UNIVERSITY OF SOUTHAMPTON

Irrigation Scheduling with Integer Programming

Tonny Tessa de Vries

Thesis submitted for the degree of Doctor of Philosophy

Department of Civil and Environmental Engineering

May 2003

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE
DEPARTMENT OF CIVIL AND ENVIRONMENTAL ENGINEERING

Doctor of Philosophy

IRRIGATION SCHEDULING WITH INTEGER PROGRAMMING

by Tonny Tessa de Vries

Water delivery arrangements in irrigation systems range from completely fixed where duration, discharge and start time are set once and seldom or never change to on-demand, where users can take water whenever they need it in whatever quantity desired. In between these two extremes lies a range of arranged-demand systems where duration, discharge and/or start time are flexible to a certain degree but are agreed upon before the irrigation event takes place.

This thesis shows how arranged-demand scheduling can be interpreted as a single or multi-machine scheduling problem. Mixed integer linear programming is used to develop a series of models that allow scheduling of irrigation turns in a wide range of irrigation systems. Given that duration and target start time of the irrigation turns are specified by the users, the objective of all models is to find a schedule such that the difference between target start time and scheduled start time are as small as possible.

Two types of models are presented that reflect different management options at tertiary level. Contiguous scheduling schedules irrigation turns back-to-back so that operational spillage and/or gate operations are minimised. In non-contiguous schedules idle time can be inserted in between irrigation turns, which allows greater opportunity to match target start times with scheduled start times. Travel times play an important role in open channel irrigation systems and need to be taken into account when determining irrigation schedules. In pressurised systems on the other hand, travel times are non-existent. Models that incorporate travel times as well as models without travel times are presented to demonstrate scheduling of irrigation turns in both open channel and pressurised irrigation systems. Irrigation systems that operate under a sequential irrigation schedule are well represented by the single machine problem, in which water is seen as a machine that processes jobs (users) one-by-one. Where users irrigate simultaneously water can be seen as consisting of several machines (or stream tubes) that each process a separate job. In simple multi-machine scheduling all users receive the same discharge and each job is processed by one stream tube. In irrigation systems where discharges vary from user to user, complex multi-machine scheduling, in which each job can be processed by more then one machine, can be applied to obtain a schedule. In this thesis models are presented that allow single machine scheduling, simple multi-machine scheduling and complex multi-machine scheduling.

An irrigation interval is not a one-off event, but part of an irrigation season. Scheduling decisions made during previous intervals can be used to influence scheduling decision made for the coming interval. By adjusting the costs of water being delivered early or tardy, it is possible to give priority to those users who were disadvantaged in earlier intervals. A method is presented that can determine appropriate earliness/tardiness costs. Finally the earliness/tardiness index is presented. This index is a measure of the fairness of a schedule or series of schedules.

Table of contents

List of figures

List of tables

Acknowledgements

I would like to thank Dr. Anwar for his supervision of this thesis. His help has been invaluable. I would also like to thank the Department of Civil and Environmental Engineering for the financial support that made this work possible.

Notation

a_b = travel time for channel section b,
b = index representing channel section 1,2,...,B,
B = total number of sections between user i and user j,
c = large constant;
C_i = scheduled completion time of job i;
C_k = scheduled completion time of the job in position k;
d_i = duration of job i;
d_{ij}^* = sequence dependent duration of job i if job i directly precedes job j;
d_j = duration of job j;
d_k = duration of job in position k;
d_m = duration of job in position m;
d_n = duration of job n;
E_i = earliness of job i;
E_{ip} = earliness of job i in interval p;
E_k = earliness of the job in position k;
E_{max} = maximum allowable earliness;
g = irrigation interval over which all jobs must be completed;
H_{ip} = 1 when user i irrigates during period p, 0 otherwise.
i = index representing job 1,2,...,N;
I_i = initial setup time
j = index representing job 1,2,...,N;
k = index representing position 1,2,...,N in a schedule;
L_b = length of channel section b,
m = index representing position 1,2,...,k in a schedule;
M = a large positive number;
n = index representing job 1,2,...,N;
N = number of jobs to be scheduled;
p = index representing interval 1,2,...,P;
P = current interval;
p_i = processing time of job i;
p_m = processing time of the job in position m;
q = size of machine (discharge of stream tube);
Q = set of jobs to be scheduled;
Q_c = channel capacity;
q_i = due date of job i;
Q_i = discharge required for job i;
q_k = due date of the job in position k;
r_i = target start time for job i;
r_k = target start time of the job in position k;
S_i = scheduled start time of job i;
S_j = scheduled start time of job j;
S_k = scheduled start time of the job in position k;
t_b = travel time for channel section b;
T_i = tardiness of job i;
t_{ij} = sequence dependent setup time from user i to user j;
T_{ip} = tardiness of job i in interval p;
t_{ji} = sequence dependent setup time from user j to user i;
T_k = tardiness of the job in position k;

T_{max} = maximum allowable tardiness;
V = set of machines available to process jobs;
v_b = velocity in channel section b,
w = index representing machine 1,2,...,W;
X_a = idle time preceding the start of the first job in the schedule;
X_{aw} = idle time preceding the start of the first job of the schedule on machine w;
X_b = idle time proceeding the end of the last job in the schedule;
X_{bw} = idle time following the end of the last job of the schedule on machine w;
X_k = idle time inserted directly before the job in position k;
X_m = idle time inserted directly before the job in position m;
Z = objective function;
α = earliness penalty cost per unit of time;
α_i = cost of earliness per unit of time for job i;
α_{ip} = earliness cost of job of job j in interval p;
β = tardiness cost per unit of time;
β_i = cost of tardiness per unit of time for job i;
β_{ip} = tardiness cost of job j in interval p;
γ_{ij} = binary variable;
δ_{ij} = binary variable;
λ_{ik} = binary variable;
λ_{im} = binary variable;
τ_{iw} = binary variable;
τ_{nw} = binary variable;
ψ_w = binary variable;
φ_{ijw} = binary variable;
φ_{njw} = binary variable;
ε_p = earliness-tardiness index;

1 Introduction/background

1.1 Irrigation

It is estimated the world's population will increase to 8 billion in the next three decades. Planners project that the food production increase needed to match population growth will have to come from irrigated agriculture. Currently 17 percent of the world's agriculture is irrigated, accounting for around 40 percent of the total food production (Skogerboe, 2000). Increasing the irrigated agricultural area however is not an easy answer to the problem of increasing food production. Water is becoming more scarce through competition from urban and industry sectors and the environment. Land suitable for (irrigated) agriculture has had to be given up to meet urban and industry demands or has been lost due to salinisation and high ground water tables. The increased productivity of irrigated agriculture will have to come from improved performance and better irrigation water management (Clyma and Reddy, 2000).

Surface irrigation -furrow, basin and border- is still the predominant method of irrigation (Clyma and Reddy, 2000). The majority of irrigation projects around the world have some type of rotation delivery (Plusquellec et al., 1994). It consists of granting each of the users (or group of users) of a tertiary canal the exclusive use of the available water for a certain duration, turning the water over to the next user when the allocated time has passed. After each user has had a turn, the cycle is repeated (Rathod and Prajapati, 1998). Rotation schedules require relatively low capital investment in infrastructure and involve the least water-agency management and operational input. They are relatively easy and cheap to implement and the required skill level of both field staff and user is low (Replogle, 1987). On the negative side rotation schedules allow very little flexibility. Flow rates, duration and start time are very often fixed for an entire growing season or even longer. Users do not always get water when needed most or may get water when it is not wanted at all. Rotation schedules can result in inefficient irrigation, over-irrigation or under-irrigation, which in their turn can lead to yield reduction, high ground water tables and/or salinisation.

At the other end of the surface irrigation spectrum is on-demand irrigation. The principle of this type of irrigation is that water is available at any time, for any duration and at any flow rate. In practise there are some limits to the maximum available flow rate

and in most on-demand systems some advance notice (usually 24-48 hours) is required before an irrigation turn can start (Burt and Styles, 2000; McCornick, 1993). The initial capital investment in infrastructure of on-demand irrigation systems is large. Sufficient system capacity is needed to meet demands if several users in a tertiary unit require water simultaneously. A major storage reservoir or pumping facilities may be needed to meet peak demands. Since requested flow rates may differ from user to user or from irrigation turn to irrigation turn, regulatory structures are needed. Staff with high skill levels are needed to operate the system, take water orders, determine delivery schedules etc. Farmers may need training to be able to order the appropriate flow rate, duration and time for irrigation turns (Merriam, 1987). The danger of inefficient irrigation, over-irrigation and/or under-irrigation still exists, but if managed properly on-demand irrigation can give a user the opportunity to grow crops efficiently and economically whilst meeting environmental demands (Cross, 2000).

Clearly, users could benefit from on-demand irrigation. Currently 93 % of the irrigation districts in Mid-Pacific Region of the United States operate on a unlimited frequency basis, i.e. users are free to choose the day they receive water (Burt and Styles, 2000). Outside the United States however, most irrigation systems still operate on rotation schedules (Burt, 2000). Many of these systems operate under protective irrigation rules, where the major objective is to spread the available water to the largest possible area to prevent drought and famine. Since the basic requirement for an on-demand supply system is the availability of unrestricted and variable amounts of water during the growing season, it is not possible to convert systems under protective irrigation to on-demand irrigation unless new water sources become available or the irrigated area is decreased. In irrigation systems where sufficient water is available, the introduction of on-demand irrigation would still require large investments in hardware (structures as regulators, spillways, etc. and communication system) and software (institutions, management and training) (Sarwar et al., 2001). The facilities required for measured and controlled delivery, which are both essential for on-demand irrigation, are not in place and their introduction would require a massive investment in physical, legal and administrative infrastructure (Perry, 2001). For most irrigation systems outside the United States on-demand irrigation facilities and investments needed for on-demand irrigation are not available. Therefore for these systems on-demand irrigation is at the moment not a viable option and improvements in the performance of water allocation and water management are more important.

1.2 Irrigation modelling

Irrigation management has received a lot of attention over the last couple of decades. Numerous models have been developed to assist managers in decision making processes at all levels, for a number of different goals and objectives, such as irrigation system layout and water allocation. To identify the different levels in an irrigation scheme it can be seen as a three tiered system: the source (reservoirs, rivers, lakes etc.), the distribution system (canals, pumping stations, division structures etc.) and the final target of fields and crops. Goals and objectives of irrigation models can be divided into three main groups:

- planning
- allocation
- scheduling

Planning is defined as "to devise, design, project, arrange beforehand" (Simpson and Weiner, 1989). In irrigation modelling planning occurs at two different levels, source/distribution and field level. At source (and distribution) level modelling is used to decide the layout of a future irrigation scheme. Models have been developed to choose the most suitable system of dams, reservoirs, river diversions and irrigation canals, making it possible to consider a number of alternative irrigation development strategies (Onta et al., 1991; Rose, 1973). At field level planning models are concerned with cropping pattern (Kumar et al., 1998; Kuo et al., 2000). Some models use of combination of both (Malek-Mohammadi, 1998; Paudyal and Das Gupta, 1990). All planning models aim to maximise profitability, whether it is at source or field level.

Allocation is defined as "to set or lay apart for a special purpose, to apportion, assign, to give one as his special portion or share" (Simpson and Weiner, 1989). In irrigation, modelling allocation refers to the process of determining how to divide the available water. This process becomes especially important in time of water scarcity. Allocation of irrigation water can be done at several levels, from source to field. Most irrigation allocation models allocate water at the source level (e.g. Dudley et al., 1971; Hannan and Coals, 1995; Kipkorir et al., 2001; Petriczek and Uhrynowski, 1986; Wardlaw, 1999), although models for canal level (Khepar et al., 2000) and field level allocation (Akhand et al., 1993) do exist. Allocation models have a number of different purposes. They can be aimed at preserving equity between users (Khepar et al., 2000), to maximize crop revenue (Kipkorir et al., 2001), to optimize reservoir levels (Ponnambalam and Adams, 1987), to

minimize yield losses (Wardlaw and Barnes, 1999) or to minimize irrigation water distribution costs (Muspratt, 1971).

A schedule is a time-table, a programme or plan of events, operations (Simpson and Weiner, 1989). In irrigation modelling, scheduling refers to when a certain irrigation related event takes places. It occurs at all levels of the irrigation system, from when to operate pumping stations to when to irrigate a certain field. At source level scheduling is concerned with when to release water from reservoirs in order to satisfy irrigation demands (Naresh and Sharma, 2000). Scheduling at distribution level is a relatively new area of interest. The main concern here is the sequencing of the users of a tertiary unit (Anwar and Clarke, 2001; Wang et al., 1995; Suryavanshi and Reddy, 1986). Field level scheduling has been studied far more extensively and numerous models have been developed to decide when to irrigate a field (Naadimuthu et al., 1999; Pleban et al., 1983). Scheduling models have a range of purposes including comparing irrigation strategies (Lembke and Jones, 1972), maximizing profits (Holzapfel et al., 1986) and minimizing yield losses (Pleban et al., 1983).

Some irrigation models combine planning, allocation and scheduling. Upcraft et al. (1996) developed a model which determines the order of irrigating (scheduling) and the duration of each irrigation (allocation). The model described by Paul et al. (2000) first plans the acreage of each crop and then allocates water to each crop. Most of the mentioned models for irrigation planning, irrigation scheduling and water allocation use optimisation techniques for solving. These techniques are also widely used in the field of Operations Research.

1.3 Operations Research

During the Second World War, British military leaders asked scientists to analyse a number of military problems such as the deployment of radar and the management of convoy and mining operations. This application of mathematics and scientific methods to military problems was called Operations Research. Today Operations Research is defined as

> *a scientific approach to decision making, which seeks to determine how best to design and operate a system, usually under conditions requiring the allocation of scarce resources* (Winston, 1994).

A large part of Operations Research involves the study of optimisation problems. The goal of this type of problem is the minimisation or maximisation of one or more objectives (Pinedo, 1995). For optimisation problems one can think of two object classes, the first of which is *limited resources* (such as land or raw materials) and the second is *activities* (such as produce stainless steel or serve costumers). Each activity consumes or possibly contributes additional resources (Schrage, 1999). Objectives can also take many shapes such as maximize profit or minimize time needed to complete a task.

A simple example of where optimisation can be of use is that of a machine painting cars. The machine has the capacity to paint a certain number of cars, but can only do one at a time. Which car is the first to be painted? If all cars are the same there might be not much difference, but what if some cars are bigger than others and there is a limited amount of paint available? Do all cars get an equal amount of paint and thus remain partially unpainted or do some cars get painted and others not? What happens if the bigger cars can be sold for a higher profit, should they get a higher priority? If a problem like this can be adequately represented in a mathematical model, it can be studied using optimisation techniques.

One of the allocation processes studied in Operations Research is the allocation of resources over time to perform a number of activities or tasks. This type of problem is called scheduling. Scheduling can only be done in a situation where the nature of the tasks to be scheduled has been described and the configuration of the available resources has been determined (Baker, 1974). Planning always precedes scheduling. First, three fundamental managerial questions need to be answered: what product or service is going to be provided, on what scale will it be provided, and what resources are available? Finding the answers to these questions is the planning phase, only once they have been answered scheduling becomes important.

There are many different types of scheduling. The example used above where a machine paints cars is one of them. This kind of scheduling is aptly called machine scheduling. In machine scheduling there is a machine capable of a certain activity (in case of the example painting) and a number of jobs (cars) need to be processed (painted). Each job has a processing time (larger cars will take longer to paint than small cars). There will be a number of limited resources (paint, time). The objective can be many things; process as many jobs as possible in a given time or process the jobs so that profit are maximised (Crama and Spieksma, 1996; Soric, 2000). Within the field of machine scheduling there

are again many different forms. The most basic form is single-machine scheduling. The single-machine problem is characterized by the following conditions:

1. a set of N independent, single-operation jobs is available for processing at time zero,
2. job descriptors are known in advance,
3. one machine is continuously available and is never kept idle while work is waiting,
4. once processing begins on a job, it is processed to completion without interruption, i.e. preemption of jobs is not allowed (Baker, 1974).

