The material below was scanned in from the original, the spelling Americanized and some typos and inaccuracies corrected. Some explanatory notes were added at the end. It is intended as a supplement to the original contained in the course reader for ISE514. Comments beyond the original are indicated by (GB)

SEQUENCING AND SCHEDULING:

An Introduction to the

Mathematics of the Job-Shop

SIMON FRENCH, B.A., M.A., D.Phil.

Department of Decision Theory University of Manchester

ELLIS HORWOOD LIMITED Publishers· Chichester

Halsted Press: a division of JOHN WILEY & SONS

New York, Chichester· Brisbane· Toronto

First published in 1982

and Reprinted in 1986 and 1987 by

ELLIS HORWOOD LIMITED

Market Cross House, Cooper Street, Chichester, West Sussex, P019 1EB, England

The publisher's colophon is reproduced from James Gillison's drawing of the ancient Market Cross. Chichester.

Distributors:

Australia and New Zealand:

Jacaranda-Wiley Ltd., Jacaranda Press, JOHN WILEY & SONS INC.

GPO Box 859, Brisbane, Queensland 4001, Australia

Canada:

JOHN WILEY & SONS CANADA LIMITED 22 Worcester Road, Rexdale, Ontario, Canada

Europe and Africa:

JOHN WILEY & SONS LIMITED

Baffins Lane, Chichester, West Sussex, England

North and South America and the rest of the world:

Halsted Press: a division of JOHN WILEY & SONS

605 Third Avenue, New York, NY 10158, USA

French. Simon . ɟ . Sequencing and scheduling. - (Ellis Horwood series in mathematics and its applications)

1. Scheduling (Management)

I. Title

658.5'1 TS157.A2

Libruy of Congress Card No. 81-3414 AACR2

ISBN 9-85312-299-7 (Ellis Horwood Ltd., Publishers - Library Edn.)
ISBN 0-8S312-364~ (Ellis Horwood Ltd., PubUshers - Student Edn.)
ISBN 0-470-27229-5 (Halsted Press)

Table of Contents

Preface

I first realized the need for a new textbook on the mathematics of scheduling when I was asked to develop and teach a course on this subject to undergraduates at Manchester University. The books already available seemed to fall into two main categories: those that, although excellent in their time, had become rather out-dated and those that were aimed primarily at research workers. Hence my intention in producing this work has been to provide an up to date introduction to this rapidly expanding and exciting field.

I have aimed this book mainly at an audience of final year undergraduate or master students in any numerate discipline, as although it is mathematically oriented it does not demand a knowledge of any specific areas of advanced mathematics. Indeed all that is required of the student is an ability to think clearly and logically. Because I am not writing for the professional mathematician I have made considerable efforts to keep the notation as simple as possible, even at the expense of some loss of brevity. Many examples have been included to ensure understanding of the theory involved, and hints and solutions provided to the problems at the end of each chapter as an aid to those studying alone.

I am optimistic that this book will find an additional audience amongst professional operational research workers, for they too have need of a basic introduction to the subject. With this group particularly in mind I have extensively surveyed current developments in the theory of scheduling, and included in the bibliography sufficient references to guide them further into the literature. However I would stress that the emphasis here is more theoretical than practical, and that I have no intention of providing a "recipe book" for those faced with a particular

scheduling problem.

The comments of many colleagues, friends, and students have been extremely helpful to me during the compilation of this book; in particular those of Professor B. W. Conolly and Dr. J. R. Walters who have read my various drafts and made many constructive criticisms and suggestions. The patience and skill of Kate Baker, who has undertaken all of the secretarial work, has been much appreciated. And finally it has indeed been a pleasure to work with my publisher, Ellis Horwood, his family and staff: their efficiency and tactful guidance have greatly smoothed the preparation of this book.

1st March, 1981

SIMON FRENCH University of Manchester

Chapter 1

Scheduling Problems

1.1 INTRODUCTORY EXAMPLE

Algy, Bertie, Charles and Digby share a flat. Each Saturday they have four newspapers delivered: The *Financial Times,* the *Guardian,* the *Daily Express,* and the *Sun.* Being a small-minded creature of habit, each member of the flat insists on reading all the papers in his own particular order. Algy likes to begin with *the Financial times* for 1 hour. Then he turns to the *Guardian* taking 30 minutes, glances at the *Daily Express* for 2 minutes and finishes with 5 minutes spent on the *Sun.* Bertie prefers to begin with the *Guardian* taking 1 hour 15 minutes; then he reads the *Daily Express* for 3 minutes, the *Financial Times* for 25 minutes, and the *Sun* for 10 minutes. Charles begins by reading the *Daily Express* for 5 minutes and follows this with the *Guardian* for 15 minutes, the *Financial Times* for 10 minutes, and the *Sun* for 30 minutes. Finally, Digby starts with the *Sun* taking 1 hour 30 minutes, before spending 1 minute each on the *Financial Times,* the *Guardian,* and the *Daily Express* in that order. Each is so insistent upon his particular reading order that he will wait for his next paper to be free rather than select another. Moreover, no one will release a paper until he has finished it. Given that Algy gets up at 8.30 a.m., Bertie and Charles at 8.45 a.m., and Digby at 9.30 a.m. and that they can manage to wash, shower, shave, and eat breakfast, while reading a newspaper, and given that each insists on reading all the newspapers before going out, what is the earliest time that the four of them can leave together for a day in the country?

The problem faced by Algy and his friends, namely in what order should they rotate the papers between themselves so that all the reading is finished as soon as possible, is typical of the scheduling problems that we shall be considering. Before describing the general structure of these problems and giving some examples that are, perhaps, more relevant to our modern society, it is worth examining this example further.

Let us write the data in a more compact form, as in Table 1.1. How might we tackle this problem? Perhaps it will be easiest to begin by explaining what is meant by a reading schedule. Quite simply it is a prescription of the order in which the papers rotate between readers. For instance, one possible schedule is shown in Table 1.2, where A, B, C, D denote Algy, Bertie, Charles, and Digby respectively. Thus Algy has the *Financial Times* first, before it passes to Digby, then to Charles, and finally to Bertie. Similarly the *Guardian* passes between Bertie, Charles, Algy, and Digby in that order. And so on.

Table 1.1—The data for the newspaper reading problem

Reader	Gets Up At	Reading Order And Times In Mins.			
Algy	8.30	F.T. (60)	G. (30)	D.E. (2)	S. (5)
Bertie	8.45	G. (75)	D.E. (3)	F.T. (25)	S. (10)
Charles	8.45	D.E. (5)	G. (15)	F.T. (10)	S. (30)
Digby	9.30	S. (90)	F.T. (1)	G. (1)	D.E. (1)

Table 1.2-A possible reading schedule

	Read by			
Paper	1st	2nd	3rd	4th
F.T.	A	D	C	B
G.	B	C	A	D
D.E.	C	B	A	D
S.	D	A	C	B

We may work out how long this reading schedule will take by plotting a simple diagram called a Gantt Diagram (see Fig. 1.1). In this we plot four time-axes, one for each newspaper. Blocks are placed above the axes to indicate when and by whom particular papers are read. For instance, the block in the top left hand comer indicates that Algy reads the *Financial Times* from 8.30 to 9.30. To draw this diagram we have rotated the papers in the order given by Table 1.2 with the restriction that each reader follows his desired reading order. This restriction means that for some of the time papers are left unread, even when there are people who are free and have not read them yet; they must remain unread until someone is ready to read them *next.* For instance, Bertie could have the *Financial Times* at 10.00 a.m., but he wants the *Daily Express* first and so leaves the *Financial Times.* Similarly the schedule is also responsible for idle time of the readers. Between 10.15 and 11.01 Charles waits for the *Financial Times,* which for all but the last minute is not being read, but Charles cannot have the paper until after Digby because of the schedule.

