1 Introduction

Scheduling problems can be understood in general as the problems of allocating resources over time to perform a set of tasks being parts of some processes, among which computational and manufacturing ones are most important. Tasks individually compete for resources which can be of a very different nature, e.g. manpower, money, processors (machines), energy, tools. The same is true for task characteristics, e.g. ready times, due dates, relative urgency weights, functions describing task processing in relation to allotted resources. Moreover, a structure of a set of tasks, reflecting relations among them, can be defined in different ways. In addition, different criteria which measure the quality of the performance of a set of tasks can be taken into account.

It is easy to imagine that scheduling problems understood so generally appear almost everywhere in real-world situations. Of course, there are many aspects concerning approaches for modeling and solving these problems which are of general methodological importance. On the other hand, however, some classes of scheduling problems have their own specificity which should be taken into account. Since it is rather impossible to treat all these classes with the same attention in a framework of one book, some constraints must be put on the subject area considered. In the case of this handbook these constraints are as follows.

First of all we focus on the problems motivated by applications from industry and service operations management as well as from case studies of real - life problems. Among others there is a detailed description of optimization procedures for acrylic-glass production and the production of helicopter parts in a flexible manufacturing system. We will describe the backbone of an efficient decision support system for airport gate scheduling as well as a flexible flow shop scheduling system in order to manufacture screws and car power brakes.

Second, we deal with deterministic scheduling problems (cf. [Bak74, BCSW86, Bru07, BT09, CCLL95, CMM67, Cof76, Eck77, Fre82, Gaw08, GK87, Len77, Leu04, LLR+93, Pin16, Rin76, RV09, TB06, TGS94, TSS94]), i.e. those in which no variable with a non-deterministic (e.g. probabilistic) description appears. Let us stress that this assumption does not necessarily mean that we deal only with static problems in which all characteristics of a set of tasks and a set of resources are known in advance. We consider also dynamic problems in which some parameters such as task ready times are unknown in advance, and we do not assume any a priori knowledge about them; this approach is even more realistic in many practical situations.

Third, we consider problems in which a set of resources always contains processors (machines). This means that we take into account the specificity of these particular resources in modeling and solving corresponding scheduling problems, but it does not mean that all presented methods and approaches are restricted to this specificity only. The main reason for which we differentiate

J. Blazewicz et al., *Handbook on Scheduling*, International Handbooks on Information Systems, https://doi.org/10.1007/978-3-319-99849-7_1

processors (we even do not call them "resources" for the convenience of a reader) is that we like to expose especially two broad (and not exclusive) areas of practical applications of the considered problems, namely computer and manufacturing systems.

After the explanation of the handbook's title, we can pass to the description of some of its deeper specificities. They can be meant as compromise we accepted in multi-objective decision situations we had to deal with before and during the preparation of the text. At the beginning, we found a compromise between algorithmic (rather quantitative) and knowledge-based (rather qualitative) approaches to scheduling. We decided to present in the first chapters the algorithmic approach, and at the end to show how it can be integrated with the approach coming from the area of Artificial Intelligence, in order to create a pretty general and efficient tool for solving a broad class of practical problems. In this way we also hopefully found a compromise between rather more computer and rather more manufacturing oriented audience.

The second compromise was concerned with the way of presenting algorithms: formal or descriptive. Basically we decided to adopt a Pascal-like notation, although we allowed for few exceptions in cases where such a presentation would be not efficient.

Next we agreed that the presented material should realize a reasonable compromise between the needs of readers coming from different areas, starting from those who are beginners in the field of scheduling and related topics, and ending with specialists in the area. Thus we included some preliminaries concerning basic notions from discrete mathematics (problems, algorithms and methods), as well as, besides the most recent, also the most important classical results.

Summing up the above compromises, we think that the handbook can be addressed to a quite broad audience, including practitioners and researchers interested in scheduling, and also to graduate or advanced undergraduate students in computer science/engineering, operations research, industrial engineering, management science, business administration, information systems, and applied mathematics curricula.

Finally, we present briefly the outline of the handbook.