Two pieces of information help to describe the job to be processed: processing time and due date. Sometimes a third one, the ready time is added. Processing time is the duration required to process a job and due date is the point in time at which a job is due to be finished. Ready time is the point in time when a job becomes available for processing.

Schedules are evaluated by the objective function. In certain situations it is important to meet due dates as closely as possible, since being late may result in angry customers and being early can cause perishable goods to deteriorate. This can be achieved by including a earliness/tardiness objective function in the mathematical model (Li, 1997; Ow and Morton, 1989). In other situations it may be more useful to keep the time a machine is processing a job to a minimum, in that case a flow time objective function has to be added to the mathematical model (Bard et al., 1993).

Machine scheduling can be used not only for problems with real machines, but in any situation where an activity takes places. An airport with just one runway can be seen as single machine scheduling problem, with the runway being the machine and the air planes coming in for landing the jobs to be scheduled. The single machine problem is only the basic form of machine scheduling. It can be expanded to a multi-machine problem, where two or more machines are capable of doing the same activity (Balakrishnan et al., 1999). In the multiple machine problem jobs need to undergo two (or more) processes at different machines, not necessarily in a certain order (Sun and Hwang, 2001). Another factor that can be added is multi periods. In this type of scheduling the effect of sequencing over a number of periods instead of just one period is studied. In many situations a certain time is needed to set up the machine so it can process the next job. In a scheduling situation this means that additional time has to be included in the schedule. Furthermore idle time between jobs can be allowed etc. A large number of possibilities exist.

1.4 Irrigation and Operations Research

A number of models have been developed for irrigation management. None of these models seem to have made the connection to Operations Research and more specifically machine scheduling or used the wealth of information available and work done on the subject. To continue in the same way would be a waste of a large source of new insights. Operations Research has been successfully applied to many different organisations as diverse as police departments and oil refineries. An irrigation department is no different from any other organisation in that scarce resources, such as water and time, have to be allocated, in this case to water users. These users in turn have their own demands, such as a preferred time to start irrigation or a certain amount of irrigation water requested. Matching theses demands within the constraints of the irrigation systems can be a very complex tasks. Operations Research may be a valuable tool to the management of an irrigation district. It allows easy and quick scheduling and can provide an insight into the effects of different scheduling policies. Hopefully it will optimise use of scarce resources such as water and labour.

1.4.1 Irrigation as single machine problem

Allocating water to users in an irrigation system can be seen as a single machine scheduling problem. There is a single activity (giving water) and there are a number of jobs (users wanting water) to be processed. In section 1.3 single machine scheduling models were characterised by the following conditions:

1. a set of N independent, single-operation jobs is available for processing at time zero,
2. job descriptors are known in advance,
3. one machine is continuously available and is never kept idle while work is waiting,
4. once processing begins on a job, it is processed to completion without interruption,

An irrigation scheme that operates under a rotational schedule, i.e. each user of a tertiary canal is granted the exclusive use of the water for a certain period, after which the water is turned over to the next user, can be seen as a single machine scheduling problem:

A. a group of N users is awaiting their irrigation turn at the start of an irrigation interval,

B. irrigation turn characteristics such as duration and desired start time are known in advance,
C. water is continuously available everywhere in the tertiary canal and is not allowed to be spilled, i.e. there is always someone irrigating,
D. once an irrigation turn has started it will be completed without interruption,

All jobs have a processing time (irrigation duration) and a ready time (start of an interval). The due date used in operations research however has to be replaced with a target start time since irrigation users do not specify when they want to finish an irrigation turn, but when they want it to start (Clemmens, 1987).

There are many possible sequences for a set of irrigation events that needs to be scheduled. As in irrigation scheduling the aim is to deliver water at the requested time, it is possible to evaluate the suitability of a schedule by its ability to match these target start times. The objective is to find the best schedule which can be found by minimizing the difference between the target start times and the actual or scheduled start time.

In Operations Research sometimes due dates become deadlines. Smith (1956) first described the scheduling problem with deadlines. In this type of problem jobs need to be completed on or before the deadline, i.e. no tardiness is allowed. In irrigation scheduling a similar problem arises. The irrigation interval is a specific period within which all jobs have to be processed, no job is allowed to be completed outside this period. The irrigation interval can be perceived as a common deadline for all jobs. But unlike the more classic single machine scheduling problems described in Operations Research, there is still a target start time (the irrigation equivalent of the due date) that needs to be considered. So although in irrigation scheduling jobs are allowed to be both early and tardy, they are not allowed to be completed past the common deadline of the irrigation interval.

1.4.2 Idle time

Baker and Scudder (1990) observed that not allowing idle time to be inserted into a schedule can have an undesired effect. It could force a job to be scheduled early whereas if idle time was allowed to be inserted the job would be processed exactly on time. Not allowing idle time to be inserted can be contrary to the objective of trying to match water

delivery with the requested start time. If idle time is allowed to be inserted into the irrigation schedule, the irrigation channel would be flowing continuously for the entire duration of the irrigation interval. Users abstract water as scheduled and when water is not being used, it spills into a drainage system and/or if possible is reused further downstream. This type of scheduling offers a greater opportunity for scheduled start time to match target start time, however at the expense of wastage. One could suggest that the channel be shut after every contiguous block of jobs is completed and reopened when the next job is scheduled to start. However, this can require an excessive number of gate operations. An alternative solution could be to remove idle time from the schedule and only allow jobs to be scheduled contiguously. If all jobs are scheduled contiguously gates would only need to be opened once, when the first users starts irrigation, and only closed one, when the last users finishes irrigation. This would reduce the need for large numbers of gate operation and prevent wastage of water. The choice whether to allow non-contiguous scheduling, where idle time is inserted in between jobs or to schedule contiguously, largely depends on the associated costs of water wastage, gate operations and the level of service an irrigation district wishes to provide.

1.4.3 Sequence dependent set-up times

In many situations a certain time is needed to set up the machine so it can process the next job. These setup times can play an important role in irrigation scheduling. In an irrigation system setup times would include the time needed for a user to close a farm outlet and the next user to open the following outlet. This time is insignificant compared to the irrigation duration and can be ignored. In open channel irrigation systems another important contribution to setup times is travel times. After irrigation at one outlet has been completed water needs to be diverted to another outlet. Depending on the distance between the outlets this time cannot be ignored. It also takes time for the channel to fill up to operating depth. The time needed for this process varies with the location of the outlets and the order in which they are operated. Figure 1.1 shows it may take a significant amount of time for water to travel from one user to another or none at all. This means that this part of the setup time is sequence dependent and cannot be ignored.

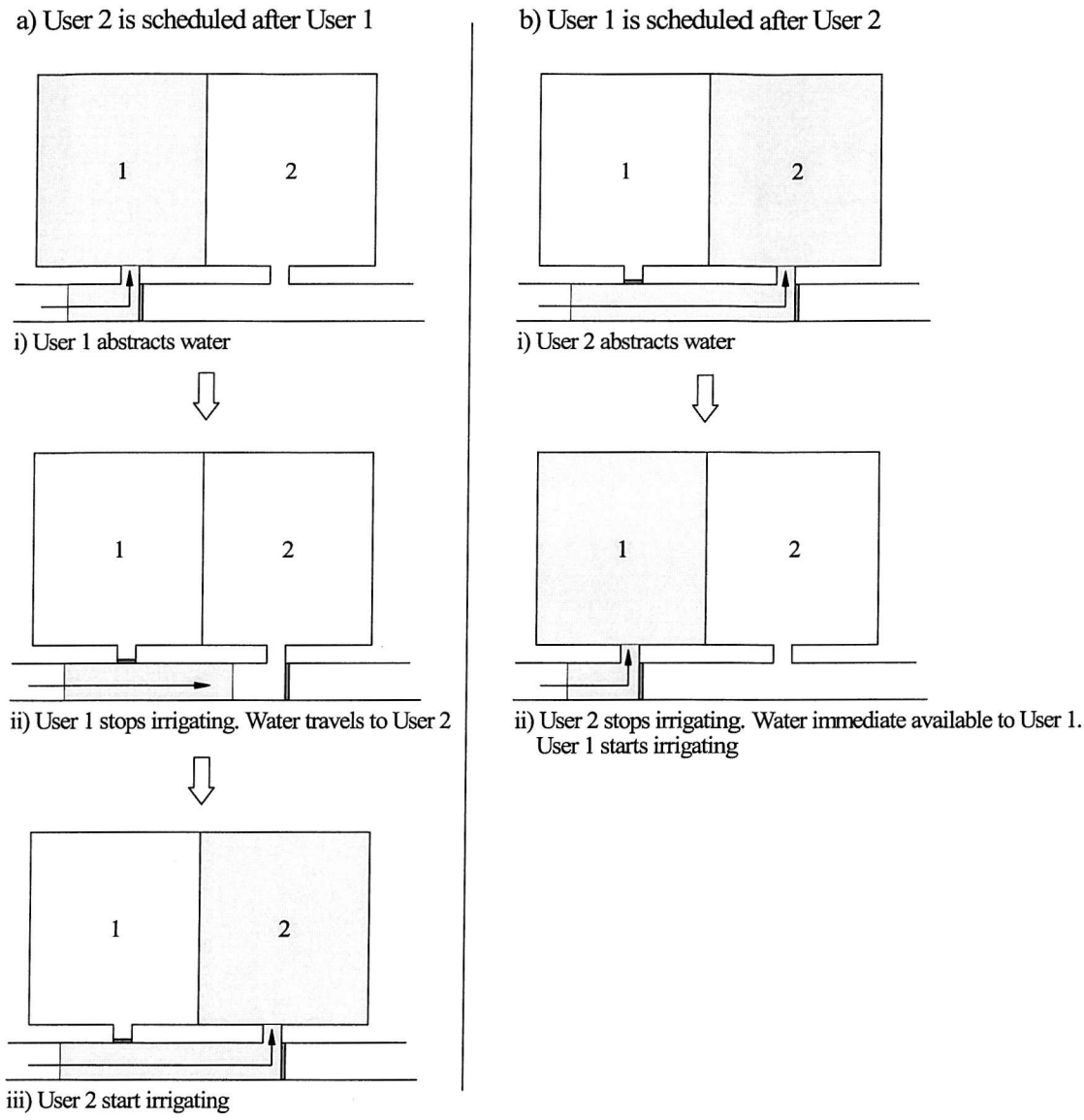


Figure 1.1 Sequence dependent setup times

1.4.4 Multi-machine scheduling

In multi-machine scheduling several machines are capable of processing any of the jobs available for processing. Water can be seen as one machine when one user receives all the water. It can also be seen as consisting of several machines if two or more users irrigate simultaneously. Wang et al. (1995) introduced the concept of stream tubes. The total discharge in a canal is composed of a number of (imaginary) stream tubes. A stream tube delivers water to only one outlet at any given time, but can supply water to a number of outlets in sequence. Figure 1.2 shows the relationship between stream tubes and outlets.

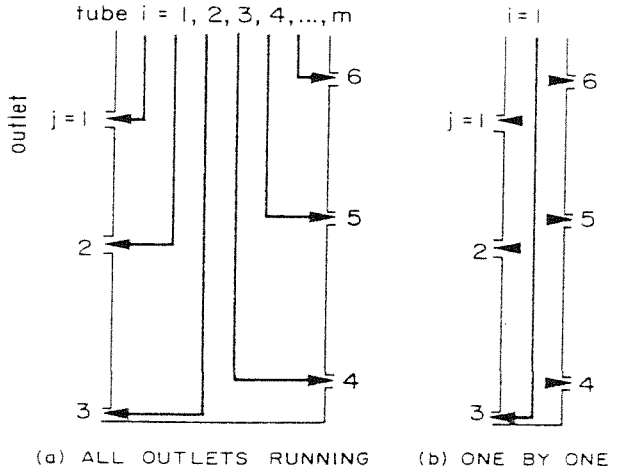


Figure 1.2 Relationship between stream tubes and outlets (from Wang et al., 1995)

Anwar and Clarke (2001) showed that stream tubes can be used to schedule a number of irrigation jobs according to their target start time. A limitation of the models used by Wang et al. (1995) and Anwar and Clarke (2001) is that the discharge of all outlets is identical. However by allowing two or more stream tubes to simultaneously supply water to one outlet it is possible to vary the discharge. In theory it is possible to divide the water in a channel into an infinite number of stream tubes. If an outlet is allowed to be simultaneously supplied by any number of stream tubes, virtually any discharge can be delivered to the outlet. In this thesis multi-machine scheduling of outlets where the discharge of all outlets is identical is referred to as simple multi-machine scheduling. When the discharge of the outlets are not identical, multi-machine scheduling is referred to as complex multi-machine scheduling.

1.4.5 Multi-interval scheduling

An irrigation interval is not a one-off sequence of irrigation events, but part of a series of similar intervals, together forming an irrigation season. Although it is important to match target start times as closely as possible during each interval, it is perhaps even more important that after an entire irrigation season all users have incurred a similar degree of earliness/tardiness. In multi-interval (or multi-period) scheduling decisions made in a previous interval can be used to influence the decisions that will be made for the current

interval. In irrigation scheduling the earliness and tardiness incurred in earlier intervals can be used to give priority to those users who have been most affected. The need for giving priority to certain users can be demonstrated by the example of an irrigation system where all users always ask for the same duration and target start time. Using optimisation techniques to determine a schedule would result in the same schedule for each interval. Consequently some users will always receive their water on time whereas others are always scheduled to receive water early or late. Giving priority to those users with the highest earliness/tardiness would result in different schedules that distribute the earliness/tardiness amongst all users. This hypothetical example may be an extreme case but unequal distribution of earliness/tardiness can result from any schedule, although to a lesser extent. To give preference to one job being processed on time over another, earliness and tardiness may be weighted. This would ensure that those jobs that already have incurred a large amount of earliness/tardiness contribute more heavily to the objective function than those jobs with a small amount of earliness/tardiness. Appropriate costs of earliness/tardiness are difficult to ascertain. James and Buchanan (1998) have described that, for typical 'industrial' applications, earliness costs can include cost of storage, cost of invested funds and cost of deterioration and tardiness costs can include the need for a product, availability of substitutes and the value of costumers. The actual value of these costs depends on who determines them. Very often costs of earliness/tardiness are set arbitrarily, in many cases by setting all costs of earliness/tardiness equal to 1 (Arkin and Roundy, 1991), which is the same as using a single interval model. Anwar and Clarke (2001) described a method to determine the earliness and tardiness costs for multi-interval scheduling. If earliness is equally as undesirable as tardiness, the earliness cost per unit of time of job j can be set equal to the tardiness cost per unit of time of job j.

2 Aim and objectives

2.1 Problem description

Water demands in irrigation systems are determined in a number of different ways, e.g. based on crop growth models or experience. Water demands consist of two parts, the requested amount of water and the desired time of delivery. The amount of water can be specified as a depth of water to be applied, a volume or as a discharge and duration of irrigation. Whatever method used to determine these demands, water deliveries need to match them as closely as possible. In irrigation systems that operate on an on-demand basis this can be achieved quite easily, but in other irrigation systems the situation is more complex. In those systems very often a rigid rotational schedule is operated with little opportunity to match demands and deliveries. By allowing some flexibility into a rotational schedule it may be possible to improve the current situation whilst retaining the simplicity of the rotational schedule. This flexibility can be achieved by creating more time within the schedule by increasing the discharge so that irrigation time becomes shorter, by allowing the irrigation interval to increase or by omitting users from the schedule. Increasing the discharge is very often not possible due to restrictions in either water availability or channel capacity. During periods of peak water use neither increasing the irrigation interval nor omitting users is an acceptable solution and a rotational schedule will almost certainly have to be operated. During other periods however, when demands decrease, both options could be used to create room for manoeuvring.

2.2 Aim

The aim of this research is to develop a series of management models for scheduling irrigation in tertiary units that operate an arranged-demand schedule. The main purpose of the models is to match water deliveries as closely as possible to water demands. To do so machine scheduling is used to schedule a number of users in a tertiary unit according to their requested start times, taking into account idle time and sequence dependent setup times. Both single machine scheduling (users irrigate sequentially) and multi-machine scheduling (users irrigate simultaneously) will be studied. The effects of scheduling over a

number of periods is also studied. A small computational experiment is conducted to determine the sensitivity of the models to the number of jobs to be scheduled.