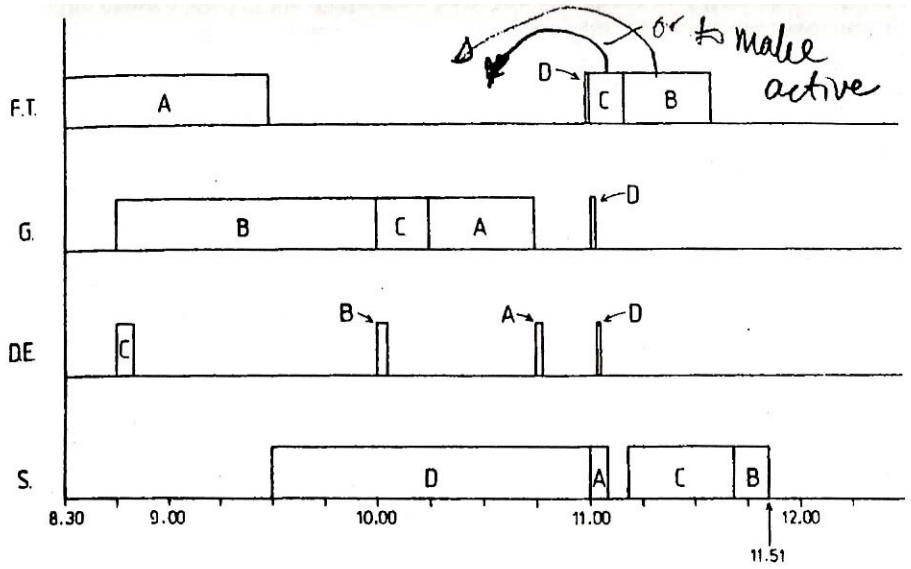

Fig.1.1—Gantt diagram for the schedule in Table 1.2.

Clearly, from the Gantt diagram, the earliest that all four can go out together is 11.51 a.m. *if* they use this schedule. So the next question facing us is can we find them a better schedule, i.e. one that allows them to go out earlier. I am leaving that question for you to consider in the first set of problems. However, before attempting those, we should consider feasible and infeasible schedules.

In Table 1.2 I simply gave you a schedule and we saw in the Gantt diagram that this schedule would work; it is possible for Algy and his flat mates to pass the papers amongst themselves in this order. But suppose I had given you the schedule shown in Table 1.3. A little thought shows that this schedule will not work. Algy is offered the *Sun* first, but he does not want it until he has read the other three papers. He cannot have any of these until Digby has finished with them and Digby will not start them until he has read the *Sun,* which he cannot have until A1gy has read it…

Table l.3-An Infeasible schedule

	Read by			
Paper	1st	2nd	3rd	4th
F.T.	D	B	A	C
G.	D	C	B	A
D.E.	D	B	C	A
S.	A	D	C	B

In scheduling theory the reading orders given in Table 1.1 are called the technological constraints. Any schedule that is compatible with these is called feasible. Thus Table 1.2 gives a feasible schedule. Infeasible schedules, such as that in Table 1.3, are incompatible with the technological constraints. Obviously, to be acceptable a solution to a scheduling problem must be feasible.

1.2 PROBLEMS

Please try these problems now before reading any further. It is true that I have given you little or no guidance on how they are to be solved. I have done so for good reason. Scheduling is a subject in which the problems look easy, if not trivial. They are, on the contrary, among the hardest in mathematics. You will not appreciate this without trying some for yourself. Solving them is relatively unimportant; I shall solve them for you shortly anyway. What is important is that you should discover their difficulty.

1. Is the following schedule feasible for Algy and his friends?

	Read By			
Paper	1st	2nd	3rd	4th
F.T.	C	D	B	A
G.	B	C	A	D
D.E.	B	C	A	D
S.	B	A	D	C

2. How many different schedules, feasible or infeasible, are there?

3. What is the earliest time that Algy and his friends can leave for the country?

4. Digby decides that the delights of a day in the country are not for him. Instead he will spend the morning in bed. Only when the others have left will he get up and read the papers. What is the earliest time that Algy, Bertie, and Charles may leave?

5. Whether or not you have solved Problems 3 and 4, consider how you would recognize the earliest possible departure time. Need you compare it explicitly with those of all the other feasible schedules, or can you tell without this process of complete enumeration of all the possibilities?

1.3 THE GENERAL JOB·SHOP SCHEDULING PROBLEM

If the theory of scheduling were simply concerned with the efficient reading of newspapers, then, of course, no one would study it. I

began with that example so that you might meet and attempt to solve a scheduling problem unhindered by the definitions and notations that are usually required. The time has come to introduce these definitions and that notation. The terminology of scheduling theory arose in the processing and manufacturing industries. Thus we shall be talking about jobs and machines, even though in some cases the objects referred to bear little relation to either jobs or machines. For instance, in the example of the last section we shall see that Algy, Bertie, Charles and Digby are 'jobs', whilst the newspapers are 'machines'. However, that is anticipating. We begin by defining the general job-shop problem. Shortly we shall show that its structure fits many scheduling problems arising in business, computing, government, and the social services as well as those in industry.

We shall suppose that we have *n* jobs $\{J., J_2, \ldots. J_n\}$ to be processed through m machines $\{M_1, M_2 \ldots M_m\}$. Some authors, particularly those writing on computer scheduling, refer to machines as processors. We shall suppose that each job must pass through each machine once and once only. The processing of a job on a machine is called an operation. We shall denote the operation on the i^{th} job by the j^{th} machine by o_{ij}. Technological constraints demand that each job should be processed through the machines in a particular order. For general job-shop problems there are no restrictions upon the form of the technological constraints. Each job has its own processing order and this may bear no relation to the processing order of any other job. However, an important special case arises when all the jobs share the same processing order. In such circumstances we say that we have a f1ow shop problem (because the jobs *flow* between the machines in the same order). This distinction between job-shops and flow-shops should be made clear by the examples below.

Each operation o_{ij} takes a certain length of time, the processing time, to perform. We denote this by p_{ij}. By convention we include in p_{ij} any time required to adjust, or set up, the machine to process this job. We also include any time required to transport the job to the machine. We shall assume that the p_{ij} are fixed and known in advance. This brings us to an important restriction, which we shall make throughout this book. We shall assume that every numeric quantity is deterministic and known to the scheduler.

We shall also assume that the machines are always available, but we shall not necessarily assume the same for jobs. Some jobs may not become available until after the scheduling period has started. We shall denote by r_j the ready time or release date of the i^{th} job, i.e. the time at which J_j becomes available for processing.

The general problem is to find a sequence, in which the jobs pass between the machines, which is

(a) compatible with the technological constraints, i.e. a feasible schedule, and

(b) optimal with respect to some criterion of performance.

We shall discuss possible criteria of performance shortly; first we shall consider some particular examples.