In **Chapter 2** basic definitions and concepts used throughout the book are introduced. One of the main issues studied here is the complexity analysis of combinatorial problems. As a unified framework for this presentation the concept of a combinatorial search problem is used. Such notions as: decision and optimization problems, their encoding schemes, input length, complexity classes of problems, are discussed and illustrated by several examples. Since the majority of scheduling problems are computationally hard, two general approaches dealing with such problems are briefly discussed: enumerative and heuristic. First, general enumerative approaches, i.e. dynamic programming and branch and bound are shortly presented. Second, heuristic algorithms are introduced and the ways of analysis of their accuracy in the worst case and on the average are described. Then, we introduce the ideas of general local search metaheuristics known under names: simulated annealing, tabu search, and ejection chains as

well as genetic algorithms. In contrast with previous approaches (e.g. hill-climbing) to deal with combinatorially explosive search spaces about which little knowledge is known a priori, the above mentioned metaheuristics, in combination with problem specific knowledge, are able to escape a local optimum. Basically, the only thing that matters, is the definition of a neighborhood structure in order to step from a current solution to a next, probably better one. The neighborhood structure has to be defined within the setting of the specific problem and with respect to the accumulated knowledge of the previous search process. Furthermore, frequently some random elements guarantee a satisfactory search diversification over the solution space. During the last years the metaheuristics turned out to be very successful in the scheduling area; specific applications will be described in Chapters 8, 9, 10 and 17.

The chapter is complemented by a presentation of the notions from sets and relations, as well as graphs and networks, which will be used in the later chapters.

In **Chapter 3** definitions, assumptions and motivations for deterministic scheduling problems are introduced. We start with the set of tasks, the set of processors (machines) and the set of resources, and with two basic models in deterministic scheduling, i.e. parallel processors and dedicated processors. Then, schedules and their performance measures (optimality criteria) are described. After this introduction, possible ways of analyzing scheduling problems are described with a special emphasis put to solution strategies of computationally hard problems. Finally, motivations for the use of the deterministic scheduling model as well as an interpretation of results, are discussed. Two major areas of applications, i.e. computer and manufacturing systems are especially taken into account. These considerations are complemented by a description of a classification scheme which enables one to present deterministic scheduling problems in a short and elegant way.

Chapter 4 deals with single-processor scheduling. The results given here are mainly concerned with polynomial time optimization algorithms. Their presentation is divided into several sections taking into account especially the following optimality criteria: schedule length, mean (and mean weighted) flow time, and due date involving criteria, such as maximum lateness, number of tardy tasks, mean (and mean weighted) tardiness and a combination of earliness and lateness. In each case polynomial time optimization algorithms are presented, taking into account problem parameters such as the type of precedence constraints, possibility of task preemption, processing and arrival times of tasks, etc. These basic results are complemented by some more advanced ones which take into account change-over cost, also called lot size scheduling, and more general cost functions. Let us stress that in this chapter we present a more detailed classification of subcases as compared to the following chapters. This follows from the fact that the algorithms for the single-processor model are useful also in more complicated cases, whenever a proper decomposition of the latter is carried out. On the other hand, its relative easiness makes it possible to solve optimally in polynomial time many more subcases than in the case of multiple processors.

Chapter 5 carries on an analysis of scheduling problems where multiple parallel processors are involved. As in Chapter 4, a presentation of the results is divided into several subsections depending mainly on the criterion considered and then on problem parameters. Three main criteria are analyzed: schedule length, mean flow time and maximum lateness. A further division of the presented results takes in particular into account the type of processors considered, i.e. identical, uniform or unrelated processors, and then parameters of a set of tasks. Here, scheduling problems are more complicated than in Chapter 4, so not as many optimization polynomial time algorithms are available as before. Hence, more attention is paid to the presentation of polynomial time heuristic algorithms with guaranteed accuracy, as well as to the description of some enumerative algorithms.

