without its significance. We intend to employ the underlying concepts proposed to more general and realistic cases. In a typical manufacturing environment amenable to Group Technology (GT), there are a group of parts which have almost similar characteristics although not exactly the same. On the other hand, there are a group of machines processing similar parts with similar processing times, but not identical. It is, thus, worthwhile to examine the extension of the perfect case to more general cases of practical significance.

First, we propose extensions to the aggregation scheme for parts, i.e. (I). Section 4.8 is devoted to the extension of aggregating parts with similar (but not identical) holding and backlogging costs, assuming that their processing times still remain equal. Section 4.9 is devoted to the extension of aggregating parts with similar processing times on resources, assuming

that the costs remain equal. Finally, section 4.10 is devoted to the extension to the aggregations scheme for resources, i.e. (II).

4.8 Extension to Unequal Costs for Parts

The first relaxation we propose is that parts need not have equal holding costs, and equal backlogging costs. We still emphasize that these costs are "close" enough to each other.

We begin with a perfect aggregation case, and then perturb the cost parameters of some parts to study the aforesaid extension. Let us assume in the perfect aggregation case we have N part families, and the first part family is composed of the first e part types, i.e., $f_1 = \{p_1, p_2, \ldots, p_e\}$. By the definition of perfect aggregation, we have :

(i) $t_{aj} = t_{bj}$; a,b = 1,2,...e, $a \neq b$, and j = 1,2,...,m

(ii) $v_1^+ = w_a^+ = w_b^+$; a,b = 1,2,...e, $a \neq b$,

(iii) $v_1^- = w_a^- = w_b^-$; a,b = 1,2,...e, $a \neq b$.

This is hereby referred to as case ($\mathcal{N}$).

We now perturb the cost parameters of the part type p_e as follows :

(a) $w_e^+ = w_e^+ + \partial w^+$,

(b) $w_e^- = w_e^- + \partial w^-$. Where ∂w^+ and $\partial w^- > 0$, and sufficiently small.

Under the restrictions of perfect aggregation, we will have N+1 part families with the following details : $f_1 = \{p_1, p_2, \ldots, p_{e-1}\}$, $f_{N+1} = \{p_e\}$, and the other families from 2 through N remain unchanged. This is hereby referred to as case ($\mathcal{N}1$).

However, in the perturbed aggregation, which we call case ($\mathcal{P}$), the composition of the families remains the same as before (case ($\mathcal{N}$)), but the cost parameters of the family f_1 change; this change is according to the proportion of parts produced : it is a weighted average of the costs of the part types constructed with the intention that actual costs be well represented by aggregate costs on the average. If r_i represents the production ratio of product type p_i within its family (see equation 8.2), then the cost parameters are as follows :

$$v_1^+ = \sum_{k=1}^{e} r_k w_k^+ ; \tag{8.1a}$$

$$v_1^- = \sum_{k=1}^{e} r_k w_k^- ; \tag{8.1b}$$

If n_i represents the long term production product type p_i, and $p_i \in f_j$, then r_i is defined as follows :

$$r_i = \frac{n_i}{\sum_{a \mid p_a \in f_j} n_a}; \qquad i = 1,2,\ldots,n \tag{8.2}$$

Now, because of the perturbation (a) and (b), and using the definitions (8.1a,b), the perturbed costs are compared with the original costs of ($\mathcal{N}$) as follows :

(a) $v_1^+{}_{(\mathcal{P})} = v_1^+{}_{(\mathcal{N})} + \partial v^+$, (b) $v_1^-{}_{(\mathcal{P})} = v_1^-{}_{(\mathcal{N})} + \partial v^-$.

Where, using (8.1 and 8.2), we can show that $\partial v^{+(-)} = r_e \, \partial w^{+(-)}$.

For instance, if all parts $p_1, p_2, \ldots, p_e$ are produced in the same production ratio (1/e), $\partial v^{+(-)} = \partial w^{+(-)}/e$.

Additionally, in a general case, as $\partial w^{+(-)} \rightarrow 0$, $\partial v^{+(-)} \rightarrow 0$.

In the following sub-sections, we first demonstrate that the optimal solution of ($\mathcal{P}$) is "close" to the optimal solution of ($\mathcal{N}1$). In other words, the optimal solution of the perturbed aggregated problem is near optimal which is obtained by treating the perturbed part as a new family, ($\mathcal{N}1$). Thereafter, we present a revised disaggregation algorithm that provides optimal disaggregation of the high level solution. Finally, we use these results to develop an iterative algorithm which solves the overall planning problem optimally.

4.8.1 Asymptotic Optimality of Extension

It is realized that the perturbed aggregation case, ($\mathcal{P}$), may not lead to an optimal solution of the overall problem. This is because the cost of a

family is only a representative average of the costs of parts composing it. In this section, we demonstrate that the solution of the perturbed aggregation, case ($\mathcal{P}$), does not differ significantly from the optimal solution of ($\mathcal{N}1$). If we observe that the solution is not highly sub-optimal, it may be a "good" approximation. Although we do not intend to address the question of sub-optimality of this approximation, we present an asymptotic proof of optimality. In other words, under a given set of conditions, we can demonstrate that the extended aggregation is near optimal.

Result 8.1 :

The extension to unequal costs for parts is near optimal under asymptotic conditions.

Proof :

First, we consider the original unperturbed problem, ($\mathcal{N}$). Let the optimal solution of problem ($\mathcal{N}$), $S_{\mathcal{N}}^*$, have the value of the criterion equal to $C_{\mathcal{N}}(S_{\mathcal{N}}^*)$. Thus, $C_{\mathcal{N}}(\bullet) \geq C_{\mathcal{N}}(S_{\mathcal{N}}^*)$, where $\bullet$ represents any feasible solution of ($\mathcal{N}$). Let the optimal solution of the perturbed problem ($\mathcal{N}1$), $S_{\mathcal{N}1}^*$, have the value of the criterion equal to $C_{\mathcal{N}1}(S_{\mathcal{N}1}^*)$. We indicate this with $C_{\mathcal{N}1}(\bullet)$ because of the change in the cost parameters. Now, owing to the fact that the corresponding costs are higher (or same) in problem ($\mathcal{N}1$) than in ($\mathcal{N}$), the value of the objective function will be higher (or same) too, i.e. $C_{\mathcal{N}1}(\bullet) \geq C_{\mathcal{N}}(\bullet)$. From which follows that $C_{\mathcal{N}1}(\bullet) \geq C_{\mathcal{N}}(S_{\mathcal{N}}^*)$.

Now, let the optimal solution to the aggregated perturbed approximation ($\mathcal{P}$), $S_{\mathcal{P}}^*$, have the value of the criterion equal to $C_{\mathcal{P}}(S_{\mathcal{P}}^*)$. Once again, we indicate this with $C_{\mathcal{P}}(\bullet)$ because of the use of representative cost parameters. Using similar arguments,we obtain $C_{\mathcal{P}}(\bullet) \geq C_{\mathcal{N}}(\bullet)$. From which follows $C_{\mathcal{P}}(\bullet) \geq C(S_{\mathcal{N}}^*)$. Now, trying to relate $C_{\mathcal{P}}(\bullet)$ to $C_{\mathcal{N}}(\bullet)$ and $C_{\mathcal{N}1}(\bullet)$ to $C_{\mathcal{N}}(\bullet)$, we obtain :

$$C_{\mathcal{P}}(\bullet) \leq C_{\mathcal{N}}(\bullet) + (q \times z \times \partial v^m) \tag{8.3}$$

$$C_{\mathcal{N}1}(\bullet) \leq C_{\mathcal{N}}(\bullet) + (q \times z \times \partial w^m) \tag{8.4}$$

where q is the maximal number of parts of family f_1 that can be produced in one elementary period, $\partial v^m = \text{Max}\{\partial v^+, \partial v^-\}$ and $\partial w^m = \text{Max}\{\partial w^+, \partial w^-\}$. Further, $\partial v^m = r_e\, \partial w^m$.

We define asymptotic conditions as follows :
The perturbation effect is small, i.e. $\partial w^{+(-)} \rightarrow 0$.

However, $\partial v^{+(-)} \rightarrow 0$ as $\partial w^{+(-)} \rightarrow 0$ (for $r_e < 1$). Thus, under asymptotic conditions, $E = q \times z \times \partial w^m \rightarrow 0$ because $q \times z$ is a constant. So we can say that $E = \varepsilon$, where $\varepsilon > 0$ and sufficiently small. Similarly, for (8.4); in the asymptotic case, we have :

$$C_{\mathcal{P}}(\bullet) \leq C_{\mathcal{N}}(\bullet) + r_e\, \varepsilon \tag{8.5}$$

$$C_{\mathcal{N}1}(\bullet) \leq C_{\mathcal{N}}(\bullet) + \varepsilon \tag{8.6}$$

Note that $S_{\mathcal{N}}^*$ remains a feasible solution of $(\mathcal{P})$. So, $C_{\mathcal{P}}(S_{\mathcal{P}}^*) \leq C_{\mathcal{P}}(S_{\mathcal{N}}^*)$. From (8.5), we obtain : $C_{\mathcal{P}}(S_{\mathcal{P}}^*) \leq C_{\mathcal{P}}(S_{\mathcal{N}}^*) \leq C_{\mathcal{N}}(S_{\mathcal{N}}^*) + r_e\, \varepsilon$. But recall that $C_{\mathcal{P}}(S_{\mathcal{P}}^*) \geq C_{\mathcal{N}}(S_{\mathcal{N}}^*)$, indicating that $C_{\mathcal{N}}(S_{\mathcal{N}}^*) \leq C_{\mathcal{P}}(S_{\mathcal{P}}^*) \leq C_{\mathcal{N}}(S_{\mathcal{N}}^*) + r_e\, \varepsilon$.

Also, $S_{\mathcal{N}}^*$ remains a feasible solution of $(\mathcal{N}1)$. So, $C_{\mathcal{N}1}(S_{\mathcal{N}1}^*) \leq C_{\mathcal{N}1}(S_{\mathcal{N}}^*)$. From (8.6), we obtain : $C_{\mathcal{N}1}(S_{\mathcal{N}1}^*) \leq C_{\mathcal{N}1}(S_{\mathcal{N}}^*) \leq C_{\mathcal{N}}(S_{\mathcal{N}}^*) + \varepsilon$. But recall that $C_{\mathcal{N}1}(S_{\mathcal{N}1}^*) \geq C_{\mathcal{N}}(S_{\mathcal{N}}^*)$, indicating that $C_{\mathcal{N}}(S_{\mathcal{N}}^*) \leq C_{\mathcal{N}1}(S_{\mathcal{N}1}^*) \leq C_{\mathcal{N}}(S_{\mathcal{N}}^*) + \varepsilon$. From which we can conclude that the solutions $S_{\mathcal{P}}^*$ and $S_{\mathcal{N}1}^*$ asymptotically lead to the same value of the criterion, i.e. they are identical solutions or alternate optima of $(\mathcal{N}1)$.

Q.E.D.

4.8.2 Revised Low Level Problem

The low level problem consists of production planning for parts, x_{ik}, from the aggregate solution, y_{jk}, of the high level. The disaggregation is performed in such a manner that the high level constraints are respected and the *value of the objective function is minimized.* Note that in this revised case, the disaggregation is required to optimize the criterion, as all parts belonging to the same family do not have equal holding costs or equal backlogging costs; this makes the revised problem more complex than the original one. Thus, when the inventory for the family is positive

(resp. negative), a judicious choice among the part types to be held (resp. backlogged) has to be made. Using the notation introduced earlier, this problem is represented as follows (*P8.1*) :

$$\text{Minimize: Cost(P)} = \sum_{k=1}^{z} \sum_{i=1}^{n} (w_i^+ \times [Q_{ik} - D_{ik}]^+ + w_i^- \times [D_{ik} - Q_{ik}]^+) \quad (8.7)$$

$$\text{Given: } D_{ik} = \sum_{a=1}^{k} d_{ia} ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad (8.8)$$

$$\text{Subject to: } Q_{ik} = \sum_{a=1}^{k} x_{ia} ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad (8.9)$$

$$y_{ik} = \sum_{b \mid p_b \in f_i} x_{bk} ; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \quad (8.10)$$

$$\text{If } U_{ik} - R_{ik} \begin{cases} \geq 0, \text{ then, } Q_{bk} - D_{bk} \geq 0 \ \forall \ b \mid p_b \in f_i \\ \leq 0, \text{ then, } Q_{bk} - D_{bk} \leq 0 \ \forall \ b \mid p_b \in f_i \end{cases} ; \quad \forall i, \text{ and } \forall k \quad (8.11)$$

$$x_{ik} \geq 0 ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad (8.12)$$

Constraint (8.10) ensures that the high level production plan for families is respected. Constraint (8.11) ensures that when the inventory level of a family is non-negative (resp. negative), then the inventory levels of each of the parts belonging to this family are also non-negative (resp. non-positive). This constraint ensures that the low level retains the same inventory sense as at the high level, which is a necessary condition for optimality. Justification of this constraint follows directly from Appendix A.3.

It is important to indicate that despite local optimization, several solutions of equal cost may exist for the disaggregation problem. Thus, it will suffice to find one of these. Solving the linear program is of course one way to obtain the optimal solution of the above problem. However, we will develop a revised disaggregation algorithm that can also optimally solve the above lpp. Section 4.8.4, details this algorithm. In section 4.8.3, we present some important properties which help us develop and prove optimality of the proposed disaggregation algorithm.

4.8.3 Properties

Let I_{sk} denote the inventory level of the family f_s under consideration, at the end of the k-th period. Note $I_{s0} = 0$, i.e. the starting inventory is zero. We characterize the entire evolution of the inventory over the horizon into sections, and for each section we present some properties which will help us compose the overall optimal solution. A section is a sub-sequence of periods.

Property 1:
Consider two consecutive elementary periods k and k+1 such that the sense of the inventory of the family changes as follows : if $I_{sk} > 0$, then $I_{s,k+1} < 0$. The parts held in period k, as well as parts backlogged in period k+1 correspond to parts due in k+1. Thus, if we respect the high level inventory level, construction of the optimal solution of the overall problem can be performed by viewing period k+1 in isolation. This follows directly from Appendix A.3.

The problem consists of finding the specific part types to hold and backlog. Let a_i denote the number of p_i parts to be held in the k-th period, and b_i denote the number of p_i parts to be backlogged in period (k+1). The problem is formulated as the following lpp (*P8.2*) :

$$\text{Minimize:} \sum_{i \mid p_i \in f_s} (w_i^+ \times a_i + w_i^- \times b_i) \tag{8.13}$$

$$\text{Subject to:} \sum_{i \mid p_i \in f_s} a_i = I_k \tag{8.14}$$

$$\sum_{i \mid p_i \in f_s} b_i = I_{k+1} \tag{8.15}$$

$$a_i + b_i \leq d_{i,k+1} \qquad i \mid p_i \in f_s \tag{8.16}$$

$$a_i, b_i \geq 0 ; \qquad i \mid p_i \in f_s \tag{8.17}$$

The objective (8.13) implies minimization of holding and backlogging costs. Constraint (8.14) (resp. 8.15) implies that the total number of part

types held (resp. backlogged) equals the inventory level for the family. Constraint (8.16) ensures that the summation of p_i parts held and backlogged doesn't exceed the demand of p_i at period (k+1). (8.17) reflect the non-negativity constraints.

Property 2:
Consider two consecutive elementary periods k and k+1 such that the sense of the inventory of the family changes as follows : if $I_{sk} < 0$, then $I_{s,k+1} > 0$. The parts held in period k+1, correspond to parts due in some period(s) $j | j > k+1$. And, the parts backlogged in period k, correspond to parts due in some period(s) $j | j < k$. This follows directly from Appendix A.3. The specific parts that correspond to each of these sets are found using properties 3 and 4 to be detailed later.

Property 3:
Consider a sequence of consecutive elementary periods $K^+ = (k,k+1,\ldots,k+j)$ with $I_{s,k-1} \leq 0$, $I_{s,k+j+1} \leq 0$, and $I_{si} > 0$ for $i = k,k+1,\ldots,k+j$. This is the longest sub-sequence of positive inventory values that includes period k. Since the bordering periods of this sequence have non-positive inventory, it follows that in the optimal solution the entire demand due at the end of the periods belonging to K^+ is produced during K^+; we can ignore other periods in the horizon. This follows directly from Appendix A.3.

In this case, we need not solve a lpp. Instead, we can employ the following procedure, which we shall assume without proof, which guarantees optimality (see [57]). If $N = \{1,2,\ldots,n\}$, we define an ordering function O1, $O1: N \rightarrow N$, with $w_{o1(i)}{}^+ \geq w_{o1(g)}{}^+$, $\forall i < g$, $i,g \in N$. That is, we order the parts in decreasing order of holding costs. We start from period k+j, and produce the parts that are due at the end of this period according to the aforesaid ordering, under the high level production constraints. The unsatisfied demand (if any) is added to the previous period (k+j-1) which would give us an updated demand for that period. The process is repeated for each preceding period until we reach period k. Note that periods k+j and k+j+1 constitute the case depicted in property 1. Once we have

determined the specific part types held at period k+j, their demand is added to the demand of part types in period k+j before we start the backward procedure. Finally, if $I_{s,k-1} < 0$, the current planned production in period k provides a partial solution to the case mentioned in property 2.

Property 4:
Consider a sequence of consecutive elementary periods $K^- = (k,k+1,\ldots,k+j)$ with $I_{s,k-1} \geq 0$, $I_{s,k+j+1} \geq 0$, and $I_{si} < 0$ for $i = k,k+1,\ldots,k+j$. This is the longest sub-sequence of negative inventory values that includes period k. Since the bordering periods of this sequence have non-negative inventory, it follows that in the optimal solution the entire demand due at the end of the periods belonging to K^- is produced during K^- or k+j+1; we can ignore other periods in the horizon. Once again, this follows directly from Appendix A.3.

In this case too, we need not solve a lpp. Instead, we can employ the following procedure, which we shall assume without proof, which guarantees optimality. If $N = \{1,2,\ldots,n\}$, we define an ordering function O2, $O2 : N \rightarrow N$, with $w_{o2(i)}^- \geq w_{o2(g)}^- \; \forall i < g$, $i,g \in N$. That is, we order the parts in decreasing order of backlogging costs. We start from period k, and produce the parts that are due at the end of this period according to the aforesaid ordering. The unsatisfied demand is added to the next period (k+1) which would yield an updated demand for that period. The process is repeated for each subsequent period until we reach k+j. The remaining demand, if any, is then produced in period k+j+1, where it can certainly be completed; this provides the remaining partial solution to the case mentioned in property 2. Note that periods k-1 and k constitute the case depicted in property 1. Once we have determined the specific part types backlogged at period k-1, their demand is added to the demand of part types at k before we start the forward procedure.

Property 5:
Consider a sequence of consecutive elementary periods $K^0 = (k,k+1,\ldots k+j)$ with $I_{si} = 0$ for $i = k,k+1,\ldots k+j$. This is the longest sub-sequence of zero

inventory value for the family that includes period k. For periods (k+1,k+2,...,k+j), zero inventory implies that the demand for each part is met exactly, which automatically solves the disaggregation problem for these periods.

Dividing the entire horizon into portions that respect one among the properties 1 through 5, we can determine the optimal solution of the disaggregation problem, and we do not have to formally solve the lpp (*P8.1*).

4.8.4 Disaggregation Algorithm

The algorithm considers one family at a time. At the beginning of the horizon, the inventory of the family is zero. Further, it is claimed that the ending inventory (final inventory at the end of the horizon) will never be positive; it will either be zero or negative (see section 4.4.3). More so, if the ending inventory is negative, it implies that some demand remains unsatisfied.

First, over the entire horizon we identify all the pairs of periods referred to in property 1. More specifically, we are interested in isolating the cases where two consecutive elementary periods k and k+1 are such that the sense of the inventory of the family changes as follows : if $I_{sk} > 0$, then $I_{s,k+1} < 0$. For each of these observations, we solve the lpp (P8.2). This lpp is of very small dimension, and can be solved very fast.

To the left of such a pair of periods lies a case corresponding to either property 2 or property 3. In case 3, we employ the backward pass procedure detailed in property 3 until we reach case 2. Further, to the right of such a pair of periods lies a case corresponding to either property 2 or property 4. In case 4, we employ the forward pass procedure detailed in property 4 until we reach case 2. This bidirectional expansion is performed around each pair which leads to the overall solution on the horizon. On the other hand, occurrences of case 5 can be handled trivially.

The process is repeated for each family. It is interesting to indicate that this disaggregation scheme remains optimal for unequal costs, and does not require assumptions relating to asymptotic proximity of costs. We can take advantage of this property to develop an iterative algorithm, that leads to the global optimality of the production planning problem.

4.8.5 An Iterative Algorithm for Global Optimality

In this section, we present a novel method for aggregate planning at the high level of the hierarchy, as well as the disaggregated solution over the low level horizon. This method is based on an iterative solution scheme that solves a modified version of both the high level problem and the low level problem, repeatedly until convergence. We also prove that this algorithm converges, and convergence occurs at the global optimum for this extention.

The first modification introduced to the high level problem is that the cost attributes associated with part families are no longer the same for all elementary periods of the high level planning horizon. The second modification is the introduction of an additional constraint, the motivation of which is detailed later. The new problem reflecting these changes is presented below (*P8.3*) :

$$\text{Minimize: } \text{Cost}(F) = \sum_{k=1}^{z} \sum_{i=1}^{N} (v_{ik}^{+} \times [U_{ik}-R_{ik}]^{+} + v_{ik}^{-} \times [R_{ik}-U_{ik}]^{+}) \qquad (8.18)$$

R_{ik} denotes the cumulative demand of part family f_i at the end of the k-th period, and is represented as :

$$R_{ik} = \sum_{a=1}^{k} \sum_{b \mid p_b \in f_i} d_{ba} ; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad (8.19)$$

$$\text{Subject to: } U_{ik} = \sum_{a=1}^{k} y_{ia} ; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad (8.20)$$

Where U_{ik} denotes the cumulative production of part family f_i at the end of the k-th period.

$$\sum_{i=1}^{N} \theta_{ij}\, y_{ik} \le \Delta; \qquad j = 1,2,\ldots,M, \text{ and } k = 1,2,\ldots,z \qquad (8.21)$$

$$y_{ik} \ge 0\,; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad (8.22)$$

$$r_{bk}(U_{ik} - R_{ik}) - r_{bk-1}(U_{ik-1} - R_{ik-1}) + d_{bk} \ge 0\,;$$

$$\forall b \mid p_b \in f_i,\ i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad (8.23)$$

Where $v_{ik}^{+(-)}$ represent the holding cost (resp. backlogging cost) of family f_i for the k-th elementary period. r_{bk} represents the proportion of parts of type p_b in the inventory of its family (say f_i, i.e. $p_b \in f_i$) at the end of the k-th period (computed from the previous iteration, $[x_{bk}]$), then :

If $U_{ik} - R_{ik} \neq 0$:

$$r_{bk} = \frac{Q_{bk} - D_{bk}}{U_{ik} - R_{ik}}; \qquad \forall b \mid p_b \in f_i,\ i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad (8.24)$$

$$\text{note :} \quad \sum_{b \mid p_b \in f_i} r_{bk} = 1; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad (8.25)$$

If $U_{ik} - R_{ik} = 0$, we can choose any ratio, $r_{bk} \ge 0$, that respects (8.25).

We also consider the cost parameters as follows :

$$v_{ik}^{+} = \begin{cases} \dfrac{\sum_{b \mid p_b \in f_i} w_b^{+}(Q_{bk} - D_{bk})^{+}}{\sum_{b \mid p_b \in f_i} (Q_{bk} - D_{bk})^{+}} & \text{if } \sum_{b \mid p_b \in f_i} (Q_{bk} - D_{bk})^{+} > 0 \\ \max_{b \mid p_b \in f_i} \{w_b^{+}\} & \text{otherwise} \end{cases} ; \quad (8.26a)$$

$$v_{ik}^{-} = \begin{cases} \dfrac{\sum_{b \mid p_b \in f_i} w_b^{-}(D_{bk} - Q_{bk})^{+}}{\sum_{b \mid p_b \in f_i} (D_{bk} - Q_{bk})^{+}} & \text{if } \sum_{b \mid p_b \in f_i} (D_{bk} - Q_{bk})^{+} > 0 \\ \max_{b \mid p_b \in f_i} \{w_b^{-}\} & \text{otherwise} \end{cases} ; \quad (8.26b)$$

The advantage of using this defintion is that at convergence, the high level criterion equals the low level criterion.

The problem associated with this modification is that the production levels, i.e. x_{ik}, and in turn the production ratios and cost parameters, (r_{ik} and $v_{ik}^{+(-)}$, respectively) are not known *a priori*. Thus, we employ an iterative scheme, which solves the high level problem with an initial guess for $[x_{ik}]^0$, followed by disaggregation to provide a new set of production levels $[x_{ik}]^1$; these are then used to determine new parameters of the high level problem. The process is repeated until convergence is reached, i.e. two consecutive solutions $[x_{ik}]^l$ and $[x_{ik}]^{l+1}$ are reasonably close.

Constraint (8.23) is introduced to ensure that the solution of a high level $[U_{jk}]$, is such that it can still be disaggregated by the ratio of the previous iteration $[r_{ik}]$. This enables us to prove convergence of this algorithm (see section 4.8.5.1).

The low level problem (*P8.1*), presented in section 4.8.2, remains unchanged. However, we present another form of the same problem below, which allows us more visibility into the iterative approach. The problem consists of finding $v_{ik}^{+(-)}$ (*P8.4*) :

$$\text{Minimize: Cost(F)} = \sum_{k=1}^{z} \sum_{i=1}^{N} (v_{ik}^{+} \times [U_{ik}-R_{ik}]^{+} + v_{ik}^{-} \times [R_{ik}-U_{ik}]^{+}) \qquad (8.27)$$

Given : y_{ik}, the production of family i in the k-th period, and

$$U_{ik} = \sum_{a=1}^{k} y_{ia} ; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad (8.28)$$

Subject to :

$$y_{ik} = \sum_{b \mid p_b \in f_i} x_{bk} ; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad (8.29)$$

$$v_{ik}^{+} = \begin{cases} \dfrac{\sum_{b \mid p_b \in f_i} w_b^{+}(X_{bk} - D_{bk})^{+}}{\sum_{b \mid p_b \in f_i} (X_{bk} - D_{bk})^{+}} & \text{if } \sum_{b \mid p_b \in f_i} (X_{bk} - D_{bk})^{+} > 0 \\ \underset{b \mid p_b \in f_i}{\text{Max}} \{w_b^{+}\} & \text{otherwise} \end{cases} \quad ; \text{(8.30a)}$$

$$v_{ik}^{-} = \begin{cases} \dfrac{\sum_{b \mid p_b \in f_i} w_b^{-}(D_{bk} - X_{bk})^{+}}{\sum_{b \mid p_b \in f_i} (D_{bk} - X_{bk})^{+}} & \text{if } \sum_{b \mid p_b \in f_i} (D_{bk} - X_{bk})^{+} > 0 \\ \underset{b \mid p_b \in f_i}{\text{Max}} \{w_b^{-}\} & \text{otherwise} \end{cases} \quad ; \text{(8.30b)}$$

$$Q_{ik} = \sum_{a=1}^{k} x_{ia} \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad (8.31)$$

$$x_{ik} \geq 0 \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad (8.32)$$

This problem can be solved optimally by the algorithm presented in section 4.8.4.

It can now be seen that the iterative approach consists of decomposing the overall problem into two parts. The first part, the high level, consists of determining production levels of the families with given cost attributes. The second part, the low level, consists of disaggregating the high level solution that leads to better cost attributes, (i.e. cost attributes that lead to a reduced value of objective function).

In the next section, we present the convergence of this iterative algorithm, and the motivation of constraint (8.23).

4.8.5.1 Convergence

Consider a feasible solution to the problem $[x_{ik}]^0$, with a value of the criterion given by C_L^0. The subsequent high level solution will have a value of the criterion C_H^1, with $C_H^1 \leq C_L^0$. This is because the aggregated

solution of $[x_{ik}]^0$ is a feasible solution of the high level problem. Although, the subsequent low level also optimizes the same criterion, it is difficult to claim that the subsequent solution C_L^1 verifies $C_L^1 \leq C_H^1$. To overcome this difficulty, we introduce constraint (8.23) which ensures that the solution at the high level can still be disaggregated according to the previous ratios. This implies that the the disaggregation based on the previous ratios, is a feasible solution of the low level problem, ensuring that $C_L^1 \leq C_H^1$. Hence, at each step of the algorithm, the value of the criterion decreases, or at least remains unchanged ($C_L^0 \geq C_H^1 \geq C_L^1 \geq C_H^2 \geq C_L^2$ ...). Since the optimal value of the overall problem is bounded, the algorithm converges (although the rate of convergence is not ascertained).

We further demonstrate that the convergence leads us to the optimal value of the criterion. At convergence, the high level and low level problems are consistently solved, i.e. the high level parameters are computed exactly according to the subsequent low level solution. Thus, we can combine the two problems to replace the cost parameters in (8.18) of (*P8.4*) by the low level definitions (8.30), and y_{ik} can be replaced in (8.20) and (8.21) by x_{bk} using (8.29). The proof that parts of the same family have the same inventory sense has been provided in Appendix A.3. This leads us to the monolithic problem, (*P4.1*). The additional constraint (8.23) is rendered trivial in this case as it is equivalent to non-negativity of production quantities (8.32). This is because, using (8.24) in (8.23), we obtain :

$$\frac{Q_{bk} - D_{bk}}{U_{ik} - R_{ik}} (U_{ik} - R_{ik}) - \frac{Q_{bk-1} - D_{bk-1}}{U_{ik-1} - R_{ik-1}} (U_{ik-1} - R_{ik-1}) + d_{bk} \geq 0$$

or $\quad (Q_{bk} - D_{bk}) - (Q_{bk-1} - D_{bk-1}) + d_{bk} \geq 0$

or $\quad x_{bk} \geq 0.$

Thus, in consistent solutions, the two problems are equivalent to the monolithic problem and the minima obtained correspond to the minima of the monolithic problem.

Q.E.D.

4.8.5.2 Algorithm

To obtain an intial solution :

1. Initialize $v_i^{+(-)} = \sum_{b \mid p_b \in f_i} r_b \, w_b^{+(-)}$, where r_b is the historical production ratio of part p_b within its family.
2. Solve the high level problem (*P4.1*).
3. Disaggregate, using the algorithm of section 4.8.4, to obtain $[x_{ik}]^0$.

Main algorithm :

c = 0; iteration number.

Repeat

c = c + 1;

Solve high level problem (*P8.3*);

Disaggregate using the algorithm of section 4.8.4 to obtain $[x_{ik}]^c$;

Until $([x_{ik}]^{c-1} = [x_{ik}]^c)$

4.9 Extension to Unequal Processing Times for Parts

The second relaxation we propose is that although parts are aggregated into part families if they have equal holding costs and equal backlogging costs, they need not have exactly the same processing requirements (i.e. they use resources with unequal processing times). We still emphasize that these processing times are "close" enough to each other.

We begin with a perfect aggregation case, and then perturb the processing times of some parts to study the proposed extension. Let us assume that in the perfect aggregation case we have N part families, and the first part family is composed of the first e part types, i.e., $f_1 = \{p_1, p_2, \ldots, p_e\}$. By the definition of perfect aggregation, we have :

(i) $t_{aj} = t_{bj}$; a,b = 1,2,...e, $a \neq b$, and j = 1,2,...,m

(ii) $v_1^+ = w_a^+ = w_b^+$; a,b = 1,2,...,e, $a \neq b$,

(iii) $v_1^- = w_a^- = w_b^-$; a,b = 1,2,...,e, $a \neq b$.

This is hereby referred to as case ($\mathcal{N}$).

We now perturb the processing times of the part type p_e as follows :

$t_{ej} = t_{ej} - \partial t_j$, $j = 1,2,\ldots,m$.

Under the restrictions of perfect aggregation, we will now have N+1 part families with the following details : $f_1 = \{p_1, p_2, \cdots, p_{e-1}\}$, $f_{N+1} = \{p_e\}$, and the other families $j = 2,3,\ldots,N$ remain unchanged. This is hereby referred to as case ($\mathcal{N}1$).

However, in the perturbed aggregation, case ($\mathcal{P}$), the composition of the families remains the same as before, but the processing times of the family f_1 change; this change is in accordance to the proportion of parts produced. It is a weighted average of the processing times of the part types constructed, with the intention that, on the average, actual processing times are well represented by aggregate processing times. If r_i represents the production ratio of product type p_i ($i = 1,2,\ldots,e$), then the processing times are as follows :

$$\tau_{1j} = \sum_{k=1}^{e} r_k t_{kj}\,; \qquad j = 1,2,\ldots,m \tag{9.1}$$

Because of the perturbation, and using the definitions (9.1), the perturbed times are compared with the original times of ($\mathcal{N}$) as follows :

$\tau_{1j(\mathcal{P})} = \tau_{1j(\mathcal{N})} - \partial\tau_j$.

Where, using (9.1), we can show that $\partial\tau_j = r_e\, \partial t_j$.

For instance, if all parts $p_1, p_2, \ldots, p_e$ are produced in the same production ratio (1/e), then $\partial\tau_j = \partial t_j / e$.

Additionally, in a general case, as $\partial t_j \to 0$, $\partial\tau_j \to 0$.

In the first sub-section, we demonstrate that the optimal solution of ($\mathcal{P}$) is "close" to the optimal solution of ($\mathcal{N}1$). In other words, the optimal solution of the perturbed aggregated problem is near optimal, which is obtained by treating the perturbed part as a new family, ($\mathcal{N}1$).

Unfortunately, for this extension, we cannot develop an optimal hierarchical algorithm. Thus, a worst case analysis of the hierarchical approach is presented in sub-section 4.2. This section provides us a

formula for the upper-bound of error that can be introduced while employing the hierarchical approach. Sub-section 4.3 is devoted to the revised low level problem formulation, while sub-section 4.4 presents the revised disaggregation algorithm. Finally, in sub-section 4.5, we present an iterative algorithm for improvement, which is based on the ideas of section 4.8.5.

4.9.1 Asymptotic Optimality of Extension

It is not difficult to appreciate that the perturbed aggregation case, ($\mathcal{P}$), may not lead to an optimal solution of the overall problem. This is because the processing times of a family are only representative averages of the processing times of the parts composing it. In this section, we demonstrate that the solution of the perturbed aggregation, case ($\mathcal{P}$), does not differ significantly from the optimal solution of ($\mathcal{N}1$). If we observe that the solution is not highly sub-optimal, this may be considered as a "good" approximation. In this section, we do not intend to address the question of sub-optimality of this approximation, but instead, we present an asymptotic proof of optimality. In other words, under a given set of conditions, we can demonstrate that the extended aggregation is near optimal. The worst case analysis is presented in the subsequent section.

Result 9.1 :

The extension to unequal processing times for parts is near optimal under asymptotic conditions.

Proof :

First, we consider the original unperturbed problem, ($\mathcal{N}$). Let the optimal solution to problem ($\mathcal{N}$), $S_{\mathcal{N}}^*$, have the value of the criterion equal to $C(S_{\mathcal{N}}^*)$. Let the optimal solution to the perturbed problem ($\mathcal{N}1$), $S_{\mathcal{N}1}^*$, have the value of the criterion equal to $C(S_{\mathcal{N}1}^*)$. Owing to the fact that the corresponding processing times are lower (or same) in problem ($\mathcal{N}1$) than in ($\mathcal{N}$), any feasible solution of ($\mathcal{N}$) remains a feasible solution of ($\mathcal{N}1$). From which follows that $C(S_{\mathcal{N}1}^*) \leq C(S_{\mathcal{N}}^*)$.

Now let the optimal solution to the aggregated perturbed approximation ($\mathcal{P}$), $S_{\mathcal{P}}^*$, have the value of the criterion equal to $C(S_{\mathcal{P}}^*)$. By using similar arguments, $C(S_{\mathcal{P}}^*) \leq C(S_{\mathcal{N}}^*)$. Relating $C(S_{\mathcal{N}1}^*)$ to $C(S_{\mathcal{N}}^*)$ and $C(S_{\mathcal{P}}^*)$ to $C(S_{\mathcal{N}}^*)$, we obtain :

$$C(S_{\mathcal{N}}^*) \leq C(S_{\mathcal{N}1}^*) + (q \times z \times w_{max}^- \times \partial t^m) \tag{9.2}$$

$$C(S_{\mathcal{N}}^*) \leq C(S_{\mathcal{P}}^*) + (q \times z \times w_{max}^- \times \partial \tau^m) \tag{9.3}$$

where q is the maximal number of parts of family f_1 that can be produced in one elementary period (in the unperturbed case), $\partial t^m = Max\{\partial t_j\}$, $\partial \tau^m = Max\{\partial \tau_j\}$ and $w_{max}^- = Max\{w_i^-\}$. Further, $\partial \tau^m = r_e\, \partial t^m$.

We define asymptotic conditions as follows :
The perturbation effect is small, i.e. , $\partial t_j \rightarrow 0$, $\forall j$.

However, $\partial \tau^m$ and $\partial t^m \rightarrow 0$ as $\partial t_j \rightarrow 0$, $\forall j$. Thus, under asymptotic conditions, $E = q \times z \times w_{max}^- \times \partial t^m \rightarrow 0$, because $q \times z \times w_{max}^-$ is a constant. So we can say that $E = \varepsilon$, where $\varepsilon > 0$ and sufficiently small. The same logic applies to (9.3); in the asymptotic case, we have :

$$C(S_{\mathcal{N}}^*) \leq C(S_{\mathcal{N}1}^*) + \varepsilon \tag{9.4}$$

$$C(S_{\mathcal{N}}^*) \leq C(S_{\mathcal{P}}^*) + r_e\, \varepsilon \tag{9.5}$$

Relation (9.4), together with $C(S_{\mathcal{N}}^*) \geq C(S_{\mathcal{N}1}^*)$, implies that $C(S_{\mathcal{N}1}^*) \leq C(S_{\mathcal{N}}^*) \leq C(S_{\mathcal{N}1}^*) + \varepsilon$. Also, (9.5), together with $C(S_{\mathcal{N}}^*) \geq C(S_{\mathcal{P}}^*)$, imply that $C(S_{\mathcal{P}}^*) \leq C(S_{\mathcal{N}}^*) \leq C(S_{\mathcal{P}}^*) + r_e\, \varepsilon$. From which we can conclude that the solutions $S_{\mathcal{P}}^*$ and $S_{\mathcal{N}1}^*$ asymptotically lead to the same value of the criterion, i.e. they are identical solutions or alternate optima of ($\mathcal{N}1$).

Q.E.D.

Alternatively, we can perturb processing times of the part type p_r in the other sense : $t_{rj(\mathcal{P})} = t_{rj(\mathcal{N})} + \partial t_j$, $j = 1,2,\ldots,m$. This leads to $\tau_{1j(\mathcal{P})} = \tau_{1j(\mathcal{N})} + \partial \tau_j$. In this case, owing to the increase in processing times, $S_{\mathcal{N}1}^*$ and $S_{\mathcal{P}}^*$ are feasible solutions of ($\mathcal{N}$) which implies that $C(S_{\mathcal{N}1}^*) \geq C(S_{\mathcal{N}}^*)$ and $C(S_{\mathcal{P}}^*) \geq C(S_{\mathcal{N}}^*)$. Relating $C(S_{\mathcal{N}1}^*)$ to $C(S_{\mathcal{N}}^*)$ and $C(S_{\mathcal{P}}^*)$ to $C(S_{\mathcal{N}}^*)$, we obtain :

$$C(S_{\mathcal{N}1}^*) \leq C(S_{\mathcal{N}}^*) + (q \times z \times w_{max}^- \times \partial t^m) \tag{9.6}$$

$$C(S_{\mathcal{P}}^*) \leq C(S_{\mathcal{N}}^*) + (q \times z \times w_{max}^- \times \partial \tau^m) \tag{9.7}$$

which leads us to the same conclusion under asymptotic conditions.

4.9.2 Worst Case Analysis of the Hierarchical Algorithm

In this section, we present the upper bound of error that can be incurred while employing the hierarchical planning algorithm. Such a demonstration is essential in that, when a monolithic approach cannot be employed, we can estimate the potential error introduced by the hierarchical approach. This ascertains the applicability of the hierarchical approach to planning. An example to demonstrate this is also presented.

We first prove two lemmas that will be used in the computation of this upper bound of error.

Lemma 1

If $A + B = C + D$ then $\max(\min(A, C)), \min(B, D)) \leq \max(A, B)$

Proof: We consider two cases: $A \geq C$ and $A < C$.

If $A \geq C$, the condition $A + B = C + D$ leads to $B \leq D$

$\max(\min(A, C), \min(B, D)) = \max(C, B) \leq \max(A, B)$

In a similar way, we can prove that in the case of $A < C$, we also obtain

$\max(\min(A, C), \min(B, D)) \leq \max(A, B)$

Q.E.D.

Lemma 2

$\max[a(A - B), b(B - A)] - \max[a(C - B), b(B - C)] \leq \max[a(A - C), b(C - A)]$

Proof:

$\max[a(A - B), b(B - A)] - \max[a(C - B), b(B - C)]$

$= \max\{ \min[a(A - C), a(A - B) + b(C - B)], \min[b(B - A) + a(B - C), b(C - A)] \}$

Since $a(A - C) + b(C - A) = [a(A - B) + b(C - B)] + [b(B - A) + a(B - C)]$,

together with lemma 1, we have:

$\max[a(A - B), b(B - A)] - \max[a(C - B), b(B - C)] \leq \max[a(A - C), b(C - A)]$

Q.E.D.

Let C (H) denote the criterion value obtained using the hierarchical approach, and C (*) denote the optimal criterion value. C^*_π is the optimal solution of problem π.

If parts that belong to the same family have equal costs, then the criterion value at the high level is the same with the criterion value of the solution obtained using the hierarchical approach. That is:

$$C(H) = \min_{x_{ia}} \sum_{k=1}^{z} \sum_{i=1}^{n} \max \left(w_i^+ \left[\sum_{a=1}^{k} x_{ia} - D_{ik} \right], w_i^- \left[D_{ik} - \sum_{a=1}^{k} x_{ia} \right] \right) \qquad (9.8)$$

subject to:

$$\sum_{i=1}^{N} \theta_{ij} x_{ik} \le \Delta; \qquad j = 1,2,\ldots,M, \text{ and } k = 1,2,\ldots,z \qquad (9.9)$$

If x^*_{ik} denotes the production planned of part i in period k for an optimal solution, we have:

$$E = C(H) - C(*) = \min_{x_{ia}} \sum_{k=1}^{z} \sum_{i=1}^{n} \max \left(w_i^+ \left[\sum_{a=1}^{k} x_{ia} - D_{ik} \right], w_i^- \left[D_{ik} - \sum_{a=1}^{k} x_{ia} \right] \right)$$

$$- \sum_{k=1}^{z} \sum_{i=1}^{n} \max \left(w_i^+ \left[\sum_{a=1}^{k} x^*_{ia} - D_{ik} \right], w_i^- \left[D_{ik} - \sum_{a=1}^{k} x^*_{ia} \right] \right)$$

$$= \min_{x_{ia}} \sum_{k=1}^{z} \sum_{i=1}^{n} \left\{ \max \left(w_i^+ \left[\sum_{a=1}^{k} x_{ia} - D_{ik} \right], w_i^- \left[D_{ik} - \sum_{a=1}^{k} x_{ia} \right] \right) \right.$$

$$\left. - \max \left(w_i^+ \left[\sum_{a=1}^{k} x^*_{ia} - D_{ik} \right], w_i^- \left[D_{ik} - \sum_{a=1}^{k} x^*_{ia} \right] \right) \right\}$$

Subject to (9.9) :

$$\sum_{i=1}^{N} \theta_{ij} x_{ik} \le \Delta; \qquad j = 1,2,\ldots,M, \text{ and } k = 1,2,\ldots,z$$

This is a minimization problem, which we denote as (*P9.1*).

In order to obtain E, we have to solve (*P9.1*) and $E = C^*_{P9.1}$

According to lemma 2, we have:

$$\max (w_i^+ [\sum_{a=1}^{k} x_{ia} - D_{ik}], w_i^- [D_{ik} - \sum_{a=1}^{k} x_{ia}]) - \max (w_i^+ [\sum_{a=1}^{k} x^*_{ia} - D_{ik}], w_i^- [D_{ik} - \sum_{a=1}^{k} x^*_{ia}])$$

$$\leq \max [w_i^+ (\sum_{a=1}^{k} x_{ia} - \sum_{a=1}^{k} x^*_{ia}), w_i^- (\sum_{a=1}^{k} x^*_{ia} - \sum_{a=1}^{k} x_{ia})]$$

$$\leq \sum_{a=1}^{k} \max [w_i^+(x_{ia} - x^*_{ia}), w_i^- (x^*_{ia} - x_{ia})]$$

Therefore, we will have $C^*_{\mathcal{P}9.1} \leq C^*_{\mathcal{P}9.2}$, if we consider problem (*P9.2*) defined as follows:

$$\min_{x_{ia}} \sum_{i=1}^{n} \sum_{k=1}^{z} \sum_{a=1}^{k} \max [w_i^+(x_{ia} - x^*_{ia}), w_i^- (x^*_{ia} - x_{ia})]$$

or, $$\min_{x_{ik}} \sum_{k=1}^{z} (z\text{-}k+1) \sum_{i=1}^{n} \max [w_i^+(x_{ik} - x^*_{ik}), w_i^- (x^*_{ik} - x_{ik})]$$

subject to (9.9).

Indeed, (*P9.2*) is equivalent to (*P9.3*), which is defined as follows :

$$\min_{x_{ik}} \sum_{k=1}^{z} (z\text{-}k+1) \sum_{i=1}^{n} u_{ik}$$

subject to (9.9), and :

$u_{ik} \geq w_i^+(x_{ik} - x^*_{ik})$ $\quad i = 1,2,\ldots,n; k = 1,2,\ldots,z$

$u_{ik} \geq w_i^- (x^*_{ik} - x_{ik})$ $\quad i = 1,2,\ldots,n; k = 1,2,\ldots,z$

Now, under the condition:

$x^*_{ik} \geq x_{ik}$ $\quad i = 1,2,\ldots,n; k = 1,2,\ldots,z$

the latter set to constraints dominate, and we have the following problem (*P9.4*), such that $C^*_{\mathcal{P}9.3} \leq C^*_{\mathcal{P}9.4}$.

$$\min_{x_{ik}} \sum_{k=1}^{z} (z\text{-}k+1) \sum_{i=1}^{n} w_i^- (x^*_{ik} - x_{ik}), \text{ or}$$

$$\min_{x_{ik}} \sum_{k=1}^{z} (z\text{-}k+1) \left[\sum_{i=1}^{n} (w_i^- x^*_{ik}) - \sum_{i=1}^{n} (w_i^- x_{ik})\right]$$

subject to (9.9) and :

$$x^*_{ik} \geq x_{ik} \qquad i = 1,2,\ldots,n;\ k = 1,2,\ldots,z$$

Let ρ_{ij} denote θ_{ij}/w_i^-, then constraint (9.9) can be written as:

$$\sum_{i=1}^{N} \rho_{ij}\, w_i^- x_{ik} \leq \Delta; \qquad j = 1,2,\ldots,M, \text{ and } k = 1,2,\ldots,z \qquad (9.10)$$

which, we can restrict by :

$$\sum_{i=1}^{N} \rho\, w_i^- x_{ik} \leq \Delta; \qquad k = 1,2,\ldots,z \qquad (9.11)$$

where $\rho = \max_{i,j}(\rho_{ij})$

This leads us to another problem (*P9.5*), as follows :

$$\min_{x_{ik}} \sum_{k=1}^{z} (z\text{-}k+1) \left[\sum_{i=1}^{n} (w_i^- x^*_{ik}) - \sum_{i=1}^{n} (w_i^- x_{ik})\right] \qquad (9.12)$$

subject to :

$$\sum_{i=1}^{N} \rho\, w_i^- x_{ik} \leq \Delta; \qquad k = 1,2,\ldots,z \qquad (9.13)$$

$$x^*_{ik} \geq x_{ik} \qquad i = 1,2,\ldots,n;\ k = 1,2,\ldots,z \qquad (9.14)$$

It is clear that $C^*_{P9.4} \leq C^*_{P9.5}$ because any feasible solution of (*P9.5*) is a feasible solution of (*P9.4*). (*P9.5*) can be solved directly.

$$C^*_{P9.5} = \sum_{k=1}^{z} (z\text{-}k+1) \left[\sum_{i=1}^{n} (w_i^- x^*_{ik}) - \min(\Delta/\rho, \sum_{i=1}^{n} w_i^- x^*_{ik})\right] \qquad (9.15)$$

$$= \sum_{k=1}^{z} (z\text{-}k+1) \max(\sum_{i=1}^{n} w_i^- x^*_{ik} - \Delta/\rho, 0) \qquad (9.16)$$

$$= \sum_{k=1}^{z} (z\text{-}k+1) \max\left(\sum_{e=1}^{N} \sum_{i \mid p_i \in f_e} w_i^- x^*_{ik} - \Delta/\rho, 0 \right) \quad (9.17)$$

$$= \sum_{k=1}^{z} (z\text{-}k+1) \max\left(\sum_{e=1}^{N} v_e^- \sum_{i \mid p_i \in f_e} x^*_{ik} - \Delta/\rho, 0 \right) \quad (9.18)$$

As for any solution of the original problem, we have:

$$\sum_{i=1}^{n} t_{ij} x^*_{ik} \le \Delta; \qquad j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \quad (9.19)$$

it is also clear that:

$$\sum_{i \mid p_i \in f_e} t_{ij} x^*_{ik} \le \Delta; \qquad e = 1,2,\ldots,N, j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \quad (9.20)$$

Setting $T_{ej} = \min_{i \mid p_i \in f_e} (t_{ij})$, then :

$$\sum_{i \mid p_i \in f_e} T_{ej} x^*_{ik} \le \Delta; \qquad e = 1,2,\ldots,N, j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \quad (9.21)$$

$$\sum_{i \mid p_i \in f_e} x^*_{ik} \le \Delta / T_{ej}; \qquad e = 1,2,\ldots,N, j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \quad (9.22)$$

Which gives us :

$$\sum_{i \mid p_i \in f_e} x^*_{ik} \le \min_j \left(\frac{\Delta}{T_{ej}} \right) = \frac{\Delta}{\max_j (T_{ej})}; \qquad e = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \quad (9.23)$$

Hence, we have:

$$E \le C^*_{\mathcal{P}9.1} \le C^*_{\mathcal{P}9.2} = C^*_{\mathcal{P}9.3} \le C^*_{\mathcal{P}9.4} \le C^*_{\mathcal{P}9.5} \le \frac{(z+1)z\Delta}{2} \max\left(\sum_{e=1}^{N} (v_e^- \tau_e) - 1/\rho, 0 \right) \quad (9.24)$$

where $\tau_e = \max_j (T_{ej})$

An Example

In this section, we present an example for which the upper bound provided by formula (9.24) is reached.