2.3 Objectives

The objectives of this research are:

1. to develop single machine irrigation management models, which allow non-contiguous (idle time between jobs is allowed) and contiguous (no idle time in between jobs) scheduling of the irrigation turns of a number of farmers, minimizing the differences between the actual start times and the requested start time;
2. to include sequence dependent setup times in the models;
3. to develop the models into multi-machine models, which allow both simple multi-machine scheduling (all outlets have an identical discharge) and complex multi-machine scheduling (discharges vary from outlet to outlet);
4. to include sequence dependent setup times in the multi-machine models;
5. to enhance the models such that they accommodate multi-period scheduling;
6. to determine the sensitivity of the models to the number of jobs to be scheduled.

A number of different models need to be developed in order to achieve the objectives mentioned. Table 2.1 shows an overview of these models and their main features. It can be seen in Table 2.1 that the models are divided into two main categories, single machine scheduling and multi-machine scheduling models. The 7 key features of the models, non-contiguous scheduling, contiguous scheduling, sequence dependent setup times (travel times), sequential irrigation, simultaneous irrigation, simple multi-machine scheduling and complex multi-machine scheduling, can be found as rows.

Table 2.1: Overview of different models and their features

(1)	Model (2)	Non- contiguous (3)	Contiguous (4)	Setup times (5)	Sequential irrigation (6)	Simultaneous irrigation (7)	Simple multi-machine (8)	Complex multi-machine (9)
Single machine models	1	✓			✓			
	2		✓		✓			
	3	✓		✓	✓			
	4		✓	✓	✓			
Multi-machine models	5	✓			✓	✓	✓	
	6		✓		✓	✓	✓	
	7	✓		✓	✓	✓	✓	
	8		✓	✓	✓	✓	✓	
	9	✓			✓	✓	✓	✓
	10		✓		✓	✓	✓	✓
	11	✓		✓	✓	✓	✓	✓
	12		✓	✓	✓	✓	✓	✓

2.4 Methodology

There is a large number of techniques available to solve optimisation problems. Perhaps the most well known are linear programming, integer programming, non-linear programming, heuristics and genetic algorithms. Winston (1994) characterised linear programming as follows:

1. a linear function of the decision variable is to be maximised or minimised. The function to be maximised or minimised is called the objective function,
2. the values of the variables must satisfy a set of constraints. Each must be a linear equation or linear inequality,
3. a sign restriction is associated with each variable. For any variable, the sign restriction specifies whether the variable is non-negative or unrestricted in sign.

Linear programming problems can be solved using the Simplex algorithm developed by George Dantzig in 1947.

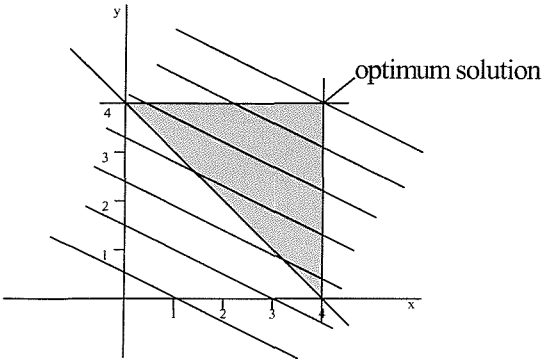
An integer programming problem, or IP, is a form of linear programming problem where some or all of the variable are required to be integers. An IP where all the variables are integers is called a pure IP and an IP where the variables must be equal to 0 or 1 is a 0-1 IP. Any integer programming problem where there is a combination of 0-1, integer and non-integer variables is called a mixed integer linear programming problem. IP's can be solved using a number of techniques, including LP relaxation, branch and bound methods and cutting plane algorithms (Winston, 1994).

Non-linear programming, which includes quadratic programming, is a term used to describe problems in which the objective function and/or one or more of the constraints are not linear. Quadratic programming has received considerable attention over the years. Townsend (1978) first formulated a single machine scheduling problem with a quadratic objective function and many have improved and extended the methods used for solving it (Alidaee et al., 1994; Della Croce et al., 1995; Mondal and Sen, 2000). However, although non-linear functions may better describe certain situations, the solution may not be the optimal solution (Winston, 1994). Figure 2.1 shows the graphical solution for a linear objective function and a quadratic objective function. It can be seen that for the linear function a global optimum is found. For the quadratic function three optima are found. Of these three only one is the global optimum, the other two are local optima. This example shows how for nonlinear functions solutions can be found that are not necessarily the optimal solution.

It can be proven that for linear and integer programming problems an optimal solution can be found. The disadvantage, especially of integer programming, is that the solution process is computationally very demanding. Many authors have solved the single machine scheduling problem by using linear/integer programming and have developed methods to cut down on computation time. Although great progress has been made, often other techniques are needed to solve larger problems.

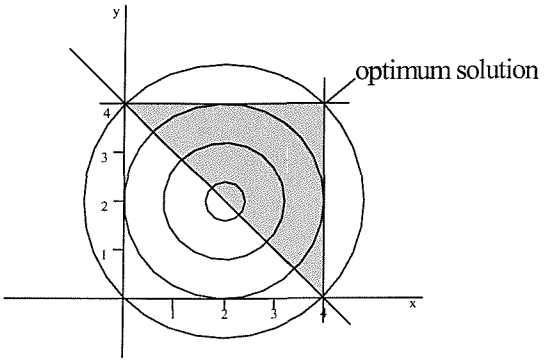
A heuristic is a search procedure, based on a number of criteria, methods and principles, that may give an optimal solution but does not guarantee to do so. The criteria, methods and principles can be rules of thumb, common sense rules drawn from experience, mathematically proven algorithms or a combination thereof.

A) contours of a linear objective function



Maximize $Z = 2x + y$
such that $x + y \geq 4$
$x \leq 4$
$y \leq 4$
$(x,y) \geq 0$

B) contours of a quadratic objective function



Maximize $Z = (x\text{-}2)^2 + (y\text{-}2)^2$
such that $x + y \geq 4$
$x \leq 4$
$y \leq 4$
$(x,y) \geq 0$

Figure 2.1 Contour lines for linear and quadratic objective functions

Genetic algorithms were developed by Holland (1975) to mimic processes observed in natural evolution. Davis (1991) described several of the general features of natural evolution as follows:

- evolution is a process that operates on chromosomes rather than on the living being that the chromosomes encode,
- natural selection is the link between chromosomes and the performance of their decoded structures. Processes of natural selection cause those chromosomes that encode successful structures to reproduce more often than those that do not,
- the process of reproduction is the point at which evolution takes place. Mutations may cause the chromosomes of biological children to be different from those of their biological parents, and recombination processes may create quite different chromosomes in the children by combining material from the chromosomes of two parents,
- biological evolution has no memory. Whatever it knows about producing individuals that will function well in their environment is contained in the gene pool - the set of chromosomes carried by the current individuals- and in the structure of the chromosome decoders.

Holland (1975) developed algorithms that manipulate a series of binary integers as if they were chromosomes. The chromosomes are manipulated, like in nature, without knowing anything about the problem to be solved and each new chromosome is evaluated for its

fitness. The result of that evaluation is used to bias the selection of chromosomes, so that a better evaluation gives a higher chance of survival.

Although heuristics and genetic algorithms can solve large problems, there is no guarantee that the obtained solution is the optimal solution or even near it. This does not need to be an issue if the problem to be solved has already been studied and benchmark solution are available to be able to compare quality. If a new kind of problem is studied it may be better to first use techniques such as linear and integer programming which provide the optimal solution, albeit for smaller problems. This allows for greater understanding of the problem and can set a series of bench marks against which the quality of the heuristics and genetic algorithms can be tested. As the study of irrigation scheduling interpreted as a machine scheduling problem is new, it is believed that in the first instance linear and integer programming are more appropriate optimisation techniques than heuristics and genetic algorithms.

2.5 Data mining and data generation

In the development of the different proposed models there will be a need for a small amount of testing, firstly to check the models actually works and secondly to be able to compare the models with each other. To be able to run any of the models one or more data sets are needed. Rardin and Uzsoy (2001) describe the following types of data sets:

1. Real world data sets
2. Random variations of real data sets
3. Published and online libraries of data sets
4. Randomly generated data sets

Real world data sets are thought to be the best, but very difficult to obtain and there may be only a limited amount of sets. Data on real irrigation systems is widely available, but most systems are not suitable for the purpose of this research because they are not operated under a rotational schedule, they are too big to be of practical use or essential information is lacking. Since no complete sets of real data are available, Option 2 where part of the data set is kept constant and part is varied randomly is also not suitable for this research. There is no known library of the necessary information, so Option 3 is not viable either. Therefore data needs to be randomly generated. However it is possible to use data from

real irrigation systems where available. All models described in this thesis need some or all of the following data:

- irrigation duration,
- irrigation interval,
- travel time between the field outlets,
- target start time.

A number of papers describe tertiary units under a rotational schedule containing useful information pertinent to this research. Khepar et al. (2000) described a tertiary unit in India with 89 users. Not all of the outlets in this tertiary unit have been fully described. Sharma and Oad (1990) described two tertiary units, one in India with 32 users and one in Pakistan with 60 users. The latter has no details on the field outlets, but for the former the following data is available: distance between field outlets, landholding size, irrigation duration, and volume of water delivered to each outlet. Latif and Sarwar (1994) described a tertiary unit in Pakistan with 20 users. The following data is available for this tertiary unit: distance from the canal outlet to field outlets, volume of water delivered and irrigation time. Bishop and Long (1983) described a tertiary unit in the Phillippines with 16 users. The following information is available for this tertiary unit: land holding size, irrigation duration, channel length, velocity in channel sections and irrigation interval. The information on the unit described by Bishop and Long (1983) is most complete, irrigation duration and irrigation interval are given and the travel times between field outlets can be determined. As this tertiary unit is irrigated under a 'bottom-up' rotation, no target start times are given. All tertiary units described here operate under a rotational schedule. Suryavanshi and Reddy (1986) described a tertiary unit in India with 8 users who are allowed to irrigate simultaneously. The following information is available for this tertiary unit: land holding size, irrigation duration, channel length, irrigation interval and discharge per outlet.

In a real world irrigation system the target start times can be expected to have a certain distribution. A large number of factors influences this distribution: users may prefer to irrigate during the day and rather not during their rest days, there may be some sort of incentive for night time irrigation and the weekly market day may not be very popular for irrigation. As far as known there has been no research into the actual distribution of target start times. As a study of the distribution of target start times is beyond the scope of this research, the target start times are assumed to be uniformly distributed along the irrigation interval.

It has been widely reported that solution times of integer programmes increase substantially with the size of the problem, e.g. Bard et al. (1993) and Fry et al. (1987). To test the sensitivity of the solution times of the models to the number of jobs to be scheduled and the data sets, a small computational experiment is used. In this experiment for an increasing number of jobs a number of different data sets are generated. The procedure for generating these data sets has been described by Anwar and de Vries (2003). This paper is included in Appendix C.

2.6 Summary

The models described in Table 2.1 first need to be defined by selecting appropriate variables and describing the objective function and constraints. Once they have been mathematically formulated mixed integer linear programming can be used to solve the model. Several different software programmes, such as Lingo, XPress^{MP} and GAMS are available for solving integer programmes. No preference exists for any of these programmes. Lingo 6 for Windows will be used as a general purpose solver as a licensed copy was made available. Initially data will be randomly generated to test the models. To apply the models to an existing situation, an irrigation system will be selected from available literature. Where necessary existing data will be supplemented by randomly generated data.

3 Irrigation as single machine scheduling[1]

An analogy can be drawn between scheduling the irrigation turns of a number of users in a tertiary unit of an irrigation systems and the machine scheduling problem from the field of Operations Research. There is a single activity (delivering water to field outlets) and there are a number of jobs (users requesting water for an outlet) to be processed. If a irrigation scheme operates under an rotational schedule, i.e. users irrigate sequentially, the water can be seen as a single machine that 'processes' the outlets (jobs) one by one. Each job has a certain duration, a target start time and there can be earliness/tardiness costs (the costs associated with delivering water before or after the target start time) for each job. These earliness/tardiness costs will become particularly important when scheduling over more than one irrigation interval and will be further discussed in Chapter 7.

Santhi and Pundarikanthan (2000) suggested that irrigation optimisation models do not take into account managerial tasks such as the minimisation of gate operations. Two models reflect different management options at the tertiary level. In the first model all jobs are scheduled non-contiguously, i.e. idle time is allowed to be inserted between jobs. In the second model all jobs are scheduled contiguously, no idle time is allowed between jobs. Although contiguous scheduling can reduce water spillage and gate operations, non-contiguous scheduling allows better matching of target start times and scheduled start times.

If the sum of the durations (commonly referred to as makespan in Operations Research) is less than the length of the interval, there are three possible variations on scheduling all jobs contiguously. Figure 3.1 shows these variations. Figure 3.1a shows all jobs scheduled such that the first job starts at the beginning of the interval and any idle time is inserted after the last job in the schedule has been completed. Figure 3.1b shows how all jobs are scheduled such that the last job is completed at the end of the interval and any idle time is inserted at the beginning of the interval. Figure 3.1c is an intermediate form where the first job is scheduled to start sometime after the start of the interval and the last job is scheduled to finished sometime before the end of the interval.

---

[1] Parts of this chapter and Chapter 7 have been written up as two companion papers (de Vries and Anwar, 2003; and Anwar and de Vries, 2003) . These papers have been accepted for publication by the ASCE Journal of Irrigation and Drainage Engineering. Copies of the papers can be found in Appendix B and C.

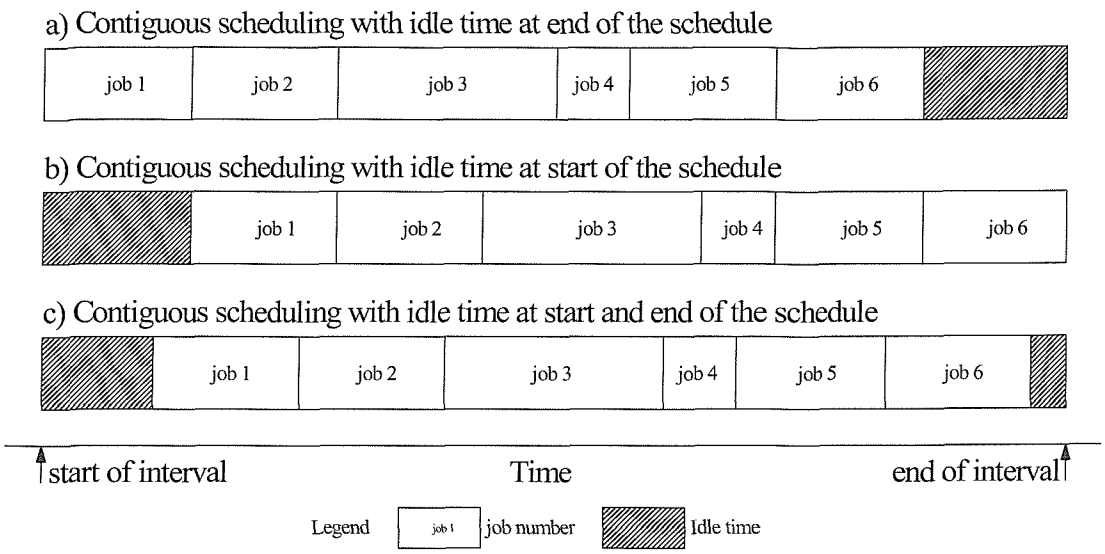


Figure 3.1 Three alternative methods for contiguous scheduling

In this chapter the models for non-contiguous and contiguous single machine scheduling are developed. An alternative formulation for these models is also presented. These alternative models can reduce the number of constraints and variables (and therefore solution times), but have a limited use. A computational experiment will show the effect of the number of jobs on the solution times. The highlighted area of Table 3.1 shows an overview of the features of the models that are developed in this chapter.