In **Chapter 6** new scheduling problems arising in the context of rapidly developing manufacturing as well as parallel computer systems, are considered. When formulating scheduling problems in such systems, one must take into account the fact that some tasks have to be processed on more than one processor at a time. On the other hand, communication issues must be also taken into account in systems where tasks (program modules) are assigned to different processors and exchange information between each other. In the chapter three models are discussed in a sequel. The first model assumes that each so-called multiprocessor task may require more than one processor at a time and communication times are implicitly included in tasks' processing times. The second model assumes that uniprocessor tasks, each assigned to one processor, communicate explicitly via directed links of the task graph. More precise approaches distinguish between coarse grain and fine grain parallelism and discuss their impact on communication delays. Furthermore, task duplication often leads to shorter schedules; this is in particular the case if the communication times are large compared to the processing times. The last model is a combination of the first two models and involves the so called divisible tasks.

Chapter 7 deals with another type of scheduling problems where the tasks are periodic in the sense that they are processed repeatedly and with given frequencies. Particularly in real-time systems designed for controlling some technical facility we are confronted with problems where sets of periodic tasks are to be processed on a single processor or on a distributed or parallel processor system. The chapter starts with a short introduction to real-time systems and discusses characteristic properties and general functional requirements of such systems. Then strategies for scheduling sets of periodic tasks on a single processor and on a multiprocessor system are presented, and the classical results for the rate monotonic and earliest deadline scheduling strategies are discussed from their properties and performance points of view. Another important issue regards runtime problems that appear if tasks use of non-preemptable (non-withdrawable) resources. Finally, several variants of the periodic task model allowing higher flexibility as compared to the simple periodic task model are presented.

In **Chapter 8** flow shop scheduling problems are described, i.e. scheduling a set of jobs (composed of tasks) in shops with a product machine layout. Thus, the jobs have the same manufacturing order. Recent local search heuristics as well as heuristics relying on the two-machine flow shop scheduling problem - which can easily be solved optimally - are considered. Some special flow shops are introduced, e.g. permutation and no-wait ones. The hybrid or flexible flow shop problem is a generalization of the flow shop in such a way that every job can be processed by one among several machines on each machine stage. In recent years a number of effective exact methods have been developed. A major reason for this progress is the development of new job and machine based lower bounds as well as the rapidly increasing importance of constraint programming. We provide a comprehensive and uniform overview on exact solution methods for flexible flow shops with branching, bounding and propagation of constraints, under two different objective functions: minimizing the makespan of a schedule and the mean flow time. For some simple cases we present heuristics with known worst case performance and then describe a branch and bound algorithm for the general case.

In **Chapter 9** we consider the open shop problem where jobs without any precedence constraints are supposed to be scheduled. Only few exact solution methods are available and we motivate our presentation with a description of optimal results for small open shop scheduling problems. We continue describing a branch-and-bound algorithm for solving this problem which performs better than any other existing algorithm. The key to the efficiency of the algorithm lies in the following approach: instead of analyzing and improving the search strategies for finding solutions, the focus is on constraint propagation based methods for reducing the search space. For the first time, many problem instances are solved to optimality in a short amount of computation time.

In **Chapter 10** job shop scheduling problems are investigated. This is the most general form of manufacturing a set of different jobs on a set of machines where each job is characterized by its specific machine order reflecting the jobs production process. We introduce the commonly used representation of the job shop scheduling problems: the disjunctive graph, and its efficient machine representation: the graph matrix, which can be generalized for any graph. The most successful branch and bound ideas are described and we will see that their branching structure is reflected in the neighborhood definitions of many local search methods. In particular tabu search, ejection chains, genetic algorithms as well as the propagation of constraints - this is closely related to the generation of valid inequalities - turned out to become the currently most powerful solution approaches. Moreover, we introduce priority rule based scheduling and describe a well-known opportunistic approach: the shifting bottleneck procedure.

Chapter 11 deals with scheduling problems where the availabilities of processors to process tasks are limited. In the preceding chapters the basic model assumes that all machines are continuously available for processing throughout the planning horizon. This assumption might be justified in some cases but it

does not apply if certain maintenance requirements, breakdowns or other constraints that cause the machines not to be available for processing have to be considered. In this chapter we generalize the basic model in this direction and discuss results related to one machine, parallel machines, and shop scheduling problems where machines are not continuously available for processing.