The example contains two parts and two machines.

$\forall$ k, $d_{1k} = 1$, $d_{2k} = d$; $t_{11} = t_{21} = p$; $t_{12} = p + d\varepsilon$ and $t_{22} = p - \varepsilon$

where $0 < \varepsilon < p$, $\forall$ i, $w_i^- = w_i^+ = 1$, and $\Delta = p(1 + d)$

If we take only one family and one cell, we find:

$$\theta_{ij} = \frac{p + d\varepsilon + dp}{1 + d}$$

With an optimal solution, we obtain:

$$x^*_{1k} = d_{1k} = 1;\ x^*_{2k} = d_{2k} = d \quad \forall\, k = 1,2,\ldots z$$

Which leads to C (*) = 0.

With the hierarchical approach, we have the production planned for the family 1 in period k.

$$y_{1k} = \frac{\Delta}{\theta_{ij}} = \frac{(1 + d)^2 p}{(1 + d)p + d\varepsilon} \text{ and } x_{2k} = d$$

$$x_{1k} = y_{1k} - x_{2k} = \frac{(1 + d)^2 p}{(1 + d)p + d\varepsilon} - d = \frac{(1 + d)p - d^2\varepsilon}{(1 + d)p + d\varepsilon}$$

$$E = C(H) - C(*) = C(H) = \frac{(z+1)\, z}{2}(1 - x_{1k}) = \frac{(1 + d)d\varepsilon}{(1 + d)p + d\varepsilon}\ \frac{(z+1)\, z}{2}$$

From the example, we have $\tau = p$, $\rho = \frac{\theta_{ij}}{w_i^-} = \frac{(1 + d)p + d\varepsilon}{1 + d}$

Applying the formula (9.24) :

$$\frac{(z+1)\, z\, \Delta}{2} \max\left(\left(\frac{w_i^-}{\tau_1} - \frac{1}{\rho}\right), 0\right) = \frac{(z+1)\, z\, \Delta}{2} \frac{d\varepsilon}{p[(1 + d)p + d\varepsilon]}$$

$$\frac{(z+1)\, z}{2} p(1 + d) \frac{d\varepsilon}{p[(1 + d)p + d\varepsilon]} = \frac{(z+1)\, z}{2} \frac{(1 + d)d\varepsilon}{p[(1 + d)p + d\varepsilon]} = E$$

That means that formula (9.24) provides the exact error incurred by employing the hierarchical approach in this case.

4.9.3 Revised Low Level Problem

The low level problem consists of production planning for parts, x_{ik}, from the aggregate solution, y_{jk}, of the high level. In this case the problem becomes more difficult because of the average representation of processing requirements by part families. Depending on the mix of parts belonging to a family, we may have an infeasible disaggregation, i.e. the resource capacities may be violated, or aggregate production may not be satisfied. This is often referred to as an inconsistency of disaggregation. Since the resource capacities are hard constraints, it naturally follows that the high level constraints of the aggregate solution, y_{jk}, may be violated in the disaggregation procedure.

We introduce the concept of a consistent disaggregation as follows. A disaggregation which satisfies the high level aggregate solution, y_{jk}, without backlogging, is called a consistent disaggregation. In other words, the cumulative disaggregate production (of the components of a family) must exceed the cumulative aggregate high level plan (for that family).

The low level problem is thus formulated in such a manner that a *consistent disaggregation* may be sought while the *value of the objective function is minimized.* Note that this low level formulation along with the high level problem is not equivalent to the formulation of the monolithic problem (section 4.3, *P4.1*). This will become clear after the mathematical formulation is presented in this section. However, in an aggregate planning environment corresponding to the high level, where the detailed demand for individual part types is not known for periods in distant future, it seems appropriate to assume such a model. Similar definitions for consistency have been supported for other problems in the literature, e.g. Gabby [28], Erschler *et al* [23]. In our case, the high level model permits backordering; therefore, it requires a different definition of consistent disaggregation. It is also true that such an aggregate problem may not have a feasible disaggregation solution. Erschler *et al* [23] provides necessary and sufficient conditions for the feasible disaggregation of a particular problem, where no backordering is permitted at the high level. To the best of our knowledge, there does not exist any work in disaggregation in which backordering is permitted at the high level. Necessary and sufficient conditions for a consistent disaggregation of such a problem can also be derived. First, we introduce the problem formulation as follows (*P9.6*):

$$\text{Minimize: } \text{Cost}(P) = \sum_{k=1}^{z} \sum_{i=1}^{n} \left(w_i^{+} \times [Q_{ik} - D_{ik}]^{+} + w_i^{-} \times [D_{ik} - Q_{ik}]^{+} \right) \quad (9.25)$$

$$\text{Given: } D_{ik} = \sum_{a=1}^{k} d_{ia} \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad (9.26)$$

$$\text{Subject to: } Q_{ik} = \sum_{a=1}^{k} x_{ia} \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad (9.27)$$

$$U_{jk} \leq \sum_{b \mid p_b \in f_j} Q_{bk} \; ; \qquad j = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad (9.28)$$

$$\text{If } U_{jk} - R_{jk} \begin{cases} \geq 0, \text{ then, } Q_{bk} - D_{bk} \geq 0 \;\forall\; b \mid p_b \in f_j \\ \leq 0, \text{ then, } Q_{bk} - D_{bk} \leq 0 \;\forall\; b \mid p_b \in f_j \end{cases} ; \quad \forall i, \text{ and } \forall k \qquad (9.29)$$

$$\sum_{i=1}^{n} t_{ij}\, x_{ik} \leq \Delta; \qquad j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \qquad (9.30)$$

$$x_{ik} \geq 0 \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad (9.31)$$

Constraint (9.28) is introduced to ensure consistency with the high level production plan for families. It can be observed that the optimization of (9.25) drives the right hand side of the constraint (9.28) as close to the left hand side as possible. In other words, when the inventory level of a family is positive, over-production of the relevant parts will lead to increased holding costs, but this is discouraged by the algorithm. On the other hand, when the inventory level of a family is negative, it would be desireable to over-produce, capacity permiting. Also, whenever possible, (9.28) will be satisfied as a strict equality for all periods; e.g. this can always be obtained in the perfect aggregation case of section 4.4 (which is in fact a condition for optimality). Constraint (9.29) ensures that when the inventory level of a family is non-negative (resp. negative), then the inventory levels of each of the parts belonging to this family are also non-negative (resp. non-positive). This constraint ensures that the low level retains the same inventory sense as the high level; justification of this constraint follows from the perfect case. Constraint (9.30) reflects the capacity constraints for the work-centers. Now we can observe that this low level formulation is in fact the monolithic problem (section 4.3, *P4.1*) along with additional constraints (9.28) and (9.29). However, in an aggregate planning environment corresponding to the high level, where the detailed demand for individual part types is not known for periods in distant future, it seems appropriate to assume such a model.

Solving the linear program is of course one way to obtain the optimal solution of the above problem. However, this is discouraged because it is computationally demanding, and defeats the advantages of the hierarchical approach. Besides, disaggregation is usually performed on a rolling basis only for a few periods and detailed demand for periods in the future may not be available. Thus, an efficient heuristic which can solve the disaggregation problem with sufficient accuracy may be acceptable. We developed a revised disaggregation algorithm that employs the principles of optimality of previous versions to efficiently solve the lpp. Section 4.8.3, details this algorithm. We employ some properties of section 4.7.3, which help us develop the proposed disaggregation algorithm.

4.9.4 Disaggregation Algorithm

The necessary and sufficient conditions for the lpp (*P9.6*) to have a feasible solution are (9.27) through (9.31). There is apparently no quick method to verify these conditions for disaggregation over the entire horizon.

The algorithm consists of two major parts. The first part, an initial assignment procedure, is very similar to the disaggregation algorithm presented in section 4.7.4. In this part, capacity constraints of resources are ignored, and a disaggregation plan is sought, in order to optimize the objective and respect the high level plan, i.e. (9.28) with strict equality. The second part of the algorithm verifies the capacity constraints (9.30). If all constraints are respected, the problem is solved. Else, redistribution of production plan is performed to respect capacity constraints (9.30), as well as to ensure consistency (9.28). This is termed as the redistribution procedure.

4.9.4.1 Initial Assignment Procedure

In this section, we present the initial assignment procedure; all properties referred to in this section correspond to those of section 4.7.3. This procedure considers one family at a time; let f_s be the family under consideration. Once again, we characterize the entire evolution of inventory values over the horizon, into sections corresponding to the

properties of section 4.7.3. Further, we define the normalized costs for parts as follows. For part p_i, let wn_i^+ and wn_i^- represent the normalized holding cost and normalized backlogging cost respectively :

$$wn_i^{+(-)} = \frac{w_i^{+(-)}}{\sqrt{\sum_j t_{ij}^2}} \tag{9.32}$$

These definitions of normalized costs replace the original costs. The reason for doing so is to define a cost measure which eliminates the bias of processing time, i.e. although a part may have a high unit cost, very high processing times may render the cost unit of processing time lower than that of some other part with lower per unit cost. Since parts are operated by more than one machine, the modulus (root square) is used for normalization.

First, we identify all pairs of periods in the horizon referred to in property 1. More specifically, we are interested in isolating the cases where two consecutive elementary periods k and k+1 are such that the sense of the inventory of the family changes as follows : if $I_{sk} > 0$, then $I_{s,k+1} < 0$. For each of these observations, we solve the lpp (*P8.2*). This lpp is of very small dimension, and can be solved very fast.

To the left of such a pair of periods lies a case corresponding to either property 2, or property 3. In case 3, we employ the backward pass procedure detailed in property 3 until we reach case 2. Further, to the right of such a pair of periods lies a case corresponding to either property 2, or property 4. In case 4, we employ the forward pass procedure detailed in property 4 until we reach case 2. This bidirectional expansion is performed around each pair which leads to the overall solution over the entire horizon. Of course, occurrences of case 5 can be handled trivially. Note that normalized costs are employed in these procedures.

The process is repeated for each family. This concludes the initial assignment procedure, that is hoped to provide a "good" starting disaggregated solution.

4.9.4.2 Redistribution Procedure

The redistribution procedure is intended to redistribute the production plan of the initial solution in order to resolve capacity conflicts (if any).

This procedure starts with the last period, and verifies the capacity required by the planned production on each of the resources. If capacities are not exceeded, we proceed to the previous period. However, if the capacities of one or more machines are exceeded, the procedure identifies the set of parts planned during the current period that utilize these machines. Let the set of machines whose capacities are exceeded be represented by $\mathcal{J}$, and the set of corresponding parts by Q. Let E_j represent the additional capacity required of work-center m_j, $m_j \in \mathcal{J}$. The parts belonging to the set Q are arranged in a increasing order of the new normalized holding cost, defined as :

$$cn_i^+ = \frac{w_i^+}{\sqrt{\sum_{j \mid m_j \in \mathcal{J}} t_{ij}^2}} \qquad \forall i \mid p_i \in Q \qquad (9.33)$$

The motivation of this cost measure is similar to the one presented before, i.e. to eliminate the bias of processing time. However, in this case only the bottleneck machines are considered for normalization, as the objective is to reduce the load on each of these machines (the other machines are irrelevant) by introducing minimal holding cost.

We begin with the first part, say p_i, with the lowest normalized holding cost, and we redistribute a demand from the current period, k, to the previous period, k-1, by the following amount :

$\text{Min}\{\forall j \mid m_j \in \mathcal{J}, E_j / t_{ij}, x_{ik}\}$, where x_{ik} represents the production planned for part p_i in the k-th period by the initial assignment procedure.

This amount represents the (minimal) production of part p_i that can be reduced in period k, such that either : (i) one of the overloaded work-centers is relieved of overload, or (ii) the entire production of p_i is

exhausted. Of course in doing so, we ensure that the constraint (9.39), relating to the parts of a family having the same sense of inventory is respected. This can be achieved by switching production amounts between parts of the same family; refer to Appendix A.3 for further details.

Then, the sets $\mathcal{J}$ and/or Q are updated, and the new normalized capacities are calculated. The process is repeated until all work-centers are relieved of their overload, i.e. $\mathcal{J} = \emptyset$.

The procedure then proceeds to the previous period with the updated planned production of parts. Observe that this algorithm is a heuristic and does not guarantee optimality; on the contrary, it may sometimes be unable to reach a solution when the necessary and sufficient conditions for feasibility are satisfied. However, when families are fairly compact, i.e. the component parts of a family do not differ significantly in their processing times, the algorithm is expected to perform well.

Note that even when the extensions related to unequal costs as well as those related to unequal processing times exist at the same time, this algorithm can still be employed.

4.9.5 An Iterative Algorithm for Improvement

In section 4.8.5, we presented an iterative solution scheme for aggregate planning at the high level of the hierarchy, followed by the disaggregated solution over the low level horizon, repeatedly until convergence to the global optimum, with the extention of unequal part costs. The modification introduced to the high level problem was that the cost attributes associated with part families are no longer the same for all elementary periods of the high level planning horizon. Instead, the cost attributes were computed from the detailed low level solution, according to the proportion of parts (belonging to the same family) in inventory.

Based on the similar idea, in this section, we present an iterative solution scheme for the extention of unequal part processing times. The

modification introduced to the high level problem is that the processing time attributes associated with part families are no longer the same for all elementary periods of the high level planning horizon. However, in this case, we have not been able to prove that the iterative scheme converges to the global optimum. Therefore, we employ the scheme in a steepest decent manner, i.e. we repeat iterations until we can improve the criterion by a significant amount (say ε) between two subsequent iterations. Although we cannot comment on the sub-optimality of this procedure, the error is expected to be much lower than that provided by (9.24).

The new problem, called (*P9.7*), is similar to (*P4.1*), with (4.4) replaced by :

$$\sum_{i=1}^{N} \theta_{ijk}\, y_{ik} \le \Delta; \qquad j = 1,2,\ldots,M, \text{ and } k = 1,2,\ldots,z \qquad (9.34)$$

Where θ_{ijk} denotes the processing time for one unit of family f_i on cell c_j for the k-th period. If R_{bk} represents the production ratio of part type p_b within its family (say f_i, i.e. $p_b \in f_i$) for the k-th period, then :

$$R_{bk} = \frac{x_{bk}}{\sum_{a \mid p_a \in f_i} x_{ak}}; \qquad \forall b \mid p_b \in f_i,\ i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad (9.35)$$

$$\text{note :} \quad \sum_{b \mid p_b \in f_i} R_{bk} = 1; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad (9.36)$$

We present the processing time parameters as follows :

$$\theta_{ijk} = \sum_{b \mid p_b \in f_i} R_{bk} t_{ba} \qquad i = 1,2,\ldots,N,\ j = 1,2,\ldots,M, \text{ and } k = 1,2,\ldots,z \qquad (9.37)$$

where, $a \mid m_a \in c_j$.
The advantage of using this defintion is that at convergence, the high level capacity contraints are equivalent to the low level ones.

Once again, the problem associated with this modification is that the production levels, i.e. x_{ik}, and, in turn, the production ratios, are not known *a priori*. Thus, we employ an iterative scheme, which solves the high level problem with an initial estimate of $[x_{ik}]^0$ followed by disaggregation to provide a new set of production levels $[x_{ik}]^1$; which are

then used to determine new parameters of the high level problem. The process is repeated until convergence is reached, i.e. two consecutive solutions $[x_{ik}]^l$ and $[x_{ik}]^{l+1}$ are reasonably close.

The low level problem remains the same as (*P9.6*) presented in section 4.9.3. This problem can be heuristically solved by the algorithm presented in section 4.9.4.

Algorithm

To obtain an intial solution :

1. Initialize $\theta_{ijk} = \sum_{b \mid p_b \in f_i} R_b t_{ba}$, where, $a \mid m_a \in c_j$ and R_b is the historical production ratio of part p_b within its family.
2. Solve the high level problem (*P9.7*).
3. Disaggregate using the algorithm of section 4.9.4 to obtain $[x_{ik}]^0$, and the corresponding value of the criterion equal to C^0.

Main algorithm :

c = 0; iteration number.

Repeat

    c = c + 1;

    Solve the high level problem (*P9.7*);

    Disaggregate using the algorithm of section 4.9.4 to obtain $[x_{ik}]^c$;

    Update θ_{ijk};

Until ($C^{c-1} - C^c > \varepsilon$; where $\varepsilon > 0$ and sufficiently small)

4.10 Extension to Aggregation of Machines

The final relaxation we propose is concerning the way in which machines are aggregated into cells. Recall that in the perfect case, aggregation of machines is performed in a manner, such that we choose to group machines which process the same set of parts with exactly the same processing times in the same cell. Two machines m_a and $m_b \in c_r$ iff : $t_{ia} = t_{ib}$; i = 1,2,...,n. Machines can also be aggregated into cells if they process parts with *similar* processing times. We still emphasize that these processing times must be "close" enough to each other.

More precisely, two machines are grouped in the same cell if the processing times for parts do not differ more than a user defined proximity factor, π. That is, if $m_a \in c_j$, then $m_b \in c_j$ iff :

$| t_{ia} - t_{ib} | \leq \pi, \; i = 1,2,\ldots,n$

Then, the processing time of a part in the cell can be derived from the maximal processing time on the machines that belong to that cell. That is, for i=1,2,...n, j=1,2,...,M :

$$\theta'_{ij} = \underset{b \mid m_b \in c_j}{\text{Max}}\{t_{ib}\}$$

where, θ'_{ij} represents the processing time of one unit of part p_i on cell c_j. In this case we choose the maximal because it defines the throughput rate achievable for the part from the cell in steady state (i.e., long term).

As in the earlier extensions, we could prove asymptotic optimality for this extension too. Furthermore, the disaggregation procedure presented in section 4.9.4 can be employed in this case too.

4.11 General Aggregation

In this section, we consolidate the ideas presented in the previous sections to address the case of general aggregation. By general aggregation, we intend the following :

(I) Parts are aggregated into part families if they have similar processing requirements, i.e., they require resources with similar processing times, and have similar holding costs and similar backlogging costs.

(II) Machines are aggregated in the same cell if they process *families* with similar processing times. Note, this implies that parts are aggregated into families, and the results of this clustering are then used for machine aggregation.

Thus, we require a general clustering scheme for classifying similar parts into families, and machines into cells. A host of clustering methods are available in the literature, which are hierarchical or non-hierarchical, and consider different criteria for grouping. In the following paragraphs, we

present clustering schemes which are consistent with our objectives of aggregation.

Part types are aggregated into part families by a modified version of the K-mean algorithm in cluster analysis (Hartigan 1975). Parts are represented in IR^{m+2} by a point. m axes are used to represent the processing time required by a part on m machines, and in addition, two axes are used to represent the holding cost and backlogging cost, respectively. Then, the K-mean algorithm is employed to determine the clusters or part families. The modification to the K-mean algorithm is that clusters are permitted iff the Euclidean distance from the center of a cluster to its corresponding points does not exceed a user specified parameter, σ. This parameter is helpful in controlling the compactness of clusters. The attributes of the families (i.e., the processing times on the m machines, and holding and backlogging costs) are computed from the center of gravity of the clusters; the "mass" of each point is the production volume/ratio of the corresponding part.

The advantages of aggregating parts into families in this manner allow for some uncertainty in the demand to be absorbed. The aggregate part family can be substituted by any part belonging to this family; this reduces variances, while retaining similar processing durations and holding costs in the schedule. Furthermore, it reduces the level of detail required for future production periods, i.e. forecasts in terms of aggregates are sufficient, and detailed part forecasts are not required.

Aggregation of machines into cells is a simple clustering procedure based on a user specified parameter, π. Details of this clustering were presented in section 4.10. Since, the process of determining clusters, based on this condition of proximity is combinatorial, we can also employ the same modified K-means algorithm presented earlier in the case of parts. In this case, machines are represented in IR^N by a point. N axes are used to represent the processing time required by the N families on a machine. Then, the K-mean algorithm is employed to determine the clusters or cells, provided the Euclidean distance from the center of a cluster to its

corresponding points does not exceed a user specified parameter, σ, and bears a relationship with π; also σ is the radius of the cluster. The processing time of a family on a cell is derived from the maximal processing on the component machines of that cell.

Disaggregation in practice is only required for the first period, after which the horizon is rolled and the high level solution is recomputed. The reasons for doing so are : (i) detailed demand in the future is not known accurately and is subject to change, and (ii) it is computationally expensive to obtain repeatedly a detailed solution at the low level.

The advantages of the generalized disaggregation algorithm (presented in section 4.9.4) which computes the disaggregation over the entire high level horizon are the following : (i) in the absence of detailed demand in the future, the algorithm may continue to consider demand for families, and (ii) it is computationally inexpensive. Observe that, if the detailed demand for future periods is unavailable, and forecasted aggregate demand is being considered, then the redistribution procedure will not identify any capacity violations.

4.11.1 Computation of family processing times on cells

In the approach for general aggregation presented so far, the processing times of a family on a cell was calculated on the basis of the long term bottle-neck machine of that cell for that family. That is, long term production volumes/ratios were employed to adopted for computing the processing times of families on component machines of a cell; the maximal among which would suggest the bottle-neck machine for production of this family. However, one can appreciate that changes in production mix of parts belonging to a family can change the bottle-neck machine. This may introduce some inconsistencies during disaggregation.

In this section, we suggest an alternative approach to calculate the processing times of families on cells. Although this is expected to result in less disaggregation inconsistencies, it may be a rather conservative or

pessimistic estimate. We present this in order that one can appreciate this problem, and we suggest an approach to circumvent it. Nevertheless, we will not adopt this in the later part of this work.

This approach is based on the following concept. If we are specified a range for the production ratio of each product type, the objective is to determine the ratio which will result in the least number of units per unit time while at least one machine is fully occupied (i.e. bottle-necked).
A feasible ratio for the component products of a family f_i is one which satisfies :

$$\sum_{b \mid p_b \in f_i} r_b = 1; \qquad i = 1,2,\ldots,N \tag{11.1}$$

$$lb_b \le r_b \le ub_b; \qquad b \mid p_b \in f_i, \text{ and } i = 1,2,\ldots,N \tag{11.2}$$

The processing time of the family f_i on the cell c_j is for a given ratio is :

$$\theta_{ij}([r]) = \max_{a \mid m_a \in c_j} \{ \sum_{b \mid p_b \in f_i} r_b \, t_{ba} \}; \quad i = 1,2,\ldots,N, \text{ and } j = 1,2,\ldots,M \tag{11.3}$$

The problem consists of determining a ratio such that $\theta_{ij}([r])$ is maximized. Then, for the specified ranges of production ratios, the processing time of the family can be assumed equal to this maximal value.

We present an algorithm that determines the processing time θ_{ij}.
$\theta_{ij} = 0$; rsum = 0;
$r_b = 0, b \mid p_b \in f_i$;
specify : $lb_b \le r_b \le ub_b$; $b \mid p_b \in f_i$;
For $a \mid m_a \in c_j$ do
    Define an ordering, Oi, of parts belonging to family f_i in non-decreasing processing time (on the current machine, m_a) order.
    That is : $t_{oi(b)a} \ge t_{oi(k)a}$, $b < k$, $b, k = 1,2,\ldots,$ card(f_i), and $p_{oi(b)}, p_{oi(k)} \in f_i$.
    For k = 1 to card(f_i) do
        if (rsum + $ub_{oi(k)} < 1$)
            $r_{oi(k)} = ub_{oi(k)}$;
            rsum = rsum + $r_{oi(k)}$;
        else

$r_{oi(k)} = 1 - rsum;$

$rsum = 1;$

break;

end;

if($\theta_{ij} < \sum_{b \mid p_b \in f_i} r_b\, t_{ba}$);

$\theta_{ij} = \sum_{b \mid p_b \in f_i} r_b\, t_{ba}$;

end.

Note, if the range for the production ratio of each product type equals [0,1], then :

$$\theta_{ij} = \max_{a \mid m_a \in c_j,\, b \mid p_b \in f_i} \{t_{ba}\}; \qquad (11.4)$$

4.12 Temporal Aggregation

In the previous sections, we assumed that the elementary periods at the different levels of the hierarchy have the same duration. However, in practice the elementary periods at the higher levels of the hierarchy are longer than those of the lower levels. Thus, along with aggregation of entities (products and machines), aggregation along the time-scale is employed as a means to reduce model complexity. Disaggregation of high level plans under this aggregation scheme the production planned over a larger interval consists of dividing among smaller time intervals. This section is devoted to the issue of temporal aggregation in hierarchical production planning.

The full benefit of temporal aggregation can be sought when accompanied with product aggregation as well; this is because while disaggregating along time, a group of products can share capacity interchangeably among themselves. This can be instrumental in reducing the conflicts introduced due to the temporal disaggregation. However, in the first section, we only consider temporal aggregation for production planning of detailed entities, in order to understand the concept. The triple aggregation case will follow in the subsequent section. We consider the monolithic problem (*P4.1*) with the hierarchical approach.

4.12.1 Hierarchical Approach for Temporal Aggregation

In this section, we present the hierarchical approach to the production planning problem (*P4.1*). The hierarchical approach consists of grouping or aggregating elementary time periods into more aggregate time periods. Thus, at the first step, a cumulative production level is computed at the end of an aggregate time period. This reduces the dimension of the planning problem, and allows planning further into the future.

Unlike the aggregations of parts and spatial aggregation, where we can decide the entities to be aggregated in a group, temporal aggregation follows a regular pattern of grouping (i.e. grouping every 'b' periods), irrespective of the demand occuring within the periods. This makes it difficult to assess optimality or develop the underlying theory. Thus, for theoretical treatment, we consider a "perfect case," where restrictive conditions permitting temporal aggregation are imposed. Later, we can extend these ideas to general cases.

By temporal aggregation in the perfect case we imply the following : Two consecutive elementary periods, s and s+1 are grouped in the same aggregate time period iff : $d_{is} = d_{is+1}$; i = 1,2,...,n. Let us assume that, at the end of the aggregation procedure we have h aggregate time periods. Let b_k represent the number of elementary periods aggregated in the k-th aggregate time period; b_k is an positive integer, i.e. $b_k \in \{1, 2, 3,...\}$.

The hierarchy consists of two levels, (i) the high level that plans production for parts over aggregate time periods, and (ii) the low level that computes the temporal disaggregation of the high level solution to determine the production plan for parts for the elementary time periods.

4.12.1.1 High Level Problem

In this case, the high level problem consists of production planning for parts over a long horizon, H; H is divided into h aggregate time periods (referred to as high level elementary periods, HLEP). The duration of the k-th high level elementary period is $ß_k$, $ß_k = b_k \times \Delta$. Let y_{ik}, $y_{ik} \in IR^+$,

denote the production planned for part p_i during the k-th HLEP; i = 1,2,...,n; k = 1,2,...,h. The production planning problem consists of determining the production, y_{ik}, in order to minimize the total weighted earliness and tardiness costs for parts over H; y_{ik} represents the production of part p_i during the k-th high level elementary period.

Before we deal with the problem formulation, we present the following assumption relating to the high level. We assume that during a high level period, the production rate of each part type remains constant. Thus, the inventory levels for the low level elementary periods belonging to a high level period can be determined based on a linear interpolation between the inventory levels of that high level period (note, this is possible because the demand rate is constant for each high level period). The criterion at the high level, representing the true value of the cost, is then computed with the aforementioned low level inventory levels. For instance, let the initial and final inventory at the k-th high level period for a part p_i be I'_{ik-1} and I'_{ik}, respectively (I'_{ik-1} and I'_{ik} can assume positive or negative values), and let there be b_k low level elementary periods in k. Then, the inventory at the end of the s-th low level elementary period of k, I_{isk}, is given as follows :

$$I_{isk} = I'_{ik-1} + \frac{(I'_{ik} - I'_{ik-1})\, s}{b_k}; \qquad s = 1, 2, \ldots, b_k \qquad (12.1)$$

or,
$$I_{isk} = U_{ik-1} - R_{ik-1} + \frac{\left(y_{ik} - \sum_{s \in k} d_{is}\right) s}{b_k}; \qquad s = 1, 2, \ldots, b_k \qquad (12.2)$$

Where, R_{ik} denotes the cumulative demand of part p_i at the end of the k-th high level elementary period, and is represented as :

$$R_{ik} = \sum_{a=1}^{k} \sum_{s \in \text{ a-th HLEP}} d_{is}; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,h \qquad (12.3)$$

and, U_{ik} denotes the cumulative production of part p_i at the end of the k-th high level elementary period (see 12.5).

The high level problem can be formally stated as the following linear programming problem (*P12.1*) :

$$\text{Minimize : Cost(H)} = \sum_{i=1}^{n} \sum_{k=1}^{h} \sum_{s=1}^{b_k} (w_i^+ \times [I_{isk}]^+ + w_i^- \times [- I_{isk}]^+) \qquad (12.4)$$

$$\text{Subject to: } U_{ik} = \sum_{a=1}^{k} y_{ia} \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,h \qquad (12.5)$$

$$\sum_{i=1}^{n} t_{ij}\, y_{ik} \leq ß_k; \qquad j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,h \qquad (12.6)$$

$$y_{ik} \geq 0 \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,h \qquad (12.7)$$

Constraint (12.6) represents the machine capacity constraints over the duration of the high level elementary periods, and (12.7) implies the non-negativity of production of parts. This problem is of smaller dimension than that of the one presented in the previous section.

4.12.1.2 Low Level Problem

The low level problem consists of production planning for parts, x_{is}, from the aggregate solution, y_{ik}, of the high level; x_{is} represents the production of part type p_i in the s-th low level elementary period. We know that a feasible disaggregation, based on the high level assumption of constant production rate during each high level period (i.e. dividing production equally among low level periods of a high level period), exists. However, an alternate disaggregation may provide even lower costs; this implies that we can relax the assumption (of constant production rate during each high level period) of the high level at the low level. Thus, disaggregation is performed in such a manner that the high level constraints are respected and the *value of the objective function is minimized.* Note that the requirement of respecting high level constraints, related to the production during each high level elementary period, leads to a decomposition at the disaggregation level. Thus, a low level problem is solved over a high level elementary period. The entire low level solution can be obtained when the low level problems are solved period-by-period on H. Thus, for a single high level period (k), the low level problem is represented as follows (*P12.2*) :

$$\text{Minimize: } \text{Cost}(L_k) = \sum_{s \in k} \sum_{i=1}^{n} (w_i^+ \times [Q_{is} - D_{is}]^+ + w_i^- \times [D_{is} - Q_{is}]^+) \quad (12.8)$$

$$\text{Given: } D_{is} = \sum_{a=1}^{s} d_{ia} ; \qquad i = 1,2,\ldots,n, \text{ and } s \in k \quad (12.9)$$

$$\text{Subject to: } Q_{is} = \sum_{a=1}^{s} x_{ia} ; \qquad i = 1,2,\ldots,n, \text{ and } s \in k \quad (12.10)$$

$$y_{ik} = \sum_{s \in k} x_{is} ; \qquad i = 1,2,\ldots,n \quad (12.11)$$

$$\sum_{i=1}^{n} t_{ij} x_{is} \leq \Delta; \qquad j = 1,2,\ldots,m, \text{ and } s \in k \quad (12.12)$$

$$x_{is} \geq 0 ; \qquad i = 1,2,\ldots,n, \text{ and } s \in k \quad (12.13)$$

Constraint (12.11) ensures that the high level production plan for parts is respected. Constraint (12.12) reflects the capacity constraints of machines over each low level elementary period. This is a linear programming problem of rather small dimension, and can be solved very fast.

4.12.2 Hierarchical Approach for Temporal, Part and Spatial Aggregation

In this section, we introduce part and machine aggregation along with the temporal aggregation as a natural extension to the previous section. This is based on the temporal aggregation theory of the previous section, as well as the part and spatial aggregation theory of sections 4.4 through 4.10. This hierarchical approach to the production planning problem (*P4.1*) consists of grouping or aggregating elementary time periods into more aggregate time periods and planning for production of families (aggregate parts) on cells (aggregate machines).

Temporal aggregation in this case implies the following : Two consecutive elementary periods, s and s+1 are grouped in the same aggregate time period iff : $\sum_{j \mid p_j \in f_i} d_{js} = \sum_{j \mid p_j \in f_i} d_{js+1}$; i = 1,2,...,N. In other words, elementary

periods are grouped if the demand for each family is identical. It can be observed that this temporal aggregation scheme is less restrictive than the one presented previously. Let us assume that, at the end of the aggregation procedure, we have h aggregate time periods. Let b_k represent the number of elementary periods aggregated in the k-th aggregate time period; b_k is an positive integer, i.e. $b_k \in \{1, 2, 3,...\}$.

Once again, the hierarchy consists of two levels, (i) the high level that plans production for families on cells over aggregate time periods, and (ii) the low level that computes the triple disaggregation of the high level solution to determine the production plan for parts (on machines) for elementary time periods.

4.12.2.1 High level problem

In this case, the high level problem consists of production planning for families on cells over a long horizon, H; H is divided into h aggregate time periods (high level elementary periods, HLEP). The duration of the k-th high level elementary period is $ß_k$, $ß_k = b_k \times \Delta$. Let y_{ik}, $y_{ik} \in IR^+$, denote the production planned for family f_i during the k-th HLEP; i = 1,2,...,N; k = 1,2,...,h. The production planning problem consists of determining the production, y_{ik}, in order to minimize the total weighted earliness and tardiness costs for families over H; y_{ik} represents the production of family f_i during the k-th high level elementary period.

Once again, we assume that during a high level period, the production rate of each family remains constant, so that inventory levels within high level periods can be linear.

Let the initial and final inventory at the k-th high level period for family f_i be I'_{ik-1} and I'_{ik}, respectively (I'_{ik-1} and I'_{ik} can assume positive or negative values), and let there be b_k low level elementary periods in k. Then, the inventory at the end of the s-th low level elementary period of k, I_{isk}, is given as follows :

$$I_{isk} = I'_{ik-1} + \frac{(I'_{ik} - I'_{ik-1})\, s}{b_k}; \qquad s = 1, 2,..., b_k \qquad (12.14)$$

$$\text{or,} \quad I_{isk} = U_{ik-1} - R_{ik-1} + \frac{\left(y_{ik} - \sum_{s \in k} \sum_{j \mid p_j \in f_i} d_{js}\right)^s}{b_k}; \quad s = 1, 2, \ldots, b_k \qquad (12.15)$$

Where, R_{ik} denotes the cumulative demand of family f_i at the end of the k-th high level elementary period, and is represented as :

$$R_{ik} = \sum_{a=1}^{k} \sum_{s \in a} \sum_{j \mid p_j \in f_i} d_{js}; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,h \qquad (12.16)$$

and, U_{ik} denotes the cumulative production of family f_i at the end of the k-th high level elementary period (see 12.18).

The high level problem can be formally stated as the following linear programming problem (*P12.3*) :

$$\text{Minimize : Cost(H)} = \sum_{i=1}^{n} \sum_{k=1}^{h} \sum_{s=1}^{b_k} (v_{isk}^{+} \times [I_{isk}]^{+} + v_{isk}^{-} \times [-I_{isk}]^{+}) \qquad (12.17)$$

$$\text{Subject to: } U_{ik} = \sum_{a=1}^{k} y_{ia}; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,h \qquad (12.18)$$

$$\sum_{i=1}^{n} \theta_{ijk}\, y_{ik} \leq \beta_k; \qquad j = 1,2,\ldots,M, \text{ and } k = 1,2,\ldots,h \qquad (12.19)$$

$$y_{ik} \geq 0; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,h \qquad (12.20)$$

Constraint (12.19) represents the cell capacity constraints over the duration of the high level elementary periods, and (12.20) implies the non-negativity of production of families. Where $v_{isk}^{+(-)}$ represents the holding cost (resp. backlogging cost) of family f_i for the s-th elementary period of the k-th high level period. These parameters have to be determined iteratively as in section 4.8.

4.12.2.2 Low Level Problem

The low level problem consists of production planning for parts, x_{isk}, from the aggregate solution, y_{ik}, of the high level; x_{isk} represents the production of part type p_i in the s-th low level elementary period which is part of the k-th high level period. In this case, the disaggregation is

performed temporally, and families are disaggregated into parts. If parts are aggregated into families as in the general case (section 4.11), owing to this triple disaggregation, we cannot consider the high level periods in isolation as before (section 4.12.1.2), i.e. we have to consider the entire horizon H.

In this case, disaggregation has to be performed in such a manner that the high level constraints are respected and the *value of the objective function is minimized*. The high level constraints consist of the production levels specified for each family during each high level elementary period.

Thus, the low level problem is represented as follows (*P12.4*) :

$$\text{Min: Cost(L)} = \sum_{i=1}^{n} \sum_{k=1}^{h} \sum_{s=1}^{b_k} (w_i^+ \times [Q_{isk} - D_{isk}]^+ + w_i^- \times [D_{isk} - Q_{isk}]^+) \quad (12.21)$$

$$\text{Given: } D_{isk} = \sum_{a=1}^{g} d_{ia} \; ; \qquad i = 1,2,\ldots,n, \text{ where } g = \sum_{e=1}^{k-1} b_e + s \quad (12.22)$$

$$\text{Subject to: } Q_{isk} = \sum_{e=1}^{k-1} \sum_{a=1}^{b_k} x_{iae} + \sum_{a=1}^{s} x_{iak};$$

$$i = 1,2,\ldots,n,\ k = 1,2,\ldots,h,\ s = 1,2,\ldots,b_k \quad (12.23)$$

$$y_{ik} = \sum_{j \mid p_j \in f_i} \sum_{s=1}^{b_k} x_{jsk} \; ; \qquad i = 1,2,\ldots,N,\ k = 1,2,\ldots,h \quad (12.24)$$

$$\sum_{i=1}^{n} t_{ij}\, x_{is} \le \Delta; \qquad j = 1,2,\ldots,m,\ k = 1,2,\ldots,h,\ s = 1,2,\ldots,b_k \quad (12.25)$$

$$x_{isk} \ge 0 \; ; \qquad i = 1,2,\ldots,n,\ k = 1,2,\ldots,h,\ s = 1,2,\ldots,b_k \quad (12.26)$$

Constraint (12.23) is the cumulative production of a part. Constraint (12.24) ensures that the high level production plan for parts is respected. Constraint (12.25) represents the capacity constraints of machines over each low level elementary period. It is obvious that this linear programming problem is rather complicated. Thus, in the following section, a heuristic algorithm to solve this problem is suggested.

4.12.2.3 Heuristic two-step disaggregation

In this section, we present a heuristic which tries to solve the low level problem with triple disaggregation. The heuristic is based on the previously developed principles of temporal, part and spatial aggregation. Hence, we will not offer any analysis of the performance of this algorithm. Further, as we tend to the perfect aggregation case (section 4.4), we asymptotically approach the optimal while employing this algorithm.

The first step of the algorithm is similar to the temporal disaggregation case. This step relaxes the high level assumption of constant production rate for each part family during each high level period (i.e. of dividing production equally among low level periods of a high level period). Thus, an alternative temporal disaggregation of production of families over low level elementary periods is sought at lower cost. In this procedure, representative cost parameters for the families are employed as follows :

$$v_i^{+(-)} = \sum_{a \mid p_a \in f_i} r_a w_a^{+(-)}; \qquad i = 1,2,\ldots,N \tag{12.27}$$

where r_a is defined as in (8.2). That is $v_i^{+(-)}$ replace $w_i^{+(-)}$ in (12.8) of the low level problem (*P12.2*). Further, θ_{ij} (as defined in section 4.11) replaces t_{ij} in (12.12).

The previous solution, which is similar to the high level solution in the case of part and spatial aggregation, serves as the input to the second step of the algorithm. Thus, the second step of the algorithm is the disaggregation procedure presented in section 4.9.4. A modification is introduced in the procedure, which verifies the production levels for families only at the high level periods, instead of each low level elementary period as computed by the first step. This modification allows for more flexibility in the dual disaggregation procedure and is expected to result in fewer disaggregation inconsistencies. Finally, this heuristic does not guarantee consistency. However, this problem can be addressed by an iterative algorithm, which modifies the processing time and cost

parameters of families for high level periods, as performed in earlier iterative schemes.

Algorithm

We assume we have the solution of the high level problem (*P12.1*), which has been computed by considering $v_i^{+(-)}$ (see 12.27) and θ_{ij} (see section 4.11). This result is the production levels of families over high level elementary periods.

Step 1 :

1.1. Initialize θ_{ij} (see section 4.11), and $v_i^{+(-)}$ (see 12.27).

1.2. Solve the low level problem (*P12.2*).

This results in the production levels of families over low level elementary periods.

Step 2 :

2.1. Disaggregate (for production of part over machines) using the algorithm of section 4.9.4 to obtain x_{ik}.

Note, in this case, consistency is defined as follows :

$$U_{jk} \leq \sum_{a=1}^{k} \sum_{s \in a} \sum_{b \mid p_b \in f_j} x_{bs} ; \qquad j = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,h \qquad (12.28)$$

That is, (12.28) replaces (9.28) in the problem (*P9.6*), or the production levels of families are verified only at each high level elementary period opposed to at each low level period.

In chapter 6, we will present an alternative approach to triple disaggregation, which is based in linear programming. This approach can address general criteria during disaggregation. However, owing to the use of linear programming, disaggregation will be obtained for only a portion of the high level horizon.

Chapter 5

Generalized Hierarchical Planning

Production planning problems differ in nature and complexity. They depend on the manufacturing system under consideration as well as the characteristics of demand. Issues relating to the manufacturing system concern : i) dimensionality, ii) type/characteristics of products and production methods, and iii) characteristics and disruptive stochastic events associated with the resources. Manufacturing systems with a large number of machines and a variety of parts tend to present more complex planning problems. Product characteristics like number of levels in the Bills-of-Materials (BOM), number of component parts types, routings, and overall lead times, have significant impact on the complexity of the planning problem. Type of production, e.g. process, mass, batch, or jobbing production has also relevance in this regard. Furthermore, the randomness of work-center failures and labor absenteeism are some other important factors.

Issues associated with the demand characteristics are : i) dynamics of the demand, e.g. trends, seasonality, ii) stochastic nature of demand, and iii) accuracy, nature and horizon of forecast. Changes in product mix, cyclicity, seasonality and trend of the demand are the demand dynamics that effectuate complexity in the planning process. On the contrary, stable and regular demand does not require sophisticated planning tools. Randomness of demand, uncertainty, frequent order cancellations, changes and expediting also impact on the complexity of the problem. Finally, the length of the horizon over which forecasts are provided, and their accuracy play an important role in planning. Forecasts that are

provided in the form of aggregates require disaggregation. Thus, planning decisions are usually preformed hierarchically.

In view of the above comments, it seems that no single planning architecture can be employed for all planning problems. The architecture of the planning system, the number of levels in the hierarchy, the criteria of relevance, and the control to be applied are problem dependent. This chapter aims at the generalization of the principles established in chapter 4 on a two-level hierarchical decomposition of a specific planning problem. We present here an approach for the design of planning hierarchies.

This chapter is organized as follows. Section 5.1 is devoted to the overview of planning decisions in the context of production planning, and some existing classifications that can be found in practice and in the literature. Section 5.2 is a justification to why production planning is hierarchical, and what are the major issues related in development of such hierarchies. Section 5.3 presents the approach for hierarchical design, and the structural issues associated with it. Section 5.4 is devoted to the underlying decomposition principles employed in obtaining a hierarchy of sub-problems. Based on this structure and these decomposition principles, we begin the design process by detailing the inputs, in section 5.5, followed by the design process, in section 5.6. Section 5.7 is devoted to the operation of the hierarchy, and finally a simplified example is presented in section 5.8 to exemplify the design process.

5.1 Overview of Planning Decisions

Planning decisions vary in terms of time-horizon, scope and focus, and have different impact on the overall functioning of the manufacturing system. Time-horizon refers to the length of time required for a strategy to have effect, or over which decisions are performed. In the context of production planning, a natural classification has been in place : short-, medium-, and long-term (or range) decisions. Short-term operations decisions may have an impact that can be measured in days or even hours. These include decisions regarding purchasing raw-materials, production

and personnel scheduling, control of quality, equipment maintenance functions, short-term inventory control issues, etc. Medium-range decisions are those whose impact can be measured in terms of weeks and months. They include demand forecasting, employment planning decisions, decisions concerning distribution of goods, and setting company targets for inventory and service levels. Aggregate production planning is an important function accomplished at medium-term. This involves the structuring of a general plan for responding to forecasted demand through some combination of work force, output, and inventory loadings. Long-term decisions over a scale measured in years include capacity levels, timing, location and scale of construction of new manufacturing facilities. Introduction of new products or processes, and locating of facilities include long-term decisions. This time-horizon based decomposition of decisions does not assume a rigid structure in different applications. The terms short, medium and long term are merely relative in their time-scale. Depending on the context, they may refer to different functions or decisions as well as time horizons.

Another classification of decisions is proposed by Anthony [5]. This hierarchical taxonomy classifies decisions as: i) strategic, ii) tactical, and iii) operational, based on their horizon and scope, as well as level of information detail, degree of uncertainty and level of management involvement, according to the most common practices in enterprises.

Strategic decisions are defined as the decisions relating to long term marketing and financial policies as well as facilities design. Tactical decisions consist of deciding the work-force and over-time levels, as well as production rates of aggregate products. Typically, this involves aggregate capacity planning. Operational decisions typically concern detailed scheduling of parts, and the assignment of workers to machines.

Abraham, *et al* [2] present a research agenda for models to plan and schedule manufacturing systems. They present a four-level framework, which is an extension of the framework proposed by Maxwell et al [59] and closely resembles that of Hax and Meal [40]. The framework is hierarchical

and presumes a corresponding hierarchical decision-making organization. Level 1 : Manufacturing systems planning, specifies and organizes the manufacturing resources necessary to meet long-term production goals, and results in facilities design. Level 2 : Production planning, takes the facilities design as given, and sets the aggregate production rates, consistent with system capacity, in order to satisfy aggregate demand in an economic manner. It also includes gross determination of reorder intervals for products and parts. Level 3 : Flow planning, is the disaggregation of the production plan. It specifies time flow of batches in a way that is consistent with the aggregate plan as well as resource constraints and demand requirements. Level 4 : Scheduling, is the implementation of the flow plan and results in the real-time sequencing and coordination of the production activities.