Table 3.1: Overview of different models and their features

	Model		Contiguous							
		Non- contiguous	idle time after all jobs	idle time before all jobs	idle time before and/or after all jobs	Setup times	Sequential irrigation	Simultaneous irrigation	Simple multi-machine	Complex multi-machine
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)
Single machine model	1	✓					✓			
	2a		✓				✓			
	2b			✓			✓			
	2c				✓		✓			
	3	✓				✓	✓			
	4		✓	✓	✓	✓	✓			

3.1 Analysis and development of non-contiguous single machine model

The non-contiguous single machine model, herein referred to as Model 1, allows scheduling of a number of jobs (user requests) according to their target start times. Let $\boldsymbol{Q}$ = {1,2,...,N} be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: duration, target start time, earliness cost and tardiness cost. In Model 1 there is one decision to be made, what is the scheduled start time of each job. If the answer to this question is known, then the schedule is known too. The objective of the model is to minimize the difference between target start time and scheduled start time. In single machine scheduling the equivalent problem is called the earliness/tardiness or E/T problem. Liaw (1999) suggested the following formulation for objective function of the single machine earliness/tardiness problem

$$Z = \min \sum_{i=1}^{N} (\alpha_i E_i + \beta_i T_i) \tag{3.1}$$

where Z = objective function; α_i = cost of earliness per unit of time for job i; E_i = earliness of job i; β_i = cost of tardiness per unit of time for job i; T_i = tardiness of job i; i = index representing job 1,2,...,N; and N = number of jobs to be scheduled. As any job can precede any other job in the schedule a variable is used to define which job precedes which

$$\begin{aligned} \delta_{ij} &= 1 \quad \text{if job } i \text{ precedes job } j \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{3.2}$$

where δ_{ij} = binary variable; and j = index representing job 1,2,...,N. It is not possible for a job to incur a negative earliness or tardiness (in fact a negative earliness is a positive tardiness and vice versa). Therefore for T_i and E_i the following constraints need to be satisfied

$$T_i \geq 0 \qquad \forall i = 1,2,...,N \tag{3.3}$$

$$E_i \geq 0 \qquad \forall i = 1,2,...,N \tag{3.4}$$

Liaw (1999) suggested the following constraint to determine the scheduled completion time of a job

$$C_i = q_i - E_i + T_i \qquad \forall i = 1,2,...,N \tag{3.5}$$

where C_i = scheduled completion time of job i; and q_i = due date of job i.

Under arranged delivery schedules users request a time when irrigation delivery is to begin (Clemmens, 1987). Therefore for irrigation scheduling it is more sensible to express constraint (3.5) in terms of target start time and scheduled start time rather then due date and completion time. Constraint (3.5) now becomes

$$S_i = r_i - E_i + T_i \qquad \forall i = 1,2,...,N \tag{3.6}$$

where S_i = scheduled start time of job i; and r_i = target start time of job i.

The final constraints described by Liaw (1999) are the capacity and availability constraints of the machine. In irrigation scheduling the water in a channel is considered to be the machine and since in a rotational system irrigation turns are taken sequentially, jobs must not be executed simultaneously. This implies that if job i precedes job j the following inequality must hold

$$S_j - S_i \geq d_i \qquad \forall i = 1,2,...,N;\ \forall j = 1,2,...,N;\ i \neq j \tag{3.7}$$

where S_j = scheduled start time of job j; and d_i = duration of job i. Similarly if job j precedes job i the following must be true

$$S_i - S_j \geq d_j \qquad \forall i = 1,2,...,N;\ \forall j = 1,2,...,N;\ i \neq j \tag{3.8}$$

where d_j = duration of job j. Constraints (3.7) and (3.8) are mutually exclusive and are called disjunctive constraints. Baker (1974) described how to accommodate these constraints by using the precedence variable described in (3.2). Constraint (3.7) now becomes

$$S_j - S_i + M(1 - \delta_{ij}) \geq d_i \qquad \forall i = 1,2,...,N;\ \forall j = 1,2,...,N;\ i \neq j \tag{3.9}$$

where M = any large positive number. Constraint (3.8) becomes

$$S_i - S_j + M\delta_{ij} \geq d_j \qquad \forall i = 1,2,...,N;\ \forall j = 1,2,...,N;\ i \neq j \tag{3.10}$$

Model 1 as it stands now allows scheduling of a number of users according to their target start time. The model does not take into account that in irrigation there is a finite period, the irrigation interval, within which all users have to start and finish their irrigation turn. Another constraint is needed to ensure all jobs are processed within the irrigation interval.

$$S_i + d_i \leq g \qquad \forall i = 1,2,...,N \tag{3.11}$$

where g = irrigation interval over which all jobs must be completed. Therefore Model 1 is defined by the objective function (3.1) and constraints (3.2), (3.3), (3.4), (3.6), (3.9), (3.10) and (3.11).

3.2 Analysis and development of contiguous scheduling model

The contiguous scheduling model, herein referred to as Model 2, is similar to Model 1 in that it allows scheduling of a number of jobs according to their target start times. The main difference is that to reduce water spillage and/or gate operations idle time between jobs is not allowed. The objective function and most of the constraints of Model 1 remain valid.

3.2.1 Contiguous single machine scheduling: Model 2a

To ensure idle time is only inserted after the last job has been processed, no job may finish later than the sum of all durations. Therefore constraint (3.11) of Model 1 must be replaced by

$$S_i + d_i \leq \sum_{n=1}^{N} d_n \qquad \forall i = 1,2,...,N \tag{3.12}$$

where d_n = duration of job n; and n = index representing job 1,2,..., N. Model 2a is therefore defined by the objective function (3.1) and constraints (3.2), (3.3), (3.4), (3.6), (3.9), (3.10) and (3.12).

3.2.2 Contiguous single machine scheduling: Model 2b

In Model 2b all job are scheduled contiguously and idle time is inserted prior to the start of the first job. Constraint (3.11) becomes

$$S_i \geq g - \sum_{n=1}^{N} d_n \qquad \forall i = 1,2,...,N \tag{3.13}$$

No job may finish outside the irrigation interval, therefore constraint (3.11) is also necessary. Model 2b is defined by the objective function (3.1) and constraints (3.2), (3.3), (3.4), (3.6), (3.9), (3.10), (3.11) and (3.13).

3.2.3 Contiguous single machine scheduling: Model 2c

In Model 2c idle time may precede the start of the first job and/or follow the end of the last job, therefore (3.11) is altered to

$$g = \sum_{i=1}^{N} d_i + X_a + X_b \tag{3.14}$$

where X_a = idle time preceding the start of the first job in the schedule; and X_b = idle time following the completion of the last job in the schedule. The following additional constraints also need to be included for Model 2c

$$S_i + d_i \leq g - X_b \qquad \forall i = 1,2,...,N \tag{3.15}$$

$$S_i \geq X_a \qquad \forall i = 1,2,...,N \tag{3.16}$$

Model 2c is therefore defined by the objective function (3.1) and constraints (3.2), (3.3), (3.4), (3.6), (3.9), (3.10), (3.14), (3.15) and (3.16).

3.3 Results and discussion

In Models 1, 2a, 2b, and 2c setup times are not considered. This is true for pressurised irrigation systems. In systems where the distance between the outlets is very small and/or the velocity in the channels very high, setup times will be very small compared to the irrigation duration and can be ignored. No information is available on such irrigation systems where setup times play no role. To demonstrate the principles of Models 1, 2a, 2b and 2c they will be applied to a (open channel) tertiary unit assuming that setup times are part of the irrigation duration.

Bishop and Long (1983) presented a procedure for setting up a rotation delivery schedule and applied it to a tertiary unit of 37.06 ha with 16 water users. Water in this tertiary unit was allocated pro rata with area at 172 min/ha and 349 min were made available for management/canal filling. This management/filling time consists of 209 minutes need to fill the channels and another 140 minutes for necessary management tasks such as adjusting of checks and gate closing. Bishop and Long (1983) suggested the irrigation to be scheduled in turn from downstream upwards, target start times were not included in their data. For the purpose of this application of the models developed the irrigation interval is assumed as 7 days (10080 minutes) and the target start times are randomly generated with a uniform distribution over the irrigation interval. The duration of each job is 172 min/ha as suggested by Bishop and Long (1983). Management/fill time is equally divided amongst all users and is added to the duration. Figure 3.2 shows a overview of the tertiary unit and Table 3.2 shows the job descriptors.

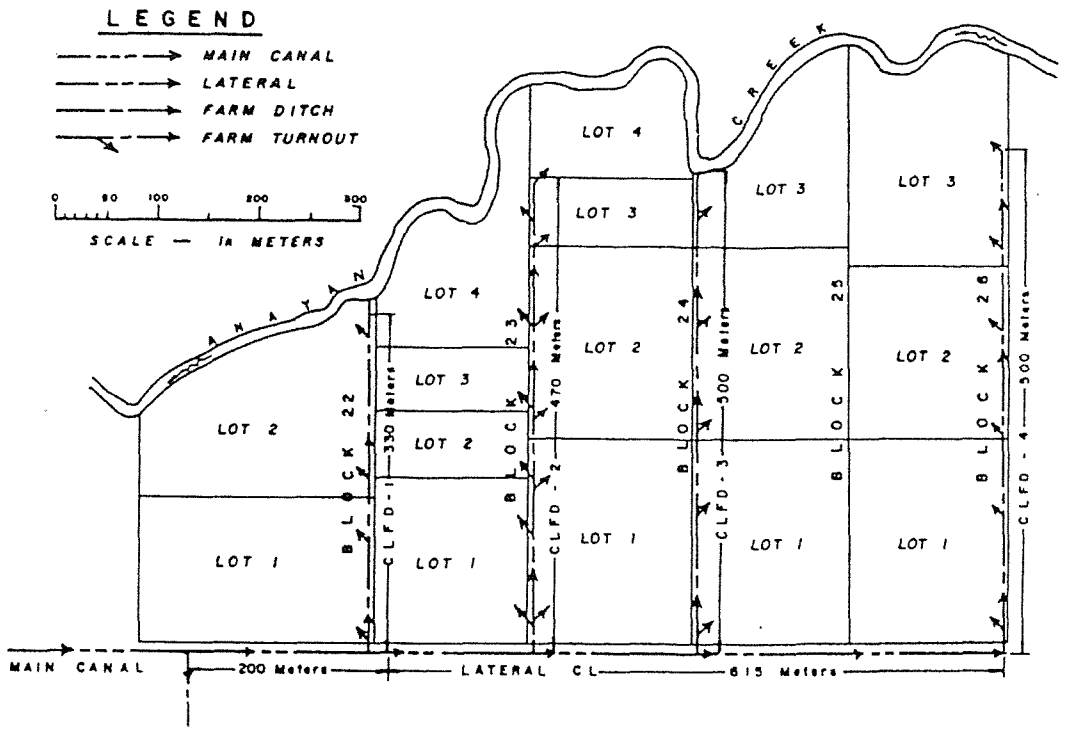


Figure 3.2 Tertiary unit, Bula project, the Phillippines (Bishop and Long, 1983)

Table 3.2: Irrigation duration and target start times for tertiary unit Bula project

Job number	Lot number[1]	Irrigation duration (minutes)	Target start times (minutes)
(1)	(2)	(3)	(4)
1	26.3	461	2140
2	26.2	451	9783
3	26.1	537	8009
4	25.3	387	8889
5	25.2	537	4841
6	25.1	537	771
7	24.4	202	3362
8	24.3	193	5635
9	23.4	360	5408
10	23.3	193	5699
11	24.2	537	1365
12	23.2	193	1503
13	24.1	537	19
14	23.1	451	9515
15	22.2	572	1206
16	22.1	572	990

[1] From Bishop and Long (1983)

Models 1, 2a, 2b and 2c were implemented in Lingo 6.0® for Windows® using data from Table 3.2 . The Lingo input files are included in Appendix A. Figure 3.3 shows a graphical representation of the schedules obtained when applying the various models.

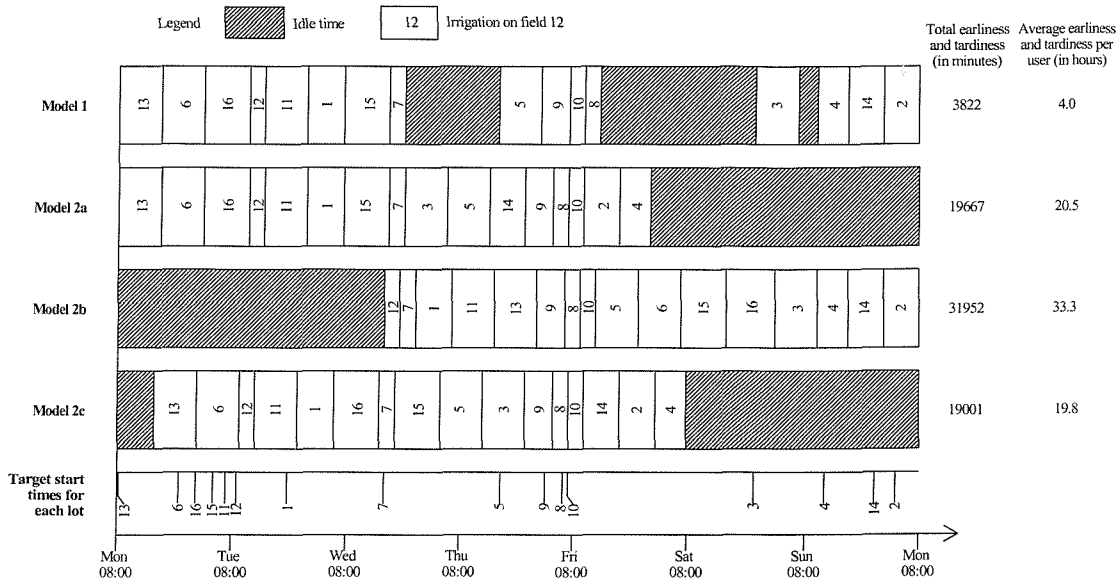


Figure 3.3 Schedules applying Models 1, 2a, 2b and 2c

For Model 1 idle time is inserted in three places in the schedule; after Lot 7 is irrigated, after Lot 8 is irrigated and again after Lot 3 is irrigated. The schedule obtained with Model 1 results in an average earliness/tardiness of 4.0 hrs/user. Figure 3.3 shows that for Model 2a all idle time is inserted after all users have finished irrigating. This schedule results in an average earliness/tardiness of 20.5 hrs/user. For Model 2b all idle time is inserted before any of the users starts irrigation. The average earliness/tardiness of this schedule is 33.3 hrs/user. Finally, for Model 2c 7.7 hours of idle time is inserted before the first user (Lot 13) and another 48.3 hours is inserted after the last user (Lot 4) has finished irrigation. This schedule results in an average earliness/tardiness of 19.8 hrs/user.

Although Model 1 gives the best results in terms of average earliness/tardiness, this model can lead either to operational spillage or an excessive number of gate operations. If operational spillage is to be avoided extra gate operations are required. It can be seen in Figure 3.3 that for this schedule three extra operations are needed to close the gate after irrigation on Lots 7, 8 and 3 have been completed and another extra three operation are needed to reopen the gate before irrigation on Lots 5, 3 and 4 can start. The number of extra gate operations depend on the number of jobs and the schedule, but can theoretically range from 2 to $2N$-2 (where N = number of jobs to be scheduled). If like in Models 2a, 2b

and 2c the gate can only be opened once, before irrigation on the first lot starts and closed once, after the last irrigation has been completed, 33% of the water will be lost due to operational spillage.

Model 2b gives the highest average earliness/tardiness. This is due to the concentration of target start times early in the interval. Model 2a and 2c perform better than Model 2b, with only a small difference between them. It is incidental that Model 2a performs better than Model 2b. This is due to the large concentration of target start times early in the interval. Model 2c can always be expected to perform at least as good as Model 2a and 2b and will often perform better. This is due to the restrictions in where jobs can be placed in Model 2a and 2b. Models 2a, 2b and 2c do not perform as well as Model 1, as Model 1 has the largest amount of freedom in where to place jobs and therefore gives the best result. However gate operations for Models 2a, 2b, and 2c are limited to opening once and closing once and spillage can be easily avoided.