In **Chapter 12** we are focused on time-dependent scheduling problems, where job processing times are functions of the job starting times. Problems of this kind compose the first group of scheduling problems with variable job processing times discussed in the handbook. Another group of such problems, with resource-dependent job processing times, is discussed in Chapter 13. We begin this chapter with a short description of basics of time-dependent scheduling and the main forms of time-dependent job processing times. Next, we review the most important results of time-dependent scheduling, diving them into groups with respect to machine environment and job processing times form. Whenever it is possible, we illustrate our presentation by examples. As in the whole handbook, we discuss only deterministic algorithms and solution methods. Apart time complexity and algorithms for time-dependent scheduling problems, we also discuss two-agent time-dependent scheduling, bi-criteria time-dependent scheduling, properties of mutually related pairs of time-dependent scheduling problems and applications of matrix methods in time-dependent scheduling.

Chapter 13 deals with resource constrained scheduling. In the first two sections it is assumed that tasks require for their processing processors and certain fixed amounts of discrete resources. The first section presents the classical model where schedule length is to be minimized. In this context several polynomial time optimization algorithms are described. In the next section this model is generalized to cover also the case of multiprocessor tasks. Two algorithms are presented that minimize schedule length for preemptable tasks under different assumptions concerning resource requirements. The last section deals with problems in which additional resources are continuous, i.e. continuously-divisible. We study three classes of scheduling problems of that type. The first one contains problems with parallel processors and tasks described by continuous functions relating their processing speeds to the resource amount allotted at a time. The next two classes are concerned with single processor problems where task processing times or ready times, respectively, are continuous functions of the allotted resource amount.

Chapter 14 is devoted to a certain scheduling model - the imprecise computation model, inspired by practical applications arising in the hard real time environment, introduced in Chapter 7. It allows trading off accuracy of computations in favor of meeting deadlines imposed on tasks. In the imprecise computation model tasks are composed of two subtasks: mandatory and optional ones. The mandatory subtask has to be completed before the deadline in order to obtain a feasible solution, but the optional subtask can be late or even left unfinished. The mandatory subtask corresponds to producing a usable but approximate result with the error modeled by the late part of the optional subtask. Completing both subtasks on time corresponds to precise result with no error. The chapter starts

with exemplary applications which motivated such scheduling problems. Then, we briefly present the general imprecise computation model and we pass, in the next section, to its special case - the late work model, which assumes that tasks consist of optional subtasks only. We present results related to single, parallel and dedicated processors. Particularly, we show a few examples of classical solution techniques introduced in Chapter 2, namely: the dynamic programming and the polynomial time approximation scheme being a family of approximation algorithms. Discussing late work problems, we refer to various topics of the scheduling theory involving for example controllable processing times, time-dependent processing times (i.e. scheduling with learning effect) or multi-agent scheduling. In the last section, some related models inspired by the imprecise computations, as well as mirror scheduling problems are mentioned.

The handbook focuses in general on deterministic scheduling models, which assume that the complete knowledge of a problem instance is provided to a decision maker, i.e. to an algorithm, in advance. **Chapter 15** introduces the fundamentals of online scheduling. The online scheduling can be considered as scheduling with incomplete information, because the decisions on executing tasks are made without knowing a complete instance of the problem, i.e. the input is being revealed to a decision maker piece-by-piece. Online scheduling models can be considered as a bridge between deterministic scheduling and stochastic scheduling, which copes with a problem input given as random variables with certain probability distributions. Online algorithms compute partial schedules whenever a new piece of information requests taking an action from them, i.e. the decisions are made without full knowledge of the future. We present basic online scheduling paradigms, such as online-over-list and online-over-time, which assume that tasks appear in the system in a given sequence or at a given time moment. We show the differences between clairvoyant and non-clairvoyant scheduling models, which reveal various amount of information on incoming tasks. Moreover, we adjust the idea of precedence constraints and preemptions, introduced in Chapter 3 for offline mode, to online mode. We present the general idea of deterministic and randomized online algorithms and describe the techniques used for evaluating their efficiency, based on the competitive analysis and on the lower bound analysis involving the adversary method. These basic ideas are illustrated with a few examples. Moreover, we introduce some enhanced online scheduling models, such as semi-online scheduling, online scheduling with advice or online scheduling with resource augmentation.