Industrial examples

Any manufacturing firm not engaged in mass production of a single item will have scheduling problems at least similar to that of the job-shop. Each product will have its own route through the various work areas and machines of the factory. In the clothing industry different styles have different requirements in cutting, sewing, pressing and packing. In steel mills each size of rod or girder passes through the set of rollers in its own particular order and with its own particular temperature and pressure settings. In the printing industry the time spent in typesetting a book will depend on its length, number of illustrations, and so on; the time spent in the actual printing will depend on both its length and the number printed; the time spent on binding will depend on the number printed; and lastly the time spent in packaging will depend both on the number printed and the book's size. Thus a printer who must schedule the production of various books through his typesetting, printing, binding and packaging departments faces a four-machine flow-shop problem; for each department is a machine and an the jobs, i.e. the books, flow from typesetting to printing to binding to packaging.

The objectives in scheduling will vary from firm to firm and often from day to day. Perhaps the aim would be to maintain an equal level of activity in all departments so that expensive skills and machines were seldom idle.

Perhaps it would be to achieve certain contractual target dates or simply to finish all the work as soon as possible. These and some other objectives will be discussed in detail in section 1.5 and in all of Chapter 2.

Algy and friends

This is a four job, four machine problem. The jobs-Algy, Bertie, Charles, and Digby-must be scheduled through the four machines-the *Financial Times,* the *Guardian,* the *Daily Express,* and the *Sun-in* order to minimize the completion time at which the processing, here reading, is finished There are ready times, namely, the times at which Algy, etc. get up. Note that in this example the technological constraints do not demand the f1at mates read the papers in the same order. Because of this we have a job shop problem. Had the constraints demanded that each read the papers in the same order, e.g. *Sun, Guardian, Daily Express* and, finally, the *Financial Times,* we would have had a flow-shop problem.

Aircraft queuing up to land

This is an *n* job, one machine problem, if we assume, as we do, that the number of aircraft arriving in a day is known. The aircraft are the jobs and the runway is the machine. Each aircraft has a ready time, namely the earliest time at which it can get to the airport's air-space and be ready to land. The objective here might be to minimize the average waiting time of an aircraft before it can land. Perhaps this average should be weighted by the number of passengers on each plane. Obviously in real life this problem models only part of the air traffic controllers' difficulties. Planes must take off too. Also it ignores the uncertainty inherent in the situation. Aircraft suffer unpredictable delays and, moreover, the time taken to land, i.e. the processing time of each aircraft, will depend on the weather.

Treatment of patients in a hospital

Again we ignore all randomness. Suppose we have *n* patients who must be treated by a consultant surgeon. Then each must be

M_1-seen in the out-patients' department,
M_2-received in the surgical ward, prepared for the operation, etc.,
M_3-be operated on.
M_4-recover, we hope, in the surgical ward.

Thus each patient (Job) must be processed through each of four 'machines' , M_1 M_2 M_3 and M_4 above. It is perhaps confusing that only one of the operations of processing a patient through a 'machine' is called a surgical operation, but in scheduling theory all are operations. Since each patient must clearly 'flow' through the 'machines' in the order M_1 M_2 M_3 M_4 this is a flow-shop problem. The objective here might be stated as: treat all patients in the shortest possible time, whilst giving priority to the most ill.

Other scheduling problems

Given these examples you should have no difficulty in fitting other practical problems into the general job-shop structure. For instance, the following all fall into this pattern:

(a) the scheduling of different programs on a computer;
(b) the processing of different batches of crude oil at a refinery;
(c) the repair of cars in a garage;
(d) the manufacture of paints of different colors.

1.4 ASSUMPTIONS

For the major part of this book we shall be making a number of assumptions about the structure of our scheduling problems. Some were mentioned explicitly above; others were implicit. Here we list all the assumptions both for further emphasis and for easy reference. Why we choose to make these assumptions, which are often quite restrictive, is discussed in section 1.7.

1. Each job is an entity. Although the job is composed of distinct operations, no two operations of the same job may be processed simultaneously.

Thus we exclude from our discussion certain practical problems, e.g. those in which components are manufactured simultaneously prior to assembly into the finished product.

2. No pre-emption. Each operation, once started, must be completed before another operation may be started on that machine.

3; Each job has m *distinct operations, one on each machine.* We do not allow for the possibility that a job might require processing twice on the same machine. Equally, we insist that each job is processed on every machine; it may not skip one or more machines. Note that

this latter constraint is *not* illusory. Although we could say that a job which skips a machine is processed upon that machine for zero time, we would still have a problem: where in the job's processing sequence should this null operation be placed. Because we do not allow pre-emption, the job could be delayed waiting for a machine which is not in fact needed.

4. No cancellation. Each job must be processed to completion.

5. The processing times are independent of the schedule. In particular we are assuming two things here. Firstly, each set-up time is sequence independent, i.e. the time taken to adjust a machine for a job is independent of the job last processed. Secondly, the times to move jobs between machines are negligible

6. In-process inventory is allowed, i.e. jobs may wait for their next machine to be free. This is not a trivial assumption. In some problems processing of jobs must be continuous from operation to operation. In steel mills one literally has to strike while the iron is hot.

7. There is only one of each type of machine. We do not allow that there might be a choice of machines in the processing of a job. This assumption eliminates, amongst others, the case where certain machines have been duplicated to avoid bottlenecks.

8. Machines may be idle.

9. No machine may process more than one operation at a time.(To a mathematician assumptions 7 and 9 are identical; for mathematically two of the same type of machine is equivalent to a single machine capable of processing two jobs simultaneously. However, we make both assumptions explicitly for those with a less mathematical view of the world.)

10. Machines never breakdown and are available throughout the scheduling period.

11. The technological constraints are known in advance and are immutable.

12. There is no randomness. In particular

(a) the number of jobs is known and fixed;
(b) the number of machines is known and fixed;
(c) the processing times are known and fixed;
(d) the ready times are known and fixed;
(e) all other quantities needed to define a particular problem are known and fixed.

Occasionally we shall relax one or two of these assumptions in specific examples. When we do, we shall state so explicitly.

1.5 PERFORMANCE MEASURES

It is not easy to state our objectives in scheduling. They are numerous, complex, and often conflicting. Mellor (1966) lists 27 distinct scheduling goals, albeit in a slightly broader context. In addition, the mathematics of our problem can be extremely difficult with even the simplest of objectives. So we shall not try to be exhaustive. Instead we shall indicate in general terms a few of the criteria by which we might judge our success. These criteria will be sufficient to motivate the mathematical definitions of the performance measures that we shall use. Whether these abstractions are so simple minded that our theory ceases to have practical relevance is discussed in section 1. 7.

Obviously we should prefer to keep promised delivery dates. Otherwise good will would surely be lost and there might be financial penalties as well. Indeed one may argue cogently that a delivery date that is not

enforced by some penalty is not truly a delivery date. We should also try to minimize the overall length of the scheduling period; because once all the jobs have been completed the machines may be released for other tasks. We should try to minimize the time for which the machines are idle; idle machines mean idle capital investment. We should try to minimize inventory costs and by these we do not mean just the cost of storing the finished product. There are also the costs of storing the raw materials and any partially

processed jobs that must wait between machines. We might try to ensure a uniform rate of activity throughout the scheduling period so that demands for labor and power are stable. Conversely it might be desirable to concentrate the activity into periods when either labor or power is particularly cheap.