In this work, we restrict ourselves to the tactical and some operational decisions as defined by Anthony. That is, strategic issues related to facilities design, and long-term marketing and financial policies, are assumed to be known or given. Furthermore, detailed operational decisions related to scheduling are not addressed. In view of the definition of Abraham et al [2], we address levels 2 through 3 in our work. The strategic decisions can be viewed as an additional adaptation type layer, that can be placed over the highest level of our suggested hierarchy.

5.2 Introduction

Production planning is usually performed hierarchically to address the complexity of global problems. The hierarchical, or specifically the multi-layer hierarchical approach consists of the following. The global problem is broken down into a series of sub-problems which are sequentially solved, in the sense that, the solution of a problem imposes constraints on the problem of the next lower level; the global solution is obtained when all the problems are solved. Thus, the initially intractable problem can be rendered pliable. The principles common to these approaches are that the higher levels are more aggregate, with longer horizons, whereas, the lower levels are more detailed, with shorter horizons.

Another important benefit of hierarchical decision making is when the system is subject to random events and uncertainty. Decision making in uncertainty could lead to frequent recomputations; monolithic models would require the entire problem to be resolved, while the hierarchical approach can gradually absorb random events without the need to resolve higher level problems. This results in large savings in computational burden apart from added stability to the overall control.

While such hierarchical methods have been applied to production systems in the past (see chapter 2), several issues related to them remain unanswered. The essential issues that draw attention in a hierarchical architecture are : (i) the construction of the hierarchy (number of levels along with the models and horizons associated with each level) depending on the manufacturing system and its complexity, (ii) controllability, i.e., the optimal control at a given level leads to a set of constraints defining a non-empty set of feasible controls at the next lower level; thus the system can be directed to a desired state by the top-down constraint propagation procedure, and (iii) consistency. In the case of similar criteria at two levels, consistency means that the optimum of the global problem lies within the set of feasible controls at the lower level generated as a consequence of the high level set of constraints; thus the optimal solution of the individual problems results in the optimal solution of the global problem. In the general case, consistency implies that the criterion chosen at each level should be related to each other and to the global criteria, in such a way that when the individual problems are optimally solved, this results in the optimal solution of the global problem. Figure 5.1 summarizes the ideas of controllability and consistency.

The objective of this chapter is to present a methodology for the construction of such a hierarchical system. The present effort is limited to the construction of hierarchical systems, and the subsequent use of the architecture for decision making. Consistency issues relating to a set of criteria have already been addressed in chapter 4. However, for general and multi-criteria problems, the consistency issue is difficult to

demonstrate. Thus, as long as the system is controllable, the hierarchical approach is considered as a "good heuristic" to otherwise intractable production planning problem.

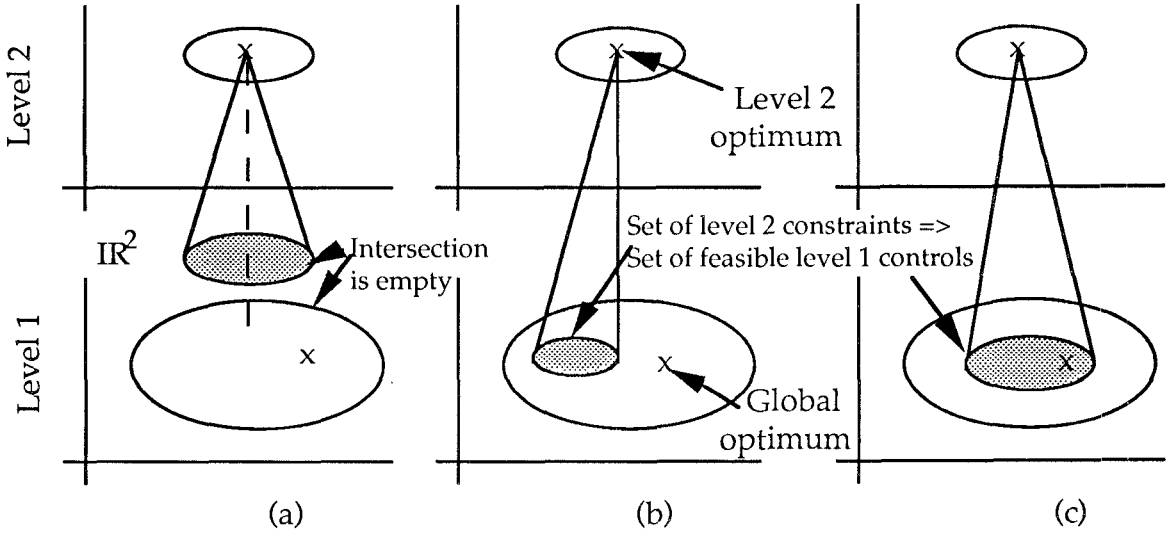


Figure 5.1 (a) uncontrollable, (b) controllable but inconsistent, (c) controllable and consistent

5.3 Approach for Hierarchical Design

Some hierarchical approaches proposed in the literature, suggest a fixed number of levels in the hierarchy for all production systems, irrespective of the complexity and characteristics of the production system. In contrast, this work suggests a generic approach for hierarchical design that depends on the physical characteristics of the production system. In our view the construction of a hierarchy is highly dependent on: (i) the complexity of the production system, (ii) the complexity of the decision making problem (number of variables and number of constraints, etc.), and (iii) frequency and nature of activities and random events. Note, that the decision making problem and the complexity of the production system are highly related. Since the need for a hierarchical approach stems from the intractability of the global problem, it is not difficult to perceive that the number of sub-problems will in fact depend on the overall complexity of the global problem and model. Furthermore, it also depend on some external factors that include the following :

(1) Capabilities of the computing facility (a digital computer) with regard to memory, speed and other characteristics.
(2) Problem solving algorithms.

These factor, relating to the solution method and related tools, are important because a *problem of a given complexity,* needs to be simplified to a different degree while employing different computers and solution algorithms; advanced computers and algorithms requiring not so much simplification (number of levels) as a simple computer and inefficient algorithms. These factors have to be taken into consideration when the problem demands solutions in "real-time." For instance, scheduling, dispatching, or real-time control problems require solutions within their defined time-frames. Solution times are dictated by the hardware and software capabilities, hence require reducing the complexity of these problems. Fortunately, we can assume for planning that decisions permit longer solution times, rendering these factors not so critical. That is, the general solution tools employed for solving planning type problems satisfy the "real-time" aspects of the planning function.

Thus, in general a complex global decision problem, D°, represented by its model M°, can be approached by a n-level hierarchy. The representation of this n-level hierarchical system is presented in figure 5.2.

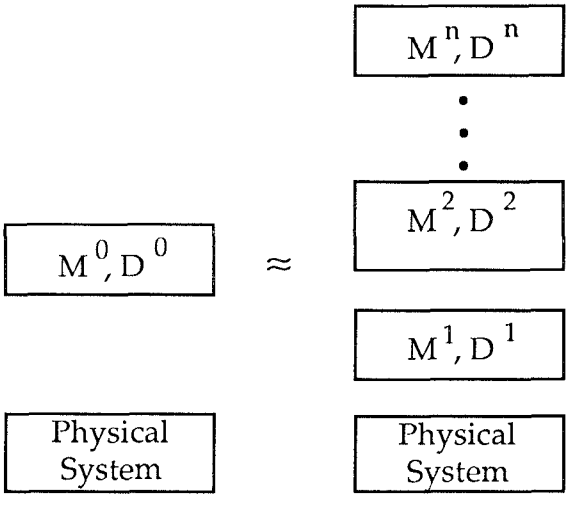


Figure 5.2 : n-level Hierarchical System

In figure 5.2, the hierarchical approach to decision making decomposes the global problem, D°, in a series of sub-problems D^i. Each D^i is solved at level i with a model M^i. The solution to the global problem is obtained when the individual problems, i.e., n through 1 of the hierarchy are solved sequentially.

Before we systematically outline the procedure for constructing a hierarchy, some preliminary remarks are necessary. Each level is associated with a model in order to characterize it, and a decision making problem. Sections 5.3.1 and 5.3.2 outline the structure of the model at each level, and the structure of the decision making problem at each level, respectively.

5.3.1 Structure of a Model at Each Level

A model consists of a set of entities. Each entity is associated with a set of attributes. Each attribute (of an entity) can be assigned a value from a value set associated with it. Entities may be related to each other. An entity is usually an identifiable "object," e.g. machines, parts; it may also be a less perceivable "object," like part families or aggregated machines. A relationship is an association among entities. For instance, an operation in a production process relates two entities : (i) a machine, and (ii) a part. Attributes are used in characterizing an entity, e.g., the contents of a buffer is an attribute of the entity "buffer." Attributes can be qualitative or quantitative variables or constants. The value of the attribute "content" can be assigned a value from the value set V = {1,2,3,4,5}, if the buffer has a maximum capacity of 5 discrete parts.

Using definitions of entity-relation models similar to Chen [15], and Hilger [43], a model can formally be defined as a four-tuple (E, R, A, V) where :

- E is a set of entities $E_i \in E$, where E_i denotes a set of entities with common characteristics. Let e denote an entity. For instance, MACHINES, PARTS are entity sets, while a specific machine is an entity 'e'.

- R is a set of relations $R_k \in R$, where R_k denotes a relationship set which is a mathematical relation among n entities, each taken from an entity set; $e_1 \in E_1, e_2 \in E_2, ..., e_n \in E_n$, i.e., $R_k = \{ [e_1, e_2, ..., e_n] \mid e_1 \in E_1, e_2 \in E_2,..., e_n \in E_n\}$. For instance, OPERATIONS is a relationship set between the set of MACHINES and PARTS.

- A is a set of attributes. An attribute $A_l \in A$ is a function which maps an entity set or a relationship set into a value set or a Cartesian product of value sets. It describes a characteristic of an entity

$A_l : E_k \rightarrow D_l$

or a relation between entities

$A_l : R_k \rightarrow D_l$.

- V is a set of domains $V_i \in V$. Where V_i denotes a value set.

Thus, using this structure, we can model each level of the hierarchy. The actual method to build the models shall be explained in section 5.6.

5.3.2 Structure of a Decision Making Problem at Each Level

Decision making consists of determining a set of optimal controls from a set of feasible controls, such that, the tasks or constraints specified are satisfied over a horizon, and some criteria are optimized. In the case of a unique quantitative criterion, optimizing implies the minimization or maximization of it, depending on the problem at hand. When several criteria are involved, optimizing can adopt several meanings, for instance : (i) minimize/maximize a sum of the weighted criteria, or (ii) allow each criterion to take up a "good value" from its range specified by a "best value" and a "worst value."

We mathematically define the decision making problem D^i as a four-tuple (T^i, U^i, C^i, H^i), where:

- T^i is a set of constraints specified by the upper level, i+1 (strategic constraints in the case of the highest level, n).

- U^i is a set of feasible controls (decisions) that can be applied.

- C^i is a set of criteria or objectives that are to be considered in D^i.

- $H^i \in R^+$, is the horizon of D^i. It is the period of time over which D^i has to be solved.

Decision making can be viewed as an optimization problem that consists of selecting a control X^i, $X^i \in U^i$, such that, the tasks or constraints T^i are satisfied within the horizon H^i, and the criteria C^i are optimized. We then say that X^i is the optimal control, or in other words, X^i is an optimal solution of D^i.

It is important to indicate that the decision making problem D^i and the model M^i are highly related. For example : (i) the criteria relevant to a level should be a function of the attributes of the entities in M^i, (ii) the values of some attributes of some entities should be modifiable within the horizon H^i by the application of a control Y^i, $Y^i \in U^i$, (iii) the value of a criterion should be modifiable within H^i also by the application of a control Y^i, $Y^i \in U^i$, etc.

5.4 Decomposition Principles

In this section, we present the principles that are employed to construct a hierarchy of such sub-problems as a means to global decision making. The decomposition principles help in decomposing the overall problem into sub-problems. Each sub-problem bears a structure of model and structure of decision making problem as presented in the previous section. We base our design of the hierarchy on these principles, which will be presented later in section 5.6.

5.4.1 Introduction to time-scale decomposition

Time-scale decomposition is a technique developed for the analysis of dynamic systems in which different components of the state vector have very different dynamics. In this decomposition, the modes of the system

are partitioned into classes, in such a way that each class is either fast or slow, with respect to the other classes (see Chow and Kokotovic [16], and Sandell et al [68]). The literature in control theory essentially treats multi-level hierarchies. In the multi-layer structures related to hierarchical management systems, the controller is decomposed into algorithms operating at different time intervals. All the layers of the controller directly affect the process but the higher ones control its slower aspects only : they intervene less frequently, with longer optimization horizons, and are based on more aggregate models. The variables manipulated at the higher layers are more aggregate. Unfortunately, this technique has not been developed substantially in the multi-layer literature.

Gershwin (1988) [35], employs the frequency separation principle as the central idea of hierarchical decomposition of production scheduling. As in other multiple time-scale systems, at one end of the scale, there are quantities that are treated as static, and other variables are divided into groups (classes) according to their speed of dynamics. Owing to this grouping, it is claimed that computation of the behavior of these systems is simplified. The essential idea is: when dealing with any dynamic quantity, treat quantities that vary slower as static; and model quantities that vary faster in a way that ignores the details of their variations (represented by averages).

Activities are grouped into sets, such that each set is represented by a characteristic frequency. The characteristic frequency of a group is much less than that of the subsequent lower group, and is much higher than that of the subsequent higher group in the hierarchy.

An important step in this direction is then the specification of the activities to be considered. The most important activities that can be considered are presented as follows.

Machine related activities :

- Failures (of different time-scales)
- Repairs

- Operations
- Set-ups
- Maintenance
- Acquisition

Product related activities :

- Lead-times
- Seasonality in demand
- Sub-contracting products

Worker related activities :

- Absenteeism (of different time-scales)
- Performing operations
- Performing set-ups
- Performing maintenance
- Training
- Hiring and Firing
- Overtime levels

It is important to indicate that an implicit relationship exists between the duration and frequency of an activity; if the duration is long, then usually its frequency is low and vice-versa.

5.4.2 Aggregation Along with Time-Scale Decomposition

In our methodology, we employ similar concepts of time-scale or frequency decomposition, blending it with the aggregation aspects at the higher levels (level ≡ layer) of the hierarchy. This is intended to also address the planning related hierarchy, that requires a broader spectrum of activities (of different time-scales) to be considered. Higher levels of the hierarchy are more aggregate (smaller dimension), allowing for longer horizons and elementary periods associated with them. This enables the higher levels to address slower aspects of the system over longer horizons, which would not have been possible without this model reduction.

Thus, in our multi-layer structure for hierarchical production management systems, the controller is decomposed into algorithms

(levels) operating at different time intervals. Higher levels control the slower aspects of the system (i.e. address activities of longer duration), they intervene less frequently (because they are concerned with less frequent random events), with longer optimization horizons, and are based on more aggregate models. All the layers of the controller directly affect the process, but progressively the lower levels address faster aspects of the system over shorter optimization horizons (and associated elementary periods), while becoming more detailed.

While the design of the hierarchical controller is based on controllable activities (controls) of different time-scales, it is also intended to address uncontrollable activities (e.g., machine failures). These uncontrollable activities, also termed as random events, are of different time-scales too. Each level of the controller treats activities (controllable of uncontrollable) with longer durations (slower) as static, and treats activities that vary faster in a way that ignores their variational details by representing them as averages. Finally, the remaining activities (neither too fast nor too slow) of comparable durations are addressed at a particular level. The controllable activities are planned at this level (controls), while the uncontrollable ones are absorbed. Another way to interpret this is that activities are grouped into sets, such that each set is represented by a characteristic duration. The characteristic duration of a group is much longer than that of the subsequent lower group, and is much shorter than that of the subsequent higher group in the hierarchy; each group of activities is addressed at one level of the hierarchy. As will be shown later, this decomposition of activities or controls also directly impacts on the calculation of the attributes of the entities at the different levels of the hierarchy.

5.5 Inputs to the Design Process

The design of the planning hierarchy requires a variety of information specifying the manufacturing system as well as the managerial goals and decisions. In this section, we present a fairly comprehensive list of characteristics that attempts to encompass a large gamut of manufacturing

and management systems. By this, we do not claim it to be exhaustive. Nonetheless, depending on the nature of the manufacturing systems and assumptions relating to it, most inputs relevant to the planning problem can be chosen among these.

5.5.1 Manufacturing System Details

The details related to the manufacturing system can essentially be classified into the following major heads :

1) Work-center details : These consist of the number and types of work-centers or machines available in the system. If work-centers are prone to failure, then the various failure modes and associated distributions of time between failures and time to repair are of relevance (also see figure 5.3). The other important characteristics can include : (i) maintenance requirements (scheduled and unscheduled), (ii) available capacity (in terms of duration of uptime), and (iii) operating and idle costs per unit time. Addition of new machines into the system can also be permitted, but it is assumed to be a decision that is beyond our framework (it is more of a systems design/strategic issue).

Given several failure modes of the machine, based on their Mean Time To Repair (MTTR), they can be classified along a time-scale as in figure 5.3. They can also be clustered, or classified into groups of failures, where each group is represented by a characteristic MTTR.

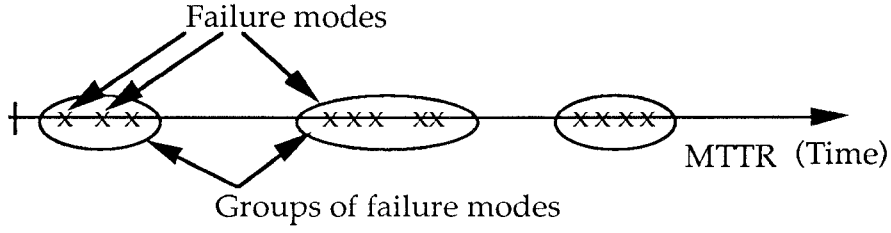


Figure 5.3 : Time-scale classification of various failure modes

2) Worker details : These details are of relevance in the case of labor intensive production. These consist of the number and types (i.e. specific trades) of workers that can be made available in the system. In this case, we

can assume that worker levels can be changed, i.e. workers can be hired and fired. Each worker type has a certain hiring and firing cost and lead time associated with it. Furthermore, each worker type has associated regular and overtime costs per unit of time. If the workers default, then the various default modes and associated distributions (and averages) of time between absenteeism and duration of absence are of relevance. Finally, the set of machines that can be operated by a particular worker type is of concern.

As in the case of machines, based on the mean duration of absence, the worker default modes can be classified along a time-scale. They can also be clustered, or classified into groups of defaults, wherein each group is represented by a characteristic absence duration.

3) Product details : These consist of the set of product types (finished products) that are manufactured in the system. For each product, the Bills-Of-Materials (BOM), identifying the set of semi-finished parts and raw materials should be known. These two sets constitute the manufactured parts in the system, which are hereby simply referred to as parts. For each part, a set of primary and alternative routings or production processes are assumed known (see below). For a unit of each product, the cost for holding it in inventory per unit time (holding cost), and the penalty cost for violating the due-date by a unit time (backlogging cost), are important attributes.

4) Routing details : These consist of the sequence of operations. The set of work-centers the operation can be performed on as well as the set-up and run times associated are important. Further, if the operation is labor intensive one, the labor time required should also be specified.

5) Historical or Forecast data : Historical information in the case of existing manufacturing systems, or forecasted projections for new manufacturing systems, about the long term production requirements (or demand) of products are essential in designing the hierarchy.

Such are the details relating to the manufacturing system that can be of interest to the design of the hierarchy. Since the planning hierarchy remains in place for a relatively long duration, as compared to the production lead-times of products, the design is based on principal choices among alternatives. That is, for designing the hierarchy we employ principal routings, perform preferred assignments of workers to machines, etc. This is in order to reduce complexity, while basing it on long term assumptions relating to the use of the hierarchy for planning. For existing systems, where proportion of assignments to alternative choices is statistically known (e.g. the long term proportion of parts produced using a particular routing among alternatives), we can also employ this information in a weighted manner. Therefore, the details presented so far may take a more simplistic form. Furthermore, from detailed part routings, *representative routings for products* are constructed as follows. The total production time (work-content) of a product at each work-center is accrued for the finished product as well all its dependent components (as indicated in its BOM) and represented in a cumulative manner. Note that in such a representative routing, the sequence of work-centers is immaterial; it only represents the total work-content required of each work-center.

5.5.2 Managerial Inputs

The details related to the managerial inputs can essentially be classified into the following major heads :

1) Controls, U^0 : Controls are decisions that can be made (by a human or the management tool) during the production planning process. As the goal of all production is the manufacture and delivery of products, a natural control is the decision to produce products : what to produce, how much to produce, and when to produce. The most common controls include : (i) decision to sub-contract products is also a relevant control, (ii) hiring or firing of workers, (iii) decisions of overtime levels, (iv) planning maintenance of resources, (v) performing set-ups, or loading parts (note, these controls that are performed more frequently than the controls

mentioned earlier), and (vi) purchasing new resources (however, it may be considered beyond the scope of the planning hierarchy in question).

Each control is associated with a period or duration of time which is elapsed before the results of the control can be observed (or expected). This duration depends on the type of the control, the entities to which the control applies, as well as the speed of the system to accomplish the action. This duration is referred to as the response time of the system to this control. For instance, production of a product has a response time of the order of its cumulative lead time. Hiring of new workers is a control which generally has a longer response time owing to the advertising, interviewing, selecting, and other activities involved. Purchasing a new machine may take even longer. Thus, each control can be assigned an order of magnitude for the response time in the particular manufacturing system under consideration. Note that the controls chosen to be included in the production planning framework should have a response time that fits within the horizon and scope of the framework. This point will be further clarified in section 5.6.

2) Criteria, C^0 : Criteria are the management objectives that have to be optimized during the production planning. They usually take the form of minimization of a cost, or maximizing profits. The criteria referred to here are quantitative in nature. Minimization of following costs can be relevant to planning : (i) production costs (costs related to the utilization of resources : machine and labor), (ii) distribution costs, i.e. cost of transporting finished products to different market locations, (iii) holding or inventory costs, i.e. cost for holding parts in inventory, (iv) backlogging costs, i.e. the penalty for not being able to deliver parts on their due dates, (v) hiring and firing costs, i.e. the costs for employing workers and laying them off, respectively, (vi) regular and overtime labor costs, and (vii) set-up costs, i.e. cost for changing the configuration of the system in order to produce another set of parts. Maximizing profits is accomplished by maximizing the difference between total revenues and total costs. The revenues are obtained by selling products (possibly with different price ranges) to different markets.

It is important to indicate that the criteria to be considered in the planning problem bear a strong relationship with the controls involved. For instance, if hiring and firing of workers is a control, then minimization of the corresponding costs should be criteria to be considered in production planning. However, if the worker level is assumed to be fixed in the problem, the hiring and firing costs are of no significance.

5.6 Design of the Planning Hierarchy

In this section, we present the design procedure for the construction of a planning hierarchy. This design procedure is attempted to be presented in an algorithmic form, but it requires significant human interaction and decision making. Thus, it cannot be viewed as an entirely automatic process. This construction is performed in a bottom-up manner, because the physical system is the only detailed information available. The basic design approach is first presented conceptually. Later, we will present a more precise and comprehensive design algorithm.

We begin at the bottom level (level 1), and consider the entities, which are physical entities, e.g. products, machines, and workers. Therefore, the model is the physical one (and known), and we need to formulate the decision making problem. As mentioned earlier, the end result of this planning hierarchy is to determine the types and number of products to produce in each elementary period of the level 1 horizon. Thus, the natural direction to be adopted is to select the appropriate elementary period, and the horizon length. The elementary period is determined based on several factors that include product lead times, intervals between shipment and inventory updates, to mention a few. For instance, if the products have lead-times of the order of minutes or hours, and the inventory is updated daily, shipments are performed and orders are accepted daily, then a day is an appropriate elementary period. Since the elementary periods at this level are usually of a fairly short duration, not all controls can be effectuated during this. Thus, only a set of controls that have a response time at least an order of magnitude less than the length of

the elementary period are considered. This elementary period is also intended to absorb random events having a response time of the order of the controls, i.e. an order of magnitude less than the length of the elementary period. Random events of much shorter response duration are considered as averages, while those of much longer response duration are considered in their actual state (and are absorbed at some other higher level of the hierarchy). This point will be further illustrated in the operation of the hierarchy (section 5.7). Note that capacities of resources at each level is computed by subtracting from the available time, a summation of the expected durations of non-productive activities (random events like failures) tying up the resources and having mean durations that are much less than the duration of the current elementary period.

The set of controls to be considered defines the complexity of the decision making problem (number of variables and constraints), after which the length of the horizon can be ascertained; this is ascertained either by the complexity of the problem or the solution time permissible. Thus, owing to the detailed nature and large dimension of level 1, its horizon is limited by the complexity of the problem. Finally, the criteria to be considered are determined based on the controls at this level. This completes the definition of the decision making problem at the bottom level. Note that at this time the next higher level is not known, so the high level constraints to this problem are not know exactly although their general form is understood.

The need to address planning at a longer horizon than the current one, and to consider all possible controls requires the development of higher levels. The essential idea is to reduce the dimension and the complexity of the current model by temporal and entity aggregation. We employ the theory developed in chapter 4 to accomplish these aggregations, in a manner that is consistent with the criteria at the bottom level. That is, attributes of product entities that bear relevance to the criterion under consideration, as well as some other related attributes are considered in the aggregation. For example, if the criteria considered are inventory and

backlogging costs, the attributes that bear relevance to these criteria are the holding cost and backlogging cost per unit product and unit time. In addition, attributes relating to the processing time on each machine are also considered. The resources are also aggregated, and their capacities are represented to reflect the absorption of the random events at level 1 as well. After aggregation, the attributes of the newly derived entities are computed. The precise details of these computations for an example are presented in the next chapter. Here, we allude to computation of some attributes of aggregate product entities as follows : (i) the holding cost and backlogging costs for are computed as in (12.27) of chapter 4, (ii) the processing times are computed as presented in section 3.11, and (iii) the long term production quantities are the summation of the long term production quantities of the component products. In this manner, we develop the model at the next upper level (level 2).

Having determined the model of the level under consideration, we need to define the decision making problem. Once again, we select the appropriate elementary period, and the horizon. The elementary period is based on the following guide-lines : it is usually a multiple of the lower level elementary period, and is no greater than the horizon at the lower level. Since the elementary periods at this level are of longer duration than those of the lower level, some additional controls can be effectuated during this. The earlier (lower level) controls are also considered, but now for the aggregate entities. Once again, the entire set of selected controls should have a response time at least an order of magnitude less than the length of the current elementary period. Furthermore, since the elementary period at this level is longer than that of the lower level, some additional random events (with a response time an order of magnitude less than the length of the current elementary period, but greater than the response times of previously considered random events) can be absorbed. Random events of much shorter response duration are considered as averages, while those of much larger response duration are considered in their actual state (and are absorbed at some other higher level of the hierarchy).

The set of controls to be considered then defines the complexity of the decision making problem (number of variables and constraints), after which the length of the horizon can be ascertained; this is ascertained by the complexity of the problem and the solution time permissible. Thus, owing to the reduced dimension at this level, the horizon is longer than that of the lower level. Finally, the criteria to be considered are determined based on the controls at this level; notice that, the form of the criteria now refer to the aggregate entities. This completes the definition of the decision making problem (DMP) at the current level.

The subsequent higher levels are designed in a similar fashion, i.e. aggregating and defining the model followed by defining the decision making problem. The process is repeated until the following conditions are obtained : (i) the desired horizon is obtained, (ii) the desired degree of aggregation is accomplished, (iii) all controls have been addressed, and (iv) all criteria have been addressed. Note that based on this design procedure, the controls as well as the criteria automatically assume a hierarchy among themselves. That will become clear in the later part of this section.

5.6.1 D^1 : Decision Making Problem at Level 1

In this section, we formally present an algorithmic form for the development of the decision making problem at the lowest level of the hierarchy. This development is in the form of rules or guide-lines and often the exact choice may have to be made by the designer, based on physical or practical considerations.

5.6.1.1 Elementary Period, Δ^1

- Δ^1 > lead time for products.
- $\Delta^1 \cong$ time interval after which orders are delivered to stock, shipments are made and inventory is updated.
- $\Delta^1 >>$ mean time for an important class of resource failures. For example, the group of failure modes to the extreme left of the time-scale in figure 5.3.

5.6.1.2 Controls, U^1

1 Production levels for products in elementary periods of duration Δ^1 .

2 Any other control u, $u \in U^0$, such that the response time of $u < \Delta^1$; preferably an order of magnitude less than Δ^1.

5.6.1.3 Criteria, C^1

1 Holding and backlogging cost.

2 Any other criteria c, $c \in C^0$, which are related to the set of controls U^1, or that undergo a change of their value due to the application of some control in U^1.

5.6.1.4 Constraints, T^1

The intrinsic constraints, e.g. resource capacities and technological constraints are known because the model M^1 has been defined and the controls U^1 are known. However, the exact higher level constraints T^1, are not known because the high level remains undefined at this time. Fortunately, the general form of these constraints can be predicted. These constraints are aggregate production constraints related to the production levels of aggregate products (families) that must be respected by the current level during disaggregation. Furthermore, aggregate labor resource levels may also represent high level constraints.

Thus, although the decision making problem is not defined precisely, its general form and complexity over an elementary period can be assessed. The next step consists of determining the length of the horizon H^1, or the number of elementary periods Δ^1 that can be included in the horizon H^1.

5.6.1.5 Horizon, H^1

Knowing U^1, and the model M^1, the complexity of the DMP is a function of H^1/Δ^1 (number of elementary periods in the horizon). Therefore, it should be chosen based on either the memory limitations of the computer, or the solution time requirements (which is << lead time of products), whichever is more constraining.

5.6.2 M^2 : Model at Level 2

In this section, we present the procedure to develop the model at level 2. Since this development procedure is similar for all subsequent levels too, we adopt a generic approach of developing a model at level i from the model at level i-1 and the decision making problem at level i-1. We refer to the product related entities like detailed products or product families, at a level i as product entities as level i. Similar references are made for machine related entities and worker related entities.

5.6.2.1 Aggregation of Product Entities

Product entities at level i-1 are grouped into more aggregate product entities at level i with the help of the modified K-mean clustering algorithm (see chapter 4). Lower level (i-1) entities are represented in IR^x by a point. The x axes are used to represent those attributes of the entities that bear relevance to the criteria at the lower level (i-1), as well as some other related attributes. For example, if the criteria considered are inventory and back-ordering costs, the attributes that bear relevance to these criteria are the holding cost and backlogging cost per unit part. In addition, attributes relating to the processing time on each machine are also considered. By choosing the parameter σ^i, in the modified K-mean algorithm, the compactness of clusters can be controlled. At any level i, almost always processing times on resources (detailed or aggregate) are concerned attributes; other attributes holding and backlogging costs per unit, production costs per unit, seasonal demand pattern, etc.

Thus, product entities at level i are formed from those at level i-1. Thereafter, the attributes of these newly formed entities have to be computed. Since these entities represent a collection of (more detailed) constituent entities, their attributes will depend on the ratio of the constituents. At the design stage, we employ historical or forecasted production ratio of constituents (see item 5 of section 5.5.1) to define the static or long term attributes of these aggregates. For example, the processing time of a product family on a machine is the weighted average of processing times of its constituent products on that machine; the weight

is the long term production volume of products (or their ratio). If r_b denotes the long term production ratio of product entity 'b' at level i-1 within its parent product entity 'a' at level i, then the holding and backlogging costs ($v_a^{+(-)}$) of the parent product entity is :

$$v_a^{+(-)} = \sum_{b \mid p_b \in f_a} r_b w_b^{+(-)}; \qquad a = 1,2,\ldots,N \qquad (6.1)$$

The processing times (θ'_{aj}) of the parent product entity on level i-1 machine entities (m_j) are :

$$\theta'_{aj} = \sum_{b \mid p_b \in f_a} r_b t_{bj}; \qquad a = 1,2,\ldots,N, j = 1,\ldots,m \qquad (6.2)$$

Finally, if n_b denotes the long term production volume of product entity p_b at level i-1, the long term production volume of the parent product entity (nf_a) is :

$$nf_a = \sum_{b \mid p_b \in f_a} n_b; \qquad a = 1,2,\ldots,N \qquad (6.3)$$

5.6.2.2 Aggregation of Machine Entities

As explained earlier, after the aggregation of product entities for level i, the processing times of the newly formed entities are computed on the machine entities of the level i-1 (see 6.2). Based on these processing times, the machine entities at level i-1 are aggregated according to the procedure mentioned in section 3.10.

Alternatively, we can employ the modified K-means clustering algorithm (see chapter 4) for grouping machines. Lower level (i-1) machine entities are represented in IR^x by a point. The x axes are used to represent the processing times on each part entity of level i (N axes in the case of M^2, where N denotes the number of families after aggregation of level 1 products). In addition, attributes related to the failure characteristics of machines can also be included. If machines have different failure characteristics, these attributes are chosen, else they are ignored. By choosing the parameter π^i (note, π^i signifies in a way, the radius of the clusters), in the modified K-mean algorithm, the compactness of clusters can be controlled. By this we mean that machine entities at level i-1 can be

aggregated into machine entities at level i, if they process product entities (of level i) with similar processing times. More precisely, two machines (or cells) are grouped in the same cell (or aggregate cell) if the processing times for products (or aggregate products) do not differ more than a user defined proximity factor. Then, the processing times (θ_{aj}) of product entities at level i on the newly formed machine entities (c_j) at level i can be derived from the maximal processing time of product entities at level i on the former machine entities at level i-1 (θ'_{ab}) as follows :

$$\theta_{aj} = \max_{b \mid m_b \in c_j} \{\theta'_{ab}\}; \qquad a = 1,2,\ldots,N, \text{ and } j = 1,2,\ldots,M \qquad (6.4)$$

Finally, the attributes of the aggregate machine entities at a level i have to be calculated. The capacities, as explained in the introduction of section 5.6, are computed by subtracting from the available time, a summation of the expected durations of non-productive activities (random events like failures) tying up the resources and having mean durations that are much less than the duration of the current elementary period at level i. Note, because of aggregation, the capacity of the aggregate machine entity is now a function of the repair state of a set of its constituents, i.e. all machines belonging to a cell should be available for the cell to be able to process parts. However, under steady state conditions (which we assume at all planning levels), existence of alternative choices, and sufficient in-process buffers, we can assume independence among the failures of the machines, i.e. the capacity can be represented by the bottleneck machine in a cell. Although this is a stringent assumption, it has the advantage of simplicity. A more complex representation of cell capacity could be determined based on the joint distributions of the failure modes (having MTTR less than the duration of the current elementary period) of the constituent machines. Thus, the choice of representation is based on the assumptions relating to the specific manufacturing system, and we defer a precise mathematical representation of this until chapter 6. The operating and idle costs of the aggregate machine entities reflect the summation of the corresponding costs of the component machine entities. This has been concluded on basis of the assumption that all machines are used by each part that visits that cell.

5.6.2.3 Aggregation of Worker Entities

For labor intensive work-centers or production systems, the aggregation of workers into aggregate worker entities is performed in the hierarchy in a manner similar to that of machines. That is, the workers assigned to the work-centers belonging to a cell are grouped to yield aggregate worker entities. The attributes of the aggregate worker entities are calculated as follows. The hiring, firing, regular and overtime costs of the aggregate worker entities reflect the summation of the corresponding costs of the component worker entities. As in the case of machine entities, this has been concluded on basis of the assumption that all workers of an aggregate operate on each part that is processed by that aggregate. The failure modes are computed as in the case of machine entities.

This completes the modeling at level i. Note that the aggregation procedures employed in developing this model are parameterized. Appropriate values of these parameters must be chosen by the designer based on the clustering results, possibly evaluated in the light of physical or practical considerations.

Finally, after the definition of the entities at the level i, the high level constraints for the level i-1, T^{i-1}, can be formulated. Thus, the decision making problem at the level i-1, D^{i-1}, can be completed at this stage.

5.6.3 D^2 : Decision Making Problem at Level 2

In this section, we present the development procedure of the decision making problem (DMP) at the level 2 of the hierarchy. Since this development procedure is similar for all subsequent levels too, we adopt a generic approach of developing a DMP at level i from the model at level i-1 and the DMP at level i-1. Once again, this development is in the form of rules or guide-lines and often the exact choice may have to be made by the designer based on physical or practical considerations.

5.6.3.1 Elementary Period, Δ^i

- Δ^i is a multiple of Δ^{i-1}.

- $\Delta^i \leq H^{i-1}$.
- $\Delta^i <$ response time of some control u, $u \in U^0 \setminus \bigcup_{j=1}^{i-1} U^j$, i.e. some control having a response time less than Δ^i and has not been addressed as yet.

5.6.3.2 Controls, U^i

1 Production levels for product entities at level i on machine entities at level i for elementary periods of duration Δ^i.

2 Any other control, u, $u \in U^0 \setminus \bigcup_{j=1}^{i-1} U^j$, such that the response time of u $< \Delta^i$; preferable an order of magnitude less than Δ^i.

5.6.3.3 Criteria, C^i

1 Holding and backlogging cost for product entities at level i.

2 Any other criteria c, $c \in C^0$, which are related to the set of controls $\bigcup_{j=1}^{i-1} U^j$, or that undergo a change of their value due to the application of some control in $\bigcup_{j=1}^{i-1} U^j$. Note that if the value of some criterion can be expressed as an average over the horizon, because the corresponding control is performed frequently over the duration of the horizon, it is ignored. In this way, some criteria can also be dropped as we proceed up the hierarchy.

5.6.3.4 Constraints, T^i

Once again, the intrinsic constraints, e.g. resource capacities and technological constraints are known because the model M^i has been defined and the controls U^i are known. However, the exact higher level constraints T^i, are not known because the high level remains undefined at this time. The general form of these constraints can be predicted. These constraints are aggregate production constraints related to the production

levels of aggregate products (families) that must be respected by the current level during the disaggregation process. Furthermore, aggregate resource capacity levels may also represent high level constraints. Thus, although the decision making problem is not defined precisely, its general form and complexity over an elementary period can be assessed.

5.6.3.5 Horizon, H^i

- $H^i > H^{i-1}$.
- Knowing U^i, and the model M^i, the complexity of the DMP is a function of H^i/Δ^i (number of elementary periods in the horizon). Therefore, it should be chosen based on either the memory limitations of the computer, or the solution time requirements (which is greater than the solution time permitted for level i-1), whichever is more constraining.

5.6.4 Repetition and Termination

Steps 6.2 and 6.3 are repeated in succession to develop the model and decision making problem, respectively at subsequent higher levels, until the following termination conditions are obtained :

- H^i > duration that can absorb the most significant disturbances, both internal as well as external (e.g. demand seasonality).
- Number of entities at level i is small enough to treat at a highly conceptual/long term planning level, i.e. the desired degree of aggregation is accomplished.
- all controls belonging to U^0 have been addressed.
- all criteria belonging to C^0 have been addressed.

The flowchart of the design process is presented in figure 5.4. In this flowchart, the block corresponding to "Finalize," refers to the finalization of the exact form of the high level constraints for the DMP D^{i-1}, which can be formulated only after the level i entities are determined.

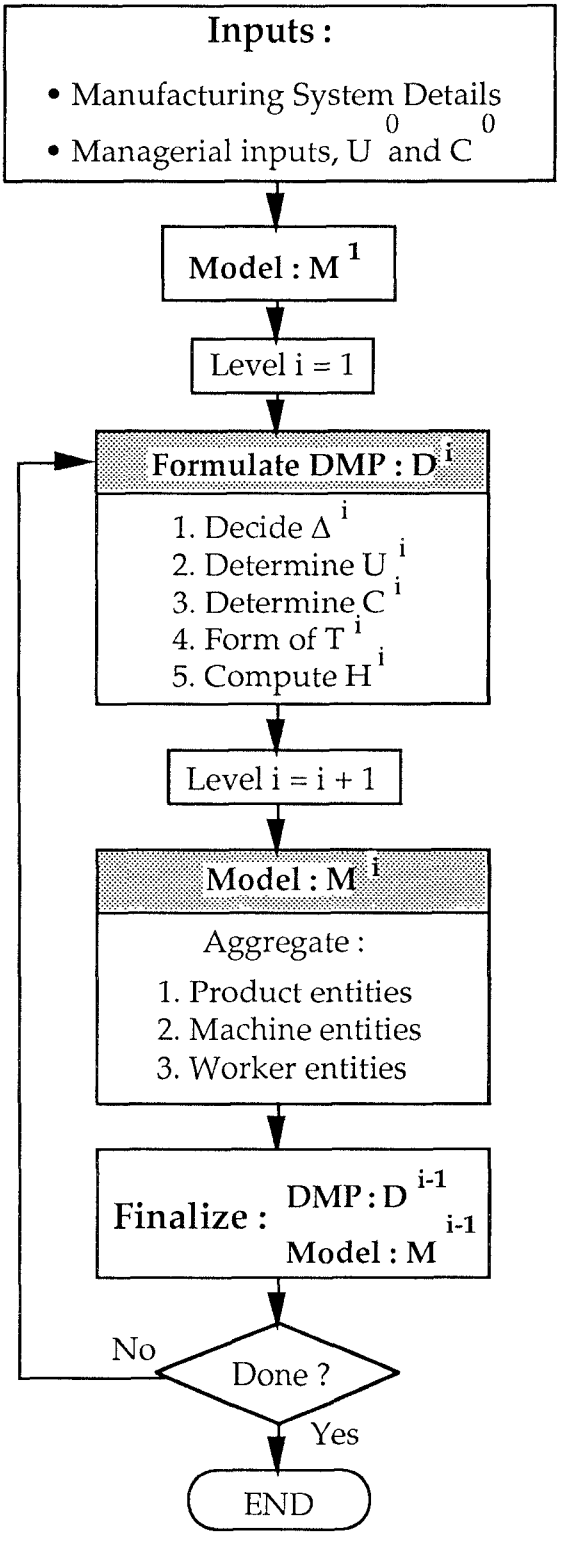


Figure 5.4 : Flowchart for the Design of the Planning Hierarchy

5.7 Operation of the Planning Hierarchy

After the construction or design of the hierarchy is complete, it can be employed for solving the planning problem and decision making. Although the decision making problems at each level and the associated solution algorithms are not defined here precisely, it is merely our intent to conceptually present the overall functioning of the hierarchy. Unlike the design process, which is bottom-up, the operation of the hierarchy is top-down. The first sub-section consists of the top-down computation of the solutions, while the second sub-section is devoted to the reaction of the hierarchy to random events.

5.7.1 Top-Down Solution Procedure to the Planning Problem

In this section, we present the operation of the hierarchy for top-down decision making in planning. We assume that the decision making problems and the associated algorithms are known at this stage. The solution procedure of the planning problem begins at the top-most level of the hierarchy, say level n. This level is the highest, hence the upper level constraints, T^n are strategic constraints in this case; they could also be constraints emanating from demand forecasting. The demand at this level is expressed in terms of the aggregate product entity at level n. For some early period(s), it can be taken from the bottom up aggregation of the detailed existing customer demand, while later periods it is forecasted demand. Given this demand, this level solves its DMP over the horizon H^n in order to optimize the criteria C^n.

The solution (usually optimal) of this problem X^n, truncated over the lower level horizon H^{n-1}, is then transmitted to the next lower level (n-1). This is in the form of its upper level constraints T^{n-1} for level n-1. This level also accepts additional information from demand forecasting regarding the proportions of level n-1 product entities composed in the level n product entities, for each elementary period Δ^{n-1}; in the absence of such information, the default is the long term/historical ratios. This information is helpful in disaggregating the aggregate production specified

by level n. The primary objectives of level n-1 are : (i) the disaggregation of this aggregate production (upper level solution) in terms of the more detailed product entities at this level, and (ii) appropriate controls corresponding to this level, i.e. controls belonging to U^{n-1}, such that the criteria C^{n-1} are optimized over H^{n-1}.

The top-down solution procedure continues for lower levels of the hierarchy in a similar manner. Each level i solves its DMP over the horizon H^i in order to optimize the criteria C^i. The solution of which, X^i, truncated over the lower level horizon H^{i-1}, is then transmitted to the lower level (i-1). This takes the form of the upper level constraints T^{i-1} for level i-1. This level also accepts additional information from demand forecasting (detailed customer demand in the case of level 1) regarding the proportions of level i-1 product entities composed in the level i product entities, for each elementary period Δ^{n-1}; the default is the long term/historical ratios. The objectives of level i-1 are the disaggregation of the upper level solution in terms of the entities at this level, as well as appropriate controls corresponding to this level (belonging to U^{i-1}), such that the criteria C^{i-1} are optimized over H^{i-1}.

The final result of this top-down computation is the production levels of product types in the elementary periods Δ^1 over the horizon H^1, under resource capacity constraints.

Rolling horizon mechanism

In a monolithic architecture, the rolling horizon mechanism is employed in order to progressively take future information into account, while only a portion of the solution is implemented. This is in a way a method to emulate an infinite horizon in practice.

The hierarchical architecture also employs this methodology, but owing to the presence of several levels, the mechanism is different. Each level in the hierarchy works on a rolling horizon basis. That is, each level recomputes its solution after every elementary period corresponding to it. In effect, only the portion corresponding to the first elementary period is

executed at every computation. We further exemplify this procedure as follows. Let all the levels compute their solutions in a top-down fashion as explained earlier. Now, only the first elementary period of the lowest level (i.e., level 1), i.e. Δ^1 is implemented on the shop-floor (or, implemented by a subsequent planning/scheduling hierarchy). Thereafter, level 1 recomputes a new solution on $[\Delta^1, \Delta^1 + H^1]$. Of course there is some vagueness for the high level (level 2) constraints for the last elementary period of level 1, because the level 2 solution is only specified after every Δ^2, where $\Delta^2 > \Delta^1$. But, this problem can be averted by assuming linearity of production during Δ^2. As soon as level 1 has implemented enough

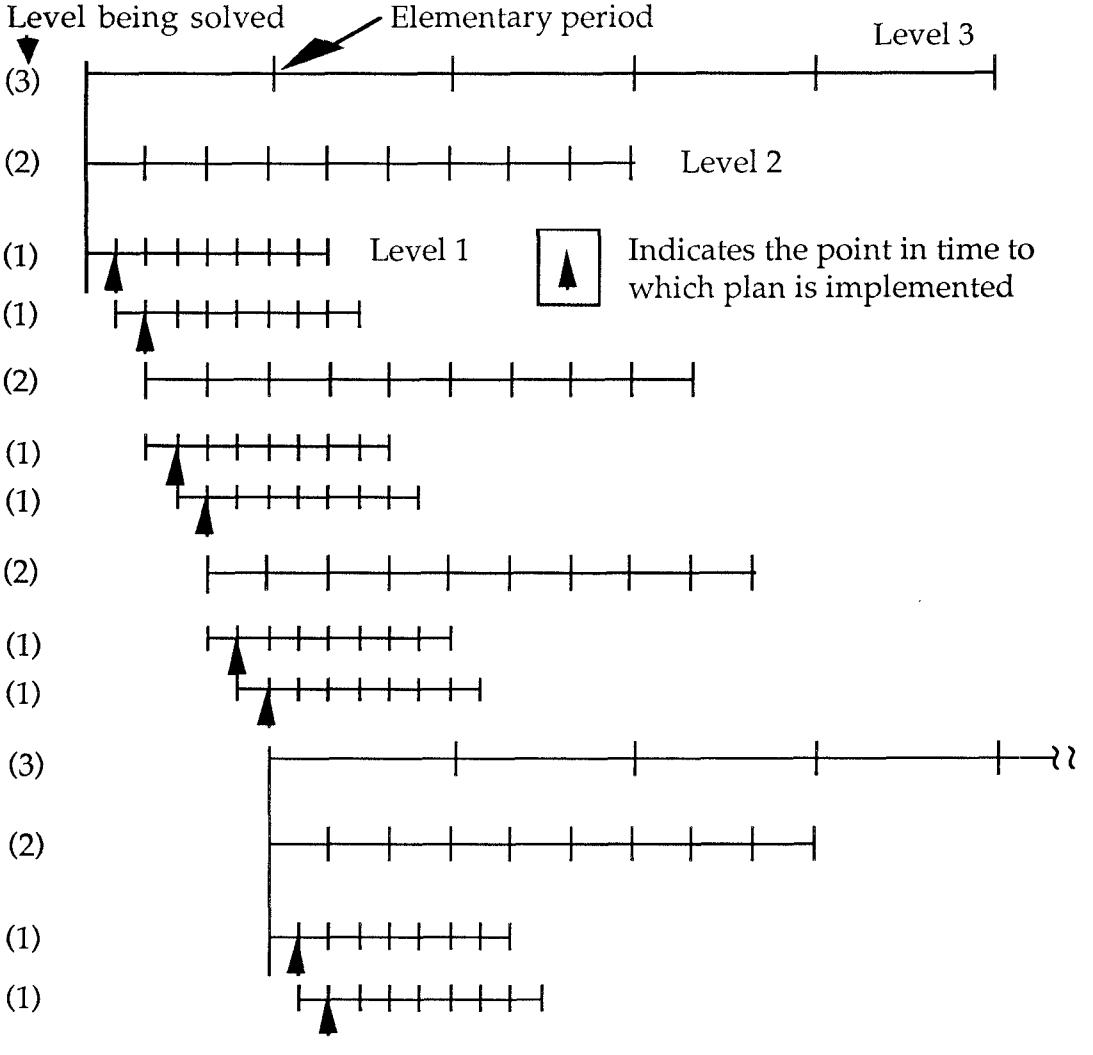


Figure 5.5 : Rolling horizon mechanism in a three-level hierarchy

elementary periods (of duration Δ^1) such that we are at the beginning of the second Δ^2 period, the level 2 solution is recomputed over $[\Delta^2, \Delta^2 + H^2]$. The process goes on for all subsequent levels of the hierarchy. Notice that, a level i problem is solved Δ^{i+1}/Δ^i times more frequently that the level i+1 problem. Which leads to the lowest level problem being solved Δ^n/Δ^1 times more frequently than the highest level problem. The rolling horizon mechanism is summarized in figure 5.5.

