Introduction

Production Management is one of the key functions in modern industries. Although other functions may be optimized, the success of a manufacturing company relies heavily on the performance of production decisions.

More than ever customers become selective with rapidly changing needs. The single international market where companies are competing strengthens this phenomenon. As a matter of fact, any manufacturing company faces such antagonist decisions as producing items with shorter and shorter life cycles, in small quantities and with the lowest possible costs. Therefore, the ability of making the right product at the right time and at the lowest possible cost, still remains the key to success for a manufacturing company. *Production Planning and Scheduling* are among the key functions allowing a manufacturing company to meet such an objective.

Production planning and scheduling can be regarded as the activities used to manage the product manufacturing process, starting from the moment demand is known or forecasted until the product has been made available for shipping.

The demand can either be expressed by customers or determined by forecast. In both cases the demand has a deadline (or due date). Since the production capacity is limited, the manufacturing needs planning so as to equilibrate the overall production load over a time horizon and so as to reduce production costs. Hence, one important issue is the effective use of the available capacity, which is achieved when carrying out good production planning and scheduling.

Based on known or forecasted demand and available capacity, *production planning aims* at building an aggregate production plan over a planning horizon. The objective here is to balance production load over a time horizon in order to reduce production costs and make an effective use of the available capacity (available resources in terms of workforce, machines, etc) while meeting the demand. Production planning uses aggregate data (i.e. product families, aggregate capacity, etc) to come up with a production plan over a medium term horizon.

Production Scheduling consists in assigning *specific* scarce resources (workforce, machines, tools, etc) to *specific* items, tasks or jobs. One should note that scheduling is used in such different domains as manufacturing, computing, aircraft landing, hospitals, etc. In a manufacturing environment, one first has to assign scarce resources to jobs and then sequence the assigned jobs on each resource. The scheduling objective, for example, might consist in building a schedule with the smallest length (minimizing makespan). It might also consist in meeting job due dates. Unlike production planning, scheduling uses accurate data to generate a ready-to-execute production plan for a short-term horizon. In

Chapter I we highlight why these two functions (production planning and scheduling functions) are handled separately.

In this project, we are mainly concerned with process industries like cosmetics, pharmaceuticals, textile and food industries. In these industries, production facilities are organized as multistage production flowshop facilities where a production stage may be made up of parallel production lines, machines or any other production facilities. Such production processes are called multiprocessor flowshop or Hybrid Flow Shop (HFS) processes. At some stages, the facilities (machines, lines, etc) are duplicated in parallel in order to increase the overall capacity of the shop floor, or in order to balance the capacities of the stages, or either to eliminate or reduce the impact of bottleneck stages on the overall shop floor capacity. A product, a task or a job consists of one operation for each stage on one of the parallel machines. Those operations have to be realized sequentially, from the first to the last stage. The multistage flowshop is considered as a hybrid flowshop if one stage at least is made up of more than one machine.

The industry concern and need for effective tools to deal with planning and scheduling functions in the HFS environment is evidenced by the increasing number of theoretical and empirical developments. In this project, we have decided to study the scheduling problem of the hybrid flowshop. Our main objective is to design effective solutions by studying several scheduling algorithms.

We now present the study conducted in this thesis. The research developed in each chapter is briefly introduced. In this introduction, to help the reader to assess the contribution of this thesis, the new developments or results with respect to the existing literature are underlined.

Chapter I states the general production planning and scheduling problem and highlights some drawbacks of classical solutions. The production planning problem in a hybrid flowshop is then presented. The main objective consists of presenting a theoretical decomposition-based approach for solving the global HFS planning and scheduling problem. The main issue in this approach is to overcome non-efficient capacity utilization while constructing a good production plan (with the lowest overall production and holding costs). The proposed approach is referred to as "*theoretical*", since no global empirical implementation has been carried out so far. Actually, this is out of the scope of this project since we have concentrated our study on an in-depth analysis of the scheduling part of the system.

At the beginning of our research, a review of the literature dealing with HFS problems (see Chapter II) has shown that the HFS scheduling problem had not been well-studied and that effective algorithms for finding good solutions for large scale problems did not exist. Also, because scheduling is so critical to the approach proposed for solving the global production planning and scheduling problem, our research has then been oriented towards the scheduling of the HFS.

In Chapter II, the HFS scheduling problem we are to study is first introduced, and then a survey of the HFS scheduling literature is presented. Based on the state of the art of general scheduling solutions, we present the research orientation we have decided to

pursue in this project, that is the study of the branch and bound algorithms through the design of branching schemes, upper bounds as well as lower bounds so as *to minimize the makespan*. These three topics are dealt with in separate chapters.