Before we can define performance measures in precise mathematical terms, we need some more definitions and notation. Remember that r_i and p_{ij} are respectively the ready time and processing times of job J_i.

d_i is the due date, i.e. the promised delivery date of J_i. It is the time by which ideally we would like to have completed J_i.

a_i s the allowance for J_i. It is the period allowed for processing between the ready time and the due date: $a_i = d_i - r_i$

W_i is the waiting time of J_i preceding its *kth* operation. By kth operation we do not mean the one performed on M_k (although it may be), but the one that comes *k*th in order of processing. Thus, if the technological constraints demand that J_i is processed through the machines in the order $M_{j(1)}, M_{j(2)}, M_{j(3)}, \ldots M_{j(m)}$, the *kth* operation is $o_{ij(k)}$ the one performed on $M_{j(k)}$. So W_{ik} is the time that elapses between the completion of J_i on $M_{j(k-1)}$ (or r_i if $k = 1$) and the start of processing on M_k.

W_i is the total waiting time of J_i. Clearly $W_i = \sum_{k=1}^{m} W_{ik}$

C_i is the completion time of J_i, i.e. the time at which processing of J_i finishes. We have the equality: $C_i = r_i + \sum_{k=1}^{m} \left(W_{ik} + p_{ij(k)} \right)$

F_i is the flow time of J_i This is defined to be the time that J_I spends in the workshop. Thus $F_i = C_i - r_i$

L_i is the lateness of J_i This is simply the difference between its completion time and its due date: $L_i = C_i - d_i$. Note that when a job is early, i.e. when it completes before its due date, L, is negative. It is often more use to have a variable that, unlike lateness, only takes non-zero values when a job is tardy, i.e. when it completes after its due date. Hence we also define the tardiness and, to be comprehensive, the earliness of a job.

T_i is the tardiness of J_i: $T_i = \max\{L_i, 0\}$.

E_i is the earliness of J_i: $E_i = \max\{-L_i, 0\}$

.

These quantities for a typical job are illustrated on the Gantt diagram shown in Fig. 1.2. There is a possibility of confusion in some of this terminology. English allows the noun 'time' two distinct meanings. It may be used to refer to an instant or to an interval. Thus ready time and completion time both refer to instants in time; whereas processing time, waiting time, and flow time refer to intervals. In remarking upon this ambiguity we hope to avoid possible confusion. In any case, the context usually ensures that the meaning is clear,

We shall often need to discuss the maximum or the mean of these quantities and it will help to have a compact notation for doing this. Let X_i be any quantity relating to J_i. Then we let $\overline{X} = 1/n \sum_{i=1}^{n} X_i$, the average over all the jobs, and $X_{max.} = \max\{X_J, X_2, \bullet\bullet\bullet, X_n\}$, the maximum over all the jobs. For instance, $\overline{F}$ is the mean flow time and C_{max} is the maximum completion time.

Next we define the idle time on machine Mj by $I_j = C_{max} - \sum_{i=1}^{n} p_{ij}$ In order to see that this definition makes sense, note that C_{max} is the time

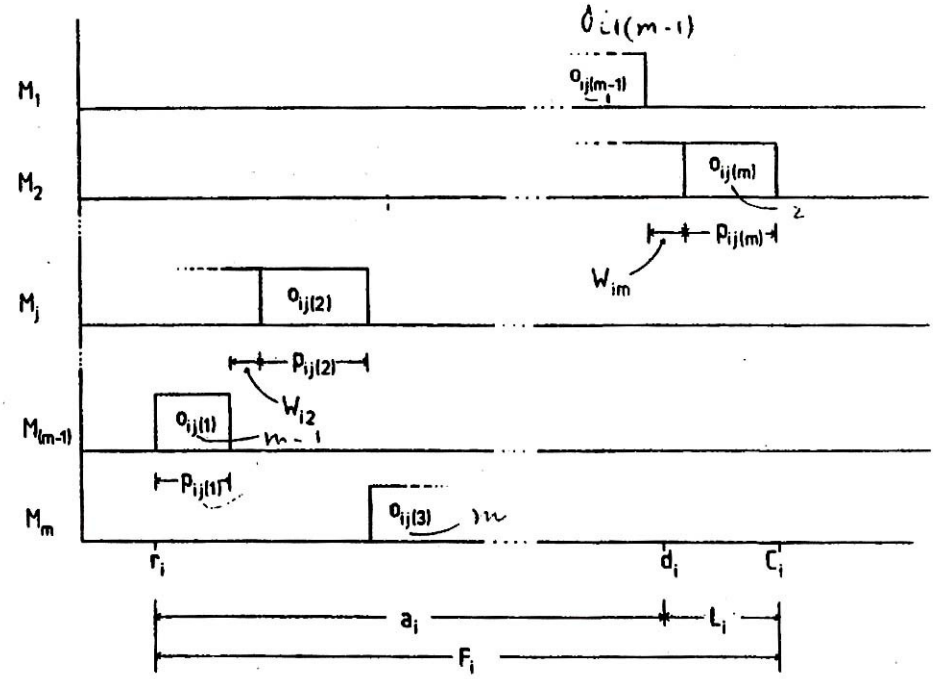

Fig. 1.2 Gantt diagram of a typical job J_i. The processing order given by the technological constraints is $(M_{(m-1)}, M_j, M_m, \ldots, M_1, M_2)$. The waiting times W_{i1}, W_{i3} are zero; W_{i2} and W_{im} are non-zero as shown. For this job $T_i = L_i$ and $E_i = 0$, since the job is completed after its due date.

when all processing ceases and $\sum_{i=1}^{n} p_{ij}$ is the total processing time on machine M_j. Their difference gives the period for which M_j is idle.

Finally we introduce some variables that count the number of jobs in various states at any given time. We let

Nw(t) the number of jobs waiting between machines or not ready for processing at time *t;*
$N_p(t)$ be the number of jobs actually being processed at time *t;*
$N_c(t)$ be the number of jobs completed by time *t;* and
Nu(t) be the number of jobs still to be completed by time *t.*

Clearly we have the following identities:

$$Nw(t) + Np(t) + N_c(t) = n \qquad \text{for all t}$$
$$Nw(t) + Np(t) = Nu(t) \qquad \text{for all t}$$
$$Nu(0) = n.$$
$$Nu(C_{max}) = 0.$$

We shall again use an over-bar to indicate an average for these quantities. The average being, of course, defined relative to the time period, e.g.

$$\overline{N}_u = \frac{1}{C_{\max}} \int_0^{C_{\max}} N_u(t)\,dt$$

Now we are in a position to define some measures of performance.

Criteria based upon completion times

The main criteria in this category are F_{max}, C_{max}, $\overline{F}$ and $\overline{C}$. Minimizing F_{max}, the maximum flow time is essentially saying that a schedule's cost is directly related to its longest job. Minimizing C_{max}, the maximum completion time, says that the cost of a schedule depends on how long the processing system is devoted to the entire set of jobs. Note that in the case where the ready times are zero C_{max} and F_{max} are identical. However, when there are non-zero ready times, C_{max} and F_{max} are quite distinct. Indeed, if one job bas an extremely late ready time it may easily happen that the job with the *shortest* flow-time completes at C_{max} It is appropriate to mention here that C_{max} is also called the total production time or the make-span. Minimizing $\overline{F}$, the mean flow-time, implies that a schedule's cost is directly related to the average time it takes to process a single job. We shall find that minimizing $\overline{C}$, the mean completion time, is equivalent to minimizing $\overline{F}$; i.e. a schedule which attains the minimum $\overline{C}$ also attains the minimum $\overline{F}$ and vice versa (see Theorem 2.2). It may seem strange that F_{max} and C_{max} are quite distinct measures of performance, whereas $\overline{F}$ and $\overline{C}$ are essentially the same. The reason for this is quite simple. The operation of taking the

maximum of a set of numbers has different properties to that of taking the mean. For a further discussion of this, see Theorem 2.2 and the remarks that follow it.