5.7.2 Reaction of the Hierarchy to Random Events

In the previous section, we presented the top-down solution procedure and the rolling mechanism. This corresponds to a perfect situation, where each lower level respects the higher level decision and no discrepancies arise during the execution of plans on the shop-floor. In practice, however, there are a number of disturbances and random events that make it difficult to strictly respect higher level decisions. This calls for a bottom-up feedback procedure, in that each level transmits the difference between the planned and accomplished states to the next higher level. This is done in order that the higher level takes this discrepancy into account during the subsequent calculation. Such bottom-up feedbacks are transmitted by a level that detects some discrepancy to its next higher level until this discrepancy is absorbed by some level. Usually such a feedback procedure originates at the bottom level of the hierarchy, due to random events on the shop floor (or at the subsequent planning/scheduling hierarchy), and is transmitted upwards until it can be absorbed by some level. Unusually high demands, or cancellation of orders occuring within the bottom level horizon also constitute random events that trigger the need for recomputation of the higher level(s).

However, there are certain other random events in the production planning environment that have a response time comparable to the duration of elementary periods at some level of the hierarchy. For example, a worker strike, or a severe machine breakdown can have durations comparable to durations of a high level elementary period of the hierarchy. In this case, when a random event arrives in the system, the

response time of the system to this event is evaluated by the some mechanism external to our system (e.g., in the case of a failure, the maintenance department estimates the time to repair). If the exact duration of this activity cannot be assessed, it is based on the expectation of the duration. This is then compared to the duration of the elementary periods of the various hierarchical levels. This activity is assigned to the level i such that Δ^i is an order of magnitude greater than the duration of this activity. Now, for all the levels including and below i (i.e. for all levels j such that $j \leq i$), the state of the system is assumed to be constant (e.g. the machine is assumed unavailable in the case of a machine failure), and solutions are recomputed top-down. On the completion of this random activity (i.e. the machine is repaired), the state of the system is revised (i.e. the machine is assumed available until the next failure of this type). The solutions are recomputed top-down from level i onwards for the revised system state. In this way, random events are addressed at an appropriate level of the hierarchy.

Thus, consistent with this methodology, and with reference to time-scale decomposition, we make the following remark. Capacities of resource entities at each level (say level i) of the hierarchy are calculated in a manner that considers all activities occupying this resource, with a duration much less than the elementary period at this level (Δ^i) on the average. That is, the resource capacity at that level is reduced by an amount that corresponds to the average duration that these activities are going to occupy the resource. If an activity of duration larger than Δ^i were to occupy this resource, the resource is assumed to have zero capacity (or assumed unavailable) in levels i through 1, until the completion of the activity. The underlying theory for this remark draws its motivation from time-scale decomposition, and the extension is that we manage to blend it with the aggregation or the multi-layer hierarchy. The overall operation of the planning hierarchy is summarized in figure 5.6.

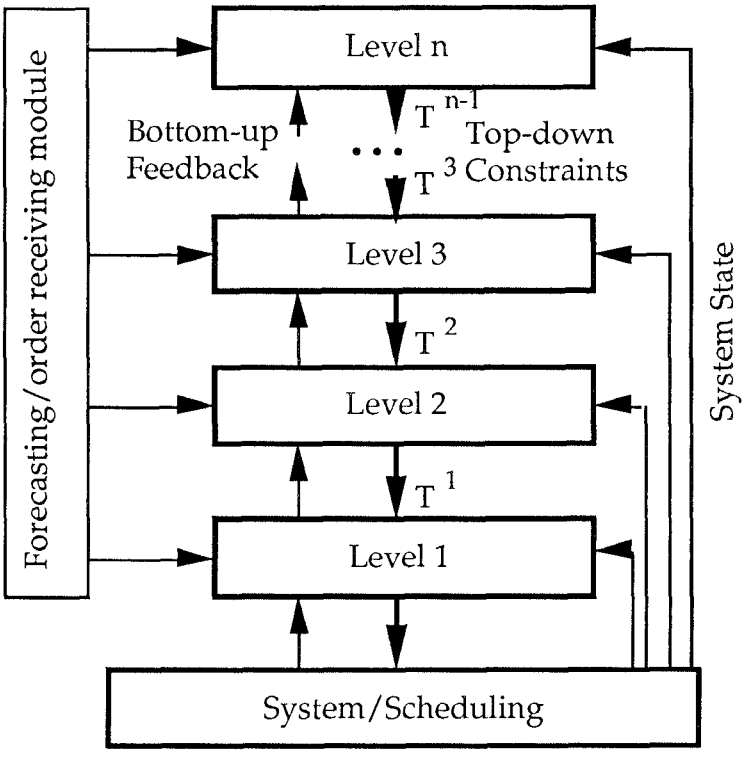


Figure 5.6 : Overall Operation of the Planning Hierarchy

5.8 A Simplified Example

In this section, we present a simplified example to demonstrate the design procedure of the planning hierarchy as presented in section 5.6. The reason we term it as simplified is because the actual details specifying the manufacturing system (as mentioned in section 5.5.1) are not detailed; instead, we assign a relative order to the relevant quantities of interest. Furthermore, the system is assumed to be deterministic. Thus, the operation of the hierarchy in the case of random events cannot be demonstrated by this example.

5.8.1 Inputs to the Design Process

In this section, we present the relevant inputs to the design of the hierarchy for the example considered. Once again, we assign a relative order of time to the relevant controls/activities of interest, while ignoring the actual details of the system.

5.8.1.1 Manufacturing system details

Here, we only present the details relating to the manufacturing system deemed essential for the design stage of this simplified example :

1) Work-center details : We consider m machines available in the system. They are reliable (or mean time between failures and mean time to repair are too short to appear at any planning level, e.g. their duration is of the order of a few minutes), and do not require to be maintained. Operating and idle costs per unit time are not considered.

2) Worker details : We assume a set of w worker types required in the system. In this case, we can assume that worker levels can be changed, i.e. workers can be hired and fired. Each worker type has a certain hiring and firing costs and lead time associated with it. Let us assign that the lead times for hiring and firing are of the order of 2-3 weeks, while overtime can be planned on less than a week (but greater than a day). Each worker type has associated regular and overtime costs per unit of time. The workers do not default. Let a working shift consist of 8 hours per day, and a maximum of 2 shifts per day are possible. However, a different set of workers must be used in each shift of a single day, i.e. the same set of workers cannot work for both shifts of the same day. The maximal overtime level is 25% of the regular hours per shift, i.e. 2 hours per shift. Overtime costs 25% higher than regular costs for all worker types.

3) Product details : We consider n product types (finished products) that are manufactured in the system. We ignore details regarding the Bills-Of-Materials (BOM) and components, and assume that a representative product routing is available for each part. A holding and backlogging cost is associated to each product type; the penalty is specified over the duration of one working day.

4) Routing details : Each product has a representative routing which reflects the cumulative work-content (processing time) required by that product type at each machine. Set-up times are ignored (or assumed too

short to consider at the planning levels). Furthermore, the time required from each worker type is also assumed known. We attribute the total lead time for products to be of the order of 10-15 minutes.

5) Historical or Forecast data : Historical information about the long term production requirements (or demand) of products is assumed available.

5.8.1.2 Managerial Inputs

The details related to the managerial inputs are as follows :

1) Controls, U^0 :
- Production of products (response time O(10-15 minutes))
- Hiring and Firing of workers (response time O(2-3 weeks))
- Decision of worker overtime levels (response time O(2-3 days))

Where, O(•) represents the order • units of time.

2) Criteria, C^0 :
- Minimization of Production costs :
  - Hiring and firing costs
  - Overtime costs
- Holding and backlogging costs for products

These criteria are the management objectives that have to be optimized during the production planning. Note that the criteria considered in the planning problem bear a strong relationship with the controls considered. For example, if hiring and firing of workers is a possible control, the corresponding costs are relevant in the objective function; otherwise, the costs are irrelevant.

5.8.2 Design of the Planning Hierarchy

We present the design procedure for the construction of a planning hierarchy in this section.

5.8.2.1 D^1 : Decision Making Problem at Level 1

1. Elementary period, Δ^1 = 1 day
   - 1 day > lead time for products, O(10-15 minutes).

- 1 day = time interval after which orders are delivered to stock, shipments are made and inventory is updated.
- 1 day >> mean time for an important class of machine failures, if any.

2. Controls, U^1

   Production levels for products in elementary periods of duration Δ^1 . Note, all other controls have response time larger than a day.

3. Criteria, C^1

   Holding and backlogging cost.

4. Constraints, T^1

The intrinsic constraints are essentially resource (machine and worker) capacities. The exact higher level constraints T^1 are not known because the high level remains undefined at this time. The general form of these constraints are aggregate production constraints related to the production levels of aggregate products (families) that must be respected by the current level during the disaggregation process. Furthermore, aggregate resource capacity levels may also represent high level constraints.

The complexity of the decision making problem can be determined in the following manner. For each elementary period, there are n variables corresponding to the production levels of products, and n variables corresponding to the inventory costs of these products. We roughly compute the number of constraints for each elementary period as follows. There are n(2) aggregate production constraints (equality constraints); where, n(i) is the number of product related entities at level i (i=2 in this case). There are m machine capacity constraints. There may be some additional constraints (see next chapter). Finally, there are the non-negativity constraints. Based on this information the complexity of the DMP over one elementary period is roughly assessed.

The next step consists of determining the length of the horizon H^1, or the number of elementary periods Δ^1 that can be included in the horizon H^1.

This should preferably be performed after the next higher level model M^2 is determined, and the exact formulation of D^1 is known.

5. Horizon, H^1 = 2 weeks (assumed)

   - Let us assume that 2 weeks nearly equal the horizon during which customer orders are firmed up and affect production/stock.
   - Let us assume that this is based on the complexity of the DMP (which is proportional to the number of elementary periods = 10 in the horizon). Let us finally assume that this is based on the memory limitations of the computer and solution time requirements (which is < 15 minutes).

5.8.2.2 M^2 : Model at Level 2

Once again, at a conceptual level, we present the procedure to develop the model at level 2. Employing the aggregation rules proposed in section 5.6.1 with appropriate values of parameters, let us assume that we obtain n(2) number of product related entities, m(2) machine related entities, and w(2) worker related entities. Let the clustering results find approval when evaluated in the light of physical or practical considerations.

Now, the high level constraints for the level 1, T^1 can be formulated. Thus, the decision making problem at the level 1, D^1 can be completed at this stage.

5.8.2.3 D^2 : Decision Making Problem at Level 2

1 Elementary period, Δ^2 = 1 week

   - Δ^2 is a multiple of Δ^1 (5 days is a multiple of 1 day).
   - $\Delta^2 \leq H^1$ (2 weeks).
   - Δ^2 < response time of decision of worker overtime levels (response time O(2-3 days)).

2 Controls, U^2

   1 Production levels for product entities at level 2 on machine entities at level 2 for elementary periods of duration Δ^2 .

   2 Decision of worker overtime levels (response time O(2-3 days)).

3 Criteria, C^2

1 Holding and backlogging cost for product entities at level 2.
2 Minimization of overtime costs.

4 Constraints, T^2

The intrinsic constraints are essentially resource (machine entities at level 2 and worker entities at level 2) capacities. As before, the exact higher level constraints T^2, are not known because the high level remains undefined at this time. The general form of these constraints are aggregate production constraints, related to the production levels of product entities that must be respected by the current level during disaggregation. Furthermore, aggregate resource capacity levels may also represent high level constraints.

The complexity of the decision making problem can be determined in the following manner. For each elementary period, there are n(2) variables corresponding to the production levels of product entities, n(2) variables corresponding to the inventory costs of these product entities, and w(2) variables corresponding to the worker overtime levels. We roughly compute the number of constraints for each elementary period as follows. There n(3) aggregate production constraints (equality constraints). There are w(2) worker capacity constraints, and w(2) worker overtime level constraints. Finally, there are the non-negativity constraints. Based on this information the complexity of the DMP over one elementary period is roughly assessed.

The next step consists of determining the length of the horizon H^2. Once again, it is preferable to ascertain this when the next higher level model M^3 is determined, and the exact formulation of D^2 is known. However, for this simplified example, we assume that the general form of these should suffice.

5 Horizon, H^2 = 3 months

- H^2 (3 months) > H^1 (2 weeks).
- Let us assume that this is based on the complexity of the DMP (which is proportional to the number of elementary periods = 12 in the horizon). Let us assume that this is based on the memory

limitations of the computer and solution time requirements (which is < 15 minutes).

5.8.2.4 M^3 : Model at level 3

Once again conceptually we present the procedure to develop the model at level 3. Employing the aggregation rules proposed in section 5.6.1 with appropriate values of parameters, let us assume we obtain n(3) number of product related entities, m(3) machine related entities, and w(3) worker related entities. Let the clustering results find approval when evaluated in the light of physical or practical considerations.

Now, the high level constraints for the level 2, T^2 can be formulated. Thus, the decision making problem at the level 2, D^2 can be completed at this stage.

5.8.2.5 D^3 : Decision Making Problem at Level 3

1 Elementary period, Δ^3 = 4 weeks (roughly 1 month)

- Δ^3 is a multiple of Δ^1 (a period of 4 weeks is a multiple of 1 week).
- $\Delta^3 \leq H^2$ (3 months).
- Δ^3 < response time of hiring and firing workers (response time O(2-3 weeks)).

2 Controls, U^3

1 Production levels for product entities at level 3 on machine entities at level 3 for elementary periods of duration Δ^3 .

2 Decision of worker overtime levels (response time O(2-3 days)).

3 Decision of hiring or firing workers (response time O(2-3 weeks)).

3 Criteria, C^3

1 Holding and backlogging cost for product entities at level 2.

2 Minimization of overtime costs.

3 Minimization of hiring and firing costs.

4 Constraints, T^3

The intrinsic constraints are essentially resource (machine entities at level 3 and worker entities at level 3) capacities. As before, the exact higher level

constraints T^3, are not known because the high level (if any) remains undefined at this time. However, we can anticipate termination at this stage because all the criteria and controls have been addressed. So, T^3 are likely to take the form of strategic constraints.

The complexity of the decision making problem can be determined in the following manner. For each elementary period, there are n(3) variables corresponding to the production levels of product entities, w(3) variables corresponding to the worker entity levels, and w(3) variables corresponding to the worker overtime levels. We roughly compute the number of constraints for each elementary period as follows. There are w(3) worker capacity constraints, w(3) worker level constraints, and w(3) worker overtime level constraints. Finally, there are the non-negativity constraints. Based on this information the complexity of the DMP over one elementary period is roughly assessed.

The next step consists of determining the length of the horizon H^3.

5 Horizon, $H^3 \approx$ 12 months (48 weeks)

- H^3 (12 months) > H^2 (3 months, or 12 weeks).
- Let us assume that this is based on the complexity of the DMP (which is proportional to the number of elementary periods = 12 in the horizon). Let us assume that this is based on the memory limitations of the computer and solution time requirements (which is < 15 minutes).

5.8.2.6 Check for Termination conditions

At this stage :

- all controls belonging to U^0 have been addressed.
- all criteria belonging to C^0 have been addressed.
- H^3 > duration that can absorb the most significant disturbances, both internal as well as external (e.g. demand seasonality over a year).
- At level 3, the desired degree of aggregation is accomplished.

Thus, the hierarchy consists of three levels, and the structure can be represented as in figure 5.7.

Lev	Δ	H	Controls, U	Criteria, C	Constraints, T
1	1 day	2 wks	Production of products	Holding & Backlogging	Aggregate Production Agg. Worker hours
2	1 wk	3 mths	Prod. of agg. products Overtime levels	Holding & Backlogging Overtime costs	Aggregate Production Number of workers
3	4 wks	12mths	Prod. of agg. products Overtime levels Hiring & Firing	Holding & Backlogging Overtime costs Hiring & Firing costs	Global worker levels (Number of shifts, & maximal overtime)

Figure 5.7 : Summary of Decision Making Problems in the Hierarchy

Chapter 6

Application to a Generic Job-Shop

In the present attempt, we examine a generic job-shop consisting of a set of work-centers, a set of labor trades, and a set of product types to produce. The major disruptive stochastic events consist of work-center failures and labor absenteeism. The overall objectives are to design a management system for planning, such that a set of criteria are optimized. In this chapter, we employ a numerical example consisting of 50 products and 34 work-centers to present the design of the planning hierarchy.

This chapter is organized as follows. Section 1 presents the inputs essential for the design and the operation of the hierarchy. Section 2 is devoted to the design of hierarchy, while section 3 presents the operation of the hierarchy in brief. In section 4, we introduce the software developed for the design and operation. Finally, section 5 present some numerical results obtained for a set of example problems.

6.1 Inputs to the Design Process

This section provides the manufacturing system details and managerial inputs that are essential for the design of the hierarchy, and its subsequent operation.

6.1.1 Manufacturing System Details

6.1.1.1 Work-Center Details

We consider a set $\mathcal{M} = \{m_1, m_2, \ldots, m_m\}$ of m work-center types in a given manufacturing system. Let mn_j represent the number of functionally similar work-centers of work-center type m_j; $j = 1,2,\ldots,m$. The work-centers

are prone to failure. We also associate with each work-center type m_j, an operating cost per unit time, mc_j; j = 1,2,...,m. We assume that this operating cost in incurred when the work-center is producing parts or during a set-up operation, however, not when the work-center is idle or under repair. For each work-center type m_j, let us assume there are nfm_j number of failure modes. For the k-th failure mode, the time between failures and the time to repair are modeled as exponentially distributed variates with means of mf^k_j and mr^k_j respectively; k = 1,2,...,nfm_j, j = 1,2,...,m. Let us also assume that the failure modes are ordered in increasing order of mean times between failures. Modeling of machine failures in this manner has been accepted in the literature (see Gershwin [35], Xie [77]).

6.1.1.2 Worker Details

Labor trades are general categories of labor like welders, machinists, foundrymen, forging workers, sheet-metal workers, etc. Each labor trade may be further classified into sub-trades; e.g., turners, grinders, drillers are sub-trades of machinists. We consider a set $\mathcal{W} = \{w_1, w_2, \ldots, w_w\}$ of w worker sub-trades in the manufacturing system. Let wn_j represent the number of workers or worker level of the same sub-trade w_j; j = 1,2,...,w. Note that, the worker level is variable. We associate with each worker sub-trade w_j a regular and overtime cost per unit time as wcr_j and wco_j respectively; j = 1,2,...,w. We also associate with each worker sub-trade w_j a hiring and firing cost per unit time (e.g. hours) of labor as whc_j and wfc_j respectively; j = 1,2,...,w. The workers may default. For each worker sub-trade w_j, let us assume there are ndm_j number of default modes. For the k-th failure mode, the time between absences and the duration of absence are modeled as exponentially distributed variates with means of wf^k_j and wr^k_j respectively; k = 1,2,...,ndm_j, j = 1,2,...,w.

For labor intensive work-centers, a worker sub-trade is capable of operating a set of work-centers. Let $\mathcal{WM}_i = \{m_i(1), m_i(2), \ldots, m_i(g_i)\}$ represent the set of g_i work-centers capable of being operated by worker sub-trade w_i; i = 1,2,...,w.

6.1.1.3 Product Details

The production process consists of transforming raw-materials into finished products, through a series of semi-finished (intermediate) products or parts. We consider three types of "parts" : (i) a set of finished products; set $\mathcal{P}_f = \{p_1, p_2, \ldots, p_{n_f}\}$ represents n_f finished product types, (ii) a set of semi-finished parts (sub-assemblies, or components); set $\mathcal{P}_s = \{p_{n_f+1}, p_{n_f+2}, \ldots, p_{n_f+n_s}\}$ represents n_s semi-finished product types, and (iii) a set of raw-materials; set $\mathcal{P}_r = \{p_{n_f+n_s+1}, p_{n_f+n_s+2}, \ldots, p_{n_f+n_s+n_r}\}$ represents n_r raw-materials (or purchased items). Semi-finished parts are recognized as inventory items, i.e., they contribute to inventory holding costs.

We define a set of manufactured or "make" parts which is the union of finished parts and semi-finished parts, and this is represented by :
$\mathcal{P}_m = \mathcal{P}_f \cup \mathcal{P}_s = \{p_1, p_2, \ldots, p_n\}$, where $n = n_f + n_s$.

We represent the Bills-Of-Materials (BOM) or product structures in the following manner. Each manufactured part requires the availability of a set (or more precisely a multi-set) of pre-requisite parts that may be raw-materials or semi-finished parts. Let $\mathcal{B}_i$ represent the set of part types required to produce part p_i, $p_i \in \mathcal{P}_m$. $\mathcal{B}_i = \{p_{c^1_i}, p_{c^2_i}, \ldots, p_{c^j_i}, \ldots, p_{c^{b(i)}_i}\}$ represents the set of b(i) part types, and let nc^j_i represent the number of each part $p_{c^j_i}$ required for producing one unit of p_i; $p_{c^j_i} \in \mathcal{P}_r$ or $\mathcal{P}_m$. Note that we have additional constraints associated with $\mathcal{B}_i$, such that it cannot include p_i or any of the parts it is used for as a component.

6.1.1.4 Routing Details

Each make part type has associated with it a set of manufacturing processes or routings. Let the set $\mathcal{R}(i) = \{R^1_i, R^2_i, \ldots, R^k_i, \ldots, R^{q(i)}_i\}$ of q(i) alternative routings be associated with part type p_i; i = 1,2,...,n. A routing is defined by a sequence of operations. The routing $R^k_i = < o^k_{i1}, o^k_{i2}, \ldots, o^k_{ij}, \ldots, o^k_{is^k_i} >$, is a sequence of s^k_i operations for the k-th routing of part type p_i; i = 1,2,...,n; k = 1,2,...,q(i). Note that several work-centers may be able to perform the same operation. Let $\mathcal{MO}^k_{ij}$ represent the set of work-center types capable of performing operation o^k_{ij}. We also associate with each

operation the corresponding set-up and processing times. Let $\sigma^k_{ij}(x)$ represent the set-up time required to configure work-center m_x, $m_x \in M$, to perform operation o^k_{ij}. Further, let $\tau^k_{ij}(x)$ be the processing time at work-center m_x to perform operation o^k_{ij} on one unit of part type p_i.

Note :

(1) if a work-center $m_x \notin \mathcal{MO}^k_{ij}$, then, $\sigma^k_{ij}(x) = \tau^k_{ij}(x) = \infty$, (or undefined).

(2) if a work-center $m_x \in \mathcal{MO}^k_{ij}$

(i) $\tau^k_{ij}(x)$ is finite and $\tau^k_{ij}(x) \in R^+$.

(ii) if work-center m_x is already configured for operation o^k_{ij} (between parts of the same batch or between batches requiring the same set-up), then, $\sigma^k_{ij}(x) = 0$, else, $\sigma^k_{ij}(x)$ is finite and $\sigma^k_{ij}(x) \in R^+$.

The information presented thus far is static input pertaining to the manufacturing system, in the sense that the values of the above attributes do not change frequently with time. It is recognized, however, that the system should be flexible enough to accommodate changes in its static elements, e.g., new products and/or process plans can be added.

6.1.1.5 Historical or Forecasted Data

To facilitate the aggregation procedures, some statistical input of historic data may be essential in determining on an average the proximity of entities. More specifically, let n_i represent the total number of products of type p_i produced in a sufficiently large horizon H; $i = 1,2,\ldots,n_f$. Let r_i represents the production ratio of product type p_i; $i = 1,2,\ldots,n_f$. Then,

$$r_i = \frac{n_i}{\sum_{k=1}^{n} n_k}; \qquad i = 1,2,\ldots,n_f \tag{1.1}$$

Further, let bs_i represent the average batch size of part type p_i, $p_i \in \mathcal{P}_m$.

If information regarding the usage ratio of alternative part routing is available, it could be of relevance too. Let u^k_i represent the usage ratio of the k-th routing of part p_i, R^k_i; $i = 1,2,\ldots,n$; $k = 1,2,\ldots,q(i)$.

This concludes the most significant manufacturing system related inputs.

6.1.2 Managerial Inputs

In this example, the details related to the managerial inputs are similar to those in the simplified example of chapter 5.

6.1.2.1 Controls, U^0

We consider the following controls :

1. Production of products (completion time O(10-15 minutes))
2. Hiring and Firing of workers (decision time O(2-3 weeks))
3. Notice for worker overtime (decision time O(2-3 days)). The overtime level can be a maximum of β ($\beta < 1$; $\beta = 0.2$ in this example) times the regular hours for each worker type.

6.1.2.1 Criteria, C^0

1. Minimization of a sub-set of Production costs :
   - Hiring and Firing costs
   - Overtime costs
2. Holding and Backlogging costs for products

For the example under consideration, these criteria have been chosen as they are believed to be among the most relevant to single plant production planning problems. Furthermore, we observe that per unit product holding costs are usually established by management by the consideration of raw-material and production costs incurred in the long term. However, we assume that the per unit product holding costs are precomputed from standard costs and assumed independent of the actual costs incurred for a specific product.

6.2 Design of the Planning Hierarchy

In this section, we present the procedure for the design of a planning hierarchy. First, we introduce some notation that will be helpful in describing the models and decision making problems at each level of the hierarchy. Some of these may have been introduced earlier, but they are repeated for the sake of completeness.

M^i : Model at level i.

Ni : number of product entity types at level i.

pi_b : product entity b at level i.

ni_b : long term production (demand) of product entity b at level i.

ri_b : long term production ratio of product entity b at level i within its parent entity at level i+1.

Mi : number of machine entity types at level i.

mi_j : machine entity j at level i.

Wi : number of worker entity types at level i.

wi_j : worker entity j at level i.

$wcri_j$: regular cost of worker entity j at level i.

$wcoi_j$: overtime cost of worker entity j at level i.

$whci_j$: hiring cost of worker entity j at level i.

$wfci_j$: firing cost of worker entity j at level i.

ti_{bj} : processing time of product entity b at level i on machine entity j at level i.

ti'_{bj} : processing time of product entity b at level i on machine entity j at level i-1.

$ci_b^{+(-)}$: holding (resp. backlogging) cost of product entity b at level i.

ei_j : efficiency of machine entity j at level i.

D^i : Decision making problem at level i.

Δ^i : Duration of an elementary period at level i.

U^i : Set of controls at level i.

C^i : Set of criteria at level i.

T^i : Set of higher level (i+1) constraints to level i.

H^i : Horizon at level i.

zi : number of elementary periods of duration Δ^i that are included in the horizon H^i, i.e. $zi \times \Delta^i = H^i$.

hi : number of level i elementary periods (Δ^i) composed in one higher level elementary periods Δ^{i+1}, i.e. $hi \times \Delta^i = \Delta^{i+1}$.

gi : number of higher level periods (Δ^{i+1}) included in H^i, i.e. gi = zi/hi.

ki : index of level i elementary periods (Δ^i).

Variables :

$rhi_{j(ki)}$: regular hours of worker entity j at level i.
$oi_{j(ki)}$: overtime hours of worker entity j at level i.
$hli_{j(ki)}$: hours hired (hiring level) of worker entity j at level i.
$fli_{j(ki)}$: hours fired (firing level) of worker entity j at level i.

Given :

$di_{b(ki)}$: demand of product entity b at level i in the ki-th period (Δ^i).
$Di_{b(ki)}$: cumulative demand of product entity b at level i at the end of ki-th period (Δ^i).
$mrhi_j$: maximal regular hours of worker entity j at level i (for Δ^i).
$rhi_{j(0)}$: initial regular hours of worker entity j at level i (for Δ^i).

6.2.1 M^1 : Model at Level 1

The model at level 1 is a physical model, and its entities are products, machines and workers. For this lowest planning level, we make some simplifying assumptions that help reduce the complexity of the system. We make principal assignments among the choice of alternatives, i.e. primary routings are chosen, and workers are assigned to preferred machines. The lower operational and scheduling levels of a hierarchical management system can take advantage of this flexibility among alternatives, but for planning purposes we restrict ourselves.

Planning Product Routings (Representative Routings)

For planning purposes, we assume that we have a single representative routing for each product type. This representative routing depicts the manufacture of this product starting directly from raw-material(s), i.e., the semi-finished part and/or sub-assembly stages are ignored; however their processing steps are included in the routing of the finished product. At first, one preferred routing (usually the first) is selected for each manufactured part. Then, the total production time (work-content) at each work-center is accrued for the finished product as well as its dependent components, and it is represented in a cumulative manner. Note that in such a representative routing, the sequence of work-centers is immaterial;

it only represents the total work-content required of each work-center. Further, the set-up times associated with each part operation are divided by the average batch size of that part to obtain a per unit set-up requirement, which is added to the processing time. Alternatively, if the historical (long term) proportion of usage of alternative routings were known, then the "preferred" routing could be constructed by the weighted average of processing times provided by each routing in the same proportion as routing usage.

Preferred Assignments

As detailed in section 6.1.1.4, each make part type has associated with it a set of routings; i.e. $\mathcal{R}(i) = \{R^1_i, R^2_i, \ldots, R^k_i, \ldots, R^{q(i)}_i\}$, represents a set of q(i) alternative routings associated with part type p_i; i = 1,2,...,n. Let the first routing be the preferred one, i.e. R^1_i. The routing $R^1_i = < o^1_{i1}, o^1_{i2}, \ldots, o^1_{ij}, \ldots, o^1_{is^1_i} >$, is a sequence of s^1_i operations for the routing of part type p_i; i = 1,2,...,n. In general, several work-centers may be able to perform the same operation. Let $\mathcal{MO}^1_{ij}$ represent the set of work-center types capable of performing operation o^1_{ij}. In the preferred assignment, $\mathcal{MO}^1_{ij}$ is a singleton set; let $\mathcal{MO}^1_{ij} = \{m_x\}$. Thus, the corresponding set-up and processing times are $\sigma^1_{ij}(x)$ and $\tau^1_{ij}(x)$, respectively.

In the following sub-section, we detail the construction of representative routings. Let t_{ix} represent the cumulative processing time for one product of type p_i on the work-center m_x; i = 1,2,...,n_f; x = 1,2,...,m.

Algorithm for representative routings or processing times

The top level call of the procedure detailed below is performed for i = 1,2,...,n_f, or i | $p_i \in \mathcal{P}_f$, i.e. finished products, and repeatedly over each work-center m_x; i.e. x = 1,2,...,m. It returns the cumulative processing time required to produce one unit of p_i from the raw-materials on work-center m_x, t_{ix}. During recursion, we may have i = n_f+1, n_f+2,...,n_f+n_s+n_r, or i | $p_i \in \{\mathcal{P}_s, \mathcal{P}_r\}$, i.e. semi-finished, or purchased parts.

```
Rep_routing_proc_time(i, x)
Begin
```

```
    If (p_i ∈ P_r)
        return(0);
    else
        PT = 0;
        for j = 1 to s^1_i do    /* Operations    */
            if(m_x ∈ MO^1_ij)
                PT = PT + σ^1_ij(x)/bs_i + τ^1_ij(x);
        end;
        return(PT + Proc_time_of_children(i, x));
End.
```

Note that, this procedure calls another procedure named Proc_time_of_children(i, x), which returns the cumulative processing time of the components of p_i on machine m_x. Proc_time_of_children(i,x) in turn calls the previous procedure Rep_routing_proc_time(•, x) for each of the components of p_i (represented by •). Thus, there is recursion, and care should be taken to avoid cyclicity while constructing the BOMs of the products.

```
Proc_time_of_children(i,x)
Begin
    CPT = 0;
    for v = 1 to b(i) do
        CPT = CPT + nc^v_i × Rep_routing_proc_time(c^v_i, x);
    end;
    return(CPT);
End.
```

A variation of this algorithm to address usage ratios of alternative routings is presented below. Here, u^k_i represents the usage ratio of the k-th routing of part p_i, R^k_i. We have,

$$\sum_{k=1}^{q(i)} u^k_i = 1; \qquad i = 1,2,\ldots,n_f + n_s \tag{2.1}$$

```
Rep_routing_proc_time(i, x)
Begin
    If (p_i ∈ P_r)
        return(0);
    else
        PT = 0;
        for k = 1 to q(i) do  /*  Routings   */
            for j = 1 to s^k_i do   /*  Operations      */
                if(m_x ∈ MO^k_ij)
                    PT = PT + (σ^k_ij(x)/bs_i + τ^k_ij(x)) × u^k_i;
            end;
        end;
        return(PT + Proc_time_of_children(i, x));
End.
```

For the example considered in this chapter, the details of the product, work-center and worker entities, and routings are presented in Appendix B.1. There are 50 product types and 34 work-centers in the system. All work-centers are assumed labor intensive, and may be composed of more than one functionally similar machine. This implies that the worker and machine entities are equivalent, i.e. the capacity available on a machine is equal to the number of labor hours of the corresponding worker type; which also leads to Wi = Mi, ∀i. There is also a constraint on the number of hours that a work-center can be operated (depending on the permissible shifts), which in turn imposes a constraint on the number of worker hours that can be planned for the corresponding worker type. Preferred assignments are represented in the routing details.

As explained earlier, the machines are subject to failure. Thus, the efficiency of each machine at level 1 is computed from the failure modes. We assume that the clustering of failure modes has been performed previously (see figure 5.3 and section 5.5.1). The response time, mt^k_j of a particular (k-th) failure mode for machine m_j is defined as the summation of its MTBF (mf^k_j) and MTTR (mr^k_j); i.e. $mt^k_j = mf^k_j + mr^k_j$. At each level i, we include (absorb) the failure modes which have a response time that is

much less than (at least an order of magnitude) the elementary period at that level, Δ^i. Thus, the efficiency reflects the absorption of these classes of failures.

$$ei_j = \frac{\sum_{\kappa} mf^{\kappa}_j}{\sum_{\kappa} mt^{\kappa}_j} \qquad \kappa \mid mt^{\kappa}_j << \Delta^i,\ j = 1,\ldots,Mi \qquad (2.2)$$

In our example, we *a priori* classify the failure modes in a manner that is consistent with the response times of controls. That is, we consider the failures which have response times comparable to that of a control in the same mode. Thus, the number of failure modes can be considered equal to the number of control classes. In this case, for level 1, we have :

$$e1_j = \frac{mf^1_j}{mf^1_j + mr^1_j} = \frac{mf^1_j}{mt^1_j} \qquad j = 1,\ldots,M1 \qquad (2.3a)$$

In general, workers can be unreliable; to effectuate production on labor intensive work-centers, both the machine as well as the worker should be present. Thus, the available capacity could be lower than the expression (2.3a). A rather conservative approach may be to consider a product of machine and worker efficiencies as :

$$e1_j = \frac{mf^1_j}{mf^1_j + mr^1_j} \times \frac{wf^1_j}{wf^1_j + wr^1_j} \qquad j = 1,\ldots,M1 \qquad (2.3b)$$

where worker w_j is assumed assigned to machine m_j. We suggest the selection of the expression that is appropriate with the assumptions relating to the particular type of production system in question. For the current example, we consider the expression (2.3a).

6.2.2 D^1 : Decision Making Problem at Level 1

1. Elementary period, Δ^1 = 1 day
   - 1 day > lead time for products, O(10-15 minutes).
   - 1 day = time interval after which orders are delivered to stock, shipments are made and inventory is updated.
   - 1 day >> mean time for an important class of machine failures.
2. Controls, U^1

Production levels for products in elementary periods of duration Δ^1. Note that, all other controls have a response time larger than a day.

3. Criteria, C^1

   Holding and backlogging cost.

4. Constraints, T^1

The exact higher level constraints T^1, are not known because the high level remains undefined at this time. These constraints consist of the following two components : (i) aggregate resource capacity levels (regular and overtime) represent high level constraints, which are intrinsic constraints relating to resource (machine and worker) capacities, and (ii) aggregate production constraints related to the production levels of aggregate products (families) that must be respected by the current level during disaggregation. The first type of aggregate resource constraints are easily transformed into local resource constraints. The second type of constraints are treated in the objective function, with high penalties for violation. This approach is adopted in order to avoid infeasible level 1 solutions that can be introduced due to disaggregation inconsistencies.

The complexity of the decision making problem can be determined in the following manner. There are z1 $\times$ N1 variables corresponding to the production levels of products in the horizon. To account for the holding as well as backlogging cost, z1 $\times$ N1 variables representing the inventory cost are introduced (see problem (*P6.1*) in section 6.2.4). There are g1 $\times$ N2 variables representing penalty costs for violating the higher level aggregate production levels; note that N2 is undefined at this stage. Now, the number of constraints are assessed as follows. There are z1 $\times$ W1 constraints corresponding to worker capacities. To account for the inventory cost being the higher of the holding and backlogging cost, 2 $\times$ z1 $\times$ N1 constraints are introduced (see problem (*P6.1*)). To account for the penalty cost being the higher of the holding and backlogging cost, 2 $\times$ g1 $\times$ N2 constraints are introduced (see problem (*P6.1*)). Finally, there are the non-negativity constraints. Based on this information the complexity of the decision making problem over one elementary period is roughly assessed.

The next step consists of determining the length of the horizon H^1, or the number of elementary periods, z1, of duration Δ^1 that can be included in the horizon H^1. This is performed in the finalization stage (see section 6.2.4), when the exact form and number of variables and constraints are known.

6.2.3 M^2 : Model at Level 2

In this section, we present the development of model M^2 from the aggregation of entities in the model M^1. In this case, all work-centers are assumed labor intensive, and may be composed of more than one functionally similar machines. Further, there is a one-to-one preferred assignment of workers to machines. Thus, we aggregate machine entities, and the worker entities follow the same aggregation scheme.

First, we aggregate products to families by employing the modified K-mean clustering algorithm presented in section 4.11 of chapter 4. Products are represented in IR^{M1+2} (IR^{36} in this case) by a point. M1 axes are used to represent the processing time required by a part on M1 machines, and in addition, two axes are used to represent the holding cost and backlogging cost, respectively. All axes are normalized in the interval [0,100]. This is to remove the bias of different orders of numerical values for costs and processing times, while determining a representation in a standard space. The clusters are permitted iff the Euclidean distance from the center of a cluster to its corresponding points does not exceed a user specified parameter, $\sigma 1$. This parameter is helpful in controlling the compactness of clusters, and is kept low (< 5 to 10 times $\sqrt{M1+2}$ for normalized axes) for the model M^2. A good clustering should possess the following characteristics : (i) appropriate reduction of dimensionality, (ii) balanced number of products to a family, and (iii) compact radii enclosing entities of the same family. Based on these criteria, a clustering which resulted in 20 families was obtained. The details of this grouping are presented in Appendix B.2. The processing times of the families on the machines and cost attributes are calculated from the center of gravity of the clusters; this time each point is assigned a weight corresponding to that product's long

term demand (see section 6.1.1.5). Now, the long term demand of a family is the summation of the long term demands of its component products. More precisely, if clustering results in N2 (20 for this example) number of families, i.e. $p2_1, p2_2,\ldots,p2_{N2}$, we have :

$$r1_i = \frac{n1_i}{\sum_{b \mid p1_b \in p2_i} n1_b}; \qquad i = 1,2,\ldots,N1 \tag{2.4}$$

$$c2_i^{+(-)} = \sum_{b \mid p1_b \in p2_i} r1_b c1_b^{+(-)} = \frac{n1_b c1_b^{+(-)}}{\sum_{b \mid p1_b \in p2_i} n1_b}; \quad i = 1,2,\ldots,N2 \tag{2.5}$$

$$t2'_{ij} = \sum_{b \mid p1_b \in p2_i} r1_b t1_{bj} = \frac{n1_b t1_{bj}}{\sum_{b \mid p1_b \in p2_i} n1_b}; \qquad i = 1,2,\ldots,N2,\ j = 1,\ldots,M1 \tag{2.6}$$

$$n2_i = \sum_{b \mid p1_b \in p2_i} n1_b; \qquad i = 1,2,\ldots,N2 \tag{2.7}$$

Note that, (2.5) provides the holding and backlogging costs of families over level 1 elementary periods (Δ^1). It is not obvious what these costs should be for the level 2 elementary periods (Δ^2). Under the assumption that the inventory sign remains unchanged during a level 2 period, we can simply multiply the above costs by the number of level 1 elementary periods in a level 2 elementary period, h1. Although, this will not make a relative difference among the families (as we are multiplying all costs by a constant), it will reflect an appropriate comparison of inventory costs to other costs (e.g., hiring/firing and overtime costs). Thus,

$$c2_i^{+(-)} = h1 \times \sum_{b \mid p1_b \in p2_i} r1_b c1_b^{+(-)}; \quad i = 1,2,\ldots,N2 \tag{2.5'}$$

Next, we aggregate machines into cells by employing the same modified K-mean algorithm presented earlier in the case of parts. In this case, machines are represented in IR^{N2} by a point. N2 axes are used to represent the processing time required by the N2 families on a machines. All axes are normalized in the interval [0,100], in order to remove the bias of different orders of numerical values for processing times, while

determining a representation in a standard space. The clusters are permitted iff the Euclidean distance from the center of a cluster to its corresponding points does not exceed a user specified parameter, $\pi 1$. This parameter is helpful in controlling the compactness of clusters, and is kept low for the model M^2. A good clustering should possess similar characteristics as presented in the case of parts. Based on these criteria, a clustering which resulted in 14 cells was obtained. The details of this grouping are presented in Appendix B.2. The processing times of the families on the cells are derived from the maximal processing time on the machines that belong to that cell. Note that, functionally similar machines are treated as parallel machines, thus, the effective processing time is derived by diving the processing time by the number of machines. This assumption suggests that we always split a batch to multiple machines, at the expense of several set-ups; we choose to neglect this detail as this may be averted at the scheduling level. Hereafter (i.e. for level 2 and all higher levels) machine entities will be considered as unique (i.e. without multiple copies). More precisely, if clustering results in M2 number of cells, i.e. $m2_1, m2_2, \ldots, m2_{M2}$, we have:

$$t2_{ij} = \max_{b \mid m1_b \in m2_j} \{t2'_{ib}/mn_b\} \qquad i = 1,2,\ldots,N2,\ j = 1,\ldots,M2 \tag{2.8}$$

where, mn_j represents the number of functionally similar machines of work-center type m_j; $j = 1,2,\ldots,M1$.

Now, the operating cost of a cell is the summation of the operating costs of its component machines.

$$mc2_j = \sum_{b \mid m1_b \in m2_j} mc1_b \qquad j = 1,2,\ldots,M2 \tag{2.9}$$

where, $mc1_b$ represents the operating cost of machine type m_b, and $mc2_j$ represents the operating cost of cell $m2_j$. This assumes that all machines of a cell are used by all parts visiting this cell, which in principle is consistent with our aggregation schemes.

The number of failure modes of the cell is the maximal number of failure modes of its component machines. The mean time between failure

(MTBF) and mean time to repair (MTTR) for each failure mode is calculated as follows :

$$mf2^k_j = \min_{b \mid m1_b \in m2_j} \left\{ \frac{mf^k_b}{mf^k_b + mr^k_b} \right\} \times \max_{b \mid m1_b \in m2_j} \{mf^k_b + mr^k_b\}$$

$$i = 1,2,\ldots,N2,\ j = 1,\ldots,M2 \qquad (2.10)$$

$$mr2^k_j = \max_{b \mid m1_b \in m2_j} \{mf^k_b + mr^k_b\} - mf2^k_j,\ i = 1,2,\ldots,N2,\ j = 1,\ldots,M2 \qquad (2.11)$$

These are formulated on the basis that the capacity of the cell is the minimal among the capacity (or availability) of the component machines, and the response time (i.e., the mean duration of a failure plus the repair) of the failure mode is the maximal among those of the component machines. Depending on the manufacturing system, and the buffers permitted between machines, we can alternately formulate (2.10) as follows:

$$mf2^k_j = \prod_{b \mid m1_b \in m2_j} \left\{ \frac{mf^k_b}{mf^k_b + mr^k_b} \right\} \times \max_{b \mid m1_b \in m2_j} \{mf^k_b + mr^k_b\}$$

$$i = 1,2,\ldots,N2,\ j = 1,\ldots,M2 \qquad (2.12)$$

This has been formulated under the assumption of zero buffers between part operations, which results in a rather pessimistic approximation of the cell capacity.

The workers follow the aggregation of machines, i.e. the workers operating the set of machines which are grouped in a cell, are grouped in to form the aggregate worker entity. Thus, the clustering results in W2 (W2 = M2) number of aggregate worker entities, i.e. $w2_1$, $w2_2$,...,$w2_{W2}$. The regular, overtime, hiring and firing costs of an aggregate worker entity is the summation of the corresponding costs of its component workers. For instance, the regular costs are computed as follows :

$$wcr2_j = \sum_{b \mid w1_b \in w2_j} wrc1_b \qquad j = 1,2,\ldots,W2 \qquad (2.13)$$

The other costs are computed similarly. This completes the definition of the model at level 2.

6.2.4 Formal Description of D^1

Having developed the model at level 2, we return to finalize the decision making problem at level 1. In this section, we formally present the problem D^1 (*P6.1*) :

$$\text{Min: } \sum_{k1=1}^{z1}\sum_{i=1}^{N1} (c1_i^{+} \times [Q1_{ik1} - D1_{ik1}]^{+} + c1_i^{-} \times [D1_{ik1} - Q1_{ik1}]^{+}) +$$

$$\sum_{k2=1}^{g1}\sum_{j=1}^{N2} (F \times c2_j^{+} \times [Q2'_{jk2} - Q2_{jk2}]^{+} + F \times c2_j^{-} \times [Q2_{jk2} - Q2'_{jk2}]^{+}) \quad (2.14)$$

$D1_{ik1}$ denotes the cumulative demand of part $p1_i$ at the end of the k1-th period (Δ^1), and is represented as :

$$D1_{ik1} = \sum_{a=1}^{k1} d1_{ia} ; \qquad i = 1,2,\ldots,N1, \text{ and } k1 = 1,2,\ldots,z1 \quad (2.15)$$

where, $d1_{ia}$ is the effective demand (for definition, see Appendix A.1) of part $p1_i$ in the a-th elementary period. $Q2_{jk2}$ denotes the cumulative production of family $p2_j$ computed (or specified) by level 2 at the end of the k2-th period (Δ^2). F is a factor >> 1 (F = 100 for the example), which is employed to magnify the penalty, or to force the level 1 to comply with the level 2 decision.

$$\text{Subject to : } Q1_{ik1} = \sum_{a=1}^{k1} x_{ia} ; \qquad i = 1,2,\ldots,N1, \text{ and } k1 = 1,2,\ldots,z1 \quad (2.16)$$

Where $Q1_{ik1}$ denotes the cumulative production of part $p1_i$ at the end of the k1-th period (Δ^1), and x_{ik1} denotes the production of product entity $p1_i$ during the k1-th period (Δ^1).

$Q2'_{jk2}$ denotes the cumulative production of family $p2_j$ at the end of the k2-th period (Δ^2), and is represented as :

$$Q2'_{ik2} = \sum_{b \mid p1_b \in p2_i} Q1_{b(k2.h1)} \qquad i = 1,2,\ldots,N2, \text{ and } k2 = 1,2,\ldots,g1 \quad (2.17)$$

$$\sum_{i=1}^{N1} t1_{ij}\, x_{ik1} \le a1_{jk1} \times mn_j \times e1_j; \qquad j = 1,2,\ldots,M1, \text{ and } k1 = 1,2,\ldots,z1 \quad (2.18)$$

$a1_{jk1}$ is the capacity in terms of time units specified by level 2. $e1_j$ is the efficiency of machine $m1_j$.

$$x_{ik1} \ge 0; \qquad i = 1,2,\ldots,N1, \text{ and } k1 = 1,2,\ldots,z1 \quad (2.19)$$

The objective function is composed of two parts : (i) inventory holding and backlogging costs for parts, and (ii) penalty for producing over or under the quantities specified by the high level. Constraint (2.18) represents the capacity constraints on each machine. Constraint (2.19) implies the non-negativity of production.