Figure ?.? shows a second set of schedules obtained when applying Models 1, 2a, 2b and 2c. New target start times have been randomly generated, irrigation durations are the same as for the set of schedules described earlier in this section.

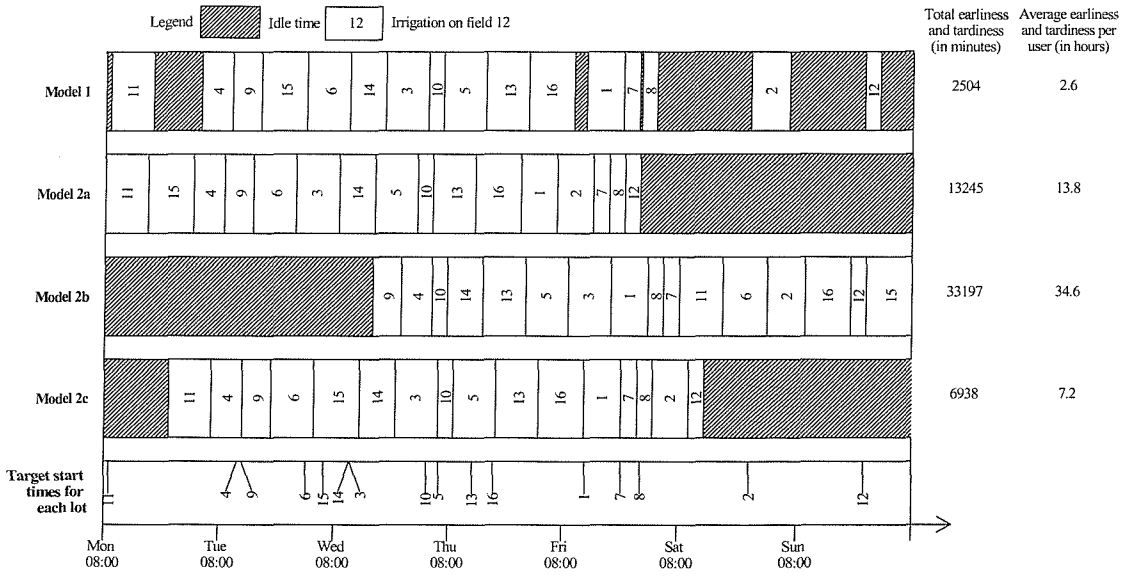


Figure 3.4 Second set of schedules applying Models 1, 2a, 2b and 2c

It can be seen in Figure ?.? that due to the larger degree of freedom Model 1 again performs better than any of the other models. As mentioned before, Model 2c can always be expected to perform at least as good as Model 2a and 2b. It can be seen in Figure ?.? that Model 2c performs considerably better than either of those two. This is due to the more even distribution of the target start times over the interval. There is still a larger

concentration of target start times in the beginning of the interval, which is why Model 2a outperforms Model 2b. It can be seen that the performance of Models 2a, 2b and to a lesser degree Model 2c, depends on the distribution of the target start times over the interval. When target start times are concentrated in the beginning of the interval Model 2a will perform better and when start times are more concentrated towards the end of the interval Model 2b will perform better.

3.4 Analysis and development of an alternative formulation

The models developed in the previous paragraph of this chapter use a 0-1 variable to define which job precedes which in a schedule. Fry and Leong (1987) formulated an mixed integer program for the E/T problem in which a 0-1 variable defines the location of the jobs in the schedule, i.e whether a job is the first in a schedule or second and so on. Again let $\boldsymbol{Q} = \{1,2,...,N\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: duration, target start time, earliness cost and tardiness cost. Although the models recognise that there can be a difference in the costs between a job being scheduled early or late, they do not allow different costs of earliness and tardiness for each individual job. This limits the use of this model to single interval scheduling as will be further explained in Chapter 7. A detailed description of the objective function and the constraints as Fry and Leong (1987) used them and how they can be adapted so as to be suitable for irrigation scheduling follows.

3.4.1 Alternative formulation of Model 1

The objective defined by Fry and Leong (1987) was to minimize total earliness and flow time rather than to minimize earliness and tardiness as in irrigation scheduling. The objective function however is only a performance measure, it does not influence the restrictions of the system behind it. That means that the constraints and variables from the model from Fry and Leong (1987) can be used together with an objective function that minimizes earliness and tardiness. The objective function for this formulation is

$$\text{minimize} \quad Z = \alpha \sum_{k=1}^{N} E_k + \beta \sum_{k=1}^{N} T_k \tag{3.17}$$

where α = earliness penalty cost per unit of time; E_k = earliness of the job in position k; β = tardiness cost per unit of time; T_k = tardiness of the job in position k; and k = index representing position 1,2,...,N. Any job, in this case an irrigation event, can be assigned to any position in the schedule, therefore the following variable is defined

$$\lambda_{ik} = \begin{array}{ll} 1 & \text{if job } i \text{ is assigned to position } k \text{ in the schedule} \\ 0 & \text{otherwise} \end{array} \tag{3.18}$$

where λ_{ik} = binary variable. Fry and Leong (1987) described a number of constraints, some of which are directly transferable to the irrigation situation, some of which need modification. The first constraint is that each job can only once be assigned to a position

$$\sum_{i=1}^{N} \lambda_{ik} = 1 \qquad \forall k = 1,2,...,N \tag{3.19}$$

Also each position can only be assigned once

$$\sum_{k=1}^{N} \lambda_{ik} = 1 \qquad \forall i = 1,2,...,N \tag{3.20}$$

The due date constraint described by Fry and Leong (1987) is as follows

$$C_k + E_k - T_k = q_k \qquad \forall k = 1,2,...,N \tag{3.21}$$

where C_k = scheduled completion time of the job in position k; and q_k = due date of the job in position k. Like for Model 1, completion time is of no interest, but start time is. So (3.21) becomes

$$S_k = r_k + T_k - E_k \qquad \forall k = 1,2,...,N \tag{3.22}$$

where S_k = scheduled start time of the job in position k; and r_k = target start time of the job

in position k.

Fry and Leong (1987) introduced a variable for idle time so that

$$C_k = \sum_{m=1}^{k} X_m + \sum_{m=1}^{k} p_m \qquad \forall k = 1,2,...,N \tag{3.23}$$

where X_m = idle time inserted directly before the job in position m; m = index representing position 1,2,...,k; and p_m = processing time of the job in position m and can be calculated with

$$p_m = \sum_{i=1}^{N} \lambda_{im} p_i \qquad \forall m = 1,2,...,k;\ k = 1,2,...,N \tag{3.24}$$

where λ_{im} = binary variable; and p_i = processing time of job i. Converting to start times (3.23) becomes

$$S_k = \sum_{m=1}^{k} X_m + \sum_{m=1}^{k-1} d_m \qquad \forall k = 1,2,...,N \tag{3.25}$$

where d_m = duration of the job in position m. Equality (3.24) becomes

$$d_m = \sum_{i=1}^{N} \lambda_{im} d_i \qquad \forall m = 1,2,...,k;\ k = 1,2,...,N \tag{3.26}$$

Also according to Fry and Leong (1987)

$$q_k = \sum_{i=1}^{N} \lambda_{ik} q_i \qquad \forall k = 1,2,...,N \tag{3.27}$$

Likewise for start times

$$r_k = \sum_{i=1}^{N} \lambda_{ik} r_i \qquad \forall k = 1,2,...,N \tag{3.28}$$

Fry and Leong (1987) used (3.23), (3.24) and (3.27) to rewrite (3.21) as

$$\sum_{m=1}^{k} X_m + \sum_{m=1}^{k}\sum_{i=1}^{N} \lambda_{im} p_i + E_k - T_k = \sum_{i=1}^{N} \lambda_{ik} q_i \qquad \forall k = 1,2,...,N \tag{3.29}$$

Similarly using (3.25), (3.26) and (3.28), (3.22) can be rewritten as

$$\sum_{m=1}^{k} X_m + \sum_{m=1}^{k-1}\sum_{i=1}^{N} \lambda_{im} d_i + E_k - T_k = \sum_{i=1}^{N} \lambda_{ik} r_i \qquad \forall k = 1,2,...,N \tag{3.30}$$

No job may be completed outside the irrigation interval. Therefore

$$\sum_{k=1}^{N} (X_k + d_k) \leq g \tag{3.31}$$

where X_k = idle time inserted directly before the job in position k; and d_k = duration of the job in position k. Therefore the alternative formulation to Model 1 based on the work by Fry and Leong (1987) is defined by the objective function (3.17) and constraints (3.18), (3.19), (3.20), (3.30) and (3.31).

3.4.2 Alternative formulation of Model 2a

In Model 2a all jobs are scheduled contiguously with idle time inserted at the end of the interval, and (3.25) is replaced by

$$S_k = \sum_{m=1}^{k-1} d_m \qquad \forall k = 1,2,...,N \tag{3.32}$$

from (3.26), (3.28) and (3.32), (3.22) becomes

$$\sum_{m=1}^{k-1}\sum_{i=1}^{N} \lambda_{im} d_i = \sum_{i=1}^{N} \lambda_{ik} r_i - E_k + T_k \qquad \forall k = 1,2,...,N \tag{3.33}$$

Model 2a now consists of the objective function (3.1) and constraints (3.18), (3.19), (3.20) and (3.33).

3.4.3 Alternative formulation of Model 2b

To ensure idle time is inserted at the beginning of an interval, as is the case for Model 2b, constraint (3.22) is replaced by

$$\sum_{m=1}^{k-1}\sum_{i=1}^{N}\lambda_{im}d_i = \sum_{i=1}^{N}\lambda_{ik}r_i - E_k + T_k - X_a \qquad \forall k = 1,2,...,N \tag{3.34}$$

The following additional constraint ensures any idle time is inserted before the first job in the schedule starts

$$\sum_{k=1}^{N} d_k = g - X_a \tag{3.35}$$

Model 2b now consists of the objective function (3.17) and constraints (3.18), (3.19), (3.20), (3.34) and (3.35).

3.4.4 Alternative formulation of Model 2c

To schedule a number of jobs so that the first job is scheduled to start some time after the start of the interval and the last job is scheduled to finish sometime before the end of the interval (Model 2c), the constraints for Model 2b are used with equality (3.35) omitted.

3.4.5 Comparison of first and alternative formulation

The first and the alternative formulation both aim at scheduling all jobs such that the differences between the target start times and the scheduled start times are minimised. Both formulation use mixed integer linear programming to find a solution. As linear programming will always result in the optimum solution and both formulations have the same objective, the results will always be the same. There is however a difference in the way both are formulated, which is reflected in the number of variables and constraints

necessary for each formulation. Table 3.3 shows the number of variables and constraints for each model with both formulations.

Table 3.3: Number of variables and constraints for all models

Model (1)	Formulation (2)	Number of variables (3)	Number of constraints (4)
1	First	N^2+3N	$2N^2+1$
	Alternative	N^2+3N	$3N+2$
2a	First	N^2+3N	$2N^2+1$
	Alternative	N^2+2N	$3N+1$
2b	First	N^2+3N+1	$2N^2+N+2$
	Alternative	N^2+2N	$3N+1$
2c	First	N^2+3N+3	N^2+2N+1
	Alternative	N^2+2N+1	$3N+1$

Table 3.3 shows the number of variables for all models is in the order of N^2. The difference in number of constraints between the two formulations explains why the alternative formulation is less computationally demanding: for the first formulation the number of constraints is in the order of N^2, but for the alternative it is only in the order of N. Figures 3.4 and 3.5 stress this difference. Figure 3.4 shows that the number of variables for Model 1 are the same for both formulations, even with an increasing number of jobs. Figure 3.5 shows that the number of constraints for the first formulation rises exponentially with the number of jobs, whereas the number of constraints for the alternative formulation rises only linearly.

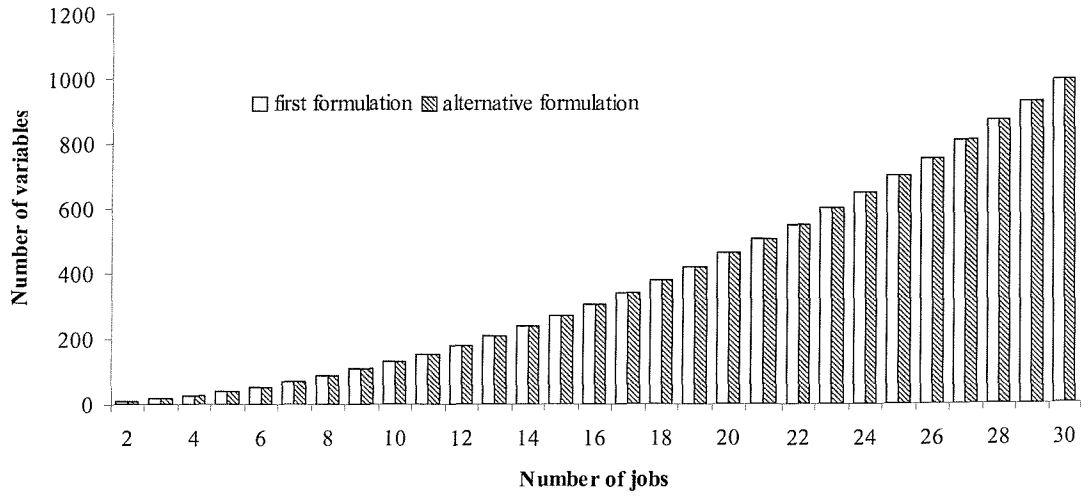


Figure 3.5 Number of variables Model 1

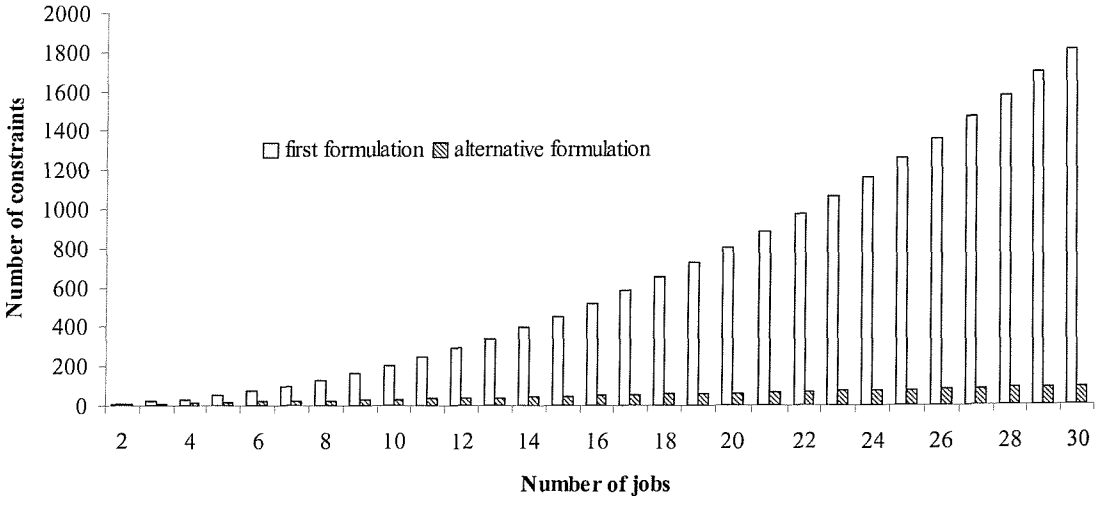


Figure 3.6 Number of constraints Model 1

However not just the number of variables and constraints influence computation time. Table 3.4 shows computation times for the problem with 16 jobs described in Section 3.3.