The graph representation of the HFS scheduling problem allows one to understand the HFS problem as well as to design scheduling solutions. In such a graph representation, the HFS can be regarded as being made up of a set of single stage subproblems with *release dates and tails*, each stage being composed of parallel machines. Therefore, a relaxation of the HFS problem into a single stage subproblem can be inferred. This relaxation is used to design lower and upper bounds on the HFS global objective function. The single stage subproblem will then be introduced, and some known results recalled. These solutions are used as submodules in the algorithms we are to develop.

Minimizing the makespan of the hybrid flowshop (HFS) scheduling problem is NP-hard. Since algorithms for finding optimal solutions in polynomial time are unlikely to exist, we therefore have decided to study heuristic solutions. The main objective in Chapter III is the design of a set of heuristics allowing to find the best possible solution. This chapter aims at studying a number of already existing and new heuristics, by presenting and comparing them on different shop configurations. These heuristics use five different approaches. The first approach uses random scheduling so as to see whether elaborate scheduling methods are worth developing or whether any randomly built solution suffices. The second approach schedules the HFS one stage at a time and uses different stage sequencing orders. The third and fourth approaches are mainly list heuristics and use several policies to assign jobs to machines. The third approach uses ideas derived from the multistage classical flowshop with a single machine per stage, while the fourth approach uses classical dispatching priority rules for jobshop scheduling problems. The fifth approach uses local search techniques. The neighborhood structure used to define exchange procedures is based on the graph representation of an HFS schedule.

All heuristics are tested on instances with different sizes in terms of number of jobs, machines and stages, and on several classes of shop configurations in terms of existence of initial release dates and final tails, as well as of bottleneck stages. Statistical analysis is carried out in order to compare the heuristics performances and to eventually select the best subset of heuristics for each shop configuration. Those selected will be used to compute upper bounds in the branch and bound algorithms we are to develop.

Our second concern is the development of lower bounds. In Chapter IV we study lower bounds on the makespan of the HFS scheduling problem. Finding the optimal preemptive schedule in a two-stage HFS is NP-hard. The relaxed single stage parallel machines subproblem with release dates and tails is also NP-hard. All existing HFS lower bounds are based on the single stage subproblem lower bounds. Some of those lower bounds are recalled and tighter lower bounds are developed.

A lower bound for the single stage subproblem is the Preemptive Bound (PB), which is the optimal solution for the preemptive case. PB is obtained in polynomial time, i.e. $O(p_{max}n^3)$, where n is the number of jobs and p_{max} is the largest processing time, but the computation complexity is too high for use in a branch and bound framework. A less time consuming but less tight lower bound is the subset bound (SSB) which can be

computed in $O(n^2)$. We introduce two tighter lower bounds and a heuristic for SSB which runs in O(nM) where M is the number of machines of the single stage subproblem.

We design a Dual Heuristic lower Bound (DHB) induced by a time indexed Linear Programming (LP) formulation where the objective is to build a non-preemptive optimal schedule. The second lower bound is obtained by constraining the preemptive assignment of jobs. It is also based on a time indexed LP formulation that produces a partially preemptive schedule (PPB). As a matter of fact, this time-indexed formulation constrains the preemptive assignment of jobs, making PPB tighter than PB, but its running time complexity is higher. The computation of PB and PPB are obtained by bisection where each step requires the solution of a transportation problem. "Preflow Push" algorithms are used to solve the corresponding network flow problem. An improved "Preflow Push" algorithm which runs faster than the original version is proposed. All lower bounds are tested on several single stage configurations using several problem data so as to study their behavior and to select the best-suited ones to be used in branch and bound algorithms.

Chapter V aims at introducing exact methods to solve this problem. Two branch and bound algorithms are studied, an improved version of the sequence enumeration branch and bound algorithm that has been proposed by Brah and Hunsucker (1991) and the generalization of the interval branching method that has been introduced by Carlier (1987) as a branching scheme for the single stage parallel machine problem with release dates and tails. Several bounding strategies are proposed.

Numerical tests on different classes of configuration are carried out so as to evaluate the performance of both methods. Thus, the improved sequence enumeration method is first compared to the original one, and is then compared to the interval method algorithm. The latter is tested with different bounding strategies. Note that these algorithms can always be used as heuristics or improvement methods when truncated after running for a certain amount of time.

Programming all the algorithms developed in this project on the computer results in thousands of lines of codes and hundreds of programs. Programming a large scheduling project on the computer is a complex and time consuming task. In the appendix of this thesis we present an Object Model for Scheduling Algorithm Implementations (OMSAI). This model allows us a better computer implementation and reduces the debugging and implementation times. *The main objective of this model is to present the general framework within which the scheduling algorithms presented in this thesis were implemented*. One can then either use, change or improve this model for future implementations or simply draw one's inspiration from this model for developing further specific algorithms. This model has allowed us a better computer implementation by reducing the debugging and implementation times.

Finally, the five chapters have been written in such a way that they can be read independently. The reader will certainly notice that some introductory explanations are repeated in some chapters. However, the cited references are put at the end of each chapter.