To solve (*P6.1*), we have to transform it into a standard linear programming form as follows (*P6.2*) :

$$\text{Min:} \quad \sum_{k1=1}^{z1}\sum_{i=1}^{N1} I_{ik1} + \sum_{k2=1}^{g1}\sum_{j=1}^{N2} P_{jk2} \quad (2.20)$$

Subject to :

$$I_{ik1} \ge c1_i^{+} \times [\sum_{a=1}^{k1} x_{ia} - \sum_{a=1}^{k1} d1_{ia}]; \quad i = 1,2,\ldots,N1, \text{ and } k1 = 1,2,\ldots,z1 \quad (2.21)$$

$$I_{ik1} \ge c1_i^{-} \times [\sum_{a=1}^{k1} d1_{ia} - \sum_{a=1}^{k1} x_{ia}]; \quad i = 1,2,\ldots,N1, \text{ and } k1 = 1,2,\ldots,z1 \quad (2.22)$$

$$P_{jk2} \ge F \times c2_j^{+} \times [\sum_{a=1}^{k2..h1} \sum_{b \mid p1_b \in p2_i} x_{ba} - Q2_{jk2}];$$

$$j = 1,2,\ldots,N2, k2 = 1,2,\ldots,g1 \quad (2.23)$$

$$P_{jk2} \ge F \times c2_j^{-} \times [Q2_{jk2} - \sum_{a=1}^{k2..h1} \sum_{b \mid p1_b \in p2_i} x_{ba}];$$

$$j = 1,2,\ldots,N2, k2 = 1,2,\ldots,g1 \quad (2.24)$$

$$\sum_{i=1}^{N1} t1_{ij}\, x_{ik1} \le a1_{jk1} \times mn_j \times e1_j; \qquad j = 1,2,\ldots,M1, \text{ and } k1 = 1,2,\ldots,z1 \quad (2.25)$$

The variables of the problem are $x_{(N1 \times z1)}$, $I_{(N1 \times z1)}$, and $P_{(N2 \times g1)}$, which are all non-negative. Finally, there are $(2 \times N1 \times z1 + 2 \times N2 \times g1 + M1 \times z1)$ number of constraints. The complexity of the problem is evident now and we can determine the length of the horizon H^1.

$z1 = 10$ implies horizon $\underline{H^1 = 2 \text{ weeks}}$. This leads to 1040 variables and 1420 constraints for our example, that can be solved using XMP [78] on a SUN/SPARC Station I+ platform in less than 20 minutes, which is claimed to be an acceptable solution time for this type of problem.

6.2.5 D^2 : Decision Making Problem at Level 2

1. Elementary period, $\Delta^2 = 5$ days = 1 week
   - Δ^2 is a multiple of Δ^1 (5 days is a multiple of 1 day).
   - $\Delta^2 \leq H^1$ (2 weeks).
   - Δ^2 < response time of decision of worker overtime levels (response time O(2-3 days)).

2 Controls, U^2

   1 Production levels for product entities at level 2 (families) on machine entities at level 2 (cells) for elementary periods of duration Δ^2 .

   2 Decision of worker overtime levels (response time O(2-3 days)).

3 Criteria, C^2

   1 Holding and backlogging cost for product entities at level 2.

   2 Minimization of overtime costs.

4 Constraints, T^2

The exact higher level constraints T^2, are not known because the high level remains undefined at this time. These constraints consist of the two following components : (i) aggregate resource capacity levels (only regular) represent high level constraints, which are intrinsic constraints relating to resource (machine and worker entity) capacities, and (ii) aggregate production constraints related to the production levels of aggregate products that must be respected by the current level during disaggregation. The first type, aggregate resource constraints, are easily transformed into

local resource constraints. The second type of constraints are treated according to the definition of consistent disaggregation (see section 4.9.3 of chapter 4, esp. (9.28)). Here, a chance for infeasible level 2 solutions (possible due to disaggregation inconsistencies) is less likely than in the previous case (D^1). This is because level 2 can plan capacity in the form of overtime levels, rendering some flexibility in capacity constraints.

The complexity of the decision making problem can be roughly assessed in a similar manner as presented in section 6.2.2. The determination of the length of the horizon H^2, or the number of elementary periods, z2, of duration Δ^2 that can be included in the horizon H^2, is performed in the finalization stage (see section 6.2.7), when the exact form and number of variables and constraints are known.

6.2.6 M^3 : Model at Level 3

In this section, we present the development of model M^3 from the aggregation of entities in the model M^2. This process follows exactly the same approach as for the development of model M^2 from model M^1.

We aggregate level 2 product entities to level 3 product entities by employing the same K-means clustering algorithm. Products are represented in IR^{M2+2} (IR^{16} in this case) by a point. M2 axes are used to represent the processing time required by level 2 product entities on M2 machine entities (cells), and in addition, two axes are used to represent the holding cost and backlogging cost, respectively. Once again, all axes are normalized in the interval [0,100] for unbiasedness. The clusters are permitted iff the Euclidean distance from the center of a cluster to its corresponding points does not exceed a user specified parameter, $\sigma 2$ ($\sigma 2 > \sigma 1$ to allow a higher degree of aggregation). A clustering which resulted in 7 part entities was obtained. The details of this grouping are presented in Appendix B.3. The processing times of the level 3 product entities on the level 2 machine entities and cost attributes are calculated from the center of gravity of the clusters; this time each point is assigned a weight corresponding to the long term demand of that entity. Now, the long term

demand of a family is the summation of the long term demands of its component products. More precisely, if clustering result in N3 (7 for this example) number of families, i.e. $p3_1$, $p3_2$,...,$p3_{N3}$, we have :

$$c3_i^{+(-)} = h1 \times h2 \times \sum_{b \mid p2_b \in p3_i} r2_b c2_b^{+(-)} = h1 \times h2 \times \frac{n2_b c2_b^{+(-)}}{\sum_{b \mid p2_b \in p3_i} n2_b};$$

$$i = 1,2,\ldots,N3 \qquad (2.26)$$

$$t3'_{ij} = \sum_{b \mid p2_b \in p3_i} r2_b t2_{bj} = \frac{n2_b t2_{bj}}{\sum_{b \mid p2_b \in p3_i} n2_b}; \qquad i = 1,2,\ldots,N3,\ j = 1,\ldots,M2 \qquad (2.27)$$

$$n3_i = \sum_{b \mid p2_b \in p3_i} n2_b \qquad i = 1,2,\ldots,N3 \qquad (2.28)$$

Next, we aggregate level 2 machine entities into level 3 machine entities by employing the same modified K-means algorithm. In this case, level 2 machine entities are represented in IR^{N3} by a point. N3 axes are used to represent the processing time required by the N3 product entities on a machine entity. Axes are normalized in the interval [0,100], for reasons explained above. The clusters are permitted iff the Euclidean distance from the center of a cluster to its corresponding points does not exceed a user specified parameter, $\pi 2$ ($\pi 2 > \pi 1$ to allow a higher degree of aggregation). A clustering which resulted in 6 machine entities was obtained. The details of this grouping are presented in Appendix B.3. The processing times of the level 3 product entities on the level 3 machine entities are derived from the maximal processing time on the level 2 machine entities that belong to that particular level 3 machine entity. Note, in this case there are no functionally similar machine entities (they are unique, i.e. without multiple copies). More precisely, if clustering results in M3 number of cells, i.e. $m3_1$, $m3_2$,...,$m3_{M3}$, we have:

$$t3_{ij} = \max_{b \mid m2_b \in m3_j} \{t3'_{ib}\} \qquad i = 1,2,\ldots,N3,\ j = 1,\ldots,M3 \qquad (2.29)$$

Now, the operating cost of a cell is the summation of the operating costs of its component machines.

$$mc3_j = \sum_{b \mid m2_b \in m3_j} mc2_b \qquad j = 1,2,\ldots,M3 \qquad (2.30)$$

The number of failure modes, and the mean time between failure (MTBF) and mean time to repair (MTTR) for each failure mode is calculated in a similar fashion as described in section 6.2.3. The aggregation of worker entities, and their resulting attributes are also calculated by the method detailed in section 6.2.3.

This completes the definition of the model at level 3.

6.2.7 Formal Description of D^2

Having developed the model at level 3, we return to finalize the decision making problem at level 2. In this section, we formally present the problem D^2 (*P6.3*) :

$$\text{Min: } \sum_{k2=1}^{z2}\sum_{i=1}^{N2} (c2_i^{+} \times [Q2_{ik2} - D2_{ik2}]^{+} + c2_i^{-} \times [D2_{ik2} - Q2_{ik2}]^{+}) + \sum_{k2=1}^{z2}\sum_{j=1}^{M2} wco2_j \times o2_{jk2} \quad (2.31)$$

$D2_{ik2}$ denotes the cumulative demand of product entity $p2_i$ at the end of the k2-th period (Δ^2), and is represented as :

$$D2_{ik2} = \sum_{a=1}^{k2} d2_{ia} \; ; \qquad i = 1,2,\ldots,N2, \text{ and } k2 = 1,2,\ldots,z2 \quad (2.32)$$

where, $d2_{ia}$ is the effective demand of product entity $p2_i$ in the a-th elementary period (Δ^2).

$$\text{Subject to : } Q2_{ik2} = \sum_{a=1}^{k2} y_{ia} \; ; \qquad i = 1,2,\ldots,N2, \text{ and } k2 = 1,2,\ldots,z2 \quad (2.33)$$

Where $Q2_{ik1}$ denotes the cumulative production of product entity $p2_i$ at the end of the k2-th period (Δ^2), and y_{ik2} denotes the production of product entity $p2_i$ during the k2-th period (Δ^2).

$$\sum_{i=1}^{N2} t2_{ij}\, y_{ik2} - e2_j \times o2_{jk2} \le a2_{jk2} \times e2_j; \; j = 1,2,\ldots,M2, \text{ and } k2 = 1,2,\ldots,z2 \quad (2.34)$$

$a2_{jk2}$ is the regular time capacity in terms of time units specified by level 3. $e2_j$ is the efficiency of the machine entity $m2_j$.

$$1/\beta \times o2_{jk2} \le a2_{jk2}; \qquad j = 1,2,\ldots,M2, \text{ and } k2 = 1,2,\ldots,z2 \quad (2.35)$$

$$\sum_{b \mid p2_b \in p3_i} Q2_{b(k3.h2)} \ge Q3_{ik3}\ ; \qquad i = 1,2,\ldots,N3, \text{ and } k3 = 1,2,\ldots,g2 \quad (2.36)$$

$Q3_{jk3}$ denotes the cumulative production of product entity $p3_j$ computed (or specified) by level 3 at the end of the k3-th period (Δ^3).

$$y_{ik2} \ge 0\ ; \qquad i = 1,2,\ldots,N2, \text{ and } k2 = 1,2,\ldots,z2 \quad (2.37)$$

$$o2_{jk2} \ge 0\ ; \qquad j = 1,2,\ldots,M2, \text{ and } k2 = 1,2,\ldots,z2 \quad (2.38)$$

The objective function is composed of two parts : (i) inventory holding and backlogging costs for product entities, and (ii) labor overtime costs. Constraint (2.34) represents the capacity constraints on each machine entity. Constraint (2.35) represents that the overtime planned cannot be more than β times the corresponding regular time specified by the upper level. (2.36) represents the "consistency constraint", that ensures consistency with the high level plan. Constraints (2.37) and (2.38) imply the non-negativity of production and overtime, respectively.

To solve (*P6.3*), we have to transform it into a standard linear programming form as follows (*P6.4*) :

$$\text{Min:} \quad \sum_{k2=1}^{z2} \sum_{i=1}^{N2} J_{ik2} + \sum_{k2=1}^{z2} \sum_{j=1}^{M2} wco2_j \times o2_{jk2} \quad (2.39)$$

Subject to :

$$J_{ik2} \ge c2_i^{+} \times [\sum_{a=1}^{k2} y_{ia} - \sum_{a=1}^{k2} d2_{ia}]\ ; \quad i = 1,2,\ldots,N2, \text{ and } k2 = 1,2,\ldots,z2 \quad (2.40)$$

$$J_{ik2} \ge c2_i^{-} \times [\sum_{a=1}^{k2} d2_{ia} - \sum_{a=1}^{k2} y_{ia}]\ ; \quad i = 1,2,\ldots,N2, \text{ and } k2 = 1,2,\ldots,z2 \quad (2.41)$$

$$\sum_{i=1}^{N2} t2_{ij}\, y_{ik2} - e2_j \times o2_{jk2} \le a2_{jk2} \times e2_j; \quad j = 1,2,\ldots,M2, \,\&\, k2 = 1,2,\ldots,z2 \quad (2.42)$$

$$1/\beta \times o2_{jk2} \le a2_{jk2}; \qquad j = 1,2,\ldots,M2, \text{ and } k2 = 1,2,\ldots,z2 \quad (2.43)$$

$$\sum_{b \mid p2_b \in p3_i} Q2_{b(k3.h2)} \geq Q3_{ik3}\ ; \qquad i = 1,2,\ldots,N3, \text{ and } k3 = 1,2,\ldots,g2 \quad (2.44)$$

The variables of the problem are $y_{(N2 \times z2)}$, $J_{(N2 \times z2)}$, and $o2_{(M2 \times z2)}$, which are all non-negative. Finally, there are $(2 \times N2 \times z2 + 2 \times M2 \times z2 + N3 \times g2)$ number of constraints. The complexity of the problem is evident now and we can determine the length of the horizon H^2.

$z2 = 12$ implies horizon $\underline{H^2 = 3 \text{ months}}$. This leads to 648 variables and 837 constraints for our example, that can be solved using XMP [78] on a SUN/SPARC Station I+ platform in less than 5 minutes, which is claimed to be an acceptable solution time for this type of problem. One can now appreciate that model reduction has permitted us to address a larger horizon, while introducing additional controls at the same time.

6.2.8 D^3 : Decision Making Problem at Level 3

1 Elementary period, Δ^3 = 4 weeks = 1 month

- Δ^3 is a multiple of Δ^1 (1 month is a multiple of 1 week).
- $\Delta^3 \leq H^2$ (3 months).
- Δ^3 < response time of hiring and firing workers (response time O(2-3 weeks)).

2 Controls, U^3

1 Production levels for product entities at level 3 on machine entities at level 3 for elementary periods of duration Δ^3 .

2 Decision of worker overtime levels (response time O(2-3 days)).

3 Decision of hiring or firing workers (response time O(2-3 weeks)).

3 Criteria, C^3

1 Holding and backlogging cost for product entities at level 2.

2 Minimization of overtime costs.

3 Minimization of hiring and firing costs.

4 Constraints, T^3

This level includes all the controls specified U^0, thus the set of higher level constraints T^3, is empty. However, T^3 can include external or

strategic constraints. An example of this is the upper bound of regular time that can be assigned to a particular machine entity.

Check for Termination conditions

At this stage :

- all controls belonging to U^0 have been addressed.
- all criteria belonging to C^0 have been addressed.
- the desired degree of aggregation is accomplished.

If :

- H^3 > duration that can absorb the most significant disturbances, both internal as well as external (e.g. demand seasonality over a year).

we have completed the design, i.e. the hierarchy will consists of 3 levels. However, if it turns out that the maximal H^3 cannot absorb the most significant disturbances, then we have to introduce another (or more) higher levels. In this case, level 3 will receive some additional higher level constraints. As a first attempt, however, we formulate the problem without the existence of a higher, i.e. fourth, level (and the associated constraints).

We formally present the problem D^3 (*P6.5*) :

$$\text{Min: } \sum_{k3=1}^{z3}\sum_{i=1}^{N3} (c3_i^{+} \times [Q3_{ik3} - D3_{ik3}]^{+} + c3_i^{-} \times [D3_{ik3} - Q3_{ik3}]^{+}) +$$

$$\sum_{k3=1}^{z3}\sum_{j=1}^{M3} wcr3_j \times rh3_{jk3} + \sum_{k3=1}^{z3}\sum_{j=1}^{M3} wco3_j \times o3_{jk3} +$$

$$\sum_{k3=1}^{z3}\sum_{j=1}^{M3} whc3_j \times hl3_{jk3} + \sum_{k3=1}^{z3}\sum_{j=1}^{M3} wfc3_j \times fl3_{jk3} \qquad (2.45)$$

$D3_{ik3}$ denotes the cumulative demand of product entity $p3_i$ at the end of the k3-th period (Δ^3), and is represented as :

$$D3_{ik3} = \sum_{a=1}^{k3} d3_{ia} ; \qquad i = 1,2,\ldots,N3, \text{ and } k3 = 1,2,\ldots,z3 \quad (2.46)$$

where, $d3_{ia}$ is the effective demand of product entity $p3_i$ in the a-th elementary period (Δ^3). $rh3_{jk3}$ and $o3_{jk3}$ represent the regular and overtime hours (or time units) of worker entity $w3_j$ during the k3-th period, respectively. $hl3_{jk3}$ and $fl3_{jk3}$ represent the hours (or time units) of worker entity $w3_j$ hired and fired during the k3-th period, respectively.

$$\text{Subject to : } Q3_{ik3} = \sum_{a=1}^{k3} z_{ia} \; ; \qquad i = 1,2,\ldots,N3, \text{ and } k3 = 1,2,\ldots,z3 \quad (2.47)$$

Where $Q3_{ik3}$ denotes the cumulative production of product entity $p3_i$ at the end of the k3-th period (Δ^3), and z_{ik3} denotes the production of product entity $p3_i$ during the k3-th period (Δ^3).

$$\sum_{i=1}^{N3} t3_{ij}\, z_{ik3} - e3_j \times rh3_{jk3} - e3_j \times o3_{jk3} \le 0;$$

$$j = 1,2,\ldots,M3, \text{ and } k3 = 1,2,\ldots,z3 \quad (2.48)$$

$e3_j$ is the efficiency of the machine entity $m3_j$.

$$rh3_{jk3} - rh3_{j\,(k3\,-\,1)} - hl3_{jk3} + fl3_{jk3} = 0;$$

$$j = 1,2,\ldots,M3, \text{ and } k3 = 1,2,\ldots,z3 \quad (2.49)$$

$$\beta \times rh3_{jk3} - o3_{jk3} \le 0; \qquad j = 1,2,\ldots,M3, \text{ and } k3 = 1,2,\ldots,z3 \quad (2.50)$$

$$rh3_{jk3} \le mrh3_j; \qquad j = 1,2,\ldots,M3, \text{ and } k3 = 1,2,\ldots,z3 \quad (2.51)$$

$$z_{ik3} \ge 0 \; ; \qquad i = 1,2,\ldots,N3, \text{ and } k3 = 1,2,\ldots,z3 \quad (2.52)$$

$$rh3_{jk3},\, o3_{jk3},\, hl3_{jk3},\, fl3_{jk3}, \ge 0 \; ; \qquad j = 1,2,\ldots,M3, \text{ and } k3 = 1,2,\ldots,z3 \quad (2.53)$$

The objective function is composed of five parts : (i) inventory holding and backlogging costs for product entities, (ii) labor regular costs, (iii) labor overtime costs, (iv) labor hiring costs, and (v) labor firing costs. Constraint (2.48) represents the capacity constraints on each machine entity. Constraint (2.49) represents the conservation of worker levels. Constraint (2.50) ensures that the overtime planned cannot be more than β times the corresponding regular time. Finally, constraint (2.51) implies that the number of regular hours cannot be more than the specified permissible limit. This limit is computed from the level 1 (physical) limit $mrh1_b$.

$$mrh2_i = \min_{b \mid m1_b \in m2_i} \{mrh1_b \times h1\} \qquad i = 1,2,\ldots,M2 \qquad (2.54)$$

$$mrh3_i = \min_{b \mid m2_b \in m3_i} \{mrh2_b \times h2\} \qquad i = 1,2,\ldots,M3 \qquad (2.55)$$

The number of initially available worker hours at level 3, $rh3_{j0}$, is also computed from level 1 in a similar fashion. That is,

$$rh2_{i0} = \min_{b \mid m1_b \in m2_i} \{rh1_{b0} \times h1\} \qquad i = 1,2,\ldots,M2 \qquad (2.56)$$

$$rh3_{i0} = \min_{b \mid m2_b \in m3_i} \{rh2_{b0} \times h2\} \qquad i = 1,2,\ldots,M3 \qquad (2.57)$$

To solve ($\mathcal{P}6.6$), we have to transform it into a standard linear programming form as follows ($\mathcal{P}6.6$) :

$$\text{Min: } \sum_{k3=1}^{z3}\sum_{i=1}^{N3} K_{ik3} + \sum_{k3=1}^{z3}\sum_{j=1}^{M3} wcr3_j \times rh3_{jk3} + \sum_{k3=1}^{z3}\sum_{j=1}^{M3} wco3_j \times o3_{jk3} +$$

$$\sum_{k3=1}^{z3}\sum_{j=1}^{M3} whc3_j \times hl3_{jk3} + \sum_{k3=1}^{z3}\sum_{j=1}^{M3} wfc3_j \times fl3_{jk3} \qquad (2.58)$$

Subject to :

$$K_{ik3} \geq c3_i^{+} \times [\sum_{a=1}^{k3} z_{ia} - \sum_{a=1}^{k3} d3_{ia}] \; ; \; i = 1,2,\ldots,N3, \text{ and } k3 = 1,2,\ldots,z3 \qquad (2.59)$$

$$K_{ik3} \geq c3_i^{-} \times [\sum_{a=1}^{k3} d3_{ia} - \sum_{a=1}^{k3} z_{ia}] \; ; \; i = 1,2,\ldots,N3, \text{ and } k3 = 1,2,\ldots,z3 \qquad (2.60)$$

$$\sum_{i=1}^{N3} t3_{ij}\, z_{ik3} - e3_j \times rh3_{jk3} - e3_j \times o3_{jk3} \leq 0;$$

$$j = 1,2,\ldots,M3, \text{ and } k3 = 1,2,\ldots,z3 \qquad (2.61)$$

$$rh3_{jk3} - rh3_{j\,(k3-1)} - hl3_{jk3} + fl3_{jk3} = 0;$$

$$j = 1,2,\ldots,M3, \text{ and } k3 = 1,2,\ldots,z3 \qquad (2.62)$$

$$\beta \times rh3_{jk3} - o3_{jk3} \leq 0; \qquad j = 1,2,\ldots,M3, \text{ and } k3 = 1,2,\ldots,z3 \qquad (2.63)$$

$$rh3_{jk3} \leq mrh3_j; \qquad j = 1,2,\ldots,M3, \text{ and } k3 = 1,2,\ldots,z3 \qquad (2.64)$$

The variables of this problem are $z_{(N3 \times z3)}$, $K_{(N3 \times z3)}$, $rh3_{(M3 \times z3)}$, $o3_{(M3 \times z3)}$, $hl3_{(M3 \times z3)}$, and $fl3_{(M3 \times z3)}$, which are all non-negative. Finally, there are $(2 \times N3 \times z3 + 4 \times M3 \times z3)$ number of constraints. The complexity of the problem is evident now and we can determine the length of the horizon H^3.

$z3 = 12$ implies horizon $\underline{H^3 = 1 \text{ year}}$. This leads to 456 variables and 456 constraints for our example, that can be solved using XMP [78] on a SUN/SPARC Station I+ platform in less than 2 minutes, which is claimed to be an acceptable solution time for this type of problem. H^3 has a duration that can absorb the most significant disturbances, both internal as well as external (e.g. demand seasonality over a year). Thus, the design of the hierarchy is complete at this stage (note that, the other termination conditions were already satisfied).

6.3 Operation of the Hierarchy

In this section, we present the operation of the hierarchy as a top-down computation procedure. First, we detail how the quantities given at the physical level (level 1) are translated into inputs for higher levels. Then, we detail the top-down solution procedure, and how the results of a particular decision level are propagated to the the subsequent lower level.

6.3.1 Translation of Physical Quantities for Higher Levels

As mentioned earlier, the operation of the hierarchy is essentially a top-down process. However, the quantities (e.g. demand, initial and maximal worker levels) that serve as inputs to the various levels of the hierarchy are specified only at the physical level. Thus, we have to translate these physical quantities into appropriate aggregate quantities.

First, the demand of the (physical) level 1 product entities is netted against the on hand inventory to obtain the effective demand, $d1_{bk1}$, $b = 1,2,\ldots,N1$, and $k1 = 1,2,\ldots,z3 \times h2 \times h1$. The definition and procedure to compute this effective demand has been detailed in Appendix A.1. The demand of the

aggregate product entities at level 2 and 3 are computed by the aggregation of $d1_{ik1}$ over time and products as follows :

$$d2_{ik2} = \sum_{a=(k2-1).h1}^{k2.h1} \sum_{b \mid p1_b \in p2_i} d1_{ba} ; \quad i = 1,2,\ldots,N2,\ k2 = 1,2,\ldots,z3 \times h2 \qquad (3.1)$$

$$d3_{ik3} = \sum_{a=(k3-1).h2}^{k3.h2} \sum_{b \mid p2_b \in p3_i} d2_{ba} ; \quad i = 1,2,\ldots,N3,\ k3 = 1,2,\ldots,z3 \qquad (3.2)$$

The maximal number of permissible regular hours for level 1 worker entities $mrh1_b$, is translated to the corresponding quantities for level 2 and 3 worker entities by (2.54) and (2.55), respectively.

Similarly, the initial number of regular hours available for level 1 worker entities $rh1_{b0}$, is translated to the corresponding quantities for level 2 and 3 worker entities by (2.56) and (2.57), respectively.

6.3.2 Solution of Level 3

Having obtained the quantities as detailed in the previous section, the first problem solved is the level 3 linear programming problem (*P6.6*).

The following serve as inputs to this problem :

1) Demand $d3_{ik3}$; i = 1,2,...,N3, k3 = 1,2,...,z3. See (3.2).

2) Maximal worker levels $mrh3_i$; i = 1,2,...,N3. See (2.55 and 2.54).

3) Initial worker levels $rh3_{i0}$; i = 1,2,...,N3. See (2.57 and 2.56).

The linear programming problem (*P6.6*) is solved, and the following quantities are obtained as a result of the optimization.

1) Production levels z_{ik3}; i = 1,2,...,N3, k3 = 1,2,...,z3.

2) Regular worker levels $rh3_{ik3}$; i = 1,2,...,N3, k3 = 1,2,...,z3.

3) Overtime worker levels $o3_{ik3}$; i = 1,2,...,N3, k3 = 1,2,...,z3.

Controls :

The control relevant to this level is the hiring and firing of level 1 worker entities, or determining the number of regular hours of level 1 worker entities. This is performed by the following backward translation :

$$rh2_{ik2} = rh3_{bk3}/h2; \qquad \forall i \mid m2_i \in m3_b,\ b = 1,2,\ldots,M3$$

$$k2 \in ((k3-1).h2,\ k3.h2] \tag{3.3}$$

$$rh1_{ik1} = rh2_{bk2}/h1; \qquad \forall i \mid m1_i \in m2_b,\ b = 1,2,\ldots,M2$$

$$k1 \in ((k2-1).h1,\ k2.h1] \tag{3.4}$$

Compare with (2.56) and (2.57). We refer to this as "backward translation", because it takes us from higher levels down to lower levels.

The control regarding the overtime levels is disregarded. This control belongs to level 2 which computes it in a more detailed manner. The reason why this was included in the level 3 optimization is that level 3 computes the solution to the overall problem, by considering all criteria in an aggregate manner. This is to obtain a balance between conflicting criteria. The lower levels of the hierarchy do not present this perspective. They take a partial solution and locally optimize it for a subset of criteria, while making the solution more detailed (in time as well as for entities). This will be further clarified as we proceed in this section.

6.3.3 Solution of Level 2

The level 2 problem is the linear programming problem (*P6.4*). In this case, the inputs are obtained from the translation procedure of 6.3.1 as well as the solution to the previous level 3 problem.

The inputs to this problem are :

1) Demand $d2_{ik2}$; i = 1,2,...,N2, k2 = 1,2,...,z2. See (3.1).

2) Regular worker levels $a2_{jk2}$ (j = 1,2,...,M2, k2 = 1,2,...,z2), specified by level 3. It is employed in (2.42) and (2.43), and is computed as follows :

$$a2_{jk2} = rh2_{jk2} \qquad j = 1,2,\ldots,M2,\ k2 = 1,2,\ldots,z2 \tag{3.5}$$

3) Production levels of level 3 product entities $Q3_{ik3}$ (i = 1,2,...,N3, k3 = 1,2,...,g2); see (2.47). It is employed in (2.44).

The linear programming problem (*P6.4*) is solved, and the following quantities are obtained as a result of the optimization.

1) Production levels y_{ik2}; i = 1,2,...,N2, k2 = 1,2,...,z2.

2) Overtime worker levels $o2_{ik2}$; i = 1,2,...,N2, k2 = 1,2,...,z2.

The Control of this problem is :
The control relevant to this level is determining the number of overtime hours of level 1 worker entities. Once again, this is performed by a backward translation :

$$o1_{ik1} = o2_{bk2}/h1; \qquad \forall i \mid m1_i \in m2_b,\ b = 1,2,...,M2$$

$$k1 \in ((k2-1).h1,\ k2.h1] \qquad (3.6)$$

At this step of the top-down computation, the total worker levels (regular plus overtime) are completely determined. The level 1 takes a partial solution in this form, and locally optimizes it for holding and backlogging costs, while making the solution more detailed (in time as well as for entities).

6.3.4 Solution of Level 1

The level 1 problem is the linear programming problem (*P6.2*). In this case, the inputs are either given, or obtained from the solution of the previous problems.

The inputs to this problem are :

1) Demand $d1_{ik1}$; i = 1,2,...,N1, k2 = 1,2,...,z1.

2) Total worker levels $a1_{jk1}$ (j = 1,2,...,M1, k1 = 1,2,...,z1), specified by level 3 and 2. It is employed in (2.25), and is computed as follows :

$$a1_{jk1} = rh1_{jk1} + o1_{jk1} \qquad j = 1,2,...,M1,\ k1 = 1,2,...,z1 \qquad (3.7)$$

3) Production levels of level 2 product entities $Q2_{ik2}$ (i = 1,2,...,N2, k2 = 1,2,...,g1); see (2.47). It is employed in (2.23) and (2.24).

The linear programming problem (*P6.2*) is solved, and the following quantities are obtained as a result of the optimization.

1) Production levels x_{ik1}; i = 1,2,...,N1, k1 = 1,2,...,z1.

At this step, the top-down computation is complete. The production levels of the (physical) level 1 product entities are determined over horizon H^1.

6.3.5 Horizon Rolling and Recomputations

In the previous sections, we detailed the one pass top-down computation procedure for the operation of the hierarchy. In practice, for continued resolution of production planning in time, the horizons are rolled. The rolling horizon concept has already been presented in section 5.7.1; it was not implemented for this example due to the absence of optimal results for comparison. Furthermore, in section 5.7.2 we addressed the reaction of the hierarchy to random events and the recomputation procedure.

The hierarchy designed in this chapter, does not lend itself easily to performance evaluation as presented in the case of chapter 3. The reason is that a monolithic problem that solves the overall problem optimally will have a very high dimension, requiring substantially more powerful linear programming software (if any exists) as well as hardware. Thus, in the lack of this comparison, rolling horizons, and stochastic events have not been implemented in the software briefly described in the following section. This could easily be performed, but it would remain purely demonstrative. Therefore, instead, in section 6.5, we present some results relating to the solution of decision making problems at various levels.

6.4 Software

As a part of this work, a menu-driven software package "DOHPMS", Design and Operation of Hierarchical Production Management Systems,

was developed in C on a SUN/UNIX platform. The algorithms were coded and tested for a variety of data, and the results obtained were satisfactory. The individual programs in DOHPMS are run through a menu based Shell on the Unix system, and are accompanied by an introduction and help features.

The software consists of the following programs :

(1) Program to generate random product, work-center and worker data.

(2) Program to generate random demand, and initial worker levels.

(3) Programs for the Design of the Hierarchy :

  (a) Clustering of products.
  (b) Clustering of work-centers.
  (c) Clustering of workers.

(4) Programs for the Operation of the Hierarchy :

  (a) Aggregation of detailed demand.
  (b) Aggregation of initial and maximal worker levels.
  (c) Generation of the Level 3 decision making problem.
  (d) Generation of the Level 2 decision making problem.
  (e) Generation of the Level 1 decision making problem.

(5) Linear program solver : XMP [77] (In Fortran-77).

(6) Programs for graphics (Using "Super Mongo", sm).

A more detailed description of the programs and the software in general has been provided in Appendix D. The next section is devoted some numerical results that have been obtained by employing this software.

6.5 Numerical Results

This section is devoted to some numerical results for the one pass top-down solution procedure, obtained for a set of random test problems. Although we do not assess the optimality of the hierarchical approach, we present here the dimensions of the linear programming problems at each level of the hierarchy, and the c.p.u. times for the solution procedures. This provides insight into the performance and applicability of the

hierarchical model. At first, test examples were created; data related to the products, machines and workers were constructed. For each example, a set of random demands was generated. The generation of the examples is detailed as follows.

The representative routings, i.e. processing times of the set of finished products on the set of machines were generated by a uniform distribution. The set of products and machines were assigned *a priori* to groups and sub-groups. That is, entities belonging to a sub-group had attributes "close" to each other (in the sense of the relevant aggregation measure), and each of the sub-groups were also "close" to each other, but more "distant" than the previous case. Table 5.1(a) presents the details of this generation; column 4 represents the perturbation of processing times between elements of a sub-group, and column 5 represents the additional perturbation of processing times between elements of a group. This results in a nice "block" incidence product-machine incidence matrix, wherein a group of products utilize one or more groups of machines (with neighborhood processing times). This structure was "polluted" by an encroachment probability (0.3) of deletion of some existing product processing times on machines and the addition of some other processing times, in order to destroy the "blocks" while retaining some underlying basic block structure. It is hoped that this generation procedure results in data that bears resemblance to some real production data (especially, to the ones that are amenable to Group Technology); see also Nagi [64]. Product attributes of holding and backlogging costs were also generated by uniform distributions, employing the similar concepts of closeness. The long term production volumes of products were uniformly distributed.

Ex.	n	Processing times	Perturbation between sub-groups	Perturbation between groups
1	50	U(0.0,2.0)	±U(0,0.1)	±U(0,0.2) (in addition to first
2	50	U(1.0,3.0)	±U(0,0.2)	±U(0,0.2) perturbation)
3	40	U(1.0,3.0)	±U(0,0.2)	±U(0,0.2)

Table 5.1(a) : Product Processing Times

Ex.	n	Holding cost	Perturbation groups, sub-gps	Backlogging cost	Perturbation groups, sub-gp	Long term Production
1	50	U(1,9)	±5.0, ±0.1	U(50,100)	±50, ±1.0	U(0.1,5.0)
2	50	U(1,9)	±5.0, ±0.1	U(50,100)	±50, ±1.0	U(0.1,8.0)
3	40	U(1,9)	±5.0, ±0.1	U(50,100)	±50, ±1.0	U(0.1,8.0)

Table 5.1(b) : Product Costs and Volume Details

The work-center data was generated as follows. The number of machines per work-center was chosen in a manner that could satisfy the long term production volume of products. For each work-center, the operating cost, the number of failure modes, the mean time between failures (MTBF) and the mean time to repair (MTTR) for each failure mode were also generated at random. The first failure mode (if any) had a response time of the order of product lead times, and each subsequent failure mode had a response time of a magnitude higher than the previous one. See Table 5.2 for details (fm stands for failure mode number).

Ex.	m	Number of machines	Operating cost	Number of failure modes	MTTR groups, sub-gp	MTBF Production
1	34	1 or 2; to	U(.16,.416)	U(1,4)	U(1*fm,2*fm)	U(10*fm,20*fm)
2	34	satisfy long	U(.16,.416)	U(1,4)	U(0.1,0.2) 10^{fm}	U(10,20) 10^{fm}
3	30	term prodtn	U(.16,.416)	U(1,4)	U(0.1,0.2) 10^{fm}	U(10,20) 10^{fm}

Table 5.2 : Work-center Details

Workers were assigned to the corresponding machine numbers, and the regular labor cost and hiring cost were generated by uniform distributions. The overtime cost was chosen to be 1.25 times the corresponding regular cost, and the firing cost was chosen to be 2.5 times (10 times for example 1) the corresponding hiring cost. These factors are consistent with normal industrial practice. The initially available number of regular worker hours was computed based on the long term part production requirements, and the maximal worker level was 1.2 times higher. Note that, it could also be computed on the basis that each machine can operate for a given number of shifts. We do not present absenteeism details, since we had decided to

employ (2.3a) for computing combined resource efficiencies. See Table 5.3 for details.

Ex.	w	Assigned Machine	Regular Op. cost	Overtime cost	Hiring cost	Firing cost
1	34	as	U(.16,.416)	1.25 Reg cost	U(1.0,2.0)	10 Hiring cost
2	34	per	U(.33,.83)	1.25 Reg cost	U(0.3,1.0)	2.5 Hire cost
3	30	number	U(.33,.83)	1.25 Reg cost	U(0.3,1.0)	2.5 Hire cost

Table 5.3 : Worker Details

For each example, a set of random demands was generated for 240 elementary periods (240 working days ≈ 1 year). For the generation of this demand a magnitude of cyclicity belonging to [0,1], was generated at random for each product. Then, a sinosoidal demand with a random phase shift, and magnitude based on the long term product production volume and cyclicity was generated. This completes the static, dynamic and historical inputs relevant to the manufacturing system.

The design of the hierarchy is performed by the aggregation of entities to higher level entities as detailed in the previous sections of this chapter. Next, we present the numerical results in table 5.4. The linear programming solutions at each level were obtained using XMP [78] on a SUN/SPARC Station I+ platform.

Ex.	Level(i)	zi	Ni	Mi	Constraints	Variables	c.p.u. time, sec
1	3	12	7	6	456	456	81.0
	2	12	20	14	837	648	281.8
	1	10	50	34	1420	1040	1493.8
2	3	12	7	6	456	456	65.0
	2	12	20	14	837	648	266.7
	1	10	50	34	1420	1040	1493.8
3	3	12	6	6	432	432	29.7
	2	12	14	12	642	480	55.5
	1	10	40	30	1156	828	278.2

Table 5.4 : Numerical results for 3 examples

Results of the top-down solution procedure of levels 3 and 2 of example 2, presented in a graphical form, are presented in figures 6.1 through 6.4. The solution of level 1 has been included in Appendix C.