Table 3.4: Computation times (in secs) for Model 1, 2a, 2b, 2c

Formulation	Model			
	1 (1)	2a (2)	2b (3)	2c (4)
First	185	>20000	>20000	>20000
Alternative	88	12074	19	8286

It can be seen that there is a considerable variation in computation time between the four models, even where there are a comparable number of variables and constraints. Comparing Model 2a and 2b of the alternative formulation shows that even for two formulations with exactly the same number of variables and constraints, using the same data set there can be a large difference in solution times. This show that solution times are sensitive to the formulation. Although from the comparison it appears the alternative formulation is computationally more efficient and would allow larger problems to be solved, there is a distinct disadvantage to this formulation. The alternative formulation does not allow for different costs of earliness and tardiness for each job. As will be shown in Chapter 7 these earliness and tardiness costs play an important role in multi-interval scheduling. The use of the alternative formulation is therefore limited to those cases where earliness/tardiness costs play no part.

3.5 Solution times

A computational experiment was conducted to examine the effect of the number of jobs on the solution time. 25 instances (data sets) each of problems with 8, 10, 12, 15, 20 and 25 jobs and 15 instances each of problems with 30 and 35 jobs were generated according to the method described by Anwar and de Vries (2003) (this paper is included in Appendix C and contains a more detailed description of the experiment). Model 1 was implemented in Lingo 6.0 ® for Windows® using the generated data sets. For the purpose of this experiment the solver was interrupted after 10^5 seconds (approximately 27.7 hours). The integer programme did not solve to a global optimum within this allocated time for 4 out of the 25 instances with 25 jobs. Similarly 3 out of 15 instances with 30 jobs and 10 out of 15 instances for 35 jobs did not solve to completion within this allocated time. Figure 3.6 shows that the solution times increase several order of magnitude with the number of jobs. Figure 3.6 shows that not only the number of jobs has an influence on the solution time, it also show that solution times vary for instances with the same formulation and number of jobs, but with different data sets. It can be seen that solving Model 1 for 20 jobs can take as little as 10 seconds or as much as 70000 seconds (~ 19.5 hours). This shows that solution times are particularly data sensitive. The exponential increase of the solution time with the number of jobs means that only smaller problems can be solved with a reasonable

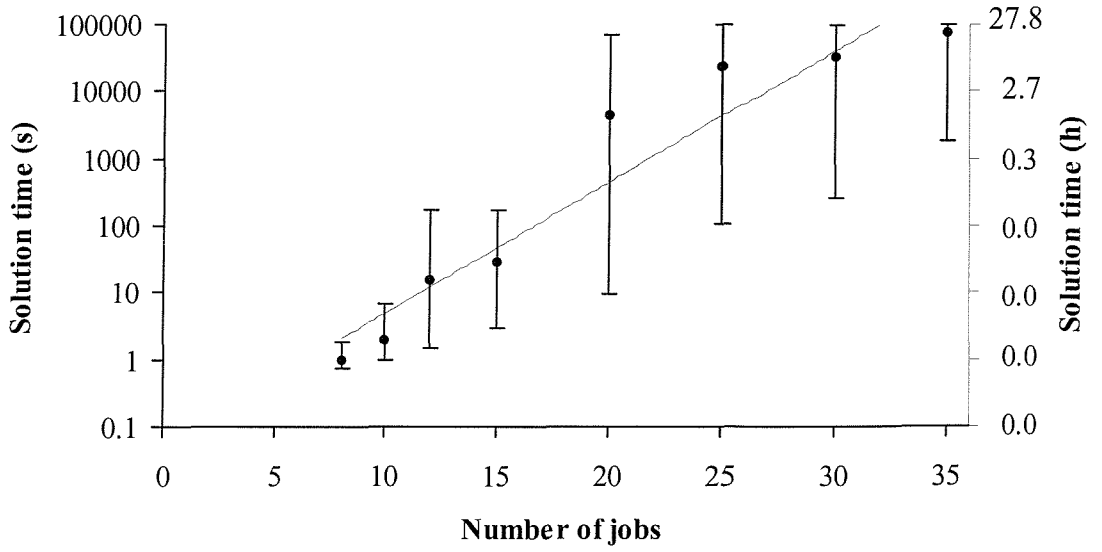


Figure 3.7 Solution times for Model 1

time. As many tertiary unit have large numbers of outlets that would need to be scheduled, different solution techniques may be needed to obtain a schedule.

The computational experiment in this section consists of a total of 150 instances of Model 1. The combined solution time of these instances is 633 hours. The other models in this chapter and the models that are developed in subsequent chapters are of an increasing complexity with more variables and constraints. Zhu and Heady (2000) reported increasing solution times with increasing complexity. It is very likely that the same will be true for the models presented here and in other chapters and it is therefore not feasible to do similar experiments with any of the other models within the time allotted for this research.

3.6 Conclusions single machine scheduling

In this chapter it is shown how irrigation scheduling at a tertiary level can be interpreted as a single machine problem. It is shown how existing single machine scheduling models for the earliness/tardiness problem can be adapted to allow irrigation scheduling. By applying the models to a tertiary unit it is shown how different management options such as non-contiguous and contiguous scheduling have an influence on the schedule and the earliness/tardiness, gate operation or operational spillage. An alternative formulation to the

models is presented and compared to the original formulation. Although solution times can decrease when applying the alternative formulations, there are limitations to its use. There are some disadvantages to the models described in this chapter:

1. Setup times are assumed to be non-existent. Although this may be true for pressurised systems, many systems irrigation systems have open channels where setup times can have a large influence on the schedule. This problem will be addressed in the next chapter, Chapter 4.
2. The models only apply to irrigation systems that operate under sequential schedule. Many irrigation systems however allow simultaneous irrigation. This problem will be addressed in Chapters 5 and 6.
3. Integer programming can be computationally very demanding and only small problems can be solved within reasonable time. Solution techniques such as heuristics or genetic algorithms may be needed to solve larger problems.
4. It is assumed the sum of the durations of all users is less then the irrigation interval. If users are allowed to request any irrigation duration, the sum of durations may become larger than the irrigation interval. How to adjust the irrigation durations such that the sum is not larger than the interval needs to be studied in future. Possible solutions are adjusting the durations pro rata or imposing an upper limit to the duration.
5. When an upstream user takes over from a down stream user, some residual volume of water is left in the channel which could be used by the down stream user. Further study is needed to study the effect of this residual volume on the irrigation duration, e.g. should the irrigation duration be reduced to account for this extra water.

4 Sequence dependent setup times

Figure ?.? shows that setup times can play an important role in open channel irrigation systems. There are two types of setup times, those which on which the sequence of the job has no influence, the sequence independent setup times, and those on which the order has an influence, the sequence dependent setup times. Sequence independent setup times would include time needed for a user to close a farm outlet and the next user to open the following outlet. This time is negligible compared to the irrigation duration and can therefore be ignored. Travel times on the other hand are, as shown in Figure ?.?, dependent on the order of irrigation and can only be ignored if they are very small compared to the irrigation duration. This is the case if the distances between outlets are very small and/or the velocity in the channel is large. However in most open channel irrigation systems neither is the case and therefore sequence dependent setup times must be taken into account.

In this chapter the models for non-contiguous and contiguous single machine scheduling with sequence dependent setup times are developed. By applying the models to a tertiary unit the significance of including sequence dependent setup times is shown. The highlighted area of Table 4.1 shows an overview of the features of the models that will be developed in this chapter.

Table 4.1: Overview of different models and their features

	Model		Contiguous							
		Non- contiguous	idle time after all jobs	idle time before all jobs	idle time before and/or after all jobs	Setup times	Sequential irrigation	Simultaneous irrigation	Simple multi-machine	Complex multi-machine
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)
Single machine model	1	✓					✓			
	2		✓	✓	✓		✓			
	3	✓				✓	✓			
	4a		✓			✓	✓			
	4b			✓		✓	✓			
	4c				✓	✓	✓			

4.1 Analysis and development of non-contiguous single machine scheduling with sequence dependent setup times

The non-contiguous single machine irrigation scheduling model with sequence dependent setup times (herein to referred to as Model 3) allows scheduling of a number of jobs according to their target start times, whilst taking into account the sequence dependent setup times. Let $\boldsymbol{Q} = \{1,2,...,N\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: duration, target start time, earliness cost, tardiness cost and sequence dependent setup times. In Model 3 the decision to be made is: what is the scheduled start time of each job. If the answer to this question is known then the schedule is known. The objective of Model 3 is to find a schedule such that the differences between the target start time and the scheduled start time is as small as possible. This can be achieved by minimising the earliness/tardiness. Therefore the objective function is

$$\text{Minimize } Z = \sum_{i=1}^{N} (\alpha_i E_i + \beta_i T_i) \quad \forall i = 1,2,...,N \tag{4.1}$$

Any job (irrigation event) can precede any other job in the schedule. To define which job precedes which, the following variable is defined

$$\begin{aligned} \delta_{ij} &= 1 \quad \text{if job } i \text{ precedes job } j \\ &= 0 \quad \text{otherwise} \end{aligned} \qquad i \neq j \tag{4.2}$$

The scheduled start time of a job is determined from the target start time, the earliness and the tardiness

$$S_i = r_i + T_i - E_i \qquad \forall i = 1,2,...,N \tag{4.3}$$

It is not possible for a job to incur a negative earliness or tardiness (in fact a negative earliness is a positive tardiness and vice versa). Therefore for T_i and E_i the following constraints need to be satisfied

$$T_i \geq 0 \qquad \forall i = 1,2,...,N \tag{4.4}$$

$$E_i \geq 0 \qquad \forall i = 1,2,...,N \tag{4.5}$$

Any two jobs cannot be serviced simultaneously. This implies that either job i precedes job j or vice versa. The following two disjunctive constraints that will enforce this constraint

$$S_j - S_i + M(1 - \delta_{ij}) \geq d_i + t_{ij} \qquad \forall i = 1,2,...,N;\ j = 1,2,...,N;\ i \neq j \tag{4.6}$$

$$S_i - S_j + M\delta_{ij} \geq d_j + t_{ji} \qquad \forall i = 1,2,..,N;\ j = 1,2,...,N;\ i \neq j \tag{4.7}$$

where t_{ij} = sequence dependent setup time from user i to user j; and t_{ji} = sequence dependent setup time from user j to user i. Every job must be completed within the irrigation interval

$$S_i + d_i \leq g \qquad \forall i = 1,2,...,N \tag{4.8}$$

In addition to the setup times needed between two jobs , there may be some setup time required to prepare for the first job to be able to start. In irrigation scheduling this initial setup time can be seen as the time needed for the water to travel from the head of the tertiary canal to the field outlet of the first user to start irrigating. This initial setup time can be incorporated into the model as follows

$$S_i \geq I_i \qquad \forall i = 1,2,...,N \tag{4.9}$$

where I_i = initial setup time. Therefore Model 3 is defined by the objective function (4.1) and constraints (4.2) through to (4.9) inclusive.

4.2 Analysis and development of contiguous single machine scheduling with sequence dependent setup times

The contiguous single machine irrigation scheduling model with sequence dependent setup times (herein to be referred to as Model 4) is similar to Model 3 in that it allows

scheduling of a number of jobs according to their target start times, whilst taking into account the sequence dependent setup times. The main difference is that, like in Model 2, to reduce water spillage and/or gate operations, idle time between jobs is not allowed. As with Model 2 there are three possible variations on scheduling all jobs contiguously. Model 4a schedules all jobs such that the first job starts at the beginning of the interval and any idle time is inserted after the last job in the schedule has been completed. In Model 4b all jobs are scheduled such that the last job is scheduled to finished at the end of the interval. Any idle time is inserted before the first job in the schedule starts. Model 4c is an intermediate form between Models 4a and 4b. In this model the first job is scheduled to start some time after the start of the interval and the last job is scheduled to finish sometime before the end of the interval. Model 3 described earlier in this chapter cannot be adapted to enable contiguous scheduling without making use of non-linear constraints. A different approach is therefore needed.

4.2.1 Contiguous scheduling with sequence dependent setup times: Model 4a

Bianco et al. (1988) described a mixed integer linear programming model that allows minimizing the maximum completion time of a number of tasks with sequence dependent setup times. This model can be adapted to enable irrigation scheduling. In this model sequent dependent setup times and durations are combined to form a new matrix of sequence dependent durations. A dummy variable with index 0 is introduced as to allow initial setup times to be included in the sequence dependent duration matrix. Let $\boldsymbol{Q} = \{0,1,...,N\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: target start time, earliness cost, tardiness cost and sequence dependent duration. In Model 4 the decision to be made is: what is the scheduled start time of each job? If the answer to this question is known then the schedule is known. The objective of Model 4 is to find a schedule such that the differences between the target start time and the scheduled start time is as small as possible. This can be achieved by minimising the earliness/tardiness. Therefore the objective function is

$$\text{Minimize } Z = \sum_{i=1}^{N} (\alpha_i E_i + \beta_i T_i) \tag{4.10}$$

Any job can directly precede any other job in a irrigation schedule. To define which job directly precedes which, the following decision variable is used

$$\gamma_{ij} = 1 \quad \text{if job } i \text{ directly precedes job } j \qquad i \neq j \qquad (4.11)$$
$$= 0 \quad \text{otherwise}$$

where γ_{ij} = binary variable ($j = 0$ if job i is the last in the sequence). The scheduled start time of a job is determined from the target start time, the earliness and the tardiness

$$S_i = r_i + T_i - E_i \qquad \forall i = 1,2,...,N \qquad (4.12)$$

It is not possible for a job to incur a negative earliness or tardiness. Therefore

$$T_i \geq 0 \qquad \forall i = 1,2,...,N \qquad (4.13)$$

$$E_i \geq 0 \qquad \forall i = 1,2,...,N \qquad (4.14)$$

Any job can be preceded by only one other job. Therefore

$$\sum_{j=0}^{N} \gamma_{ij} = 1 \qquad \forall i = 0,1,...,N \qquad (4.15)$$

Similarly any job can be followed by only one other job

$$\sum_{i=0}^{N} \gamma_{ij} = 1 \qquad \forall i = 0,1,...,N \qquad (4.16)$$

No job should start before the previous job has finished. Therefore

$$S_i + d_{ij}^{*} \gamma_{ij} + M(\gamma_{ij} - 1) \leq S_j \qquad \forall i = 0,1,...,N;\ \forall j = 1,2,...,N;\ i \neq j \qquad (4.17)$$

where d_{ij}^{*} = sequence dependent duration of job i if job i directly precedes job j ($j = 0$ if job i is the last in the sequence, d_{0j}^{*} = the initial setup time if the sequence starts with job

j). To ensure that idle time is only inserted after the last job has been processed, no job may finish later than the sum of all durations

$$S_i + \sum_{j=0}^{N} (d_{ij}^* \gamma_{ij}) \leq \sum_{i=0}^{N} \sum_{j=0}^{N} (d_{ij}^* \gamma_{ij}) \qquad \forall i = 0,1,...,N \tag{4.18}$$

Model 4a is defined by the objective function (4.10) and constraints (4.11) - (4.18).

4.2.2 Contiguous scheduling with sequence dependent setup times: Model 4b

Model 4b is similar to Model 4a with the difference that in Model 4b all jobs are scheduled contiguously and idle time is inserted prior to the start of the first job. The amount of idle time inserted into the schedule depends on the sum of the sequence dependent durations

$$X_a = g - \sum_{i=0}^{N} \sum_{j=0}^{N} d_{ij}^* \gamma_{ij} \tag{4.19}$$

Idle time is inserted before the first job in the schedule starts therefore

$$S_i \geq X_a \qquad \forall i = 0,1,...,N \tag{4.20}$$

Model 4b is defined by the objective function (4.10) and constraints (4.11) - (4.17), (4.19) and (4.20).

4.2.3 Contiguous scheduling with sequence dependent setup times: Model 4c

Model 4c again closely resembles Model 4a, with the difference that idle time may precede the start of the first job in the schedule and/or follow after the completion of the last job, therefore (4.18) is replaced with

$$S_i + \sum_{j=0}^{N} d_{ij}^* \gamma_{ij} \leq \sum_{i=0}^{N} \sum_{j=0}^{N} d_{ij}^* \gamma_{ij} + X_a \qquad \forall i = 0,1,...,N \tag{4.21}$$

Constraint (4.20) is also necessary to ensure correct insertion of idle time in Model 4c. Model 4c is therefore defined by the objective function (4.10) and constraints (4.11) - (4.17), (4.20) and (4.21).