- Although we shall seldom do so, some authors have considered weighted measures, which recognize that some jobs are more important than others. Thus they suggest that we minimize weighted averages,

$$\sum_{i=1}^{n} \alpha_i C_i \quad \text{or} \quad \sum_{i=1}^{n} \beta_i F_i$$

where $\alpha_1, \alpha_2 \ldots \alpha_n$ and $\beta_1, \beta_2 \ldots \beta_n$ are weighting factors usually summing to 1.

Criteria based upon due dates

Since the cost of a schedule is usually related to how much we miss target dates by, obvious measures of performance are $\overline{L}$, L_{max}, $\overline{T}$, and T_{max}; i.e. the mean lateness, the maximum lateness, the mean tardiness, and the maximum tardiness respectively. Minimizing either L or $L_{max.}$ is appropriate when there is a positive reward for completing a job early t and that reward is larger the earlier a job is. Minimizing either $\overline{T}$ or T_{max} is appropriate when early jobs bring no reward; there are only the penalties incurred for late jobs.

Sometimes the penalty incurred by a late job does not depend on how late it is; a job that completes a minute late might just as well be a century late. For instance, if an aircraft is scheduled to land at a time after which it will have exhausted its fuel then the results are just as catastrophic whatever the scheduled landing time. In such cases, a reasonable objective would be to minimize n_t the number or tardy jobs, i.e. the number of jobs that complete after their due dates.

Criteria based upon the Inventory and utilization costs

Here we might wish to minimize $\overline{N}_W$, the mean number of jobs waiting for machines, or $\overline{N}_u$, the mean number of unfinished jobs. Both of these measures are roughly related to the in-process inventory costs. We might be more concerned with minimizing $\overline{N}_c$, the mean number of completed jobs, because doing this will, in general, reduce the inventory costs of the finished goods. If our aim is to ensure the most efficient use of the machines, then we might choose to maximize $\overline{N}_p$ the mean number of jobs actually being processed at any time. Alternatively we might seek the objective of efficient machine use by minimizing $\overline{I}$, I_{max} the mean or the maximum machine idle time. (N.B. For idle times the mean and the maximum are taken over the machines, not the jobs.)

Finally, we note a classification of performance measures into those that are regular and those that are not. A regular measure R is simply one

that is non-decreasing in the completion times. Thus R is a function of $C_1, C_2, \ldots C_n$ such that

$$C_1 \leq C_1', C_2 \leq C_2', \ldots C_n \leq C_n' \text{ together} \Rightarrow R(C_1, C_2, \ldots C_n) \leq R(C'_1, C'_2, \ldots C'_n).$$

The rationale underlying this definition is as follows: Suppose that we have two schedules such that under the first all the jobs complete no later than they do under the second. Then under a regular performance measure the first schedule is at least as good as the second. Note that we seek to minimize a regular measure of performance.

$\overline{C}$, C_{max}, $\overline{F}$, F_{max}, $\overline{L}$ L_{max}, $\overline{T}$, T_{max} and n_T are all regular measures of performance, as a number of simple arguments show. For instance, consider the performance measure C_{max} Here we have:

$R(C_1,C_2,..C_n) = C_{max} = \max(C_1,C_2,..C_n)$

Let $C_1 \leq C_1^{'}, C_2 \leq C_2^{'}, ... C_n \leq C_n^{'}$. Then

$R(C_1,C_2,..C_n) = \max(C_1,C_2,..C_n)$

$\leq \max(C'_1,C'_2,..C'_n) = R(C'_1,C'_2,..C'_n)$

Hence C_{max} is a regular performance measure.

1.6 CLASSIFICATION OF SCHEDULING PROBLEMS

It will be convenient to have a simple notation to represent types of jobshop problems. We shall classify problems according to four parameters: *n/m/A/B.*

n is the number of jobs.

m is the number of machines.

A describes the flow pattern or discipline within the machine shop.

When *m* = 1, *A* is left blank. *A* may be

F for the flow-shop case, i.e. the machine order for all jobs is the same.

P for the permutation flow-shop case. Here not only is the machine order the same for all jobs, but now we also restrict the search to schedules in which the job order is the same for each machine. Thus a schedule is completely specified by a single permutation of the numbers 1, 2, ... *n,* giving the order in which the jobs are processed on each and every machine.

G the general job-shop case where there are no restrictions on the form of the technological constraints.

B describes the performance measure by which the schedule is to be evaluated. It may take, for instance, any of the forms discussed in the previous section.

As. an example: $n/2/F/C_{max}$ is the n job, 2 machine, flow-shop problem where the aim is to minimize make-span.

In using four parameters we follow Conway, Maxwell. and Miller (1967). Other authors, notably Lenstra (1977), Rinnooy Kan (1976), and Graham *et al.* (1979), introduce further parameters, but their discussions extend over a much larger family of problems than we shall be considering.

1.7 REAL SCHEDULING PROBLEMS AND THE MATHEMATICS OF THE JOB-SHOP

One need not have encountered scheduling problems in practice to realize that they are vastly more complex than those of the job-shop as we have defined them. Few obey many, much less all, of the assumptions that we have made. The cost of a schedule is seldom well represented by a function as simple as $\overline{C}$ or T_{max}. It might be expected, therefore, that there would be little practical relevance of the theory which we will develop. In fact, our analysis is not at all irrelevant and it would be wise to pause and consider why.

Firstly let us examine assumptions 1 to 11 of section 1.4. (Assumption 12 differs in nature from the others and will be discussed separately.) These assumptions are undoubtedly restrictive. They limit the structure of the job-shop problem greatly. They define quite

precisely which routings of jobs are allowable and which are not; what the capacities and availabilities of the machines are; etc. It is very easy to imagine practically occurring problems where some of these restrictions are totally inappropriate. So it is comforting to find that these assumptions are not necessary to the development of a mathematical theory of scheduling; rather they are typical of the assumptions which we might make. We could have chosen another set of assumptions, and so defined another family of scheduling problems, and then developed a theory of scheduling for the family. Had we done so, we would have encountered the same difficulties, the same forms of argument, the same mathematical techniques, and essentially the same results as we shall here. The reason for choosing the job-shop family is that it leads to a presentation of the theory which is particularly coherent and, furthermore, is not encumbered with a confusion of caveats and provisos needed to cover special cases. Once the job-shop family has been studied, it will be an easy matter to follow developments of the theory in other contexts. To help in this, Chapter 12 is a brief survey of such developments; it defines many other families of scheduling problems and references the literature appropriate to them.