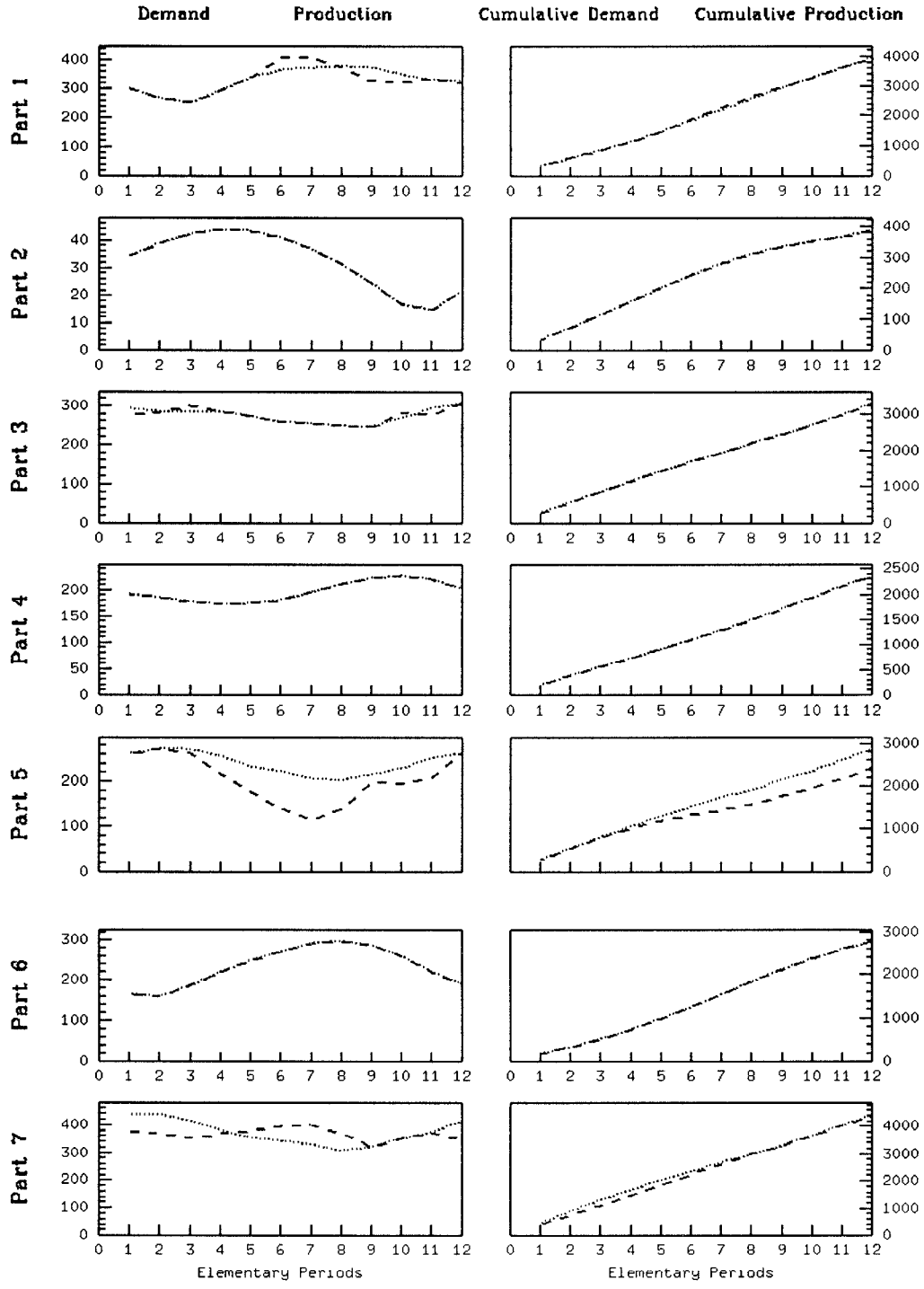


Figure 6.1 : Production Levels of Level 3 Product Entities

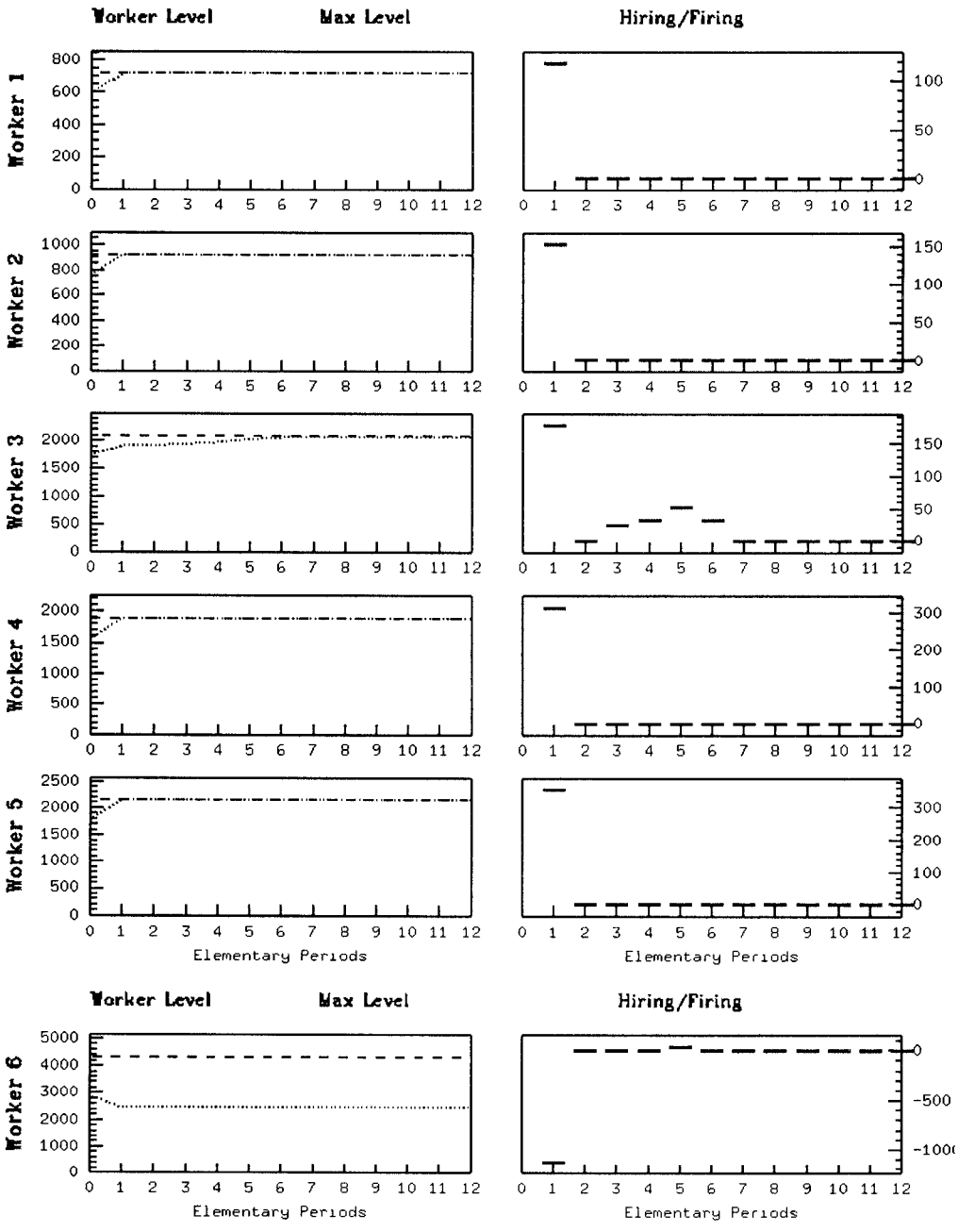


Figure 6.2 : Worker Levels of Level 3 Worker Entities

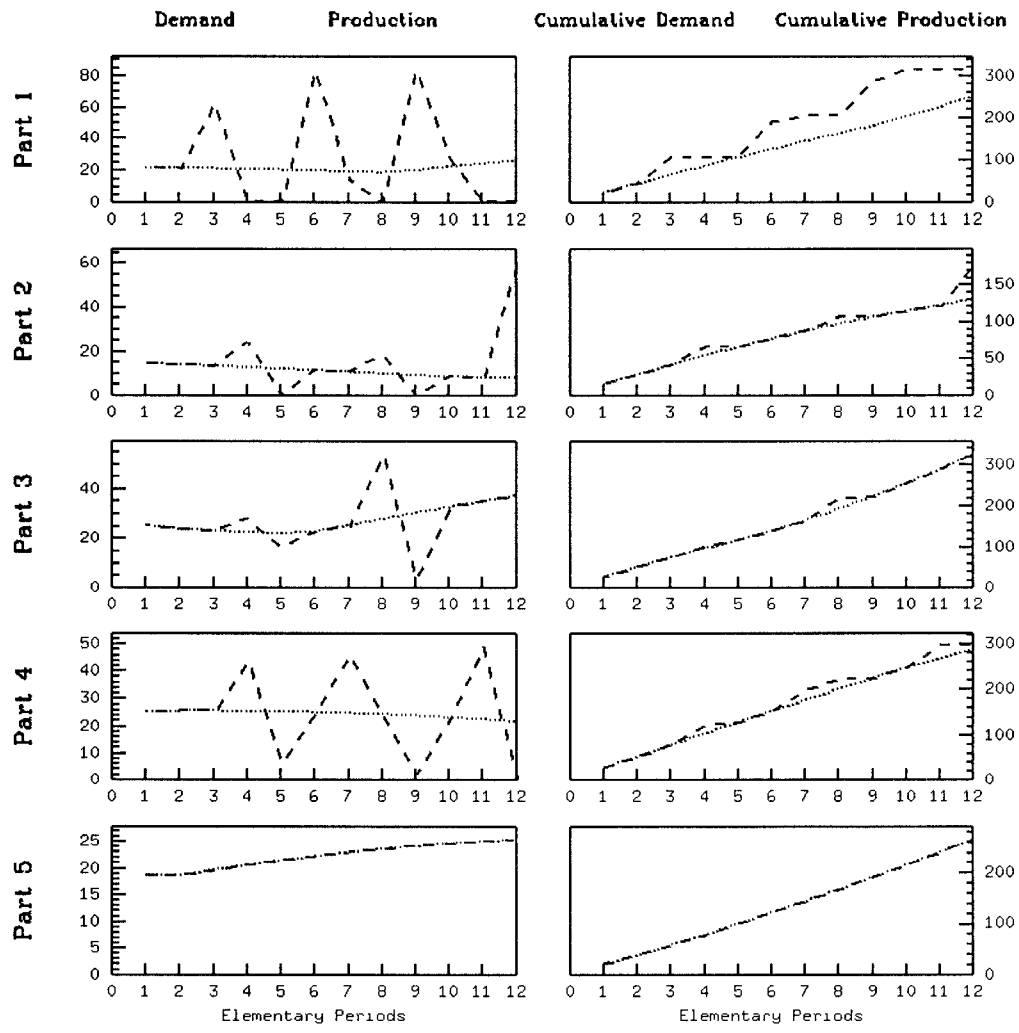


Figure 6.3(a) : Production Levels of Level 2 Product Entities

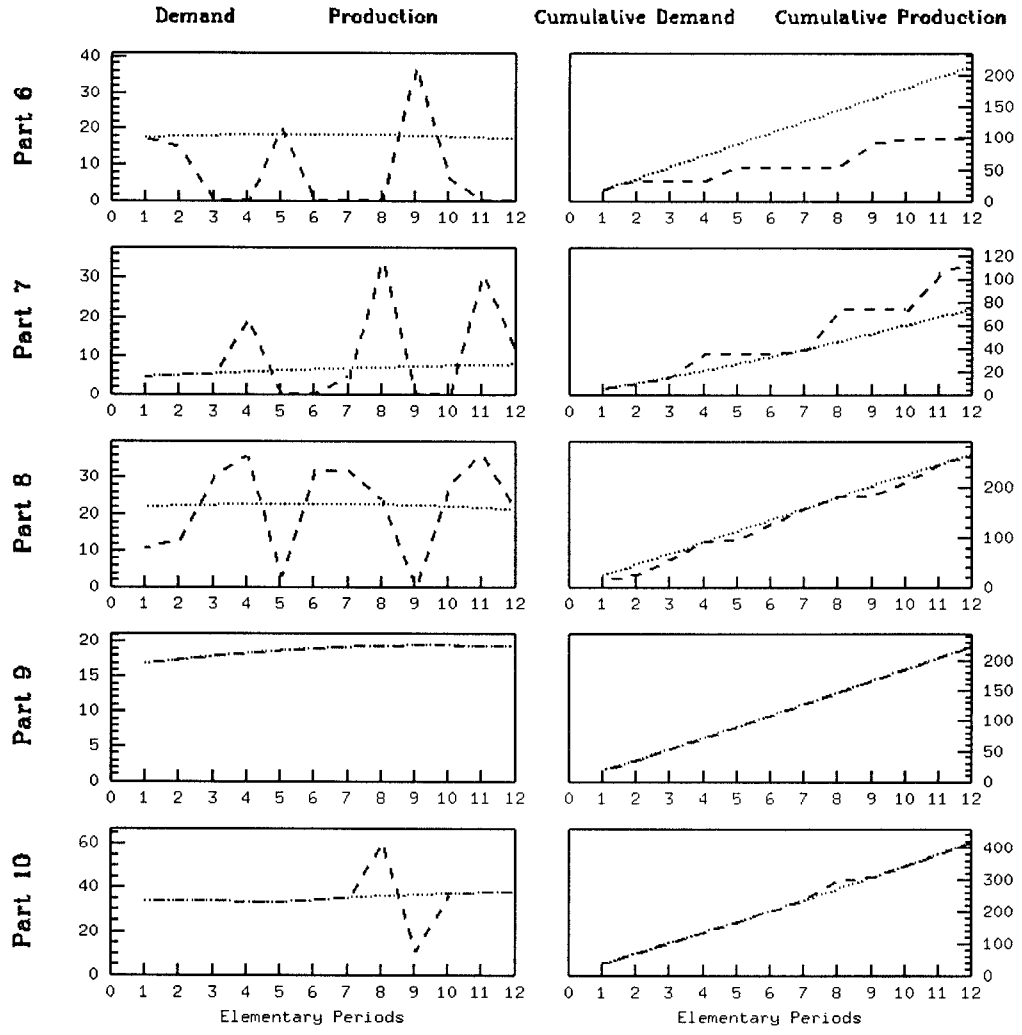


Figure 6.3(b) : Production Levels of Level 2 Product Entities

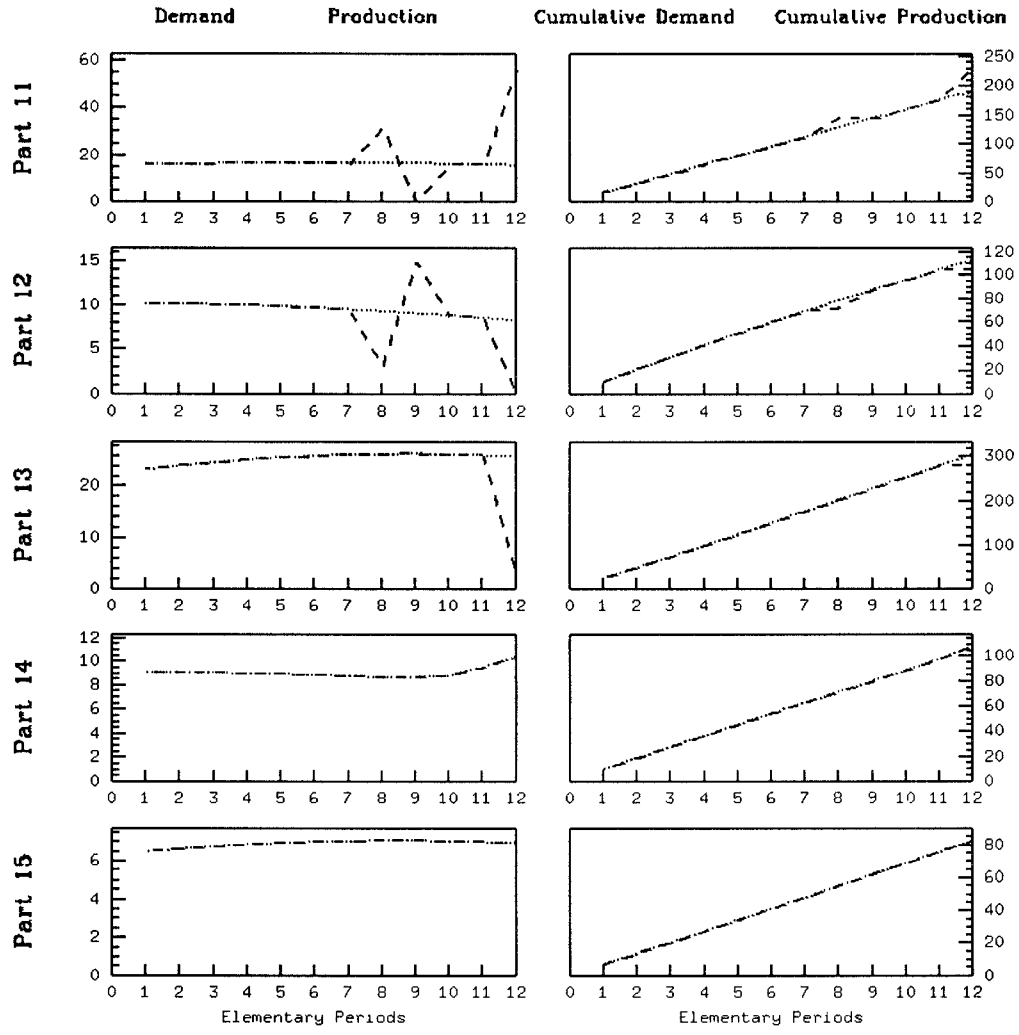


Figure 6.3(c) : Production Levels of Level 2 Product Entities

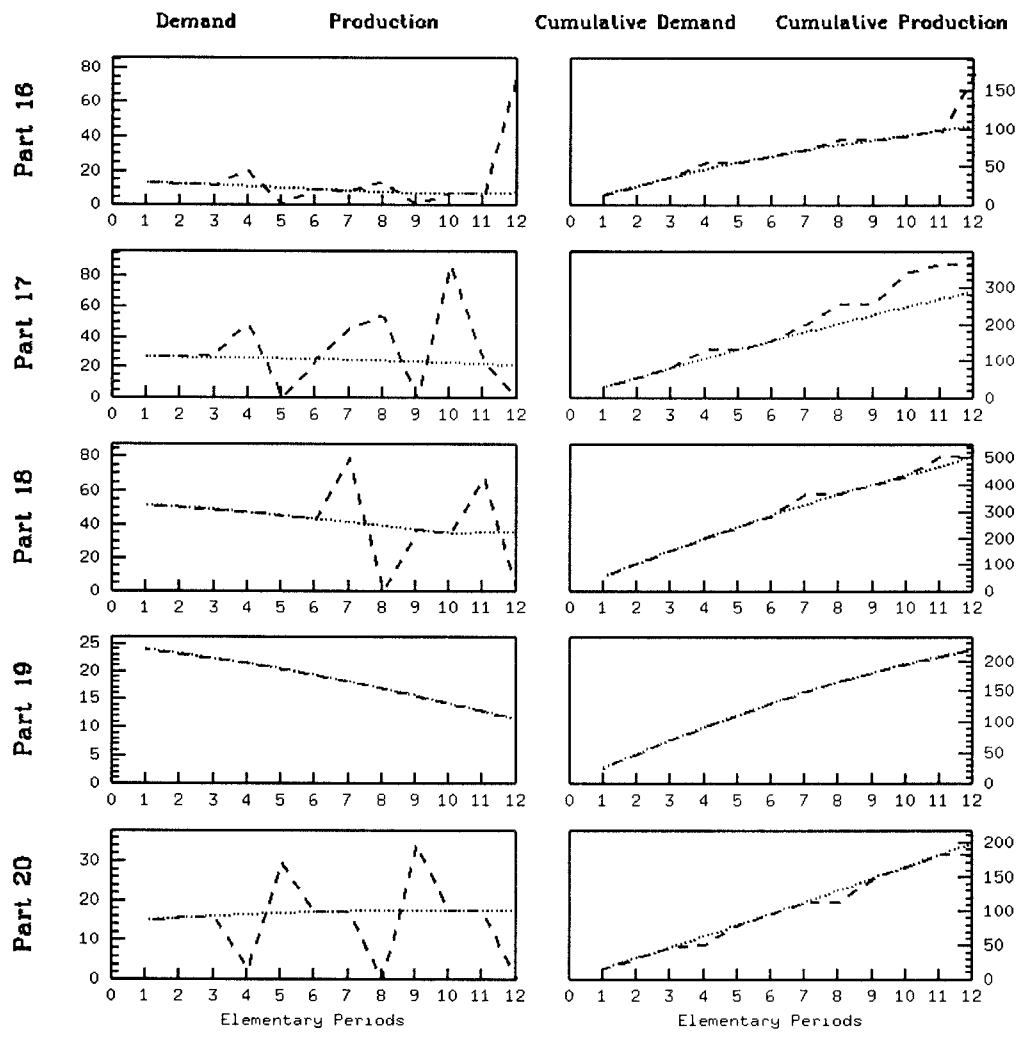


Figure 6.3(d) : Production Levels of Level 2 Product Entities

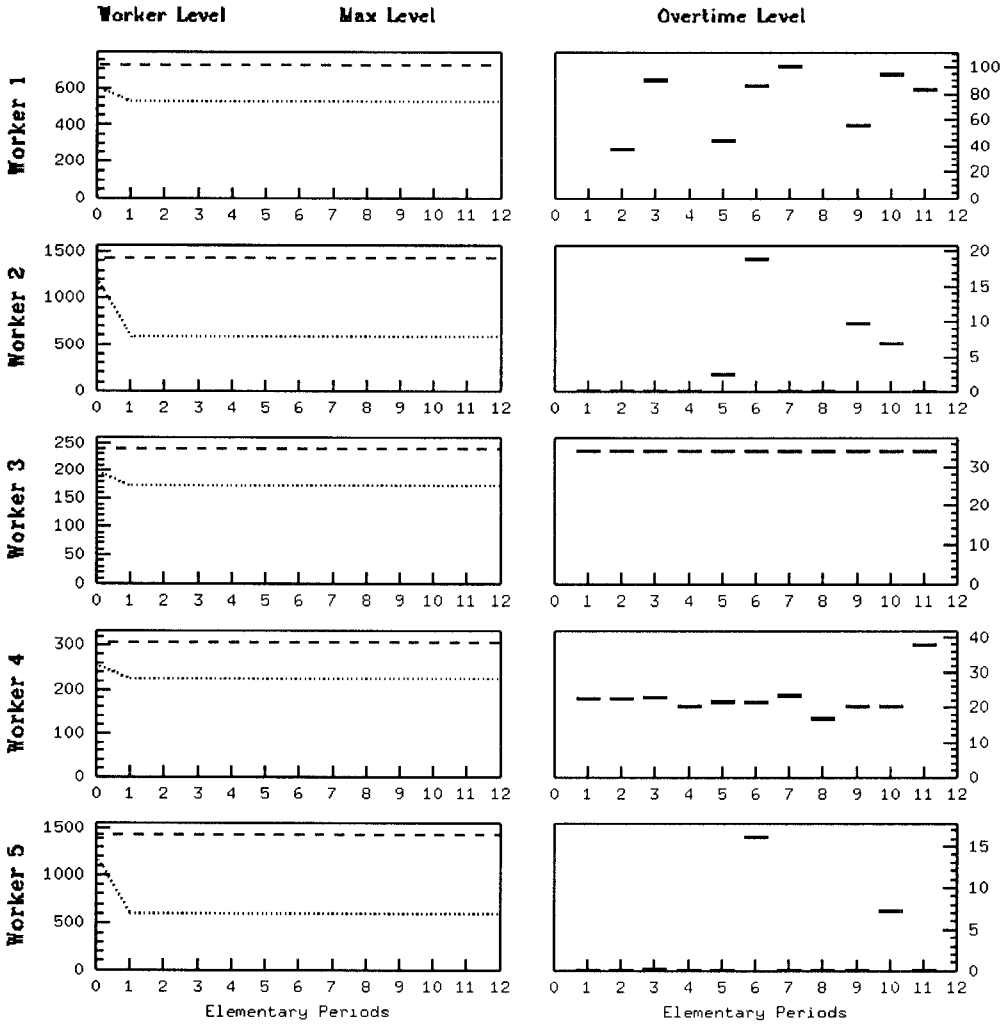


Figure 6.4(a) : Overtime Levels of Level 2 Worker Entities

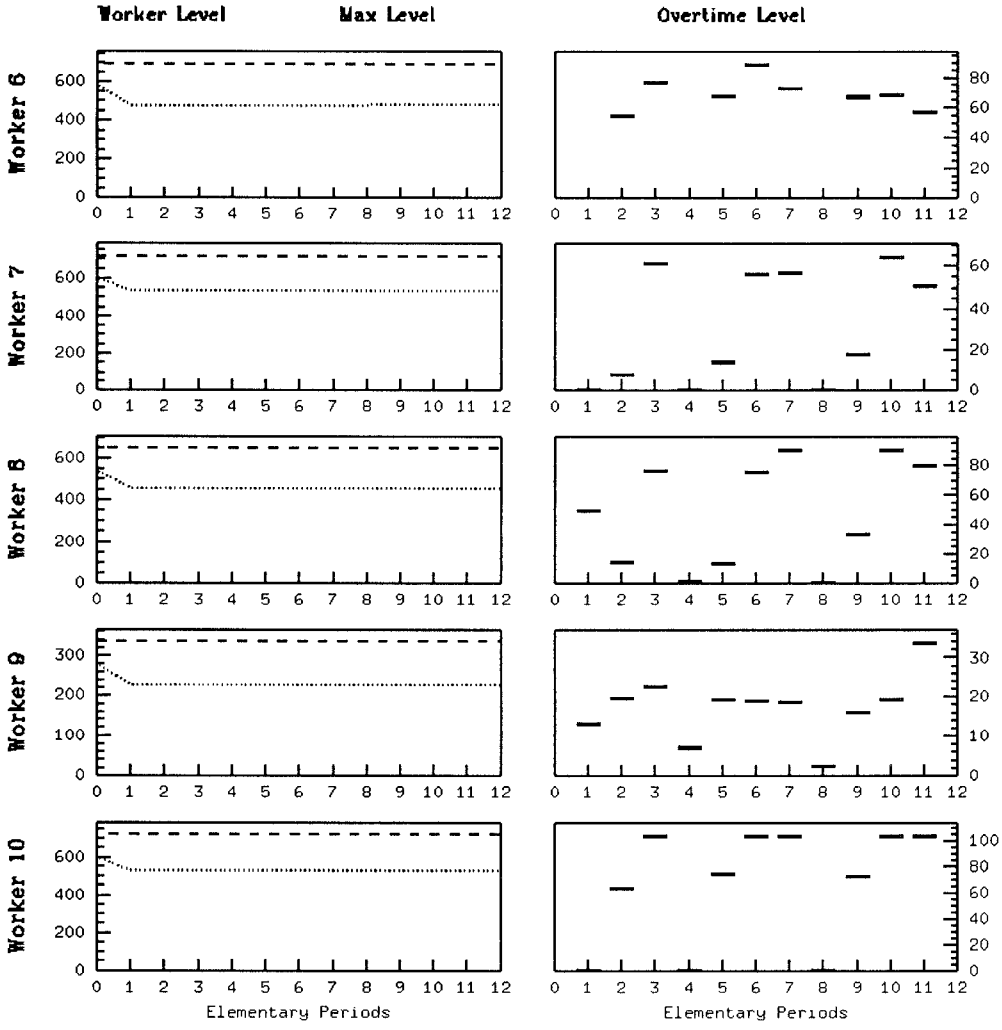


Figure 6.4(b) : Overtime Levels of Level 2 Worker Entities

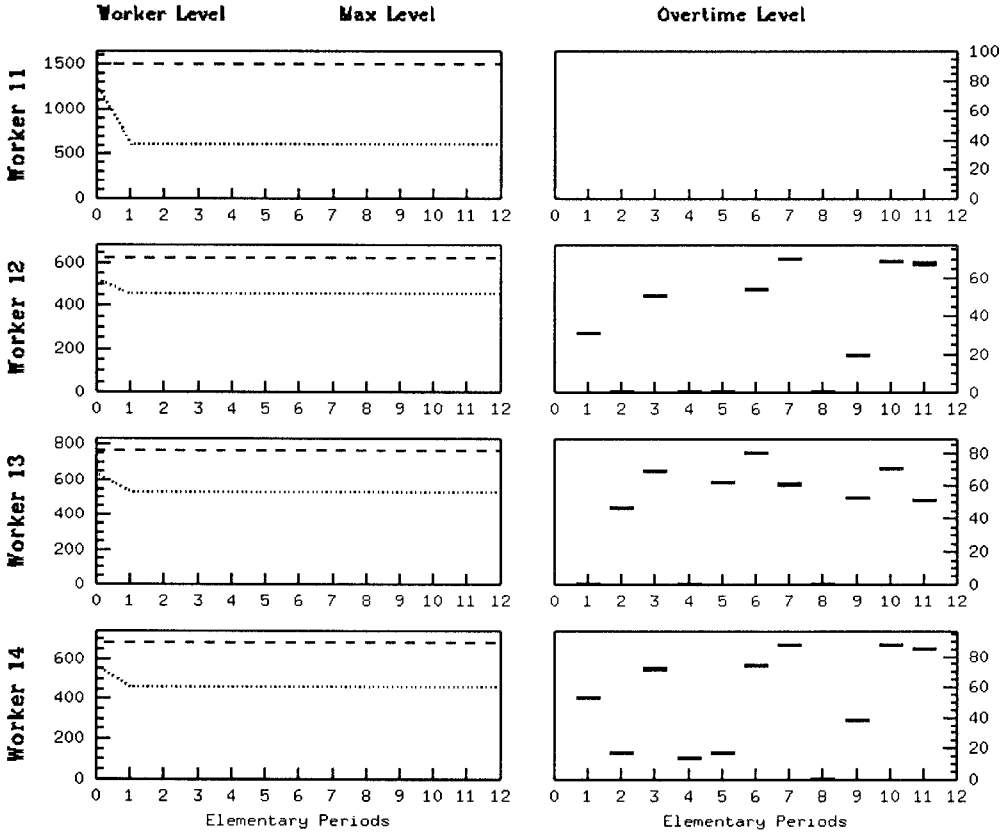


Figure 6.4(c) : Overtime Levels of Level 2 Worker Entities

Chapter 7

Conclusions & Recommendations for Further Work

This work is an attempt to define efficient architectures and operation modes for production management. The motivation is derived from the criticality of proper management for an overall desired performance of a production system. Lack of existing work and literature addressing some important facets of production planning problems in a generic fashion serves as the primary impetus to this work.

A multi-layer hierarchical approach to production planning problems is adopted, wherein the architecture (in particular, the number of levels) is strongly based on the specific physical production system and the complexity of the decision making problem at hand. We advocate the need for a problem specific hierarchy as opposed to a fixed hierarchy to serve as a panacea for all production planning problems. The scope of production management problems we have addressed, and the purpose of the corresponding hierarchy, is to perform typical tactical level planning decisions : all the way from aggregate production planning over a long horizon, down to the detailed production planning over a relatively short horizon. Aggregate production planning involves the structuring of a general plan for responding to forecasted demand through the combination of work force, output, and inventory loadings. More specifically, tactical decisions consist of deciding work-force and over-time levels, as well as production rates of aggregate products. They include employment planning decisions, and setting company targets for inventory and service levels. Furthermore, a production system is subject

to both endogenous (e.g., resource failures) as well as exogenous (e.g., unscheduled orders) random events of different time-scales. The planning decisions at various hierarchical levels envisage an expectation of uncontrollable events to be tying-up the resources. Addressing these new aspects in production planning and suggesting problem specific hierarchies emerge as unique contributions of this work. We focus on the design and operation aspects of such Hierarchical Production Management Systems (HPMS).

Regarding the design of the HPMS, we start with a given physical system, and then we identify the factors influencing the problem complexity, and detail the relevant inputs required. We develop product and machine aggregation schemes for a planning problem with holding and backlogging penalties. In some specific cases, we prove optimality of such schemes. Then, we extend the theory to more general aggregations, and present worst cases analyses for some more general schemes. The consistency and controllability issues in multi-layer hierarchies have been addressed in the aggregation/disaggregation schemes. Temporal aggregation is also introduced in the theory. These three aggregation dimensions for model reduction have been developed and blended with the time-scale decomposition of activities, in order to provide a solid theoretical foundation of the architecture. Thus, a systematic stepwise design approach for the construction of the hierarchy has been presented.

The design process results in the appropriate number of layers, their models, entities, definitions, planning horizons, and domains. Each level is modeled by a set of pertinent entities to that level, the links between these entities, a set of attributes associated with each entity, and the value set associated with each attribute. This is followed by the formal definition of the Decision Making Problem (DMP) at each level, along with a set of possible controls (decisions), a set of constraints, and one or more optimality criteria over a given horizon. The lengths of horizons are derived in such a way that each level is able to absorb most of the random events associated with it, under the existing hardware (memory,

depending on the DPM dimension) and software (solution time permissible for the DMP) limitations.

The operation of the hierarchy in an unreliable environment is also explained. In particular, the rolling horizon mechanism, and the reaction of the hierarchy to random events has been detailed. Solution algorithms to resolve some related optimization problems (especially, disaggregation problems) have been presented. The mechanism for top-down constraint propagation is an important aspect for the operation of the hierarchy. Problem solutions are attempted in a sequential manner; the solution of any problem in sequence determines some parameters in the subsequent problem. We propose the propagation of constraints in a manner that does not render the subsequent problem infeasible, while ensuring consistency of criteria.

This work does not cover the entire gamut of planning problems and their related assumptions. Employing the concepts proposed, the hierarchy can be extended to both higher or strategic level decisions, as well as lower or operational level decisions. Strategic decisions have lead-times that are generally higher than tactical decisions, and are performed at a higher degree of abstraction (aggregation). Thus, they lend themselves adaptable to the methodology presented in this work. Encompassing operational decisions may require extensions to the methodologies presented here, or even developing some other ones. Finally, the execution involves bottom-up feedbacks, and a revision of the plan if necessary. This closed loop real time control process can be implemented in future. Operation can be numerically studied by the development of a simulation environment that emulates resource failures and exogenous random events. Performance evaluation against a monolithic approach with powerful linear programming codes for small problems can possibly be performed.

It is hoped that this methodology can be applied to other types of large-scale complex decision making problems (such as communication system).

Considering more examples, and validating their architectures can be further justification to the methodology developed here.

On a different note, we propose some links of this work with research aiming at controlling the manufacturing information flow among manufacturing application systems (Computer Aided Design, Computer Aided Process Planning, Manufacturing Resources Planning and Shop Floor Control). The latter has lead to the development of the INformation System for Integrated Manufacturing (INSIM) in the CIM Laboratory, University of Maryland at College Park [38]. As effective management relies on the availability and exchange of a wide variety of reliable information, most of which is already present in the distributed database of a CIM architecture built by INSIM, the further integration of decision making seems to be a natural course. Integration of the decision making architecture with the CIM environment will eliminate conflicts and reduce redundancy in data input, apart from providing it with timely and correct information. Our recommendation consists of the control of information flow between each of the key manufacturing applications at the factory level and the decision making architecture. Common data entities form the basis of this integration, which involves identification of inputs, outputs, and interactions of the HPMS with existing CIM databases.

In our suggested approach, we classify information types to be exchanged in the following manner :

(1) Statistical information :

The aggregate models of the hierarchy and their algorithms depend primarily on statistical information about product mix, work-center and worker failure characteristics. Most CIM systems maintain history files that can store and provide this information on request.

(2) Static information :

Bills of materials, routings, work-center details, and other static information present in MRP databases can be obtained on request.

(3) Dynamic information :

Information about customer orders, forecasts, machine states, worker states, and others, constitute the dynamic information requiring varied speed of exchange. Maintaining information integrity with speedy exchange presents challenging research issues.

In summary, the recommended work consists of the knowledge acquisition, to establish the rules for information flow between HPMS and the CIM architectures. This would be followed by modeling, verification and implementation of the Knowledge Base Systems (KBS) employing the methods detailed in Harhalakis *et al* [38].

It is believed that such improved tools in a manufacturing environment for companywide decision making and management, which are well integrated with other functions, can provide an answer to the much required competitive edge to the U.S. manufacturing sector.

Appendix A

A.1

In this section, we demonstrate that the production planning problem for parts with initial inventory can be converted into one with zero initial inventory and a corresponding "net demand".

Result : The linear programming problems ($\mathcal{P}0$) and ($\mathcal{P}1$) are equivalent, or in other words the x_{ik} solves ($\mathcal{P}0$) iff x_{ik} solves ($\mathcal{P}1$).

We represent the problem with initial inventory, ($\mathcal{P}0$) as follows :

$$\text{Minimize: } Cost(P) = \sum_{k=1}^{z} \sum_{i=1}^{n} w_i^+ \times [Q_{ik} - D_{ik}]^+ + w_i^- \times [D_{ik} - Q_{ik}]^+ \quad \text{(A.1.1)}$$

where : $[\bullet]^+ = Max\{0, \bullet\}$.

$$\text{Again, } D_{ik} = \sum_{a=1}^{k} d_{ia} ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad \text{(A.1.2)}$$

$$\text{Subject to: } Q_{ik} = Q_{i0} + \sum_{a=1}^{k} x_{ia} ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad \text{(A.1.3)}$$

$$\sum_{i=1}^{n} t_{ij} x_{ik} \le \Delta; \qquad j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \quad \text{(A.1.4)}$$

$$x_{ik} \ge 0 ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad \text{(A.1.5)}$$

Let us define :

$$Q'_{ik} = Q_{ik} + Q_{i0}; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad \text{(A.1.6)}$$

$$D''_{ik} = D_{ik} - Q_{i0}; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad \text{(A.1.7)}$$

Now we define D'_{ik} as the net cumulative demand of part type p_i at the end of the k-th elementary period, detailed as follows :

$$D'_{ik} = [D''_{ik}]^+; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.1.8)}$$

The corresponding net demand is then represented by d'_{ik}, see (A.1.22).

Now consider a problem without initial inventory ($\mathcal{P}1$) :

$$\text{Min. : Cost(P')} = \sum_{k=1}^{z} \sum_{i=1}^{n} w_i^+ \times [Q'_{ik} - D'_{ik}]^+ + w_i^- \times [D'_{ik} - Q'_{ik}]^+ \qquad \text{(A.1.9)}$$

$$\text{Again, } D'_{ik} = \sum_{a=1}^{k} d'_{ia} ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.1.10)}$$

$$\text{Subject to: } Q'_{ik} = \sum_{a=1}^{k} x'_{ia} ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.1.11)}$$

$$\sum_{i=1}^{n} t_{ij}\, x'_{ik} \le \Delta; \qquad j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.1.12)}$$

$$x'_{ik} \ge 0 ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.1.13)}$$

Demonstration :

Using (A.1.6) and (A.1.7), (A.1.1) can be written as follows :

$$\text{Min.: Cost(P)} = \sum_{k=1}^{z} \sum_{i=1}^{n} w_i^+ \times [Q'_{ik} - D''_{ik}]^+ + w_i^- \times [D''_{ik} - Q'_{ik}]^+ \qquad \text{(A.1.14)}$$

Using the relation

$$D''_{ik} = [D''_{ik}]^+ - \alpha_{ik}; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.1.15)}$$

where $\alpha_{ik} \ge 0$, with (A.1.8), (A.1.14) becomes :

$$\text{Min.: } \sum_{k=1}^{z} \sum_{i=1}^{n} w_i^+ \times [Q'_{ik} - D'_{ik} - \alpha_{ik}]^+ + w_i^- \times [D'_{ik} - Q'_{ik} - \alpha_{ik}]^+ \qquad \text{(A.1.16)}$$

Consider $D''_{rs} > 0$ for some r and s $\Rightarrow \alpha_{rs} = 0 \Rightarrow$ we obtain

$$w_r^+ \times [Q'_{rs} - D'_{rs}]^+ + w_r^- \times [D'_{rs} - Q'_{rs}]^+ \qquad \text{(A.1.17)}$$

Consider $D''_{rs} \le 0$ for some r and s $\Rightarrow \alpha_{rs} \ge 0$, $D'_{ik} = 0 \Rightarrow$ we obtain

$$w_r^+ \times [Q'_{rs} - \alpha_{rs}]^+ + w_r^- \times [-Q'_{rs} - \alpha_{rs}]^+ \qquad \text{(A.1.18)}$$

Observing that $Q'_{rs}, \alpha_{rs} \geq 0 \; \forall r,s$ implies that the second term equals zero, and the first term can be written as follows :

$$w_r^+ \times Q'_{rs} - w_r^+ \times \alpha_{rs} \qquad \text{(A.1.19)}$$

Thus, combining (A.1.17), (A.1.18) and (A.1.19), more generally, we obtain:

$$\forall r,s \quad w_r^+ \times [Q'_{rs} - D'_{rs}]^+ + w_r^- \times [D'_{rs} - Q'_{rs}]^+ - w_r^+ \times \alpha_{rs} \qquad \text{(A.1.20)}$$

Thus, (A.1.16) becomes :

$$\sum_{k=1}^{z} \sum_{i=1}^{n} w_i^+ \times [Q'_{ik} - D'_{ik}]^+ + w_i^- \times [D'_{ik} - Q'_{ik}]^+ - \sum_{k=1}^{z} \sum_{i=1}^{n} w_r^+ \times \alpha_{rs} \qquad \text{(A.1.21)}$$

Observe that the last summation is a constant, which can be eliminated from the objective function. This leads us to (A.1.9). Also, constraints (A.1.11) follows directly from the definition (A.1.6). Hence (*P1*) and (*P0*) are equivalent problems.

Q.E.D.

Computation of the net demand

(A.1.7) and (A.1.8), define D'_{ik} as the net cumulative demand of part type p_i at the end of the k-th elementary period, as follows :

$$D'_{ik} = [D_{ik} - Q_{i0}]^+; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.1.7,8)}$$

The corresponding net demand, represented by d'_{ik}, i = 1,2,...,n, can be calculated as follows :

$$d'_{ik} = \begin{cases} D'_{ik} - D'_{ik-1}, & k = 2,\ldots,z \\ D'_{ik}, & k = 1 \end{cases} \qquad \text{(A.1.22)}$$

Henceforth, we will only consider problems with zero initial inventory. Without loss of generality, we can use d_{ik} and d'_{ik} interchangeably.

A.2

In this section, we demonstrate the production planning problem for parts on individual machines is equivalent to planning parts on cells, provided the aggregation of machines into cells is performed in the following manner.

Machines belong to the same cell, if they process the same set of parts with exactly the same processing times. The set of M cells, $M \leq m$, is represented by $\mathcal{C} = \{c_1, c_2, \ldots, c_M\}$. Two machines m_a and $m_b \in c_r$ iff :
(i) $t_{ia} = t_{ib}$; $i = 1,2,\ldots,n$.

Let τ_{ij} represent the processing time of one unit of part p_i in cell c_j; $\tau_{ij} = t_{ia}$ such that $m_a \in c_j$.
Once again, consider the problem of production planning of parts on machines ($\mathcal{P}0$) as follows :

$$\text{Minimize: Cost(P)} = \sum_{k=1}^{z} \sum_{i=1}^{n} w_i^+ \times [Q_{ik} - D_{ik}]^+ + w_i^- \times [D_{ik} - Q_{ik}]^+ \quad \text{(A.2.1)}$$

$$\text{Given } D_{ik} = \sum_{a=1}^{k} d_{ia} \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad \text{(A.2.2)}$$

$$\text{Subject to: } Q_{ik} = \sum_{a=1}^{k} x_{ia} \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad \text{(A.2.3)}$$

$$\sum_{i=1}^{n} t_{ij} \, x_{ik} \leq \Delta; \qquad j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \quad \text{(A.2.4)}$$

$$x_{ik} \geq 0 \; ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \quad \text{(A.2.5)}$$

Result : Considering the constraints (A.2.4) for machines belonging to the same cell, gives rise to identical constraints for all machines, hence, one constraint is sufficient.
Proof :
Say machines m_a and $m_b \in c_r$; we also know that $t_{ia} = t_{ib}$; $i = 1,2,\ldots,n$. Hence,

$$\sum_{i=1}^{n} t_{ia} \, x_{ik} \leq \Delta; \qquad k = 1,2,\ldots,z \quad \text{(A.2.4.a)}$$

$$\sum_{i=1}^{n} t_{ib} \, x_{ik} \leq \Delta; \qquad k = 1,2,\ldots,z \quad \text{(A.2.4.b)}$$

are identical.

Hence, $\forall a \mid m_a \in c_j$, *it is sufficient to represent one capacity constraint :*

$$\sum_{i=1}^{n} \tau_{ij}\, x_{ik} \leq \Delta; \qquad j = 1,2,\ldots,M, \text{ and } k = 1,2,\ldots,z \qquad (A.2.6)$$

Thus, (A.2.6) replaces (A.2.4).

Q.E.D.

A.3

In this section, we demonstrate an important property of an optimal solution to the production planning problem for parts, provided the aggregation of parts into part families is performed in the following manner.

Parts belong to the same family if they have exactly the same processing requirements, i.e., they use identical resources with same processing times. In addition, they have equal holding costs, and equal backlogging costs. *The set of N part families, $N \leq n$, is represented by $\mathcal{F} = \{f_1, f_2, \ldots, f_N\}$. Two* parts p_a and $p_b \in f_r$ *iff :*

(i) $t_{aj} = t_{bj}$, $j = 1,2,\ldots,m$

(ii) $w_a^{+} = w_b^{+} = v_r^{+}$

(iii) $w_a^{-} = w_b^{-} = v_r^{-}$

Once again, consider the problem of production planning of parts on machines *($\mathcal{P}0$) as follows :*

$$\text{Minimize: } Cost(P) = \sum_{k=1}^{z} \sum_{i=1}^{n} w_i^{+} \times [Q_{ik} - D_{ik}]^{+} + w_i^{-} \times [D_{ik} - Q_{ik}]^{+} \qquad (A.3.1)$$

$$\text{Given } D_{ik} = \sum_{a=1}^{k} d_{ia}\ ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad (A.3.2)$$

$$\text{Subject to: } Q_{ik} = \sum_{a=1}^{k} x_{ia}\ ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad (A.3.3)$$

$$\sum_{i=1}^{n} t_{ij}\, x_{ik} \leq \Delta; \qquad j = 1,2,\ldots,m, \text{ and } k = 1,2,\ldots,z \qquad (A.3.4)$$

$$x_{ik} \geq 0 ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.3.5)}$$

Let us define the cumulative demand, R_{ik}, and cumulative production, U_{ik}, for families as follows :

$$R_{ik} = \sum_{a=1}^{k} \sum_{b \mid p_b \in f}^{k} d_{ba} ; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.3.6)}$$

$$U_{ik} = \sum_{a=1}^{k} \sum_{b \mid p_b \in f_i}^{k} x_{ba} ; \qquad i = 1,2,\ldots,N, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.3.7)}$$

Statement : In an optimal solution of ($\mathcal{P}0$), when the inventory level of a family is non-negative (resp. negative), then the inventory levels of each of the parts belonging to this family are also non-negative (resp. non-positive).

$$\text{If } U_{ik} - R_{ik} \begin{cases} \geq 0, \text{ then, } Q_{bk} - D_{bk} \geq 0 \ \forall \ b \mid p_b \in f_i \\ \leq 0, \text{ then, } Q_{bk} - D_{bk} \leq 0 \ \forall \ b \mid p_b \in f_i \end{cases} ; \quad \forall i, \text{ and } \forall k \qquad \text{(A.3.8)}$$

Proof :

Let us consider two parts, p_a and $p_b \in f_r$. Let us define $L_{ak} = Q_{ak} - D_{ak}$.

Hypothesis H1 : In the optimal solution S1 of ($\mathcal{P}0$), $\exists\ k \mid L_{ak} > 0$ and $L_{bk} < 0$.

Consider $\mu = \text{Min}\{L_{ak}, - L_{bk}\}$

During the periods $j \mid j \leq k$, we can replace the most recent production of μ parts of p_a by μ parts of p_b. Note, since these parts share exactly the same production capacity, this is always possible. Let L'_{aj} represent the new values of L_{aj} in the alternate solution.

For periods $j \mid j < k$, we cannot increase the value of the criterion (if not decrease it). This is obvious for the following reasons :

(i) For periods $j \mid j < k$, and $L'_{aj} = L_{aj}$, i.e. unaffected periods, the criterion remains unchanged.

(ii) For periods $j \mid j < k$, and $L'_{aj} < L_{aj}$, i.e. affected periods, the criterion value cannot increase. Note that $L'_{aj} \geq 0$, as we will never be required to reach negative inventory. Among these periods if $\exists j \mid L_{bj} \geq 0$, the criterion value remains unchanged, as both parts have equal holding costs.

However, if $\exists j \mid L_{bj} < 0$, the criterion value can in fact be reduced owing to the need for lesser backlogging of p_b. In any case, we cannot increase the value of the criterion for periods $j \mid j < k$.

For period k, we can decrease the value of the criterion by the following amount :
$v_r^+ \times \mu + v_r^- \times \mu$.

During the periods $j \mid j > k$, we can replace the most recent production of μ parts of p_b by μ parts of p_a, unless of course the horizon ends. Once again, since these parts share exactly the same production capacity, this is always possible. Now for periods $j \mid j > k$, we cannot increase the value of the criterion for the following reasons :
(i) For periods $j \mid j > k$, and $L'_{xj} = L_{xj}, \forall\ x \mid p_x \in f_r$, i.e. periods for which we have been able to return to the original solution, the criterion remains unchanged.
(ii) For periods $j \mid j > k$, and $L'_{xj} \neq L_{xj}, \forall\ x \mid p_x \in f_r$, i.e. affected periods, the criterion value cannot increase.
We start with $j = k + 1$; let the production of p_b for this period be π. If $\pi \geq \mu$, our point is proven. If $0 \leq \pi < \mu$, it implies that in the original solution $L_{bj} < 0$. Now regarding the inventory status of p_a in the new solution :
(a) If $L'_{aj} > 0$, we can decrease the objective function once again, because, we have to hold less p_a and backlog less p_b.
(b) If $L'_{aj} \leq 0$, we will increase backlogging for p_a by no more than $v_r^- \times \pi$, but, we will decrease backlogging cost of p_b by $v_r^- \times \pi$. Thus, the cost cannot increase.
The procedure continues till we have returned to the original solution path, or we arrive at the end of the horizon. It is important to indicate that L_{bj} remains negative as long as we do not reach the original solution path, so we will never introduce extra holding cost for p_b before we reach the original solution path.

Thus, we have an alternate feasible solution with a lower value for the criterion. This refutes the hypothesis H1. Therefore, in an optimal

solution of ($\mathcal{P}0$), parts belonging to the same family have inventory levels in the same sense, which implies (A.3.8).

Q.E.D.

Comment : A similar procedure to the one employed in this proof can also be used to develop an iterative algorithm for the disaggregation problem.

A.4

In this section, we demonstrate the equivalence of the hierarchical approach to the problem ($\mathcal{P}1$) (see section 4), i.e. problem ($\mathcal{P}2$) followed by ($\mathcal{P}3$) is equivalent to problem ($\mathcal{P}1$).

Consider ($\mathcal{P}1$) :

$$\sum_{k=1}^{z}\sum_{i=1}^{n}\left(w_i^{+}\times\left[\sum_{a=1}^{k}x_{ia}-\sum_{a=1}^{k}d_{ia}\right]^{+}+w_i^{-}\times\left[\sum_{a=1}^{k}d_{ia}-\sum_{a=1}^{k}x_{ia}\right]^{+}\right) \qquad (A.4.1)$$

$$\text{Subject to: } \sum_{i=1}^{n} t_{ij}\,x_{ik}\le\Delta; \qquad j=1,2,\ldots,m, \text{ and } k=1,2,\ldots,z \qquad (A.4.2)$$

$$x_{ik}\ge 0\,; \qquad i=1,2,\ldots,n, \text{ and } k=1,2,\ldots,z \qquad (A.4.3)$$

Consider problem ($\mathcal{P}2$) :

$$\sum_{k=1}^{z}\sum_{i=1}^{N}\left(v_i^{+}\times\left[\sum_{a=1}^{k}y_{ia}-\sum_{a=1}^{k}\sum_{b\,|\,p_b\in f_i}d_{ba}\right]^{+}+\right.$$

$$\left.v_i^{-}\times\left[\sum_{a=1}^{k}\sum_{b\,|\,p_b\in f_i}d_{ba}-\sum_{a=1}^{k}y_{ia}\right]^{+}\right) \qquad (A.4.4)$$

$$\text{Subject to: } \sum_{i=1}^{N}\theta_{ij}\,y_{ik}\le\Delta; \qquad j=1,2,\ldots,M, \text{ and } k=1,2,\ldots,z \qquad (A.4.5)$$

$$y_{ik}\ge 0\,; \qquad i=1,2,\ldots,N, \text{ and } k=1,2,\ldots,z \qquad (A.4.6)$$

problem ($\mathcal{P}3$) :

$$y_{ik}=\sum_{b\,|\,p_b\in f_i}x_{bk}\,; \qquad i=1,2,\ldots,N, \text{ and } k=1,2,\ldots,z \qquad (A.4.7)$$

$$\text{If } \sum_{a=1}^{k} y_{ia} - \sum_{a=1}^{k} \sum_{b \mid p_b \in f_i} d_{ba} \begin{cases} \geq 0, \text{ then, } \sum_{a=1}^{k} x_{ba} - \sum_{a=1}^{k} d_{ba} \geq 0 \ \forall\, b \mid p_b \in f_i \\ \leq 0, \text{ then, } \sum_{a=1}^{k} x_{ba} - \sum_{a=1}^{k} d_{ba} \leq 0 \ \forall\, b \mid p_b \in f_i \end{cases} ;$$

$$\forall i, \forall k \qquad \text{(A.4.8)}$$

$$x_{ik} \geq 0 ; \qquad i = 1,2,\ldots,n, \text{ and } k = 1,2,\ldots,z \qquad \text{(A.4.9)}$$

Demonstration :

A : (A.4.6), (A.4.7) and (A.4.9) imply (A.4.3).

B : (A.4.5), and (A.4.7) imply :

$$\sum_{i=1}^{N} \theta_{ij} \sum_{b \mid p_b \in f_i} x_{bk} \leq \Delta$$

or

$$\sum_{i=1}^{N} \sum_{b \mid p_b \in f_i} t_{bj}\, x_{bk} \leq \Delta$$

which is equivalent to (A.4.2) because each part belongs to one family.

C : Consider (A.4.4) with (A.4.7) :

$$\sum_{k=1}^{z} \sum_{i=1}^{N} (v_i^{+} \times [\sum_{a=1}^{k} \sum_{b \mid p_b \in f_i} x_{bk} - \sum_{a=1}^{k} \sum_{b \mid p_b \in f_i} d_{ba}]^{+} +$$

$$v_i^{-} \times [\sum_{a=1}^{k} \sum_{b \mid p_b \in f_i} d_{ba} - \sum_{a=1}^{k} \sum_{b \mid p_b \in f_i} x_{bk}]^{+})$$

Changing the order of summation :

$$\sum_{k=1}^{z} \sum_{i=1}^{N} (v_i^{+} \times [\sum_{b \mid p_b \in f_i} \left(\sum_{a=1}^{k} x_{bk} - \sum_{a=1}^{k} d_{ba} \right)]^{+} +$$

$$v_i^{-} \times [\sum_{b \mid p_b \in f_i} \left(\sum_{a=1}^{k} d_{ba} - \sum_{a=1}^{k} x_{bk} \right)]^{+})$$

Now, by (A.4.8), knowing that the parts belonging to the same family have similar inventory sense, i.e. all non-negative, or all non-positive, it follows that :

$$\sum_{k=1}^{z}\sum_{i=1}^{N}\Big(v_i^{+}\times\sum_{b\,|\,p_b\in f_i}\Big[\sum_{a=1}^{k}x_{bk}-\sum_{a=1}^{k}d_{ba}\Big]^{+}+$$

$$v_i^{-}\times\sum_{b\,|\,p_b\in f_i}\Big[\sum_{a=1}^{k}d_{ba}-\sum_{a=1}^{k}x_{bk}\Big]^{+}\Big)$$

Observing that all parts belonging to the same family have equal penalty costs, we obtain :

$$\sum_{k=1}^{z}\sum_{i=1}^{N}\sum_{b\,|\,p_b\in f_i}\Big(w_b^{+}\times\Big[\sum_{a=1}^{k}x_{bk}-\sum_{a=1}^{k}d_{ba}\Big]^{+}+w_b^{-}\times\Big[\sum_{a=1}^{k}d_{ba}-\sum_{a=1}^{k}x_{bk}\Big]^{+}\Big)$$

Since each part belongs to one family, this is equivalent to (A.4.1).

A, B and C imply equivalence.

Q.E.D.