4.3 Results and discussion single machine scheduling with setup times

The tertiary unit described by Bishop and Long (1983) is used for application of the models developed in this chapter. No detailed information, other than the map in Figure 3.2, of this tertiary unit is available to determine the exact measurements of fields and channels or the placement of field outlets, all of which are needed to determine the travel times between field outlets. Figure 4.1 shows a schematic representation of the tertiary unit. Field outlets are assumed to be in the upstream corner of each field. Table 4.2 gives the dimensions of each field as determined from Figure 4.1. Table 4.3 gives the velocity in the channel sections, as given by Bishop and Long (1983).

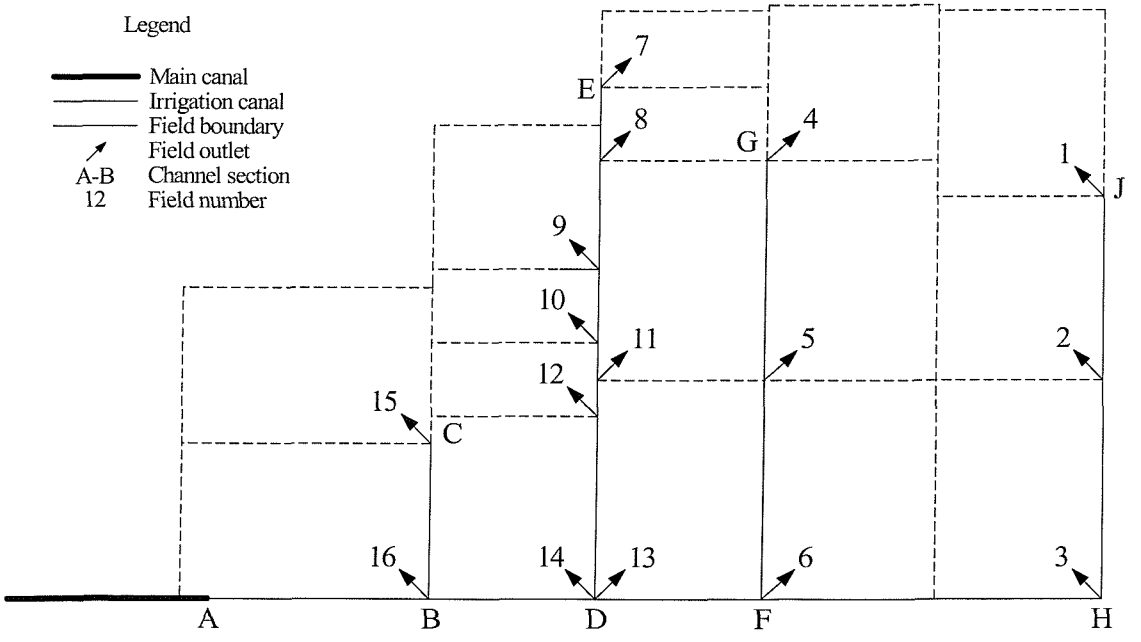


Figure 4.1 Schematic representation of a tertiary unit, Bula project, Phillippines (from Bishop and Long, 1983)

Table 4.2: Approximate field dimensions tertiary unit Bula project

Field number (1)	Length (m) (2)	Width (m) (3)	Area (ha) (4)
1	200	150	3.00
2	167	150	2.50
3	170	150	2.55
4	142	150	2.13
5	200	150	3.00
6	200	150	3.00
7	70	150	1.05
8	67	150	1.01
9	131	150	1.97
10	67	150	1.01
11	200	150	3.00
12	67	150	1.01
13	200	150	3.00
14	167	150	2.50
15	142	225	3.20
16	142	225	3.20

Table 4.3: Velocity in channel sections

Channel section (1)	Velocity (m/s) (2)
A-B	0.650
B-C	0.456
B-D	0.223
D-E	0.682
D-F	0.223
F-G	0.229
F-H	0.223
H-J	0.229

The sequence dependent setup times can be determined as follows

$$t_{ij} = \sum_{b=1}^{B} t_b \qquad \forall i = 1,2,...,N;\ \forall j = 1,2,...,N;\ i \neq j \tag{4.22}$$

where t_b = travel time for channel section b; b = index representing channel section 1,2,...,B; and B = total number of sections between user i and user j. Bishop and Long

(1983) suggested that the travel time for each channel section can be determined by

$$t_b = \frac{L_b}{0.7 v_b} \tag{4.23}$$

where L_b = length of channel section b; and v_b = average velocity in channel section b. Multiplying the velocity by 0.7 compensates for the lower advance velocity that generally exist when initially filling a channel and will also compensate for the additional time needed to fill the channel to operating depth (Bishop and Long, 1983). The method described here to calculate travel times is empirical. To obtain more exact travel times hydro-dynamic modelling may be required, e.g. De Bièvre et al. (2003) use hydro-dynamic modelling to calculate travel times.

Table 4.4 shows the travel times for the tertiary unit, determined with (4.22) and (4.23).

Table 4.4: Travel times (in minutes) from outlet i to outlet j

		to outlet j															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
from outlet i	1	-	0	0	42	21	0	16	14	10	8	7	6	0	0	7	0
	2	17	-	0	42	21	0	16	14	10	8	7	6	0	0	7	0
	3	38	21	-	42	21	0	16	14	10	8	7	6	0	0	7	0
	4	69	52	31	-	0	0	16	14	10	8	7	6	0	0	7	0
	5	69	52	31	21	-	0	16	14	10	8	7	6	0	0	7	0
	6	69	52	31	42	21	-	16	14	10	8	7	6	0	0	7	0
	7	85	68	47	57	36	16	-	0	0	0	0	0	0	0	7	0
	8	85	68	47	57	36	16	2	-	0	0	0	0	0	0	7	0
	9	85	68	47	57	36	16	6	3	-	0	0	0	0	0	7	0
	10	85	68	47	57	36	16	8	6	2	-	0	0	0	0	7	0
	11	85	68	47	57	36	16	9	7	4	1	-	0	0	0	7	0
	12	85	68	47	57	36	16	10	8	5	2	1	-	0	0	7	0
	13	85	68	47	57	36	16	16	14	10	8	7	6	-	0	7	0
	14	85	68	47	57	36	16	16	14	10	8	7	6	0	-	7	0
	15	101	83	63	73	52	31	32	30	26	24	23	21	16	16	-	0
	16	101	83	63	73	52	31	32	30	26	24	23	21	16	16	7	-

The highlighted area in Table 4.4 shows an example of the travel time from Outlet 3 to Outlet 4. It can be seen in Figure 4.1 that when Outlet 3 is running channel sections A-B,

B-D, D-F and F-H are filled. When irrigation at Outlet 3 stops and Outlet 4 takes over water needs to be diverted at F and travel to Outlet 4 (at G). The travel time is the time it will take to fill channel section F-G, and is calculated to be 42 minutes.

Bishop and Long (1983) gave the irrigation duration as 172 min/ha. In their scheduling method 30 min/day was set apart for management purposes. As management time is usually not considered to be a component of travel time (e.g. Khepar et al., 2000; Latif and Sarwar, 1994) this time is added to the irrigation duration. The irrigation duration is given in Table 4.5. Target start times as generated for the application of Models 1 and 2 in Chapter 3 are used here too and are given in Table 4.5. The initial setup times for each job can also be determined with (4.22) and (4.23) and are given in Table 4.5.

Table 4.5: Job parameters

Job number (1)	Irrigation duration (min) (2)	Target start times (min) (3)	Initial setup time (min) (4)
1	453	2140	108
2	443	9783	91
3	529	8009	70
4	379	8889	80
5	529	4841	59
6	529	771	39
7	194	3362	39
8	185	5635	37
9	352	5408	33
10	185	5699	31
11	529	1365	30
12	185	1503	29
13	529	19	23
14	443	9515	23
15	563	1206	15
16	564	990	7

Models 3, 4a, 4b and 4c were implemented in Lingo® for Windows® using the data from Table 4.4 and 4.5. The Lingo input files are included in Appendix A. Table 4.6 shows the schedule obtained when applying Model 3. The first row of the table shows that Field 13 is the first to be irrigated. At 08:00 the setup time for this field is scheduled to start. This

means that at this time the water is at the main inlet of the tertiary channel and starts travelling to the field inlet of Field 13. The travel time from the main inlet to Field 13 is 23 minutes and irrigation can start at 08:23.

Table 4.6: Irrigation schedule Model 3

	Setup time		Irrigation			
Field	Start		On		Off	
(1)	(2)		(3)		(4)	
13	Mon	08:00	Mon	08:23	Mon	17:13
6	Mon	17:13	Mon	17:29	Tue	02:18
16	Tue	02:18	Tue	02:18	Tue	11:42
12	Tue	11:42	Tue	12:03	Tue	15:08
11	Tue	15:08	Tue	15:09	Tue	23:58
1	Wed	01:16	Wed	01:23	Wed	08:56
15	Wed	08:56	Wed	09:03	Wed	18:27
7	Wed	18:52	Thu	09:03	Thu	12:17
5	Thu	15:52	Thu	16:08	Fri	00:57
9	Fri	00:57	Fri	01:07	Fri	06:59
10	Fri	06:59	Fri	06:59	Fri	10:04
8	Fri	10:04	Fri	10:10	Fri	13:15
3	Sat	21:15	Sat	21:29	Sun	06:18
4	Sun	09:16	Sun	09:47	Sun	16:06
14	Sun	16:06	Sun	16:06	Sun	23:29
2	Sun	23:29	Mon	00:37	Mon	08:00

Figure 4.2 shows a graphical representation of the last two days of the schedules obtained with Model 1 and Model 3. As explained in Appendix D Model 1 has to be slightly adjusted before it can be directly compared to Model 3.

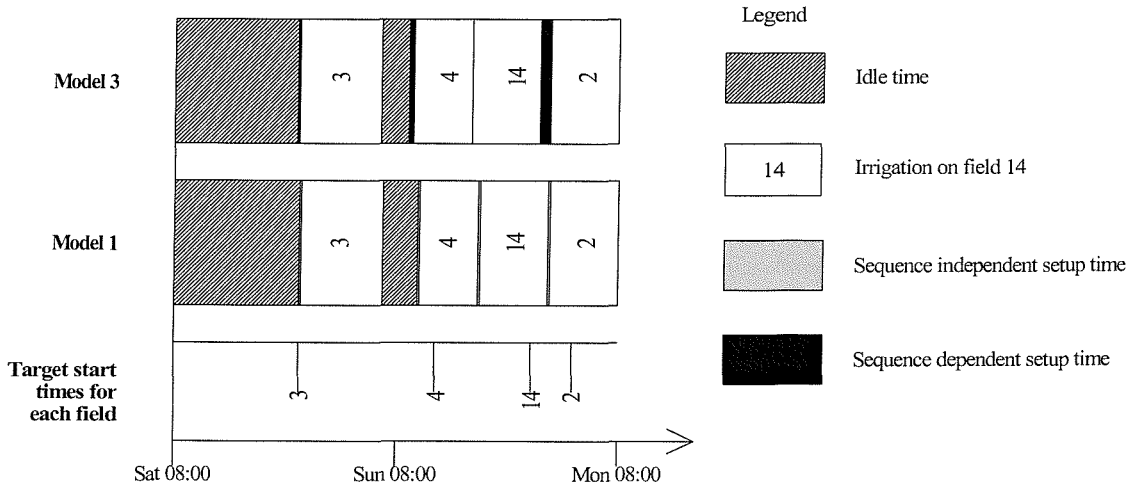


Figure 4.2 Comparison Model 1 and Model 3

It can be seen in Figure 4.2 that an equal amount of setup time, 13 minutes, is inserted before each job for Model 1. For Model 3 the inserted setup times vary from 0 minutes between Job 4 and Job 14 to 68 minutes between Jobs 14 and 2. If Model 1 was implemented rather than Model 3 the schedule does not always allow enough time for the water to reach the field inlet of the next user to irrigate. For instance the travel time between Job 14 and 2 is 68 minutes, if only 13 minutes is available as in Model 1, the users would have to wait 55 minutes for the water to reach the field and either to finish irrigating on the agreed time and loose out on irrigation time or to continue irrigating beyond the agreed time and disadvantage other users. Neither is an acceptable course of action. Model 3 takes into account the actual travel time between users, water is therefore delivered the scheduled start time and the irrigation can proceed according to the schedule.

As reported in section 3.5, solution times increase with the complexity of the model that is being solved. The formulation for Models 4a, 4b and 4c is more complex as that for Model 3. Although it is possible to solve Model 3 for 16 users, for Model 4a, 4b and 4c this resulted in excessively large solution times (> 10 days). The tertiary unit described by (Bishop and Long, 1983) has been modified to allow the models to be solved for a smaller number of jobs. The 6 users furthest away from the inlet on the main canal (users 1 through 6) have been excluded. The irrigation interval has been shortened to 5.5 days and new target start times were generated randomly. Figure 4.3 shows a graphical representation of the schedules obtained for Models 4a, 4b, and 4c. Model 3 was also run with the 6 further users removed and is included in Figure 4.3 too.

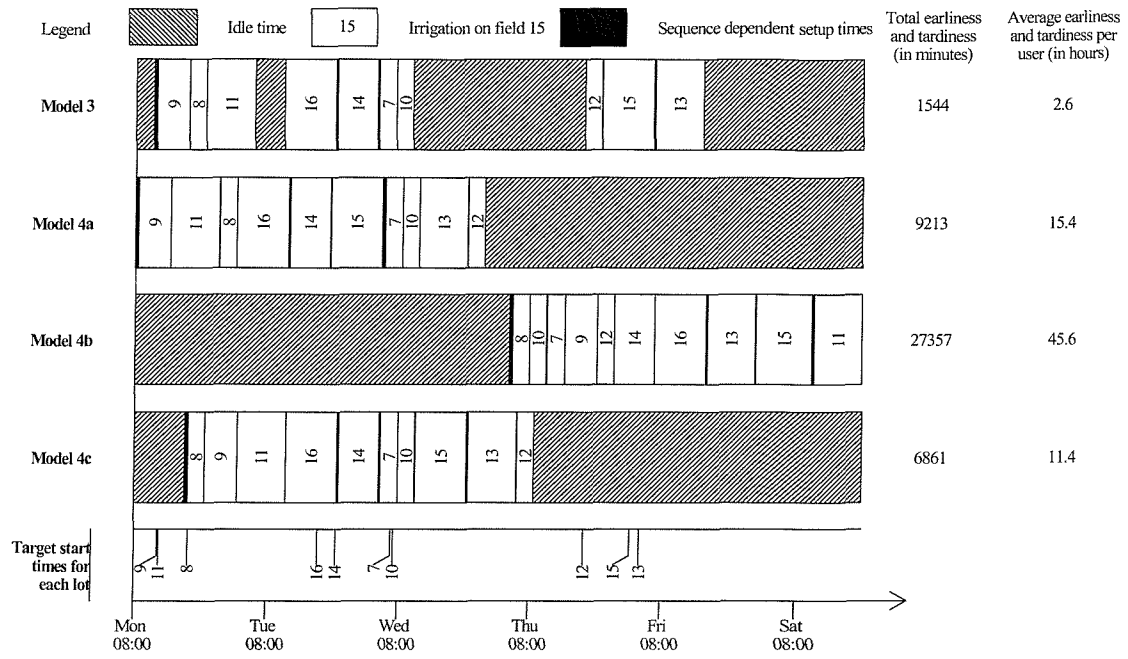


Figure 4.3 Schedules applying Models 3, 4a, 4b, 4c

4.4 Conclusions single machine scheduling with sequence dependent setup times

In this chapter the models for non-contiguous and contiguous single machine scheduling with sequence dependent setup times are developed. It is shown that sequence dependent setup times can have an influence on the schedule. By applying the models to a tertiary unit the importance of including sequence dependent setup times is demonstrated. It should be noted that travel times are not constant over time. Discharge and velocity in the channel can vary due to external influences such as rain or shortage of water at the source and can vary from interval to interval or even within an interval. This means that travel times may need to be recalculated to reflect these changes in discharge and/or velocity. There are some disadvantages/shortcomings to the models described in this chapter:

1. It is assumed that users irrigate sequentially, but in many irrigation systems simultaneous irrigation is allowed. This problem will be addressed in the next two chapters, Chapter 5 and 6.
2. Solution times increase with the number of jobs. As a result thereof only small problems can be solved with the models developed in this chapter. Other solution techniques such as heuristics or genetic algorithms may be required to solve larger problems.