Assumption 12 and, to a small extent, Assumption 10, stand quite

distinct from the rest. They confine attention to non-random problems, that is problems in which *all* numerical quantities are known and fixed in advance. There is no uncertainty. Because the number of jobs and their ready times are known and fixed, we call our problems **static**. Because the processing times and all other parameters are known and fixed, we call our problems **deterministic**. Problems in which jobs arrive randomly over a period of time are called **dynamic**. Problems in which the processing times, etc. are uncertain are called **stochastic**. It may be argued that all practical scheduling problems are both dynamic and stochastic, if for no other reason than that all quantities are subject to some uncertainty. In fact, in many problems the randomness is quite obvious. For instance, it may be impossible to predict exactly when jobs will become available for processing, e.g. aircraft arriving at an airport's airspace; it may be impossible to predict processing times exactly, e.g. during routine maintenance it will not be known which parts have to be replaced until they have been examined and that examination is one of the operations of the maintenance process; it may be impossible to predict the availability of machines, for some may have significant breakdown rates; and so on. Given that most problems have dynamic and stochastic elements, why do we confine ourselves to static, deterministic cases?

To begin with, there are problems in which any randomness is quite clearly insignificant, i.e. the uncertainty in the various quantities is several orders of magnitude less than the quantities themselves. Indeed, as microprocessors and industrial robots enter production lines, we may expect a greater degree of certainty in processing times. Secondly, we cannot study the dynamic and stochastic families of problems until we have understood the static, deterministic family. An awareness of the techniques for scheduling jobs when there is no uncertainty involved will point us towards the solution of stochastic problems. Finally, there is a rather negative reason for omitting the study of dynamic and stochastic problems from this book. To have included it would have required a deeper knowledge of probability theory and statistics than appropriate to an introductory text. The literature of the dynamic stochastic problem is briefly surveyed and referenced in Chapter 12.

So we shall accept all twelve assumptions of section 1.4 without further question, and that leaves us just one more point to discuss here: is our choice of performance measures too limited to allow the representation of the scheduling goals that arise in practice. The first point to note is that in using a performance measure such as $\overline{T}$ we are not assuming that the cost of a schedule is directly proportional to $\overline{T}$, i.e. that the cost is a positive linear

function of $\overline{T}$. What we are assuming is that the cost is an increasing function of $\overline{T}$, i.e. a function such that the cost increases when and only when $\overline{T}$ increases; the increases in cost and $\overline{T}$ need not be proportional. Under this condition minimizing $\overline{T}$ minimizes the cost. Thus in restricting the choice of performance measures to those listed in section 1.5, we are not restricting the form of cost function quite as much as we might think. Nonetheless, we have limited our choice and we should consider the implications of this.

We are limiting ourselves not because it is wrong to want to minimize, say C_{max} but because in any real problem we would also want to minimize *L,* I, etc. The total cost of a schedule is a complex combination of processing costs, inventory costs, machine idle-time costs, and lateness penalty costs. In other words, each of our performance measures represents only a component of the total cost. A schedule which minimizes a component cost may be very poor in terms of the total cost.

In fact, our study will lose little by its failure to consider total costs. We shall discover that even with simple performance measures scheduling problems can be extremely difficult. Their solution usually requires heuristic or approximate methods (see Chapters 10 and 11), and these methods may easily be modified to minimize total costs. Moreover, the insight necessary to enable such modifications will come from our earlier work on problems with simple performance measures.

Gupta and Dudek (1971), and Rinnooy Kan (1976, pp. 24-28) discuss the form of the total scheduling cost. Also Van Wasenhove and Gelders (1980) have introduced the concept of efficiency, i.e. Pareto optimality, into the field of scheduling, and we shall study this briefly in Chapter 4.

1.8 ALGY, BERTIE, CHARLES, AND DIGBY'S FLAT REVISITED

Please do not read this section until you have tried Problems 1.2.

Usually I shall only sketch the solutions to the problems that I set (see pp. 207-229). However, the problems concerning Algy and his friends are so important to the development of the theory that we should pause and examine their solution in some detail.

1. Is the given schedule, feasible? The short answer is no, and we may discover this in a number of ways. We might try to draw a Gantt diagram of the schedule and find that it is impossible to place some of the blocks without conflicting with either the schedule or the technological constraints. Alternatively we might produce an argument similar to, but more involved than, that by which we showed the schedule in Table 1.3 to be infeasible. It will be agreed though that neither of these methods is particularly straightforward and, moreover, that the thought of extending either to larger

problems is awesome. What we need is a simple scheme for checking the schedule operation by operation until either a conflict with the technological constraints is found or the whole schedule is shown to be feasible. The following illustrates such a scheme.

Firstly we write the schedule and the technological constraints side by side as below. You should ignore the 'circles' for the time being. We shall imagine that we are operating the schedule. We shall assign papers to readers as instructed. As the papers are read we shall circle the operations to indicate that they are completed and pass the papers to their next readers. Either we shall meet an impasse or we shall show that the schedule is feasible. (Note that I have subscripted the circles in the order in which they are entered.)

Schedule					Technological Constraints			
	Read By							
Paper	1st	2nd	3rd	4th	Reader	Order of Reading Papers		
F.T.	C	D	B	A	A	F.T.	G.	D.E.	S.
G.	B	C	A	D	B	G1	DE2	F.T.	S.
D.E.	B	C	A	D	C	DE3	G4	F.T5.	S.
S	B	A	D	C	D	S.	F.T.	G.	D.E.

We begin in the top left hand comer of the schedule. Charles is given the *Financial Times,* but will not read it until he has read the *Dailv Express* and the *Guardian.* So we must leave this operation uncompleted and un· circled. Proceeding down the schedule, Bertie is given the *Guardian* and we see from the technological constraints that he is immediately ready to read it. So we circle this operation both in the schedule and in the technological constraints. Next Bertie is given the *Daily Express* and we see that, now he has read the *Guardian,* he is immediately ready to read it. So this is the second operation to be circled. The *Sun* we see is also assigned to Bertie, but he is not ready to read it so we leave this operation uncircled. We continue by returning to the top line of the schedule and by repeatedly working down the schedule, checking the leftmost uncircled operation in each line to see if it may be performed. Thus we show that Charles may read the *Daily Express,* the *Guardian,* and the *Financial Times* without any

conflict with the technological constraints. The position is now as shown with five operations circled and we can see quite clearly that an impasse has been reached. Each paper is assigned to a reader who does not wish to read it yet. Hence the schedule is quite clearly infeasible.

It will transpire that we need to check for feasibility only very infrequently since the algorithms and methods that we shall study are designed so that they cannot produce infeasible schedules. For this reason I shall not write out the steps of the checking scheme explicitly; they should be clear enough from the example anyway. If you need to check a schedule for a more conventional problem

based upon jobs and machines, you should have no difficulty in translating the method from the present context; just remember that here Algy and friends are the jobs, while the papers are the machines.

2. How many different schedules, feasible or infeasible, are there? A schedule for the general *n/m/G/B* job-shop problem consists of *m* permutations of the *n* jobs. Each permutation gives the processing sequence of jobs on a particular machine. Now there are *n!* different permutations of *n* objects and, since each of the *m* permutations may be as different as we please from the rest, it follows that the total number of schedules is $(n!)^m$. In the problem facing Algy and his friends $n = 4$ and $m = 4$. So the total number of schedules is $(4!)4 = 331,776$.