Appendix B

B.1 Problem Data

Product details for 50 products :

P No.,	Holding cost,	Backlogging cost,	Production
1	9.334	72.472	7.072
2	3.168	88.745	6.723
3	2.927	89.483	6.811
4	5.238	70.868	1.674
5	5.288	50.940	6.749
6	12.089	56.561	7.125
7	10.208	69.108	4.447
8	5.177	49.249	7.136
9	9.677	57.455	2.832
10	4.851	93.963	3.641
11	9.343	56.818	2.869
12	11.918	55.574	5.138
13	4.070	49.227	3.727
14	10.073	70.535	4.679
15	11.686	86.305	4.967
16	10.000	71.549	1.503
17	9.212	73.438	3.494
18	4.957	67.958	2.983
19	3.994	49.137	5.615
20	4.246	47.690	4.064
21	11.957	56.836	5.646
22	12.099	55.578	5.392
23	3.216	89.754	4.110
24	9.484	73.002	6.760
25	7.939	26.443	4.008
26	9.201	74.924	2.931
27	8.695	62.441	6.668
28	3.937	46.483	6.705
29	4.723	97.730	2.150
30	11.708	86.173	5.671
31	11.535	84.254	3.560
32	12.177	55.535	2.752
33	5.036	71.330	2.011
34	10.120	70.409	3.364
35	7.982	27.653	5.908
36	8.678	60.344	6.304
37	4.964	68.112	7.017
38	10.130	71.032	5.308
39	7.883	26.434	6.104

```
40     5.099  68.614 6.991
41     9.425  22.555 4.448
42     10.106 69.508 3.734
43     5.298  70.267 1.674
44     9.414  74.394 3.378
45     5.081  70.355 2.004
46     4.636  95.774 7.140
47     11.819 61.394 6.575
48     4.312  77.329 4.270
49     3.959  76.072 5.057
50     9.406  56.616 2.227
```

Work-center details for 34 work-centers :

```
Number of Work-centers
34
M No., Quantity, Op cost, #FM, FMi(MTBF,MTTR)..
 1  1  0.302 1  145.256 1.501
 2  1  0.343 4  105.772 1.198   1757.626   17.764 17794.650  177.911
    175473.641  1754.784
 3  1  0.320 1  149.126 1.649
 4  1  0.207 3  142.214 1.398   1523.617   15.232 18545.814  185.617
 5  1  0.188 4  110.525 1.176   1758.682   17.694 17791.840  177.988
    175473.953  1754.816
 6  2  0.266 2  185.930 1.994   1137.352   11.400
 7  2  0.246 2  177.487 1.835   1295.645   13.058
 8  1  0.346 4  107.453 1.078   1751.226   17.614 19460.678  194.734
    176830.531  1768.381
 9  1  0.376 1  157.834 1.531
 10    2      0.259       2     186.625   1.989  1202.960   12.001
 11    1      0.261       3     140.975   1.507  1529.668   15.236
    18548.766   185.620
 12    2      0.335       2     175.565   1.824  1293.281   12.990
 13    1      0.338       1     152.229   1.558
 14    1      0.176       1     136.745   1.375
 15    1      0.259       4     113.412   1.267  1666.942   16.745
    18508.115   185.161 161332.766 1613.389
 16    1      0.183       1     136.665   1.449
 17    1      0.174       3     146.693   1.535  1597.549   15.974
    18297.221   183.097
 18    2      0.315       2     183.330   1.831  1299.955   13.072
 19    1      0.307       4     110.667   1.193  1749.740   17.564
    19453.900   194.617 176833.625 1768.400
 20    1      0.334       3     136.661   1.420  1524.459   15.228
    18546.102   185.480
 21    2      0.192       2     185.003   1.833  1301.187   12.981
 22    1      0.348       1     149.476   1.567
 23    1      0.236       3     149.241   1.612  1607.926   16.140
    17075.383   170.834
 24    1      0.238       4     104.038   1.202  1759.144   17.669
    17786.566   177.868 175471.172 1754.694
 25    1      0.192       4     111.366   1.130  1746.857   17.494
    19454.346   194.598 176834.406 1768.343
 26    1      0.169       3     145.678   1.587  1605.007   16.056
    17081.492   170.887
 27    2      0.229       2     187.436   2.034  1134.585   11.390
```

```
 28    1      0.234       4     103.301   1.201   1763.438   17.684
    17792.885   178.034 175469.219 1754.773
 29    2      0.372       2     194.024   2.040   1133.207   11.422
 30    2      0.229       2     179.848   1.820   1301.918   13.097
 31    1      0.349       4     105.789   1.225   1748.238   17.505
    19456.852   194.721 176835.156 1768.403
 32    2      0.204       2     193.291   1.961   1200.837   11.980
 33    1      0.319       3     143.646   1.543   1522.934   15.301
    18552.512   185.517
 34    1      0.314       3     141.735   1.440   1432.157   14.478
    16841.576   168.459
```

Worker details for 34 workers :

```
Number of Workers
34
W No., Machine, Reg cost, O/T cost, Hire cost, Fire cost
 1      1   0.365  0.456  0.516  1.290
 2      2   0.560  0.700  0.684  1.710
 3      3   0.714  0.893  0.668  1.669
 4      4   0.753  0.941  0.424  1.059
 5      5   0.652  0.815  0.783  1.957
 6      6   0.688  0.861  0.389  0.973
 7      7   0.826  1.032  0.671  1.679
 8      8   0.438  0.547  0.812  2.031
 9      9   0.623  0.778  0.523  1.308
 10     10  0.767  0.959  0.950  2.375
 11     11  0.410  0.513  0.423  1.058
 12     12  0.616  0.770  0.415  1.037
 13     13  0.547  0.684  0.649  1.622
 14     14  0.757  0.946  0.857  2.142
 15     15  0.649  0.811  0.839  2.097
 16     16  0.746  0.933  0.379  0.947
 17     17  0.809  1.012  0.373  0.932
 18     18  0.689  0.861  0.689  1.722
 19     19  0.446  0.557  0.438  1.094
 20     20  0.376  0.470  0.441  1.102
 21     21  0.792  0.990  0.669  1.672
 22     22  0.400  0.501  0.669  1.672
 23     23  0.446  0.557  0.715  1.787
 24     24  0.579  0.724  0.703  1.758
 25     25  0.677  0.846  0.428  1.071
 26     26  0.830  1.038  0.669  1.672
 27     27  0.619  0.773  0.654  1.635
 28     28  0.809  1.011  0.800  2.000
 29     29  0.609  0.761  0.891  2.226
 30     30  0.363  0.453  0.352  0.880
 31     31  0.560  0.700  0.310  0.774
 32     32  0.728  0.910  0.933  2.332
 33     33  0.690  0.862  0.705  1.763
 34     34  0.392  0.490  0.671  1.677
```

Routing details (Incidence Matrix of cumulative processing times) :

```
Number of Parts
50
Number of Machines
34
```

Part details : Number, <(machine,time), (), ..>

1 0.0 0.0 0.0 1.7 0.0 2.5 2.5 0.0 0.0 2.4 1.6 2.4 0.0 0.0 0.0 0.0 1.8 2.8 0.0 1.6
2.9 0.0 1.7 0.0 0.0 1.9 2.7 0.0 2.4 2.8 0.0 2.5 1.7 1.3

2 0.0 0.0 0.0 1.8 0.0 2.6 2.7 0.0 0.0 2.7 1.4 2.4 0.0 0.0 0.0 0.0 1.6 2.6 0.0 1.4
2.6 0.0 1.8 0.0 0.0 1.8 2.8 0.0 2.5 2.5 0.0 2.5 1.6 1.7

3 0.0 0.0 0.0 1.7 0.0 2.6 2.4 0.0 0.0 2.7 1.5 2.6 0.0 0.0 0.0 0.0 1.4 2.7 0.0 1.5
2.4 0.0 1.9 0.0 0.0 2.0 2.6 0.0 2.6 2.6 0.0 2.5 1.6 1.6

4 1.7 0.0 1.1 1.7 0.0 2.6 2.8 0.0 1.1 2.3 1.7 2.5 1.2 1.5 0.0 1.3 1.7 2.3 0.0 1.6
2.5 1.2 2.0 0.0 0.0 1.4 2.6 0.0 2.6 2.8 0.0 2.7 1.8 1.6

5 0.0 0.0 0.0 1.8 0.0 2.2 2.7 0.0 0.0 2.5 1.7 2.6 0.0 0.0 0.0 0.0 1.6 2.7 0.0 1.7
2.5 0.0 1.7 0.0 0.0 1.8 2.5 0.0 2.6 2.5 0.0 2.5 1.6 1.7

6 0.0 2.4 0.0 1.6 2.2 2.5 2.7 2.6 0.0 2.3 1.4 2.6 0.0 0.0 1.9 0.0 1.6 2.6 2.4 1.8
2.8 0.0 1.7 2.5 2.1 1.9 2.8 2.5 2.7 2.9 2.5 2.5 1.9 1.8

7 1.4 2.5 1.3 0.0 2.4 2.4 3.0 2.5 1.0 2.4 0.0 2.6 1.2 1.5 2.1 1.1 0.0 2.6 1.9 0.0
2.9 0.8 0.0 2.5 2.2 0.0 2.5 2.5 2.3 2.9 2.2 2.6 0.0 0.0

8 0.0 0.0 0.0 1.6 0.0 2.4 2.5 0.0 0.0 2.6 1.9 2.6 0.0 0.0 0.0 0.0 1.7 2.5 0.0 1.9
2.7 0.0 1.6 0.0 0.0 1.6 2.5 0.0 2.7 2.6 0.0 2.7 1.3 1.7

9 0.0 2.7 0.0 1.7 2.7 2.6 2.5 2.5 0.0 2.6 1.6 2.7 0.0 0.0 2.0 0.0 1.6 2.4 2.6 1.8
2.5 0.0 1.7 2.3 2.4 1.7 2.6 2.7 2.3 2.6 2.4 2.7 1.9 1.5

10 1.4 0.0 0.9 1.8 0.0 2.8 2.6 0.0 0.8 2.4 1.8 2.4 1.1 1.6 0.0 1.4 1.5 2.8 0.0 1.7
2.4 1.0 2.0 0.0 0.0 1.4 2.5 0.0 2.7 2.5 0.0 2.4 1.6 1.6

11 0.0 2.4 0.0 1.8 2.5 2.5 2.7 2.4 0.0 2.9 2.0 2.6 0.0 0.0 2.3 0.0 1.9 2.8 2.5 1.9
2.5 0.0 1.6 2.5 2.5 1.7 2.6 2.6 2.5 2.4 2.7 2.5 1.8 1.6

12 0.0 2.3 0.0 1.7 2.4 2.6 2.4 2.4 0.0 2.4 1.5 2.7 0.0 0.0 2.2 0.0 1.8 2.7 2.4 1.6
2.9 0.0 1.6 2.4 2.3 2.0 2.6 2.4 2.4 2.9 2.4 2.4 1.9 1.7

13 0.0 0.0 0.0 1.6 0.0 2.8 2.6 0.0 0.0 2.4 1.5 2.7 0.0 0.0 0.0 0.0 1.5 2.5 0.0 1.5
3.0 0.0 1.9 0.0 0.0 1.9 2.4 0.0 2.8 2.5 0.0 2.5 1.6 1.7

14 1.3 2.2 1.3 0.0 2.4 2.6 2.9 2.4 0.9 2.5 0.0 2.5 0.9 1.3 2.0 1.2 0.0 2.6 2.0 0.0
2.7 0.9 0.0 2.4 2.2 0.0 2.3 2.5 2.4 2.6 2.2 2.6 0.0 0.0

15 0.0 2.6 0.0 1.5 2.3 2.5 2.7 2.1 0.0 2.7 1.8 2.4 0.0 0.0 2.4 0.0 1.8 2.6 2.4 1.8
2.7 0.0 1.7 2.4 2.2 1.6 2.6 2.6 2.5 2.5 2.2 2.4 1.7 1.5

16 1.5 2.6 1.3 0.0 2.5 2.3 2.6 2.3 1.0 2.4 0.0 2.4 1.0 1.3 2.3 1.2 0.0 2.6 2.3 0.0
2.8 1.1 0.0 2.4 2.3 0.0 2.3 2.5 2.2 2.6 2.2 2.4 0.0 0.0

17 0.0 0.0 0.0 1.5 0.0 2.6 2.5 0.0 0.0 2.5 1.6 2.4 0.0 0.0 0.0 0.0 1.6 2.7 0.0 1.5
2.6 0.0 1.4 0.0 0.0 1.9 2.6 0.0 2.6 2.7 0.0 2.5 1.6 1.3

18 1.4 0.0 1.1 1.6 0.0 2.5 2.8 0.0 1.1 2.6 1.8 2.8 1.0 1.5 0.0 1.4 1.8 2.6 0.0 1.8
2.7 1.2 1.8 0.0 0.0 1.6 2.7 0.0 2.6 2.8 0.0 2.7 1.7 1.5

19 0.0 0.0 0.0 1.6 0.0 2.6 2.6 0.0 0.0 2.5 1.6 2.6 0.0 0.0 0.0 0.0 1.7 2.6 0.0 1.6
2.8 0.0 1.6 0.0 0.0 1.6 2.7 0.0 2.7 2.5 0.0 2.3 1.4 1.6

20 0.0 0.0 0.0 1.7 0.0 2.7 2.3 0.0 0.0 2.7 1.8 2.9 0.0 0.0 0.0 0.0 1.6 2.8 0.0 1.4
2.7 0.0 1.6 0.0 0.0 1.6 2.7 0.0 2.8 2.5 0.0 2.4 1.6 1.7

21 0.0 2.5 0.0 1.9 2.4 2.4 2.4 2.5 0.0 2.6 1.6 2.7 0.0 0.0 2.1 0.0 1.7 2.7 2.3 1.4
2.8 0.0 1.4 2.4 2.1 1.8 2.8 2.4 2.4 2.6 2.2 2.7 2.0 1.6

22 0.0 2.3 0.0 1.7 2.2 2.5 2.7 2.2 0.0 2.4 1.6 2.8 0.0 0.0 2.0 0.0 1.6 2.7 2.5 1.5
2.6 0.0 1.7 2.3 2.4 1.8 2.7 2.3 2.7 2.9 2.3 2.6 1.9 1.9

23 0.0 0.0 0.0 1.9 0.0 2.4 2.7 0.0 0.0 2.6 1.6 2.4 0.0 0.0 0.0 0.0 1.7 2.8 0.0 1.5
2.4 0.0 1.9 0.0 0.0 2.0 2.6 0.0 2.5 2.5 0.0 2.7 1.7 1.7

24 0.0 0.0 0.0 1.7 0.0 2.4 2.7 0.0 0.0 2.3 1.6 2.4 0.0 0.0 0.0 0.0 1.7 2.6 0.0 1.8
2.8 0.0 1.8 0.0 0.0 1.8 2.7 0.0 2.5 2.5 0.0 2.4 1.5 1.4

25 0.0 2.2 0.0 1.6 2.2 2.6 2.4 2.2 0.0 2.9 1.7 2.5 0.0 0.0 2.2 0.0 1.7 2.5 2.4 1.9
2.5 0.0 1.8 2.6 2.3 1.8 2.4 2.7 2.4 2.5 2.3 2.7 1.7 1.4

26 0.0 0.0 0.0 1.6 0.0 2.6 2.8 0.0 0.0 2.5 1.5 2.2 0.0 0.0 0.0 0.0 1.6 2.7 0.0 1.7
2.8 0.0 1.5 0.0 0.0 1.9 2.4 0.0 2.7 2.7 0.0 2.3 1.8 1.5

27 1.6 2.5 1.2 0.0 2.4 2.9 2.4 2.0 1.1 2.4 0.0 2.5 1.4 1.4 2.2 1.4 0.0 2.8 2.2 0.0
2.4 1.2 0.0 2.1 2.3 0.0 2.7 2.4 2.7 2.7 2.3 2.5 0.0 0.0

28 0.0 0.0 0.0 1.7 0.0 2.5 2.4 0.0 0.0 2.4 1.5 2.6 0.0 0.0 0.0 0.0 1.5 2.8 0.0 1.4
2.6 0.0 1.6 0.0 0.0 1.8 2.6 0.0 2.6 2.6 0.0 2.6 1.4 1.7

29 1.6 0.0 1.3 1.5 0.0 2.6 2.8 0.0 0.9 2.4 2.0 2.4 1.1 1.7 0.0 1.1 1.5 2.5 0.0 1.6
2.4 0.9 2.0 0.0 0.0 1.8 2.5 0.0 2.5 2.6 0.0 2.7 1.5 1.5

30 0.0 2.3 0.0 1.5 2.1 2.5 2.4 2.1 0.0 2.7 1.8 2.6 0.0 0.0 2.4 0.0 1.7 2.5 2.4 1.5
2.5 0.0 1.4 2.3 2.5 1.5 2.5 2.5 2.4 2.5 2.5 2.6 1.7 1.7

```
31 0.0 2.5 0.0 1.8 2.1 2.4 2.7 2.1 0.0 2.7 1.6 2.5 0.0 0.0 2.5 0.0 1.8 2.4 2.4 1.6
   2.5 0.0 1.7 2.2 2.4 1.6 2.5 2.7 2.4 2.5 2.4 2.8 1.8 1.5
32 0.0 2.5 0.0 1.7 2.4 2.4 2.5 2.3 0.0 2.5 1.4 2.9 0.0 0.0 2.2 0.0 1.4 2.9 2.6 1.7
   2.6 0.0 1.5 2.4 2.1 1.9 2.8 2.4 2.5 2.7 2.3 2.7 1.7 1.8
33 1.3 0.0 1.0 1.8 0.0 2.4 2.6 0.0 1.1 2.6 1.6 2.6 1.2 1.5 0.0 1.5 1.8 2.6 0.0 1.8
   2.8 1.3 1.7 0.0 0.0 1.6 2.5 0.0 2.5 2.8 0.0 2.6 1.6 1.6
34 1.6 2.3 1.1 0.0 2.3 2.4 2.7 2.5 1.0 2.6 0.0 2.4 1.0 1.3 2.2 1.3 0.0 2.8 1.9 0.0
   2.7 0.8 0.0 2.3 2.1 0.0 2.3 2.6 2.3 2.7 2.3 2.4 0.0 0.0
35 0.0 2.5 0.0 1.6 2.5 2.5 2.5 2.1 0.0 2.6 1.4 2.7 0.0 0.0 2.4 0.0 1.5 2.7 2.4 1.6
   2.6 0.0 1.8 2.5 2.3 1.8 2.5 2.4 2.5 2.7 2.2 2.8 2.7 1.6
36 1.6 2.2 1.2 0.0 2.1 2.7 2.3 2.1 1.1 2.2 0.0 2.5 1.3 1.5 2.4 1.5 0.0 2.8 2.6 0.0
   2.7 1.3 0.0 2.4 2.1 0.0 2.8 2.7 2.8 2.5 2.0 2.3 0.0 0.0
37 1.7 0.0 1.3 1.5 0.0 2.7 2.7 0.0 1.1 2.4 1.7 2.7 1.1 1.6 0.0 1.4 1.6 2.4 0.0 1.5
   2.8 1.1 1.7 0.0 0.0 1.5 2.7 0.0 2.7 2.7 0.0 2.6 1.8 1.6
38 1.3 2.3 1.3 0.0 2.6 2.4 2.6 2.4 1.1 2.7 0.0 2.6 1.1 1.5 2.4 1.1 0.0 2.9 2.1 0.0
   3.0 0.8 0.0 2.4 2.1 0.0 2.3 2.6 2.3 2.9 2.1 2.4 0.0 0.0
39 0.0 2.3 0.0 1.4 2.3 2.4 2.5 2.2 0.0 2.7 1.5 2.5 0.0 0.0 2.3 0.0 1.5 2.5 2.2 1.9
   2.4 0.0 1.9 2.7 2.4 1.9 2.3 2.4 2.2 2.4 2.1 2.6 1.9 1.4
40 1.6 0.0 1.4 1.8 0.0 2.7 2.9 0.0 1.1 2.6 1.7 2.7 1.0 1.4 0.0 1.6 1.8 2.4 0.0 1.7
   2.6 1.1 1.8 0.0 0.0 1.4 2.7 0.0 2.5 2.7 0.0 2.7 1.7 1.5
41 1.5 0.0 1.0 1.5 0.0 2.5 2.7 0.0 1.2 2.4 1.7 2.5 1.1 1.8 0.0 1.1 1.9 2.5 0.0 1.6
   2.8 1.2 1.5 0.0 0.0 1.8 2.5 0.0 2.9 2.6 0.0 2.7 1.5 1.8
42 1.5 2.4 1.1 0.0 2.6 2.3 2.9 2.2 0.9 2.4 0.0 2.3 1.1 1.5 2.2 1.4 0.0 2.7 2.2 0.0
   2.7 1.1 0.0 2.5 2.0 0.0 2.5 2.4 2.2 2.6 2.3 2.5 0.0 0.0
43 1.4 0.0 1.2 1.8 0.0 2.6 2.7 0.0 1.3 2.4 1.5 2.7 1.1 1.4 0.0 1.6 1.8 2.7 0.0 1.6
   2.7 1.2 2.0 0.0 0.0 1.6 2.4 0.0 2.5 2.8 0.0 2.7 1.5 1.4
44 0.0 0.0 0.0 1.7 0.0 2.6 2.7 0.0 0.0 2.5 1.8 2.4 0.0 0.0 0.0 0.0 1.6 2.6 0.0 1.8
   2.9 0.0 1.7 0.0 0.0 1.5 2.5 0.0 2.5 2.6 0.0 2.6 1.8 1.2
45 1.3 0.0 1.3 1.7 0.0 2.7 2.9 0.0 1.0 2.6 1.7 2.6 1.2 1.5 0.0 1.6 1.5 2.6 0.0 1.8
   2.8 1.0 1.9 0.0 0.0 1.6 2.5 0.0 2.8 2.7 0.0 2.6 1.8 1.4
46 1.6 0.0 1.2 1.5 0.0 2.7 2.5 0.0 1.0 2.6 1.9 2.4 1.2 1.5 0.0 1.1 1.7 2.6 0.0 1.5
   2.5 1.1 1.8 0.0 0.0 1.7 2.7 0.0 2.8 2.4 0.0 2.7 1.7 1.7
47 1.3 2.6 1.2 0.0 2.4 2.6 2.3 2.2 1.0 2.7 0.0 2.6 1.0 1.3 2.2 1.4 0.0 2.6 2.2 0.0
   2.7 1.1 0.0 2.5 2.5 0.0 2.4 2.8 2.5 2.6 2.4 2.7 0.0 0.0
48 1.2 2.6 1.1 0.0 2.7 2.7 2.4 2.5 1.1 2.8 0.0 2.5 1.1 1.5 2.4 1.5 0.0 2.7 2.1 0.0
   2.7 0.8 0.0 2.4 2.4 0.0 2.6 2.5 2.5 2.4 2.2 2.2 0.0 0.0
49 1.2 2.6 1.2 0.0 2.7 2.6 2.5 2.5 1.2 2.6 0.0 2.5 0.9 1.4 2.4 1.5 0.0 2.7 2.0 0.0
   2.8 1.1 0.0 2.6 2.1 0.0 2.3 2.5 2.7 2.5 2.3 2.5 0.0 0.0
50 0.0 2.5 0.0 1.4 2.5 2.4 2.7 2.3 0.0 2.7 1.7 2.6 0.0 0.0 2.0 0.0 1.7 2.8 2.3 2.0
   2.7 0.0 1.8 2.4 2.3 1.8 2.6 2.6 2.6 2.5 2.2 2.7 1.7 1.4
```

B.2 Aggregation Details : Level 2

Product Aggregation details for Level 2 :

```
Number of Families
 20
Family details : Family Number, parts
 1   3   28
 2   11   50
 3   6   12   21   22   32
 4   25   39
 5   15   30   31
 6   27
 7   44
 8   18   33   41   43
 9   48   49
 10   4   37   40   45
 11   9   35
 12   14   47
```

```
 13  7  38
 14  1  17  26
 15  24
 16  10  29  46
 17  13  19  20
 18  2  5  8  23
 19  16  34  42
 20  36
```

Work-center Aggregation details for Level 2 :

```
Number of Cells
 14
Cell details : Cell Number, machines
 1  17  33
 2  6  18  27  29  30
 3  3  9  13  22
 4  1  16
 5  7  10  12  32
 6  34
 7  20
 8  5  8  24
 9  14
 10  4  11  23
 11  21
 12  15  19  25  31
 13  26
 14  2  28
```

Aggregate Product details for 20 product entities at Level 2 :

```
F No., Holding cost, Backlogging cost, Production
 1      3.43   68.15  13.52
 2      9.37   56.73  5.10
 3      12.04  56.11  26.05
 4      7.91   26.44  10.11
 5      11.66  85.74  14.20
 6      8.69   62.44  6.67
 7      9.41   74.39  3.38
 8      6.81   50.75  11.12
 9      4.12   76.65  9.33
 10     5.06   68.83  17.69
 11     8.53   37.31  8.74
 12     11.09  65.19  11.25
 13     10.17  70.15  9.76
 14     9.27   73.25  13.50
 15     9.48   73.00  6.76
 16     4.71   95.59  12.93
 17     4.09   48.72  13.41
 18     4.33   67.19  24.72
 19     10.09  70.22  8.60
 20     8.68   60.34  6.30
```

Aggregate Work-center details for 14 work-centers entities at Level 2 :

```
Number of Cells
14
C No., Quantity, Op cost, #FM, FMi(MTBF,MTTR)..
```

```
1  1   0.49   3   145.13 3.09   1581.66   31.87  18368.70   369.33
2  1   1.41   2   186.14 9.92   1250.87   64.14
3  1   1.38   1   152.92 6.44
4  1   0.48   1   143.73 3.02
5  1   1.04   2   187.35 7.90   1257.54   51.16
6  1   0.31   3   141.74 1.44   1432.16   14.48  16841.58   168.46
7  1   0.33   3   136.66 1.42   1524.46   15.23  18546.10   185.48
8  1   0.77   4   108.18 3.52   1724.28   52.53  19077.15   578.26
   173346.22  5252.69
9  1   0.18   1   136.74 1.37
10     1   0.70   3   146.23 4.63   1576.31   47.76  18183.04   551.34
11     1   0.19   2   185.00 1.83   1301.19   12.98
12     1   1.11   4   109.80 4.88   1698.16   69.15  18884.42
   767.15 171634.34  6969.22
13     1   0.17   3   145.68 1.59   1605.01   16.06  17081.49   170.89
14     1   0.58   4   104.56 2.41   1745.79   35.33  17618.36
   354.20 173736.19  3492.23
```

Aggregate Worker details for 14 workers entities at Level 2 :

```
Number of Worker Groups
14
G No., Cell, Reg cost, O/T cost, Hire cost, Fire cost
 1      1   1.50   1.87   1.08   2.69
 2      2   2.97   3.71   2.98   7.44
 3      3   2.28   2.86   2.51   6.27
 4      4   1.11   1.39   0.89   2.24
 5      5   2.94   3.67   2.97   7.42
 6      6   0.39   0.49   0.67   1.68
 7      7   0.38   0.47   0.44   1.10
 8      8   1.67   2.09   2.30   5.75
 9      9   0.76   0.95   0.86   2.14
 10     10  1.61   2.01   1.56   3.90
 11     11  0.79   0.99   0.67   1.67
 12     12  2.33   2.91   2.01   5.04
 13     13  0.83   1.04   0.67   1.67
 14     14  1.37   1.71   1.48   3.71
```

Routing details (Incidence Matrix of cumulative processing times) Level 2:

```
Number of Families
20
Number of Cells
14
Family details : Number, <(cell,time), (), ..>
1 1.501 1.375 0.000 0.000 1.300 1.650 1.450 0.000 0.000 1.751 1.250
   0.000 1.901 0.000
2 1.813 1.400 0.000 0.000 1.406 1.513 1.944 2.500 0.000 1.869 1.293
   2.413 1.744 2.600
3 1.901 1.407 0.000 0.000 1.357 1.758 1.601 2.424 0.000 1.716 1.378
   2.420 1.877 2.407
4 1.821 1.250 0.000 0.000 1.390 1.400 1.900 2.660 0.000 1.860 1.220
   2.360 1.860 2.440
5 1.760 1.268 0.000 0.000 1.350 1.580 1.630 2.310 0.000 1.750 1.285
   2.425 1.560 2.585
6 0.000 1.450 1.400 1.600 1.250 0.000 0.000 2.400 1.400 0.000 1.200
   2.300 0.000 2.500
```

```
7 1.800 1.300 0.000 0.000 1.350 1.200 1.800 0.000 0.000 1.800 1.450
   0.000 1.500 0.000
8 1.840 1.360 1.218 1.422 1.355 1.623 1.690 0.000 1.605 1.692 1.379
   0.000 1.680 0.000
9 0.000 1.350 1.154 1.500 1.346 0.000 0.000 2.700 1.446 0.000 1.377
   2.400 0.000 2.600
10 1.760 1.355 1.321 1.615 1.406 1.538 1.623 0.000 1.500 1.791 1.347
   0.000 1.462 0.000
11 1.765 1.334 0.000 0.000 1.384 1.432 1.665 2.565 0.000 1.768 1.284
   2.465 1.768 2.565
12 0.000 1.300 1.242 1.317 1.329 0.000 0.000 2.458 1.300 0.000 1.350
   2.375 0.000 2.675
13 0.000 1.450 1.300 1.346 1.391 0.000 0.000 2.509 1.500 0.000 1.477
   2.263 0.000 2.554
14 1.705 1.376 0.000 0.000 1.283 1.343 1.596 0.000 0.000 1.627 1.401
   0.000 1.900 0.000
15 1.700 1.350 0.000 0.000 1.350 1.400 1.800 0.000 0.000 1.800 1.400
   0.000 1.800 0.000
16 1.639 1.361 1.155 1.544 1.308 1.639 1.573 0.000 1.561 1.890 1.227
   0.000 1.632 0.000
17 1.614 1.379 0.000 0.000 1.360 1.658 1.512 0.000 0.000 1.683 1.413
   0.000 1.683 0.000
18 1.645 1.316 0.000 0.000 1.321 1.700 1.643 0.000 0.000 1.759 1.284
   0.000 1.776 0.000
19 0.000 1.361 1.135 1.539 1.385 0.000 0.000 2.465 1.387 0.000 1.359
   2.283 0.000 2.496
20 0.000 1.400 1.300 1.600 1.250 0.000 0.000 2.400 1.500 0.000 1.350
   2.600 0.000 2.700
```

B.3 Aggregation Details : Level 3

Product Aggregation details for Level 3 :

```
Number of Families
 7
Family details : Family Number, parts
 1  1  18
 2  7
 3  8  10  16
 4  14  15  17
 5  4  5  11
 6  2  3
 7  6  9  12  13  19  20
```

Work-center Aggregation details for Level 3 :

```
Number of Cells
 6
Cell details : Cell Number, machines
 1  3
 2  4  9
 3  6
 4  8  12  14
 5  1  7  10  13
 6  2  5  11
```

Aggregate Product details for 7 product entities at Level 3 :

```
F No., Holding cost, Backlogging cost, Production
 1  4.01    67.53  38.24
 2  9.41    74.39  3.38
 3  5.42    72.30  41.74
 4  7.25    63.43  33.67
 5  9.69    54.79  33.05
 6  11.60   56.21  31.15
 7  8.90    68.07  51.91
```

Aggregate Work-center details for 6 work-centers entities at Level 3 :

```
Number of Cells
6
C No., Quantity, Op cost, #FM, FMi(MTBF,MTTR)..
 1  1   1.38    1   152.92 6.44
 2  1   0.66    1   142.30 4.45
 3  1   0.31    3   141.74 1.44   1432.16    14.48  16841.58   168.46
 4  1   2.46    4   103.94 10.74  1627.89    153.23 17971.13   1684.28
    163303.95  15299.61
 5  1   1.69    3   140.18 10.68  1514.73    109.34 17476.74   1261.29
 6  1   2.64    2   176.86 19.20  1190.10    124.91
```

Aggregate Worker details for 6 workers entities at Level 3 :

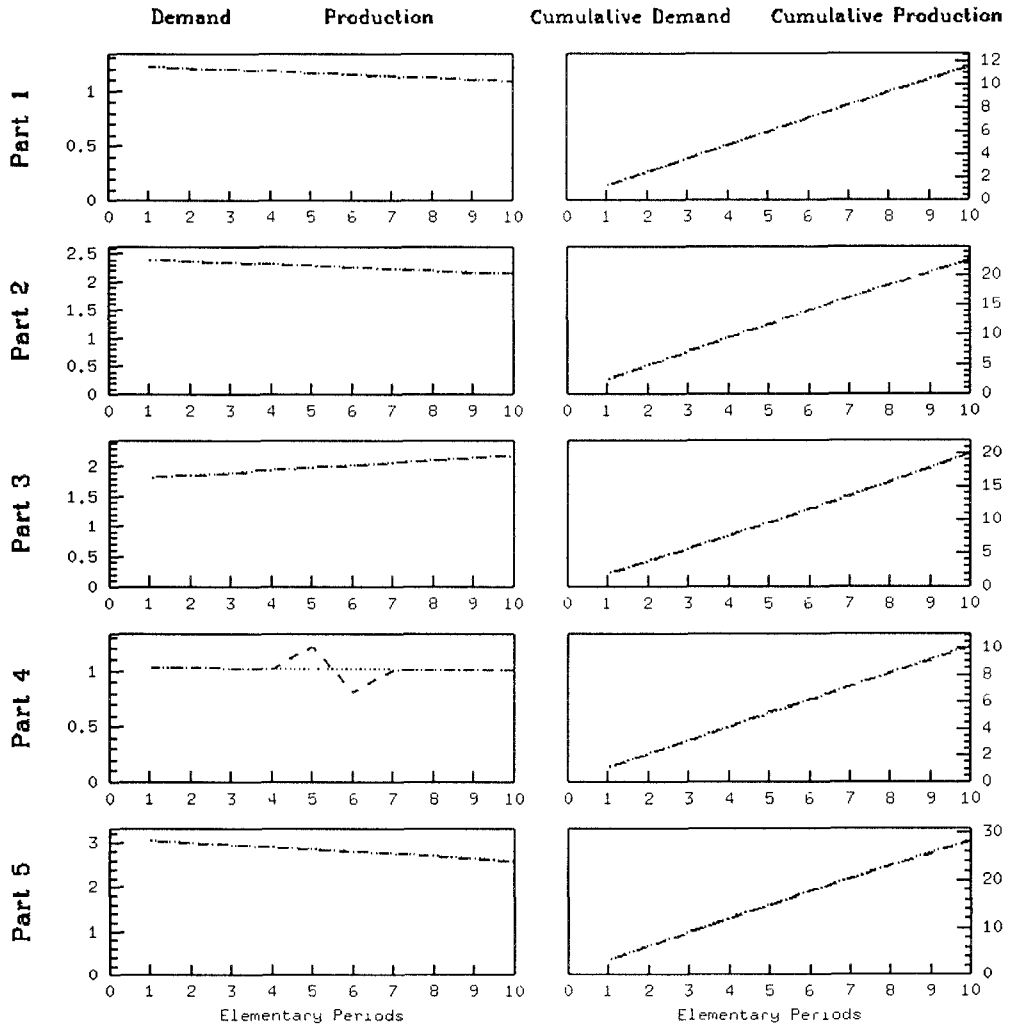
```
Number of Worker Groups
6
G No., Cell, Reg cost, O/T cost, Hire cost, Fire cost
 1  1   2.28    2.86    2.51    6.27
 2  2   1.87    2.34    1.75    4.38
 3  3   0.39    0.49    0.67    1.68
 4  4   5.37    6.71    5.79    14.50
 5  5   4.32    5.39    3.75    9.36
 6  6   6.70    8.37    6.62    16.53
```

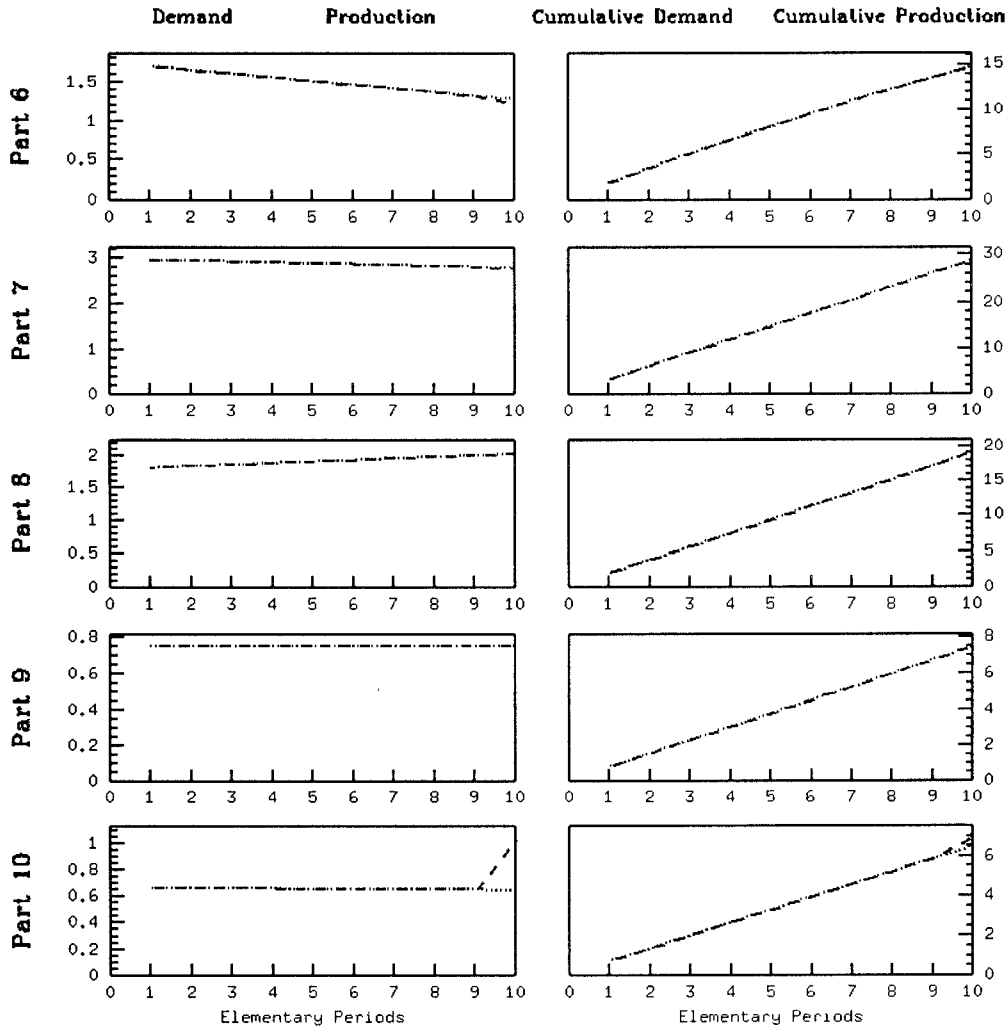
Routing details (Incidence Matrix of cumulative processing times) Level 3:

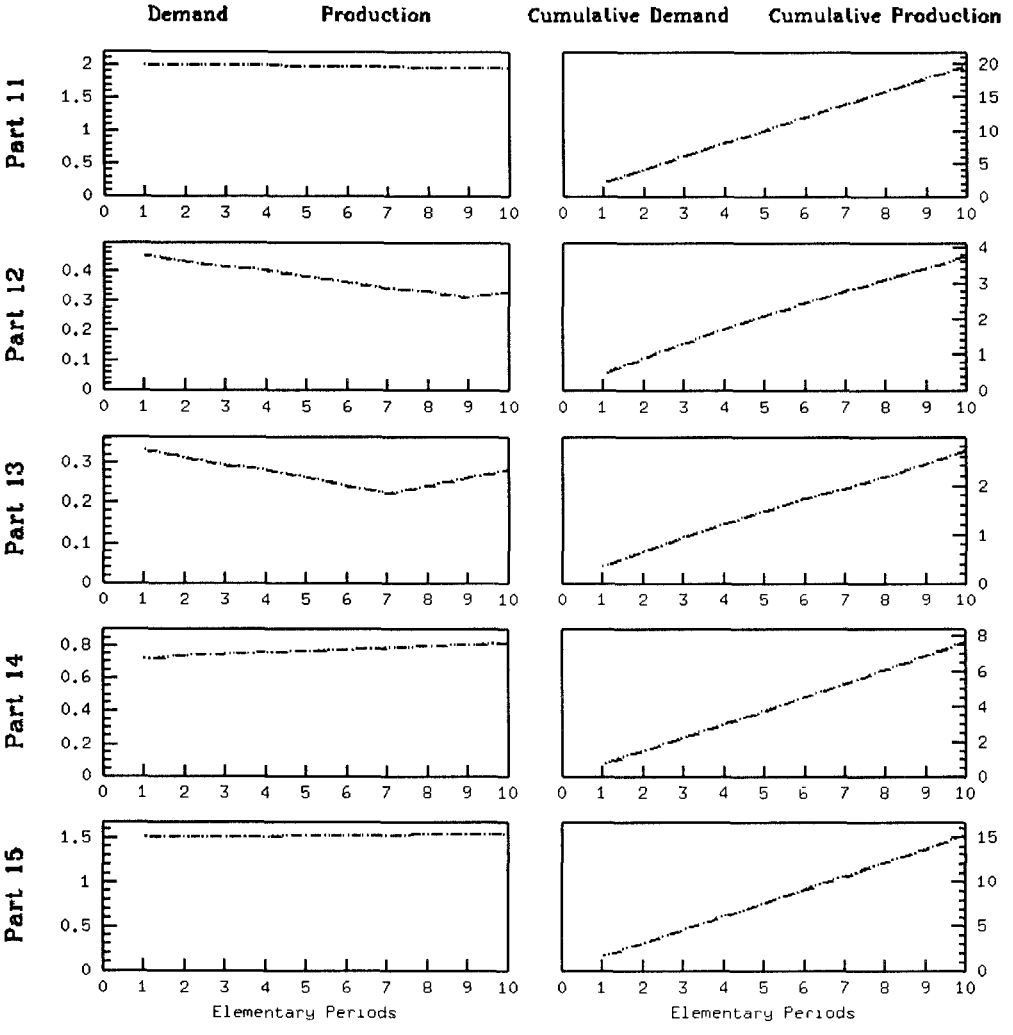
```
Number of Families
7
Number of Cells
6
Family details : Number, <(cell,time), (), ..>
1 0.000 0.000 1.682 0.000 1.820 1.337
2 0.000 0.000 1.200 0.000 1.800 1.450
3 1.242 1.547 1.592 0.000 1.795 1.362
4 0.000 0.000 1.480 0.000 1.793 1.406
5 0.000 0.000 1.486 2.535 1.788 1.371
6 0.000 0.000 1.718 2.439 1.887 1.406
7 1.247 1.463 0.000 2.590 0.000 1.379
```

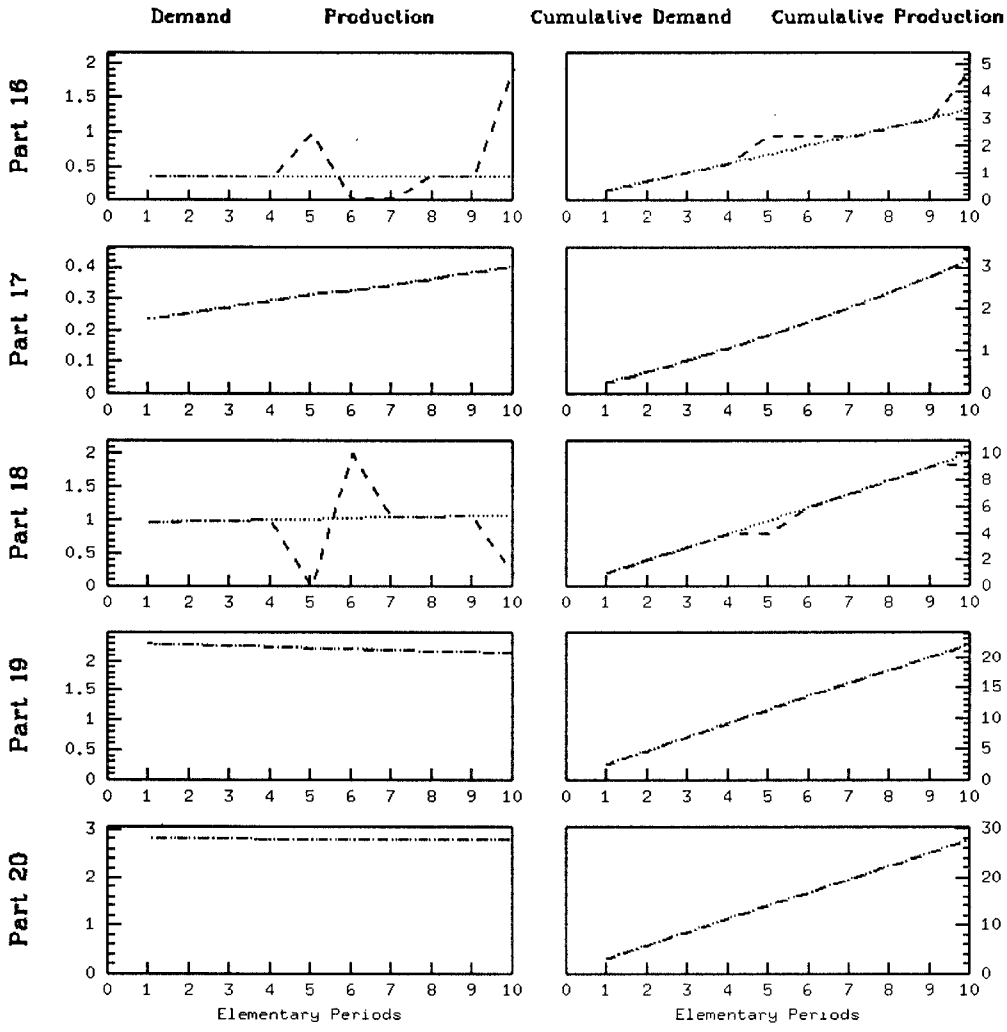
Appendix C

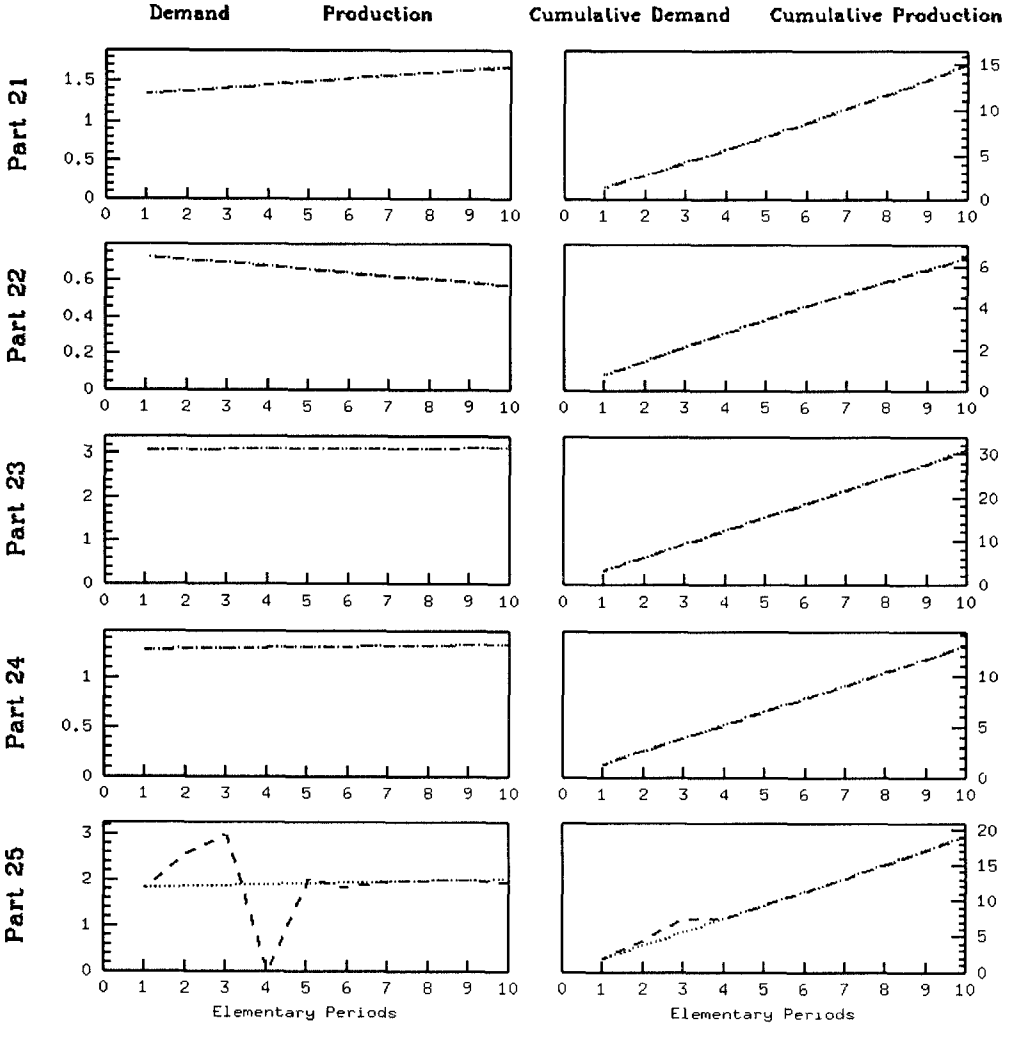
Results of Level 1

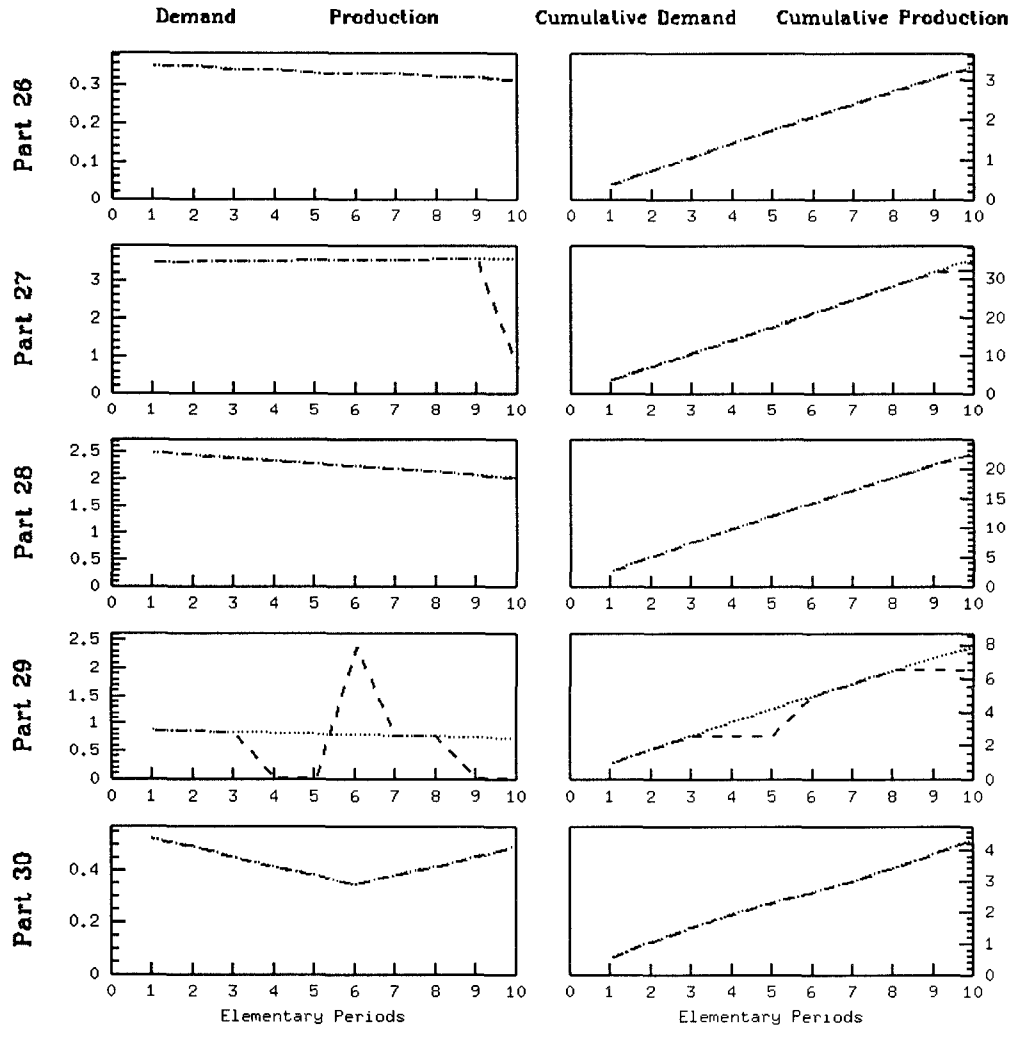


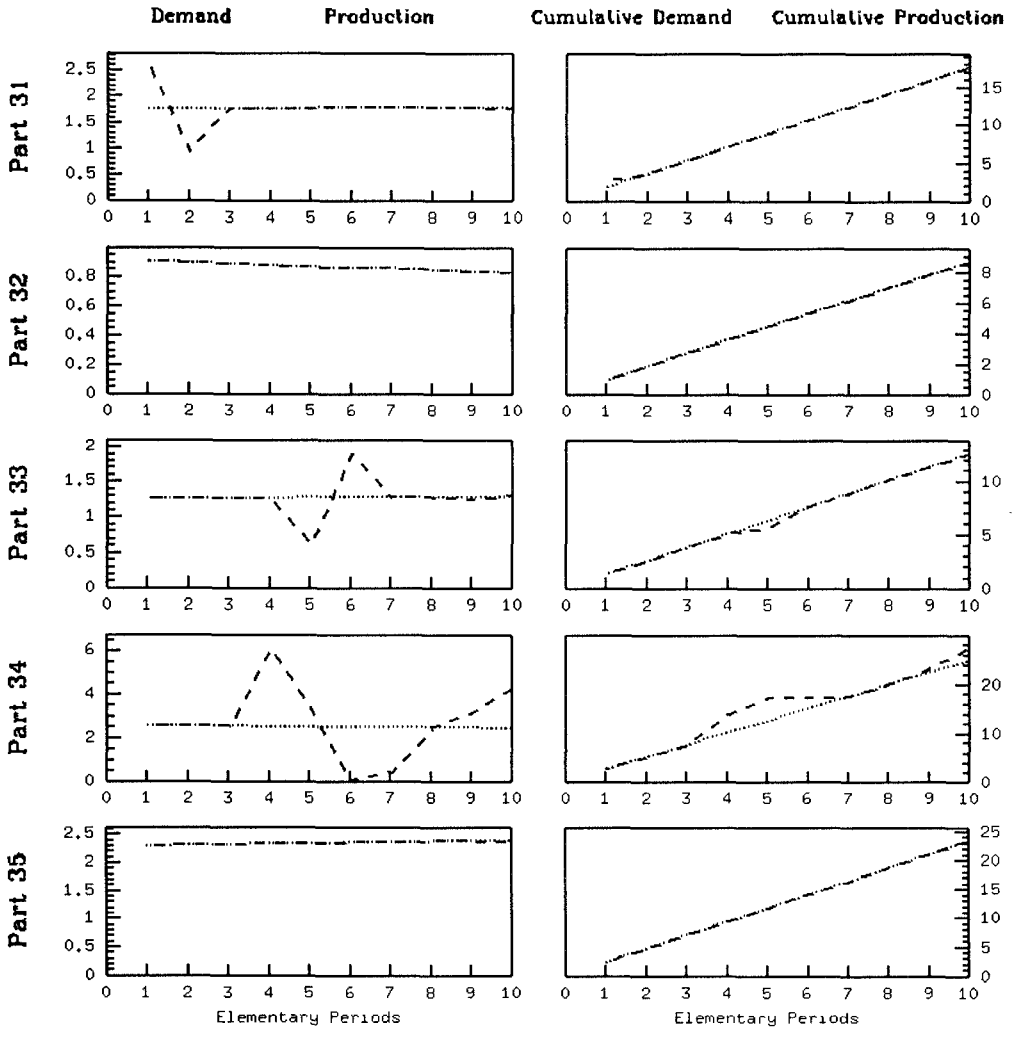

Demand
Production
Cumulative Demand
Cumulative Production
Part 6
Part 7
Part 8
Part 9
Part 10
Elementary Periods
Elementary Periods