Constraint propagation is the central topic of **Chapter 16**. It is an elementary method for reducing the search space of combinatorial search and optimization problems which has become more and more important in the last decades. The basic idea of constraint propagation is to detect and remove inconsistent variable assignments that cannot participate in any feasible solution through the repeated analysis and evaluation of the variables, domains and constraints describing a specific problem instance. We describe efficient constraint propagation methods also known as consistency tests for the disjunctive scheduling problem (DSP) applications of which will be introduced in machine scheduling chap-

ters 8 to 10. We will further present and analyze both new and classical consistency tests involving a higher number of variables. They still can be implemented efficiently in a polynomial time. Further, the concepts of energetic reasoning and shaving are analyzed and discussed.

The other contribution is a classification of the consistency tests derived according to the domain reduction achieved. The particular strength of using consistency tests is based on their repeated application, so that the knowledge derived is propagated, i.e. reused for acquiring additional knowledge. The deduction of this knowledge can be described as the computation of a fixed point. Since this fixed point depends upon the order of the application of the tests, we first derive a necessary condition for its uniqueness. We then develop a concept of dominance which enables the comparison of different consistency tests as well as a method for proving dominance.

Chapter 17 is devoted to problems which perhaps closer reflect some specific features of scheduling in flexible manufacturing systems than other chapters do. Dynamic job shops are considered, i.e. such in which some events, particularly job arrivals, occur at unknown times. A heuristic for a static problem with mean tardiness as a criterion is described. It solves the problem at each time when necessary, and the solution is implemented on a rolling horizon basis. The next section deals with simultaneous assignment of machines and vehicles to jobs. This model is motivated by the production of helicopter parts in some factory. First we solve in polynomial time the problem of finding a feasible vehicle schedule for a given assignment of tasks to machines, and then present a dynamic programming algorithm for the general case. In the last section we are modeling manufacturing of acrylic-glass as a batch scheduling problem on parallel processing units under resource constraints. This section introduces the real world problem and reveals potential applications of some of the material in the previous chapters. In particular, a local search heuristic is described for constructing production sequences.

Chapter 18 serves two purposes. On one hand, we want to introduce a quite general solution approach for scheduling problems as they appear not only in manufacturing environments. On the other hand, we also want to survey results from interactive and knowledge-based scheduling which were not covered in this handbook so far. To combine both directions we introduce some broader aspects like computer integrated manufacturing and object-oriented modeling. The common goal of this chapter is to combine results from different areas to treat scheduling problems in order to answer quite practical questions. To achieve this we first concentrate on the relationship between the ideas of computer integrated manufacturing and the requirements concerning solutions of the scheduling problems. We present an object-oriented reference model which is used for the implementation of the solution approach. Based on this we discuss the outline of an intelligent production scheduling system using open loop interactive and closed loop knowledge-based problem solving. For reactive scheduling we suggest to use concepts from soft computing. Finally, we make some proposals concerning the integration of solution approaches discussed in the preceding chap-

ters with the ideas developed in this chapter. We use an example to clarify the approach of integrated problem solving and discuss the impact for computer integrated manufacturing.