It is worthwhile pausing to consider the implications of this rather startling number. Here we have a very small problem: only 4 'machines' and 4 'jobs'. Yet the number of possible contenders for the solution is enormous. We cannot hope to solve the problem by the simple expedient of listing all the possible schedules, eliminating the infeasible, and selecting the best of those remaining. To be fair, we might be able to check through the 331,776 possibilities in a reasonable time on a fast computer. If 1000 schedules were checked each second (which would be very fast), the computer would solve the problem in just over 51 minutes. But suppose that a guest were staying in the flat so that there were 5 readers. The number of schedules would now be $(5!)4 = 2.1 \times 10^8$ and the computer would take over 57 hours to solve this new problem! The very size of these numbers indicates the very great difficulty of scheduling problems. To have any chance at all of solving them we must use subtlety. But even with the most subtle methods available we shall discover that some problems defy practical solution; to solve them would literally take centuries.

3. *What is the earliest time at which Algy and his friends may leave for the country?* Perhaps the easiest way for us to approach this problem is to look back at the schedule given in Table 1.2 and see if we can improve upon it in any obvious way. Looking at the Gantt diagram (Fig. 1.1) and, in particular the row for the *Sun,* we see that it is the *Sun* that is finished last. Moreover,

it is left unread between 11.05 when Algy finishes it and 11.11 when Charles is ready for it. Thus 6 minutes are apparently wasted. Is there another schedule which does not waste this time, one which ensures that the *Sun* is read continuously? Well yes, there is. Consider Table 1.4. This has the Gantt diagram shown in Fig. 1.3. Note that now the *Sun* is read continuously and that it is the last paper to be finished. Thus under this schedule the earliest time at which they can leave for the country is 11.45. Moreover, a little thought will convince most people that this is an optimal schedule. Everybody starts reading as soon as they can and once started the *Sun* is read without interruption. There seems to be no slack left in the

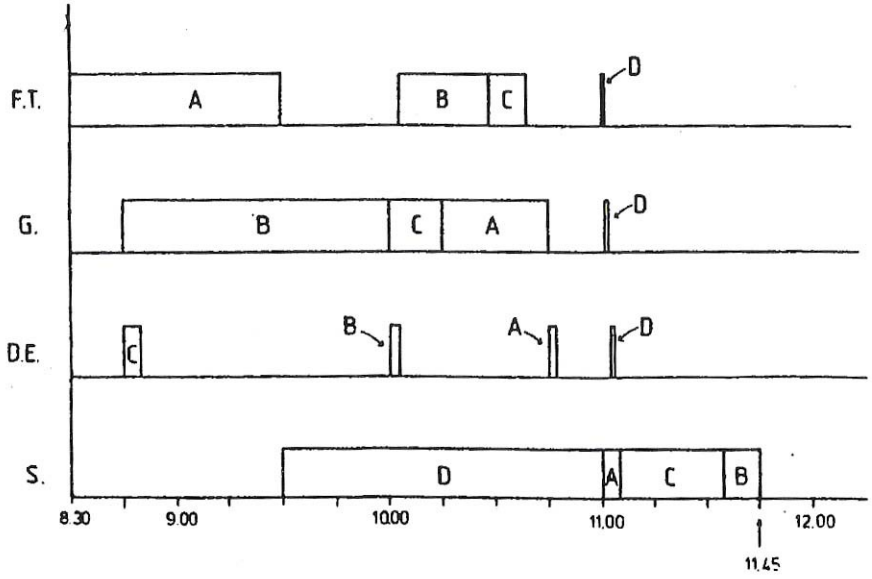

Fig. 1.3—Gantt diagram for the schedule in Table 1.4.

system. But there is. Consider the schedule in Table 1.5. This schedule leads to the Gantt diagram shown in Fig. 1.4 and we see that all reading is now completed by 11.30, 15 minutes earlier than allowed by the schedule in Table 1.4. So that schedule is clearly not optimal. How has this improvement been achieved? Compare the rows for the *Sun* in the two Gantt diagrams (Figs 1.3 and 1.4). What

we have done is 'leap-frogged' the block for Charles over those (or Digby and Algy. Because Charles can be ready for the *Sun* at 9.15, if he is allowed the other papers as he wants them, we gain 15 minutes. Moreover, it is possible to schedule the other readers, Algy, Bertie, and Digby so that this gain is not lost. The moral of all this is that in scheduling you often gain overall by not starting a job on

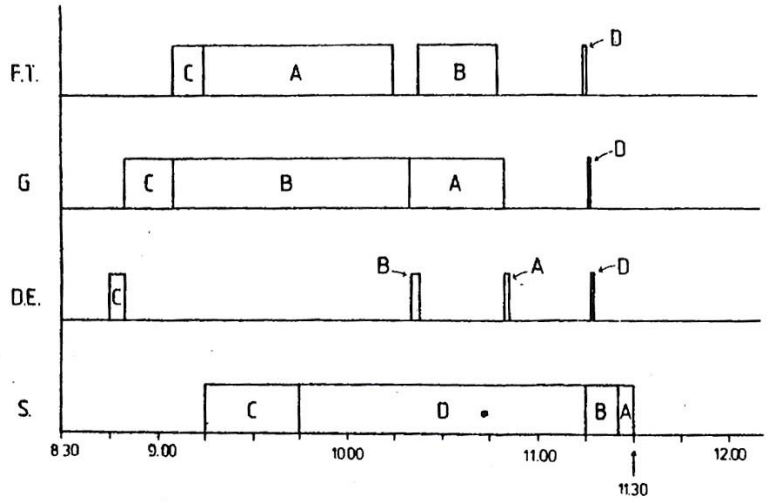

Fig. 1.4—Gantt diagram for the schedule in Table 1.5.

a

machine as soon as you might. Here Algy and Bertie wait for the Financial Times and Gwzrdian respectively. They could snatch up these papers as soon as they get up, but their patience is a virtue rewarded.

It turns out·that the schedule in Table 1.5 is optimal; no other schedule allows them to leave the flat earlier. To see this we consider four mutually exclusive possibilities: Algy reads the Sun before anyone else; Bertie does; Charles does; or Digby does. At the earliest Algy can be ready to read the Sun at 10.02. (Check this from Table 1.1.) The earliest times at which

. Bertie, Charles and Digby can be ready are 10.28,9.15, and 9.30 respec-

,1 tively.Thus.j~'=_~&!iumeJhat, once started, the Sun is read continuously f\ taking 2 hr. 15 min. in total, then the earliest time at which all reading can ; finish in the four cases is 12.17, 12.43, 11.30 and 11.45 respectively. Note J that these are lower bounds on the completion times. For instance, a ~. schedule which gives Algy the Sun first might not finish at 12.17 either

\.~ because other papers continue to be read after the Sun is finished or)'fd because it is not possible for the Sun to be read continuously. So the

earliest pos,~j~~'me for any schedule to finish is min{12.17, 12.43, 11.30,

/11.45} = 41.30. able 1.5 gives a sche~ule }ompleting at. 11.30; it must, the~~eiotC,be- ptimal.j)~ aw t?17-. ~ ~.~ 4UJ+' .?

J .- The structure of the above argument deserves special emphasis for it

will be developed into a powerful solution technique known as .branch and bound (see Chapter 7). We had a particular schedule which completed finally at a known time. To show that this schedule was optimal, We considered all possible schedules and divided them into four disjoint classes. We worked out for each class the earliest that any schedule within that class could complete, Le. we found a lower bound appropriate to each of these classes. We then noted that our given schedule completed at the lowest of the lower bounds. Thus no other schedule could complete before it and so it had to be an optimal schedule.