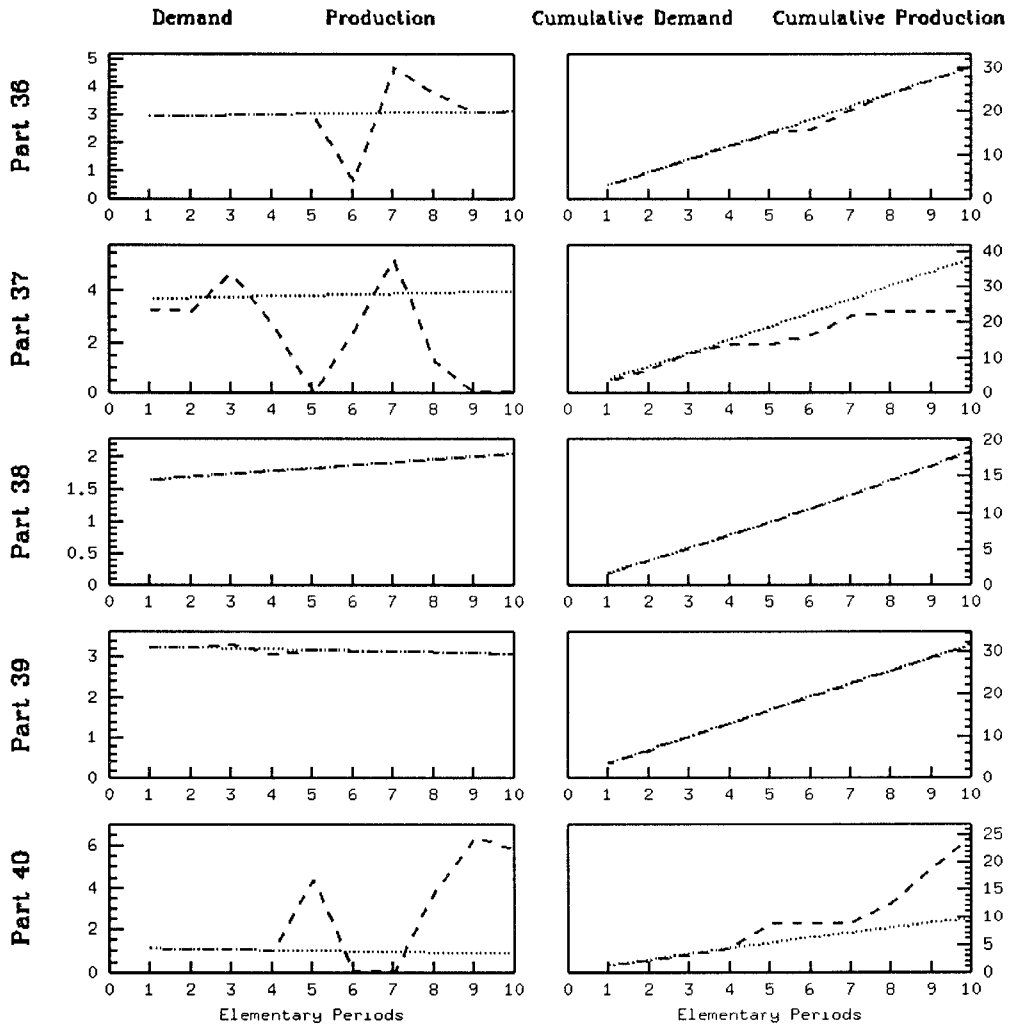

Demand
Production
Cumulative Demand
Cumulative Production
Part 11
Part 12
Part 13
Part 14
Part 15
Elementary Periods
Elementary Periods

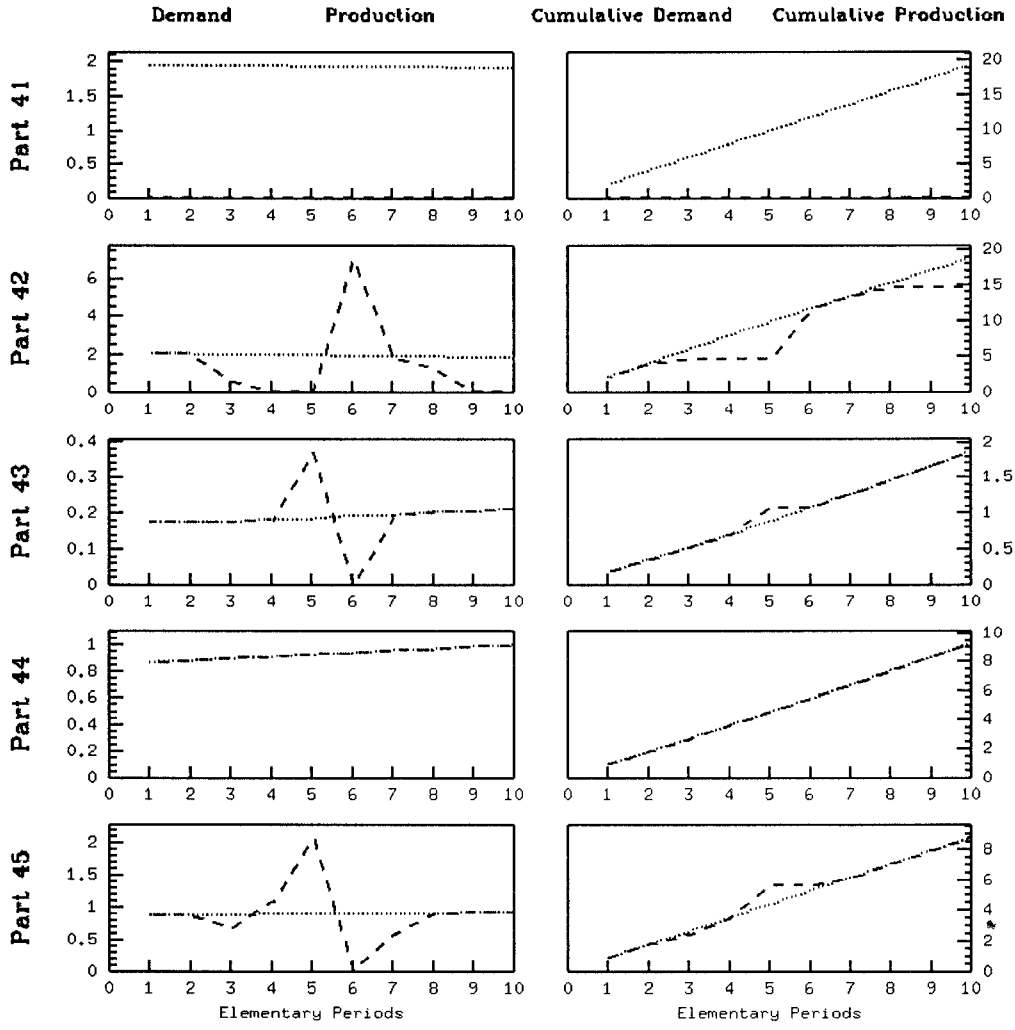

Demand
Production
Cumulative Demand
Cumulative Production
Part 16
Part 17
Part 18
Part 19
Part 20
Elementary Periods
Elementary Periods

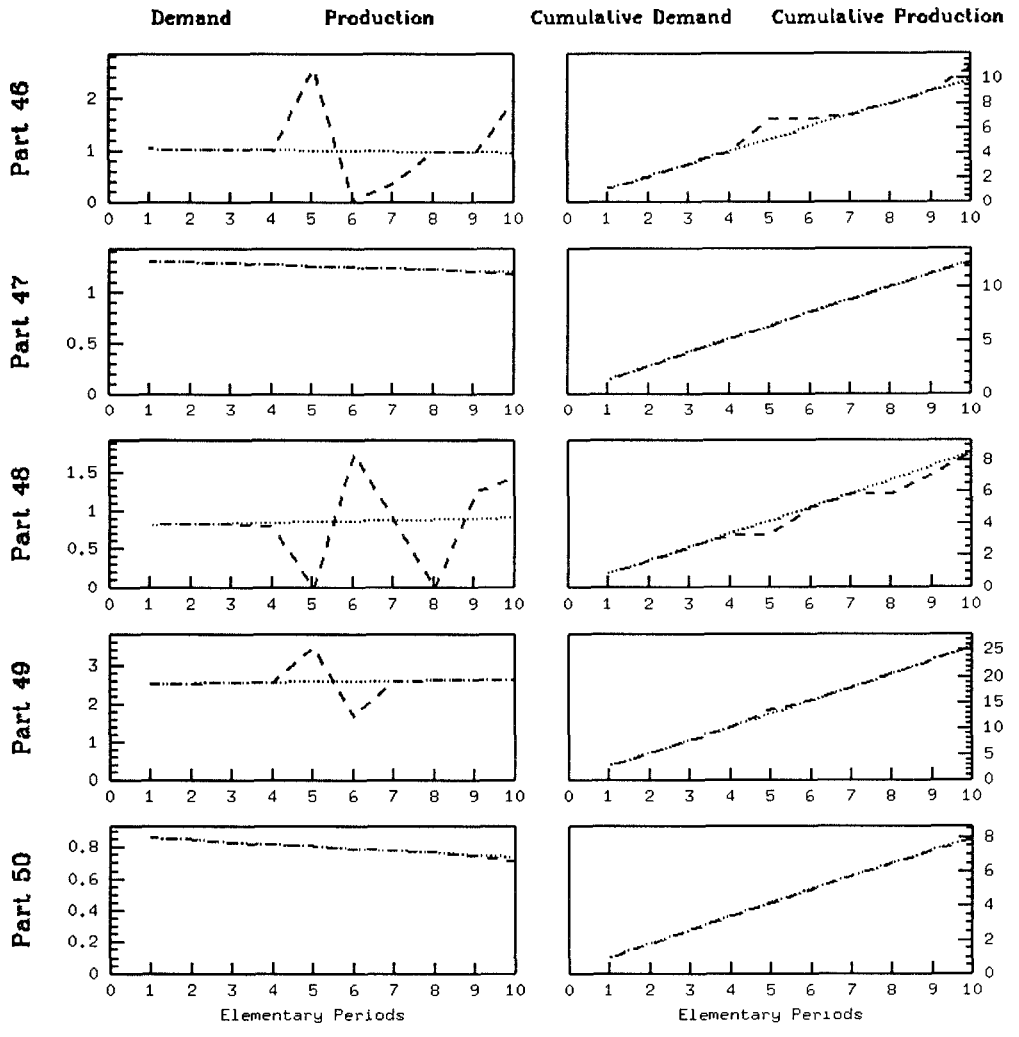

Demand
Production
Cumulative Demand
Cumulative Production
Part 21
Part 22
Part 23
Part 24
Part 25
Elementary Periods
Elementary Periods


Demand
Production
Cumulative Demand
Cumulative Production
Part 26
Part 27
Part 28
Part 29
Part 30
Elementary Periods
Elementary Periods


Demand
Production
Cumulative Demand
Cumulative Production
Part 31
Part 32
Part 33
Part 34
Part 35
Elementary Periods
Elementary Periods


Demand
Production
Cumulative Demand
Cumulative Production
Part 36
Part 37
Part 38
Part 39
Part 40
Elementary Periods
Elementary Periods


Demand
Production
Cumulative Demand
Cumulative Production
Part 41
Part 42
Part 43
Part 44
Part 45
Elementary Periods
Elementary Periods


Demand
Production
Cumulative Demand
Cumulative Production
Part 46
Part 47
Part 48
Part 49
Part 50
Elementary Periods
Elementary Periods

Appendix D

Operation and Instruction Manual -

Design and Operation of Hierarchical Production Management Systems

D.1 Introduction

Design and Operation of Hierarchical Production Management Systems (DOHPMS) is a software package for the design and operation of a three level hierarchy (based on chapter 6 : Application to a generic Job-Shop) . It has 4 modules:

1. Entry of Production System's Data (ED)
2. Design of the HPMS (DHPMS)
3. Operation of the HPMS (OHPMS)
4. Review of Results (RR)

One can use either or all of these modules. Each module consists of a set of programs and options that can be executed independently. However, for the complete execution, all modules should be used according to the sequence in the main menu. Each program responds when invoked, and provides a suggestion about the next course of action. Thus, a few runs through this system will be enough to provide the user with an insight into the functioning and the capabilities of the package.

Any time the user finishes execution of a program, DOHPMS can be quit or continued with the next option. In case the user has exited DOHPMS and would like to go back in, previous operations need not need be repeated. The use of this package assumes that the user is familiar with the concepts of hierarchical design and operation.

Help is available for selected topics & programs. The call for help from outside of DOHPMS is by "$ helpme {argument}" ($ is the system prompt). If the argument does not match with those available, 'helpme' suggests the available topics for choice.

The programs are coded in C and a few in Fortran-77. The programs otherwise portable, the integrated modules are written in Bourne Shell (a standard UNIX command interpreter). In addition there are some

graphics, so the best environment to run this package is on the 'x-windows' on a SUN/UNIX platform.

The main menu of DOHPMS is presented in figure D.1.

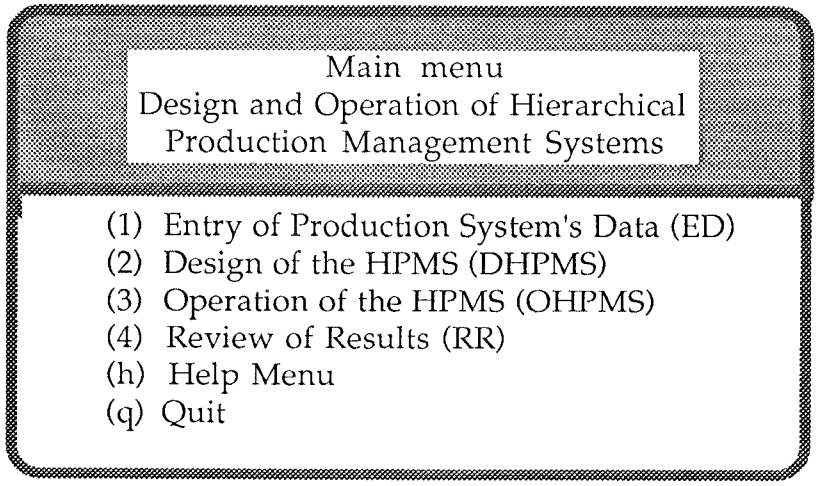


Figure D.1 : Main menu of DOHPMS

D.2 Programs

The software consists of the following programs :

(1) Program to generate random product, work-center and worker data.

(2) Program to generate random demand, and initial worker levels.

(3) Programs for the Design of the Hierarchy :

- (a) Clustering of products.
- (b) Clustering of work-centers.
- (c) Clustering of workers.

(4) Programs for the Operation of the Hierarchy :

- (a) Aggregation of detailed demand.
- (b) Aggregation of initial and maximal worker levels.
- (c) Generation of the Level 3 decision making problem.
- (d) Generation of the Level 2 decision making problem.
- (e) Generation of the Level 1 decision making problem.

(5) Linear program solver : XMP [77] (In Fortran-77).

(6) Programs for graphics (Using "Super Mongo", sm).

D.3 Modules

D.3.1 Entry of Data Module

The Entry of Data module allows a guided creation of information required for the design of the hierarchy. This information is primarily the production system related details of products, work-centers and labor, and their associated attributes. There are essentially four files that must be created (see Outputs below). This module provides format instructions and fields required to be input. It also provides an example file in each case. It is recommended to change that file for easier input, rather than creating a new one.

This module also includes a program for the random creation of these data. The user inputs the number of products and machines, and attributes are generated from some distributions. One can change the program 'createx' if the distributions or their parameters need to be altered.

Programs : create, createx.
Generate random manufacturing system related data.

Outputs : part-ex#, part-ex#.his, machine-ex#, worker-ex#.

Inputs : None

Outputs : part-ex#, part-ex#.his, machine-ex#, worker-ex#.
Note that, ex# stands for the example number; this can be numerals or characters up to 10 field places.

The editing of text files can be performed by an editor of one's choice.

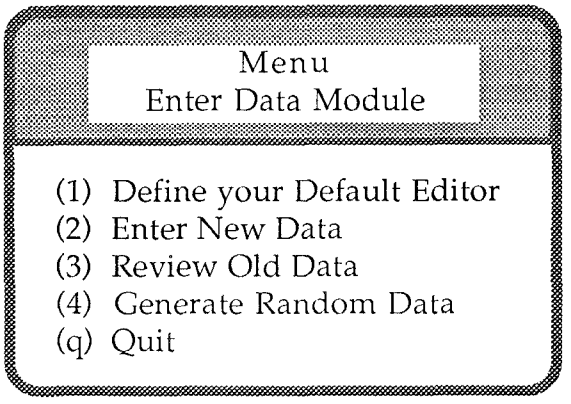


Figure D.2 : Menu of Enter Data Module of DOHPMS

D.3.2 Design of HPMS

The Design module of DOHPMS is intended to design the hierarchy. It essentially consists of programs to cluster or aggregate product entities, machine entities, and worker entities into their respective aggregates. The sequence of aggregations to develop a higher level model is indicated in the main menu (i.e., Products, machines, and workers). The clustering algorithms are a modified version of K-mean cluster analysis; it is parameterized and starts with a random initial set of points. Therefore, the programs may need to be re-run with different 'seeds' till a good clustering is obtained. These programs can also permit the clustering results from text files (obtained by a previous aggregation, or company data); however, these results should conform to the aggregation scheme presented in Chapter 4.

The design process ends with the creation of an 'architecture' file. This file contains information about the number of elementary periods (EP) per horizon and number of EPs per high level EP. Most of this information may not be known at this stage, but one can start with an estimated value. Note that none of the design programs have to be re-run in case the architecture file is changed.

Programs : clusters, cmcs, cws.

(1) "clusters" : Aggregates products into families, or product entities to more aggregate product entities.

(2) "cmcs" : Aggregates work-centers into cells, or machine entities to more aggregate machine entities.

(3) "cws" : Aggregates workers into aggregates, or worker entities to more aggregate worker entities.

Inputs :

(1) clusters : part-ex#, part-ex#.his (1st level of aggregation)
fam2-ex#, fam2-ex#.his (2nd level of aggregation)

(2) cmcs : machine-ex# (1st level of aggregation)
cell-ex# (2nd level of aggregation)

(3) cws : worker-ex#, cell-ex#.res (1st level of aggregation)
group-ex#, cell2-ex#.res (2nd level of aggregation)

Outputs :

(1) clusters : fam-ex#, fam-ex#.his, fam-ex#.res (1st level)
fam3-ex#, fam3-ex#.his, fam3-ex#.res (2nd level)

(2) cmcs : cell-ex#, cell-ex#.res (1st level of aggregation)
cell2-ex#, cell2-ex#.res (2nd level of aggregation)

(3) cws : group-ex# (1st level of aggregation)
group2-ex# (2nd level of aggregation)

(4) arch-ex#.

An editor is required to create the architecture file.

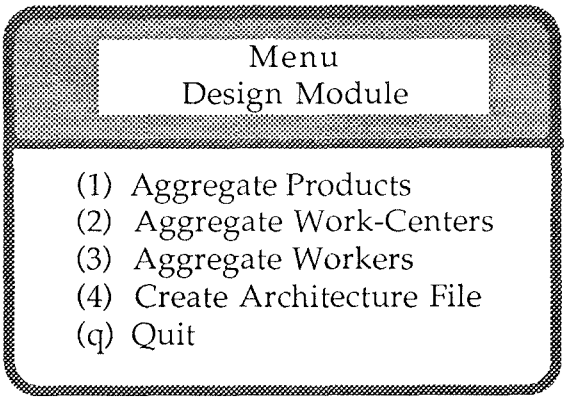


Figure D.3 : Menu of Design Module of DOHPMS

D.3.3 Operation of HPMS

The Operation module of DOHPMS creates and solves the various Decision Making Problems (DMP) of the hierarchy. The operation of the hierarchy is a top-down process. First, the of production demand, and the initial and maximal worker levels are input. Then, the detailed data are translated to higher levels according to the aggregation results as well as the architecture information ('arch-ex#'). The solution procedure begins with setting up the level 3 problem, and solves it. The results of this level serve as inputs to the subsequent level. The operation module sets up the linear programming problems at each level, and solves them using XMP [77].

Programs : gendem, gendemx, aggdem, aggiwl, create_iwl,
sx3, x3, sx2, x2, sx1, x1.

(1) "gendem" and "gendemx"

Generates random sinosoidal demand for products with random cyclicity magnitude and phase shift.

(2) "create_iwl"

Creates initial and maximal worker levels based on the long term production volumes of products.

(3) "aggdem"

Aggregates demand of detailed entities to demand of aggregate entities over aggregate time elementary periods.

(4) "aggiwl"

Aggregates initial and maximal worker levels of detailed worker entities to corresponding quantities for aggregate entities over aggregate time elementary periods.

(5) "sx*" (sx3, sx2, sx1)

Generates lpp of Level *.

(6) "x*" (x3, x2, x1)

Solves lpp of Level *.

Inputs :
(1) gendem(x) : part-ex#.his, part-ex#
(2) aggdem : demand1-ex#, arch-ex#, fam-ex#.res, fam3-ex#.res
(3) create_iwl : worker-ex#, part-ex#.his, part-ex#
(4) aggiwl : iwlev1-ex#, maxwlev1-ex#, arch-ex#, cell(2)-ex#.res
(5) sx* : all the above files.
(6) x* : data*.

(2) Given : demand1-ex#, demand2-ex#, demand3-ex#, iwlev3-ex#, maxwlev(3 and 2)-ex#, fam4-ex#, fam2-ex#, part-ex#, and arch-ex#.

Outputs :
(1) gendem(x) : demand1-ex#
(2) aggdem : demand2-ex#, demand3-ex#
(3) create_iwl : iwlev1-ex#, maxwlev1-ex#
(4) aggiwl : iwlev2-ex#, iwlev3-ex#, maxwlev2-ex#, maxwlev3-ex#
(5) sx* programs develop the lpp in 'data*'
(6) x* solves lpp : prod3.res, prod2.res, prod1.res, worklev3.res, worklev2.res.

The linear programming software is "XMP".

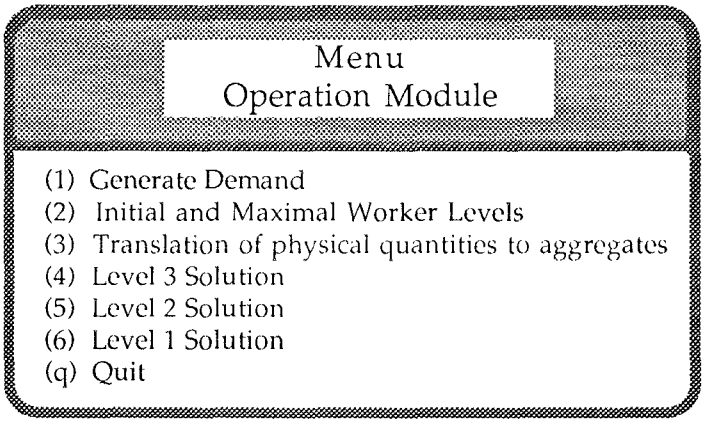


Figure D.4 : Menu of Operation Module of DOHPMS

D.3.4 Review of Results

The Review of Results module allows viewing of the computational results of the top-down operation of the hierarchy. Solutions to the Decision Making Problems (DMP) at each level of the hierarchy can be viewed in a graphical form. Each level provides curves for the production of the product entity relevant to that level, over the corresponding elementary periods (EP). In addition, the other controls (if any) are also displayed.

For each level, we have the details as follows :
(1) Level 3 displays : (i) Demand and Production over EPs, (ii) Cumulative Demand and Cumulative Production over EPs, (iii) Worker Levels over EPs, and (iv) Hiring/Firing Control over EPs.
Note that these details are related to the entities corresponding to Level 3.
(2) Level 2 displays : (i) Demand and Production over EPs, (ii) Cumulative Demand and Cumulative Production over EPs, (iii) Worker Levels over EPs, and (iv) Overtime Levels (a Control) over EPs.
Note that these details are related to the entities corresponding to Level 2.
(1) Level 3 displays : (i) Demand and Production over EPs, and (ii) Cumulative Demand and Cumulative Production over EPs for detailed products.

Programs : g3, g2, g1.
(1) "g*" (g3, g2, g1)
Displays graphical of level * solution

Inputs :
(1) Results : prod3.res, prod2.res, prod1.res, worklev3.res, worklev2.res.
(2) Given : demand1-ex#, demand2-ex#, demand3-ex#, iwlev3-ex#, maxwlev(3 and 2)-ex#, fam4-ex#, fam2-ex#, part-ex#, and arch-ex#.

The underlying Plotting software is "Super Mongo" or sm.
The results can be displayed through X-WINDOW graphical tool.

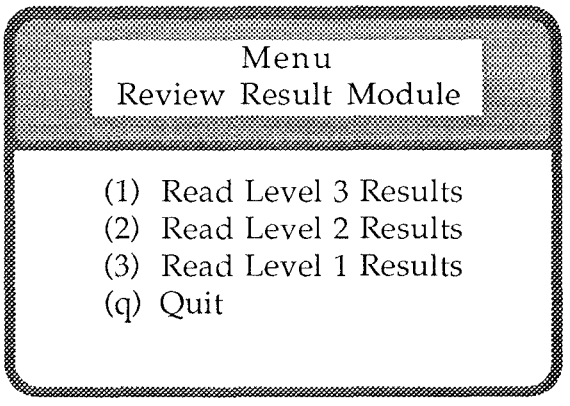


Figure D.5 : Menu of Review Result Module of DOHPMS

D.4 Getting help from DOHPMS - 'helpme'

A feature for obtaining online help has also been provided for limited topics : (i) functions of modules, and (ii) programs, and their functions.

The help feature is composed of the following parts :

(1) The main one called "helpme" An executable file in C - Shell
(2) The child "helpme.sh" An executable file in Bourne - Shell
(3) "helpdir" A directory containing the following text files:
- module_ed, module_des, module_op , module_rev
- programs_ed, programs_des , programs_op , programs_rev

Help can be obtained by typing "helpme" at the system prompt, which may optionally be followed by an argument about the specific topic. When helpme is invoked without an argument, or the argument is "dohpms", then the main shell (helpme) calls the program "helpme.sh". This is an interactive menu based procedure, that provides the user with help to a topic as per his response to the options put forward. If "helpme" is called with an argument that corresponds to one of the text files in the help directory ('helpdir'), then the text of that specific topic is displayed. If an unrecognized argument is supplied after "helpme", then "helpme" displays a menu of the topics about which help exists, and then calls the program "helpme.sh".

References

[1] Abdul-Razaq, T.S. and Potts, C.N. (1988) : "Dynamic Programming State-Space Relaxation for Single-Machine Scheduling," Journal of Opl. Res. Soc., Vol. 39, No. 2, pp 141-152, 1988.

[2] Abraham, C., Dietrich, B., Graves, S., Maxwell, W. and Yano, C. (1985): "A Research Agenda for Models to Plan and Schedule Manufacturing Systems," presented at an NSF workshop as "Scheduling the Factory of the Future: A Research Planning Session," Decision Sciences Department, University of Pensylvania, March 1985, and Boston TIMS/ORSA meeting (1985).

[3] Akella, R., Choong, Y. and Gershwin, S.B. (1984) : "Performance of Hierarchical Production Scheduling Policy," IEEE Transactions on Components, Hybrids, and Manufacturing Technology, Vol. CHMT-7, No. 3, 1984.

[4] AMICE Vol. 1 (1989) : "Open System Architecture for CIM," Springer-Verlag, Berlin, 1989.

[5] Anthony, R.N. (1965) : "Planning and Control Systems : A Framework for Analysis," Harvard University, Graduate School of Business Administration, MA, 1965.

[6] Armstrong, R.J. and Hax , A.C. (1977) : "A Hierarchical Approach for a Naval Tender Job-Shop Design," Applied Mathematical Programming, Chap. 10, S.P. Bradley, A.C. Hax and T.L. Magnanti. Addison-Wesley, 1977.

[7] Axsater, S. (1981) : "Aggregation of Product Data for Hierarchical Production Planning," Operation Research, Vol. 29, No. 4, 1981.

[8] Baker, K.R. (1974) : "Introduction to Sequencing and Scheduling," Wiley, 1974.

[9] Baker, K.R. and Scudder, G.R. (1990) : "Sequencing with Earliness and Tardiness Penalties : A Review," Operations Research, Vol. 38, No. 1, pp. 22-36, 1990.

[10] Bitran, G.R., Haas, E.A. and Hax, A.C. (1981) : "Hierarchical Production Planning : A Single Stage System," Operations Research, Vol. 29, No. 4, 1981.

[11] Bitran, G.R., Haas, E.A. and Hax, A.C. (1982) : "Hierarchical Production Planning : A Two Stage System," Operations Research, Vol. 30, No. 2, 1982.

[12] Bitran, G.R. and Hax, A.C. (1977) : "On the Design of Hierarchical Planning Systems," Decision Sciences, Vol. 8, 1977.

[13] Bitran, G.R. and Hax, A.C. (1981) : "Disaggregation and Resource Allocation using Convex Knapsack Problems with Bounded Variables," Management Science, Vol. 27, No. 4, 1981.

[14] Bonnevie, A. and Krzesinski, P. : "A Double Approach for Analysis and Design of Production Systems," ESPRIT'86, Results and Achievements, pp. 469-478, 1987.

[15] Chen, P, P-S. (1976) : The Entity-Relationship Model - Towards a Unified View of Data," ACM Transactions on Database Systems, Vol. 1, No. 1, pp 9-36, 1976.

[16] Chow, J.H. and Kokotovic, P.V. (1985) : "Time Scale Modeling of Sparse Dynamic Networks," IEEE Transactions on Automatic Control, Vol. AC-30, No. 8, 1985.

[17] Conway, R.W., Maxwell, W.C. and Miller, L.W. (1967) : "Theory of Scheduling," Addison-Wesley, 1967.

[18] Davis, W.J. and Jones, A.T. : "A real-time production scheduler for a stochastic manufacturing environment," Int. J. Computer Integrated Manufacturing, Vol. 1, No. 2, pp 101-112, 1988.

[19] Dempster, M.A.H., Fisher, M.L., Lageweg, B., Jansen, L., Lenstra, J.K. and A.H.G. Rinnoy Kan (1981) : "Analytical evaluation of Hierarchical Planning Systems," Operations Research, Vol. 29, No. 4, 1981.

[20] Doumeingts, Guy (1985) : "How to Decentralize Decisions through GRAI Model in Production Management," Computers in Industry 6, 1985.

[21] Doumeingts, Guy (1986) : "Artificial Intelligence Concept Techniques for Computer Integrated Manufacturing," Flexible Manufacturing Systems: Methods and Studies, edited by Kusiak, A., Elsevier Science Publishers B.V., 1986.

[22] Emmons, H. (1969) : " One-Machine Scheduling to Minimize certain Functions of Job Tardiness," Operations Research, Vol. 17, pp 701-715, 1969.

[23] Erscheler, J., Fontan, G. and Merce, C. (1986) : "Consistency of the Disaggregation Process in Hierarchical Planning," Operations Research, Vol. 34, No. 3, 1986.

[24] Fox, M. and Smith, S.F. (1984): "ISIS: A Knowledge-Based System for Factory Scheduling," Expert Systems, Vol. 1, No. 1, pp. 25-49, 1984.

[25] Fox, M., Greenberh, M., Sathi, A., Mattic, J. and Rychnener, M. (1983): "CALLISTO: An Intelligent Project Management System," working paper, Intelligent Systems Lab, The Robotics Institute, Carnegie-Mellon University, November, 1983.

[26] French, S. (1982) : "Sequencing and Scheduling," Ellis Horwood Limited (1982).

[27] Fry, T.D., Armstrong, R.D. and Blackstone, J.H. (1987) : "Minimizing Weighted Absolute Deviation in Single Machine Scheduling," IIE Transactions, Vol. 10, No. 4, pp 445-450, 1987.

[28] Gabby, H. (1975) : "A Hierarchical Approach to Production Planning," TR-120, Operations Research Center, M.I.T, Cambridge, MA, 1975.

[29] Garey, M.R. and Johnson, D.S. (1979) : "Computers and Intractability : A Guide to the Theory of NP-Completeness," W.H. Freeman and Co., New York, 1979.

[30] Garey, M.R., Tarjan, R.E. and Wilfong, G.T. (1988) : "One-Processor Scheduling with Symmetric Earliness and Tardiness Penalties," Math. Opns. Res., Vol 13, pp. 330-348.

[31] Gelders, L. and Kleindorfer, P.R. (1974) : "Coordinating Aggregate and Detailed Scheduling in the One-Machine Job Shop: Part I. Theory," Operations Research, Vol. 22, No. 1, pp. 46-60, 1974.

[32] Gelders, L. and Kleindorfer, P.R. (1975) : "Coordinating Aggregate and Detailed Scheduling in the One-Machine Job Shop: Part I. Theory," Operations Research, Vol. 23, No. 2, pp. 312-324, 1975.

[33] Gelders, L.F. and van Wassenhove, L.N. (1982): "Hierarchical Integration in Production Planning : Theory and Practice," Journal of Operations Management, Vol. 3, No. 1, 1982.

[34] Gershwin, S.B. (1986) : "Stochastic Scheduling and Set-ups in Flexible Manufacturing Systems," Proceedings of the 2nd ORSA/TIMS Conference on Flexible Manufacturing Systems : OR Models and Applications, K. Stecke and R. Suri eds., 1986.

[35] Gershwin, S.B. (1988) : "Hierarchical Flow Control : A Framework for Scheduling and Planning Discrete Event Manufacturing Systems," IEEE Proceedings : Special Issue on Discrete Event Systems.

[36] Graves, S.C. (1982) : "Using Lagrangean Techniques to solve Hierarchical Production Planning Problems," Management Science, Vol. 28, No. 3, 1982.

[37] Gupta, S.K. and Kyparisis, J. (1987) : "Single Machine Scheduling Research," OMEGA Int. J. Mgmt. Sci., Vol. 15, No. 3, pp. 207-227, 1987.

[38] Harhalakis, G., Lin, C. P., Mark, L. and Muro, P. (1991) : "Formal Representation, Verification and Implementation of Rule Based INformation Systems for Integrated Manufacturing ()," Technical Report TR 91-19, Systems Research Center, University of Maryland, College Park, 1991.

[39] Hartigan, J.A. (1975) : "Clustering Algorithms," John Wiley & Sons, 1975.

[40] Hax, A.C. and Meal, H.C. (1975) : "Hierarchical Integration of Production Planning and Scheduling," in Studies in the Management Sciences, M.A. Geisler, ed., Vol. 1, Logistics, North Holland - American Elsevier, 1975.

[41] Hildebrant, R.R. (1980) : "Scheduling and Control of Flexible Manufacturing Systems whose Machines are Prone to Failure," Ph.D. Thesis, M.I.T. Dept. of Astronautics and Aeronautics, August 1980.

[42] Hilderbrant, R.R. and Suri, R. (1980) : "Methodology and Multi-level Algorithm for Scheduling and Real-time Control of Flexible Manufacturing Systems," Proceedings of 3rd. International Symposium on Large Engineering Systems, Memorial University of Newfoundland, July, 1980.

[43] Hilger, J. (1988) : "Langage de Production: Description Coherente du Court Terme (premier partie)," INRIA, France, Research Report No. 912, 1988.

[44] Hillion, H., Meier, K. and Proth, J.M. (1987) : "Production Subsystems and Part-families: The Tope Level Model in Hierarchical Production Planning Systems," Operational Research'87, G.K. Rand, ed., Elsevier Science Publishers B.V. (North-Holland), © IFORS, 1988.

[45] Jackson, R.H.F. and Jones, A.W.T. : "Hierarchical Control and Real-time Optimization in Automated Manufacturing Systems," National Bureau of Standards Technical Report NBSIR 86-3503, 1986.

[46] Jones, A.T. and Mclean, C. : "A proposed Hierarchical Control Model for Automated Manufacturing Systems," Journal of Manufacturing Systems, Vol. 5, No. 1, pp. 15-25, 1986.

[47] Jorysz, H.R. and Vernadat, F.B. (1990) : "CIM-OSA Part 1 : Total Enterprise Modelling and Functional View," Int. J. Computer Integrated Manufacturing, Vol. 3, No. 3, pp. 144-156, 1990.

[48] Jorysz, H.R. and Vernadat, F.B. (1990) : "CIM-OSA Part 2 : Information View," Int. J. Computer Integrated Manufacturing, Vol. 3, No. 3, pp. 157-167, 1990.

[49] Karmarkar, U.S. (1981) : "Equalization of Run-Out Times," Operations Research, Vol. 29, No. 4, 1981.

[50] Klittich, M. (1990) : "CIM-OSA Part 3 : Integrating Infrastructure-the Operational Basis for Integrated Manufacturing Systems," Int. J. Computer Integrated Manufacturing, Vol. 3, No. 3, pp. 168-180, 1990.

[51] Kimemia, J. (1982) : "Hierarchical Control of Production in Flexible Manufacturing Systems," Ph.D. Thesis, M.I.T. Dept of Electrical Engineering and Computer Science, 1982.

[52] Kimemia, J. and Gershwin, S.B. (1983) : "An Algorithm for the Computer Control of a Flexible Manufacturing System," IIE Transactions, Vol. 15, No. 4, pp 353-362, 1983.

[53] Kosanke, K. and Klevers, T. (1990) : "CIM-OSA: architecture for enterprise integration - A report on current developments," Computer-Integrated Manufacturing Systems, Vol. 3, No. 1, 1990.

[54] Krajewski, L.J. and Ritzman, L.P. (1977) : "Disaggregation in manufacturing and service organizations: survey of problems and research," Decision Sciences, Vol. 8, No. 1, 1977.

[55] Lakshminarayan, S., Lakshman, R., Papineau, R.L. and Rochette, R. (1978) : "Optimal Single-Machine Scheduling with Earliness and Tardiness Penalties," Opns. Res., Vol 26, pp. 1079-1082, 1978.

[56] Lawler, E.L. (1977) : "A "Pseudopolynomial" Algorithm for Sequencing Jobs to Minimize Total Tardiness," Ann. Discrete Math., Vol 1, pp. 331-342, 1977.

[57] Libosvar, C. (1988) : "Hierarchical Production Management : The Flow Control Layer," Ph.D. Thesis, University of Metz, France, 1988.

[58] Maimon, O.Z. and Gershwin, S.B. (1988) : "Dynamic Scheduling and Routing for Flexible Manufacturing Systems that have Unreliable Machines," Operations Research, Vol. 36, No. 2, 1988.

[59] Maxwell, W., Muckstadt, J.A., Thomas, J. and VanderEecken, J. (1983): "A Modeling Framework for Planning and Control of Production in Discrete Parts Manufacturing Systems and Assembly Systems," Interfaces, Vol. 13, 1983.

[60] Meier, K. (1989) : "Commande Hierarchisee d'un Systeme de Production," Ph.D. Thesis, University of Metz, France, 1989.

[61] Mesarovic, M.D., Macko, D. and Takahara, Y. (1970) : "Theory of Hierarchical , Multilevel, Systems," Academic Press, Inc. New York, 1970.

[62] Morton, T.E., Mark, F. and Sathi, A. (1984): "PATRIARCH; A Multilevel System for Cost Accounting, Planning, Scheduling," working paper GSIA, Carnegie-Mellon University, May, 1984.

[63] Morton, T.E. and Smunt, T.L. (1986) : "A Planning and Scheduling System for Flexible Manufacturing," Flexible Manufacturing Systems: Methods and Studies, edited by Kusiak, A., Elsevier Science Publishers B.V., 1986.

[64] Nagi, R., (1988) : "Selection and Layout of Facilities for Cellular Manufacturing Systems," M.S. Thesis, University of Maryland, 1988.

[65] O'Grady, P.L. (1986) : "Controlling Automated Manufacturing Systems," Kogan Page Ltd, 1986.

[66] Ow, P.-S. and Morton, T.E. (1989) : "The Single Machine Early/Tardy Problem," Management Science, Vol. 35, No. 2, pp 177-191, 1989.

[67] Roboam, M., Zanettin, M. and Pun, L. (1989) : "GRAI-IDEF0-Merise (GIM): Integrated methodology to analyse and design manufactuing systems," Computer Integrated Manufacturing Systems, Vol. 2, No. 2, 1989.

[68] Sandell, N.R., Varayia, P., Athans, M.A. and Safonov, M. (1978) : "A Survey of Decentralized Control Methods for large Scale Systems," IEEE Transactions on Automatic Control, Vol. AC-23, No. 2, 1978.

[69] Sen, T. and Gupta, S.K. (1984) : "A State-of-the-Art Survey of Static Scheduling Research Involving Due Dates," OMEGA, Vol 12, No. 1, pp. 63-76, 1984.

[70] Shaw, M.J.P. and Whinston, A.B. (1986) : "Applications of Artificial Intelligence to Planning and Scheduling in Flexible Manufacturing," Flexible Manufacturing Systems: Methods and Studies, edited by Kusiak, A., Elsevier Science Publishers B.V., 1986.

[71] Sidney, J. (1977) : "Optimal Single-Machine Scheduling With Earliness and Tardiness Penalties," Opns. Res., Vol 25, pp. 62-69, 1977.

[72] Smith, W.E. (1956) : "Various Optimizers for Single-Stage Production," Nav. Res. Logist. Quart., Vol 3, pp. 59-66, 1956.

[73] Smith, S.F., Ow, P.S. and Matthys, D.C. (1989) : "OPIS : An Opportunistic Factory Scheduling System," Proceedings Int. Symposium for Computer Scientists, Bejing, China, August, 1989.

[74] Stecke, K. E. (1986) : "Useful Models to Address FMS Operating Problems," Flexible Manufacturing Systems: Methods and Studies, edited by Kusiak, A., Elsevier Science Publishers B.V., 1986.

[75] Villa, A., Mosca, R. and Murari, G. (1986) : "Expert Control Theory: A Key for Solving Production and Control Problems in Flexible Manufacturing," Proceedings of the 1986 IEEE Conference on Robotics and Automation, San Francisco, CA.

[76] Xie, X.-L. (1989) : "Real Time Scheduling and Routing for Flexible Manufacturing Systems with Unreliable Machines," Recherche opérationelle/Operations Research, Vol. 23, No. 4, pp 355-374, 1989.

[77] Xie, X.-L. (1989) : "Controle Hierarchique d'un Systeme de Production soumis a Perturbations," Ph.D. Thesis, University of Nancy I, France, 1989.

[78] XMP (1981) or Marsten, R. E. : "The design of the XMPLP Library," Transactions on Mathematical Software, Vol. 7, No. 4, Dec 1981.

[79] Zoller, K. (1971) : "Optimal Disaggregation of Aggregate Production Plans," Management Science, Vol. 17, No. 8, 1971.