Chapter 19 presents some examples of applying the scheduling theory for solving problems arising in logistics. Since logistics is strictly related with transportation, for the sake of completeness, we first present a short survey of various variants of the famous vehicle routing problem, which concerns designing routes for a fleet of vehicles to supply a set of customers. Since this problem is widely studied in the literature, we provide numerous references, which can direct the readers deeply interested in this field. Then we describe three problems arising in various modes of transportation, in overland, air and maritime transportation: the concrete delivery problem, the flight gate scheduling problem, and the berth and quay crane allocation problem, respectively. Based on these examples we show various aspects of incorporating scheduling theory in logistics. The concrete delivery problem is a vehicle routing problem in which schedules for concrete mixer vehicles are constructed to deliver concrete from depots to customers, taking into account the specificity of the perishable material being transported. On this example we show the process of modelling complex real world problems, providing the graph model and - based on it - the mixed integer programming model. Then we present a selected approach for the flight gate scheduling problem, concerning assigning aircrafts serving flights to airport gates. The described method is an example of transforming a scheduling problem to a related combinatorial problem (clique partitioning in this case), in order to utilize known algorithms to solve new problems. Finally, on the example of the berth and quay crane allocation problem, in which berths and cranes have to be assigned to ships for loading/unloading containers transported by them, we show a direct application of the scheduling models to solve this logistic case. For all three mentioned problems we provide a rich set of references for readers deeper interested in these and related subjects.

References

Bak74 K. Baker, *Introduction to Sequencing and Scheduling*, J. Wiley, New York, 1974.

BCSW86 J. Błażewicz, W. Cellary, R. Słowiński, J. Węglarz, *Scheduling under Resource Constraints: Deterministic Models*, J. C. Baltzer, Basel, 1986.

Bru07 P. Brucker, *Scheduling Algorithms*, Springer, 5th ed., Berlin, 2007.

BT09 K. R. Baker, D. Trietsch, *Principles of Sequencing and Scheduling*, J. Wiley, New Jersey, 2009.

CCLL95 P. Chretienne, E. G. Coffman, J. K. Lenstra, Z. Liu (eds.), *Scheduling Theory and its Applications*, J. Wiley, New York, 1995.

CMM67 R. W. Conway, W. L. Maxwell, L. W. Miller, *Theory of Scheduling*, Addison-Wesley, Reading, Mass., 1967.

Cof76 E. G. Coffman, Jr. (ed.), *Scheduling in Computer and Job Shop Systems*, J. Wiley, New York, 1976.

Eck77 K. Ecker, *Organisation von parallelen Prozessen*, BI-Wissenschaftsverlag, Mannheim, 1977.

Fre82 S. French, *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop*, Horwood, Chichester, 1982.

Gaw08 S. Gawiejnowicz, *Time-Dependent Scheduling*, Springer, Berlin-Heidelberg, 2008.

GK87 S. K. Gupta, J. Kyparisis, Single machine scheduling research, *Omega-Int. J. Manage. Sci.* 15, 1987, 207-227.

Len77 J. K. Lenstra, *Sequencing by Enumerative Methods*, Mathematical Centre Tract 69, Amsterdam, 1977.

Leu04 J. Y.-T. Leung (ed.), *Handbook of Scheduling: Algorithms, Models and Performance Analysis*, Chapman & Hall/CRC, Boca Raton, 2004.

LLR+93 E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, D. B. Shmoys, Sequencing and scheduling: Algorithms and complexity, in: S. C. Graves, A. H. G. Rinnooy Kan, P. H. Zipkin (eds.), *Handbook in Operations Research and Management Science, Vol. 4: Logistics of Production and Inventory*, Elsevier, Amsterdam, 1993.

Pin16 M. Pinedo, *Scheduling: Theory, Algorithms, and Systems*, 5th ed., Springer, New York, 2016.

Rin76 A. H. G. Rinnooy Kan, *Machine Scheduling Problems; Classification, Complexity and Computations,* Martinus Nijhoff, The Hague, 1976.

RV09 Y. Robert, F. Vivien (eds.), *Introduction to Scheduling*, CRC Press, Boca Raton, 2009.

TB06 V. T'kindt, J.-C. Billaut, *Multicriteria Scheduling: Theory, Models and Algorithms*, 2nd ed., Springer, Berlin, 2006.

TGS94 V. S. Tanaev, V. S. Gordon, Y. M. Shafransky, *Scheduling Theory. Single-Stage Systems*, Kluwer, Dordrecht, 1994.

TSS94 V. S. Tanaev, Y. N. Sotskov, V. A. Strusevich, *Scheduling Theory. Multi-Stage Systems*, Kluwer, Dordrecht, 1994.