4. What is the earliest time at which Algy, Bertie and Charles may leave without Digby? 11.03. I leave with you both the problem of finding a schedule to achieve this and the problem of showing such a schedule to be optimal. However, I will give you one hint. Use a bounding argument like that above except that you should consider who is first to read the Guardian, not the Sun.

5. How does one prove a schedule to be optimal? Need one resort to complete enumeration? For the particular scheduling problem facing Algy and friends we now know that complete enumeration is unnecessary. However, the solution of Problem 1.2.3 involved a certain element of luck, or rather relied on knowing the answer before we started. No straightforward logical argument led to the schedule in Table 1.5. I just produced it rather like a magician pulling a rabbit from a hat. Moreover, the bounding argument

machine as soon as you might. Here Algy and Bertie wait for the Financial Times and Guardian respectively. They could snatch up these papers as soon as they get up, but their patience is a virtue rewarded.

It turns out that the schedule in Table 1.5 is optimal; no other schedule allows them to leave the flat earlier. To see this we consider four mutually exclusive possibilities: Algy reads the Sun before anyone else; Bertie does; Charles does; or Digby does. At the earliest Algy can be ready to read the Sun at 10.02. (Check this from Table 1.1.) The earliest times at which Bertie, Charles and Digby can be ready are 10.28, 9.15, and 9.30 respectively. Thus, if we assume that, once started, the Sun is read continuously taking 2 hr. 15 min. in total, then the earliest time at which all reading can finish in the four cases is 12.17, 12.43, 11.30 and 11.45 respectively. Note that these are lower bounds on the completion times. For instance, a schedule which gives Algy the Sun first might not finish at 12.17 either because other papers continue to be read after the Sun is finished or because it is not possible for the Sun to be read continuously. So the earliest possible time for any schedule to finish is min{12.17, 12.43, 11.30, 11.45} = 11.30. Table 1.5 gives a schedule completing at. 11.30; it must, therefore be optimal.

The structure of the above argument deserves special emphasis for it will be developed into a powerful solution technique known as .branch and bound (see Chapter 7). We had a particular schedule which completed finally at a known time. To show that this schedule was optimal, we considered all possible schedules and divided them into four disjoint classes. We worked out for each class the earliest that any schedule within that class could complete, i.e., we found a lower bound appropriate to each of these classes. We then noted that our given schedule completed at the lowest of the lower bounds. Thus no other schedule could complete before it and so it had to be an optimal schedule.

4. What is the earliest time at which Algy, Bertie and Charles may leave without Digby? 11.03. I leave with you both the problem of finding a schedule to achieve this and the problem of showing such a schedule to be optimal. However, I will give you one hint. Use a bounding argument like that above except that you should consider who is first to read the Guardian, not the Sun.

5. How does one prove a schedule to be optimal? Need one resort to complete enumeration? For the particular scheduling problem facing Algy and friends we now know that complete enumeration is unnecessary. However, the solution of Problem 1.2.3 involved a certain element of luck, or rather relied on knowing the answer before we started. No straightforward logical argument led to the schedule in Table 1.5. I just produced it rather like a magician pulling a rabbit from a hat. Moreover, the bounding argument

that I used to show optimality relied heavily on the, structure of this particular problem, namely that the optimal schedule allowed the Sun to be read continuously. (Why is this particular feature important to the argument?) In short, I have been able to solve this problem for you simply because I set it. So the question remains: in general is it necessary to use complete enumeration to solve scheduling problems? The answer is the rest of this book.

1.9 GENERAL READING

Scheduling theory, as I have said, is a surprisingly difficult subject. Consequently there are few elementary introductions. Conway, Maxwell, and Miller (1967) is, perhaps, the simplest reference with Baker (1974) a close second. The papers of Mellor (1966) and Bakshi and Arora (1969) introduce the job shop scheduling problem, but then go little further. There are several collections of papers available: notably Muth and Thompson (1963), Elmaghraby (1973), Coffman (1976), and a special issue of Operations Research (1978, Volume 26, No.1). At a more advanced level are the monographs of Lenstra (1977) and Rinnooy Kan (1976). Graham et al. (1979) give an excellent, but mathematically terse, survey of the known results in scheduling theory as they stood at the end of 1977. Klee (1980) reviews the whole area of combinatorial optimization, of which scheduling theory forms a part.

1.10 PROBLEMS

1. Choose a practical scheduling problem that you know about from your own experience. Try to fit it into the structure of the general job shop. Consider carefully whether your problem obeys each assumption given in section 1.4 and discuss carefully what performance criterion is appropriate.

2. Show that for any job, Jj:

Li = Fi - ai = Ci - ri - ai = Ci - di and hence deduce that:

L = $\overline{F} - \overline{a} = \overline{C} - \overline{r} - \overline{a} = \overline{C} - \overline{d}$

Show that Li = Ti – Ei.

4. (i) Show that $\overline{F}$, Tmax and nT are all regular measures of performance.

(ii) Show that minimizing $\overline{E}$ or Emax does not correspond to a regular measure. Why might it be sensible to want to minimize $\overline{E}$ or Emax ?

5. Is the following schedule feasible for Algy and his friends?

	Read By			
Paper	1st	2nd	3rd	4th
F.T.	A	C	B	D
G.	C	A	B	D
D.E.	C	A	B	D
S.	A	D	C	B

that I used to show optimality relied heavily on the, structure of this particular problem, namely that the optimal schedule allowed the Sun to be read continuously. (Why is this particular feature important to the argument?) In short, I have been able to solve this problem for you simply because I set it. So the question remains: in general is it necessary to use complete enumeration to solve scheduling problems~ The answer is the rest of this book.

1.9 GENERAL READING

Scheduling theory, as I have said, is a surprisingly difficult subject. Consequently there are few elementary introductions. Conway, Maxwell, and Miller (1967) is, perhaps, the simplest reference with Baker (1974) a close second. The papers of Mellor (1966) and Bakshi and Arora (1969) introduce the job~shop scheduling problem, but then go little further. There are several collections of papers available: notably Muth and Thompson (1963), Elmaghraby (1973), Coffman (1976), and a special issue of Operations Research (1978, Volume 26, No.1). At a more advanced level are the monographs of Lenstra (1977) and Rinnooy Kan (1976). Graham et al. (1979) give an excellent, but mathematically terse, survey of the known results in scheduling theory as they stood at the end of 1977. Klee (1980) reviews the whole area of combinatorial optimization, of which scheduling theory· forms a part.

1.10 PROBLEMS

1. Choose a practical scheduling problem that you know about from your own experience. Try to fit it into the structure of the general job shop. Consider carefully whether your problem obeys each assumption given in section 1.4 and discuss carefully what performance criterion is appropriate.

2. Show that for any job, Jj:

L; = Fi - a; = C; - r; - a; = C; - d; and hence deduce that:.

L = 17 - a = C - r - a = C - a.

Show that L; = T; - E;.

4. (i) Show that 'P, T m •• ' and n T are all regular measures of performance.

(ii) Show that minimizing E or Em •• does not correspond to a regular measure. Why might it be sensible to want to minimise E or Em •• ?

5. Is the following schedule feasible for Algy and his friends?