5 Simple multi-machine scheduling

In single machine scheduling water is seen as a machine that can be used by at most one user at a time. This makes it a suitable analogy for rotational irrigation where users use the water sequentially. There are however many irrigation systems where two or more users irrigate simultaneously. Wang et al. (1995) introduced the concept of stream tubes. One stream tube supplies water to only one user at a time, but can supply several users in sequence. As water can divided in to more than one (imaginary) stream tube, more than one user can be supplied at a time. Each stream tube can be seen as a separate machine and multi-machine scheduling can be used to find the irrigation schedule for a number of users.

The work by Wang et al. (1995) focussed on the decision of whether to operate the outlets of a tertiary unit simultaneously or in sequence. Figure 5.1a shows how irrigating simultaneously requires a larger canal capacity (economically undesirable) and the availability of sufficient water resources. The large flow required to allow simultaneous irrigation can exceed the capacity of the channel. Figure 5.1b shows how running the outlets sequentially allows the use of a smaller channel, but the total running time of a channel may exceed the irrigation interval. Figure 5.1c shows the optimal schedule, wherein all outlets are supplied with water within the irrigation interval, using the smallest possible discharge. Anwar and Clarke (2001) further developed the work by Wang et al. (1995) by showing how the concept of stream tubes can be used to schedule a number of jobs according to their target start times. This formulation was limited to non-contiguous scheduling. A limitation of the work by both Wang et al. (1995) and Anwar and Clarke (2001) is that the discharges of all outlets are assumed to be identical. By allowing each outlet to receive water from more than one stream tube at the same time, it is possible to vary the discharge to that outlet. Stream tubes supplying water to outlets can be seen as machines processing jobs. If more than one machine is available to process jobs, multi-machine scheduling can be used to find the optimal schedule. The case where all outlets receive the same discharge will be referred to as simple multi-machine scheduling. The case where the discharges vary from outlet to outlet and more than one machine may be needed to process that job will be referred to as complex multi-machine scheduling. Multi-machine scheduling, whether simple or complex, can offer a degree of flexibility in a schedule. If canal capacity and available water resources permit, the discharge in a channel

can be increased. This can improve the timeliness of water deliveries, which is of importance to irrigation districts where penalties are imposed if water is not delivered on time.

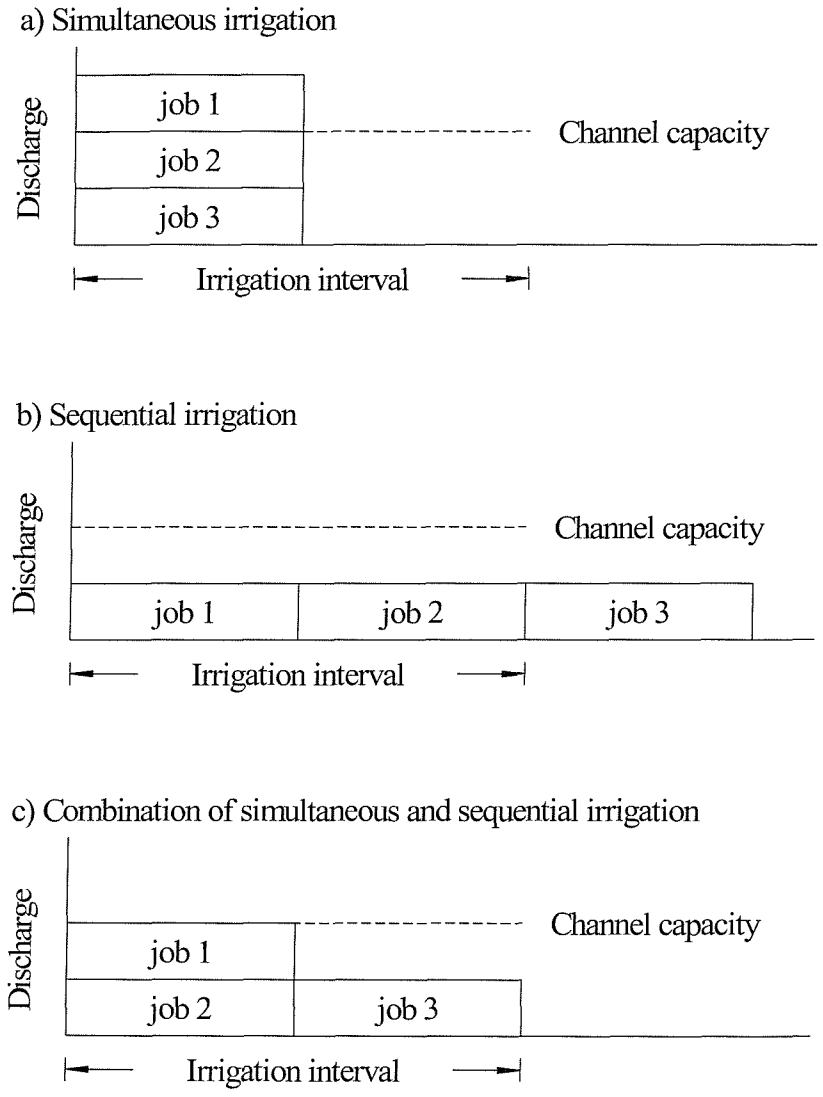


Figure 5.1 Simultaneous and sequential irrigation

In this chapter the models for simple multi-machine scheduling (non-contiguous and contiguous), with and without sequence dependent setup times, will be developed. The non-contiguous multi-machine model achieves the same results as the work done by Anwar and Clarke (2001). However the formulation by Anwar and Clarke (2001) does not allow contiguous scheduling nor complex multi-machine scheduling. Models for complex multi-machine scheduling and complex multi-machine scheduling with setup times will follow in the following chapter, Chapter 6. A series of alternatives to the simple multi-

machine scheduling will also be presented. These models can reduce the number of constraints and variables and therefore the solution times, but have a limited use. The issue of timeliness of deliveries is discussed in this chapter and constraints are presented that allow the concept of timeliness to be incorporated into any of the multi-machine models. Table 5.1 shows an overview of the models that will be developed in this chapter.

Table 5.1: Overview of different models and their features

	Model		Contiguous							
		Non- contiguous	Idle time after all jobs	Idle time before all	Idle time before and/or after all	Setup times	Sequential irrigation	Simultaneous irrigation	Simple multi-machine	Complex multi-machine
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)
Multi-machine scheduling	5	✓					✓	✓	✓	
	6a		✓				✓	✓	✓	
	6b			✓			✓	✓	✓	
	6c				✓		✓	✓	✓	
	7	✓				✓	✓	✓	✓	
	8a		✓			✓	✓	✓	✓	
	8b			✓		✓	✓	✓	✓	
	8c				✓	✓	✓	✓	✓	
	9	✓					✓	✓	✓	✓
	10		✓	✓	✓		✓	✓	✓	✓
	11	✓				✓	✓	✓	✓	✓
	12		✓	✓	✓	✓	✓	✓	✓	✓

5.1 Analysis and development of non-contiguous simple multi-machine scheduling

The simple non-contiguous multi-machine schedule, herein referred to as Model 5, allows scheduling of a number of jobs (irrigation events) according to their target start times. In

this model more than one machine (stream tube) is available to process jobs, i.e. users are allowed to irrigate simultaneously. A fictitious job with index 0 is used to simplify writing of the constraints. Let $\boldsymbol{Q} = \{0,1,2,...,N\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: duration, target start time, earliness cost, tardiness cost and sequence independent setup time. Also let $\boldsymbol{V} = \{1,2,...,W\}$ be a set of machines available to process jobs. All jobs require equal discharges and the size of each machine is equal to this discharge. In Model 5 there are two decisions to be made, which machine processes which job and what is the scheduled start time of each job. If the answers to these questions are known, the schedule is known too. The objective of the model is to find a schedule such that every job starts as close as possible to the target start time. The objective function consists of two terms. The first aims to find the sequence of jobs and scheduled start time for each job so that every job starts as close as possible to the target start time. This is achieved by minimising the penalties incurred when a job is either early or tardy. The second term aims to limit the wastage of water by reducing the discharge in the channel. This can be done by minimizing the number of stream tubes used. Multiplying the second term by a large constant will ensure that minimizing the discharge has priority over the minimizing the earliness/tardiness. This constant will have to be chosen such that the second goal will be several orders of magnitude larger than the first (Anwar and Clarke, 2001). The objective function can be written as

$$\text{minimize } Z = \sum_{i=1}^{N} (\alpha_i E_i + \beta_i T_i) + c \sum_{w=1}^{W} \psi_w \tag{5.1}$$

where c = large constant; w = index representing machine 1,2,...,W; W = number of available machines; and ψ_w = binary variable, which is used to define whether a machine is being activated, i.e. processes one or more jobs

$$\begin{aligned} \psi_w &= 1 \quad \text{if machine } w \text{ is activated} \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{5.2}$$

Any job can directly precede any other job on a machine, therefore a variable is used to define which job precedes which other job on what machine

$$\varphi_{ijw} = 1 \quad \text{if job } i \text{ directly precedes job } j \text{ on machine } w \qquad i \neq j \qquad (5.3)$$
$$= 0 \quad \text{otherwise}$$

where φ_{ijw} = binary variable. A job can be processed by any machine, so a variable is used to define which job is processed by which machine

$$\tau_{iw} = 1 \quad \text{if job } i \text{ is processed by machine } w \qquad (5.4)$$
$$= 0 \quad \text{otherwise}$$

where τ_{iw} = binary variable.

The scheduled start time of a job is determined from the target start time, the earliness and the tardiness

$$S_i = r_i + T_i - E_i \qquad \forall i = 1,2,...,N \qquad (5.5)$$

It is not possible for a schedule to incur a negative earliness or tardiness (in fact a negative tardiness is the same as a positive earliness and vice versa), therefore the following constraints need to be satisfied

$$T_i \geq 0 \qquad \forall i = 1,2,...,N \qquad (5.6)$$

$$E_i \geq 0 \qquad \forall i = 1,2,...,N \qquad (5.7)$$

Each job can be processed by only one machine

$$\sum_{w=1}^{W} \tau_{iw} = 1 \qquad i = 1,2,...,N \qquad (5.8)$$

Each job can at most precede one other job

$$\sum_{j=1}^{N} \varphi_{ijw} \leq \tau_{iw} \qquad \forall i = 0,1,...,N;\ \forall w = 1,2,...,W;\ i \neq j \qquad (5.9)$$

Each job (with the exception of job 0) must follow one other job

$$\sum_{i=0}^{N} \varphi_{ijw} = \tau_{jw} \qquad \forall j = 1,2,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{5.10}$$

No job is allowed to start before the previous job on the same machine has finished

$$S_j - S_i - M\varphi_{ijw} \geq d_i - M \qquad \forall i = 0,1,...,N;\ \forall j = 1,2,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{5.11}$$

Each job should be finished within the interval

$$S_i + d_i \leq g \qquad \forall i = 1,2,...,N \tag{5.12}$$

A machine is activated if it processes at least one job (supplies at least one outlet with water)

$$\sum_{i=1}^{N} \tau_{iw} \leq \psi_w W \qquad \forall w = 1,2,...,W \tag{5.13}$$

The capacity of a channel may not be exceeded

$$q \sum_{w=1}^{W} \psi_w \leq Q_c \tag{5.14}$$

where q = size of machine (discharge of stream tube); and Q_c = channel capacity. For each machine the sum of durations must be less than or equal to the length of the interval

$$\sum_{i=1}^{N} d_i \tau_{iw} \leq g \qquad \forall w = 1,2,...,W \tag{5.15}$$

This constraint is not absolutely essential to the model, as it achieves the same as constraint (5.12), but is found to greatly reduce solution times. Model 5 is therefore defined by the objective function (5.1) and the constraints (5.3) - (5.15).

5.2 Analysis and development contiguous simple multi-machine scheduling

The contiguous simple multi machine scheduling model, herein referred to as Model 6, is based on Model 5. The objective function and constraints described for Model 5 remain valid. Some extra constraints are necessary to ensure correct insertion of idle time.

5.2.1 Contiguous simple multi-machine modelling: Model 6a

In Model 6a all idle time is inserted after the last job in the schedule has been completed, therefore no job should finish later than the sum of durations of all jobs processed on the same machine

$$S_i + d_i - M(1 - \tau_{iw}) \leq \sum_{i=1}^{N} d_i \tau_{iw} \qquad \forall i = 1,2,...,N;\ \forall w = 1,2,...,W \tag{5.16}$$

Model 6a is therefore defined by the objective function (5.1) and the constraints (5.3) - (5.16).

5.2.2 Contiguous simple multi-machine scheduling: Model 6b

Model 6b is similar to Model 6a with the difference that all idle time is inserted before the first job in the schedule starts. The amount of idle time inserted on each machine depends on the duration of the jobs processed by that machine. Therefore

$$X_{aw} = g - \sum_{i=1}^{N} d_i \tau_{iw} \qquad \forall w = 1,2,...,W \tag{5.17}$$

where X_{aw} = idle time preceding the start of the first job of the schedule on machine w. Idle time on each machine is inserted before the all jobs. Therefore

$$S_i + M(1 - \tau_{iw}) \geq X_{aw} \qquad \forall i = 1,2,...,N;\ \forall w = 1,2,...,N \tag{5.18}$$

Model 6b is therefore defined by the objective function (5.1) and the constraints (5.3) - (5.15), (5.17) and (5.18).

5.2.3 Contiguous simple multi-machine modelling: Model 6c

In Model 6c idle time is allowed to be inserted both before the first job in the schedule starts and after the last job in the schedule is finished. The total amount of idle time inserted (before and after all jobs) on a machine depends on the duration of the jobs processed by that machine

$$g = \sum_{i=1}^{N} d_i \tau_{iw} + X_{aw} + X_{bw} \qquad \forall w = 1,2,...,W \tag{5.19}$$

where X_{bw} = idle time following the end of the last job in the schedule on machine w. If idle time is inserted at the end of a schedule this may only be done after all jobs have been completed

$$S_i + d_i - M(1 - \tau_{iw}) \leq g - X_{bw} \qquad \forall i = 0,1,...,N;\ \forall w = 1,2,...,W \tag{5.20}$$

If idle time is inserted at the beginning of a schedule this may only be done before the first job start. Therefore constraint (5.18) is also necessary. Model 6c is therefore defined by the objective function (5.1) and the constraints (5.3) - (5.15) and (5.18) - (5.20)

5.3 Simple multi-machine modelling with sequence dependent setup times

Sequence dependent setup times for multi-machine scheduling is not as straightforward to incorporate as for single machine scheduling. In single machine scheduling, although the discharge can vary from interval to interval, it is constant during an interval. Sequence dependent setup times for single machine scheduling are defined as the time needed for the water in the channel to be diverted from one outlet to another. In multi-machine scheduling water is not diverted from just one outlet to only one other outlet, it is diverted from several outlets to several other outlets, all at different times. As a result hereof the discharge will vary, along the length of the channel as well as during the course of the interval. Figure 5.2 shows these spatial and temporal fluctuations. Figure 5.2 shows a channel with three outlets, A, B and C. Outlet B is located downstream of outlet A and

outlet C is located downstream of both outlet A and B. Figure 5.2a shows that at time T only outlet A is running, the channel is empty downstream of this outlet. Figure 5.2b shows that after a while, at time $T + \Delta t$, outlet A has stopped running, the water has been diverted and now outlet B is running. The sequence dependent setup time in this situation is the time needed for the water to travel (through the empty channel) from outlet A to outlet B.

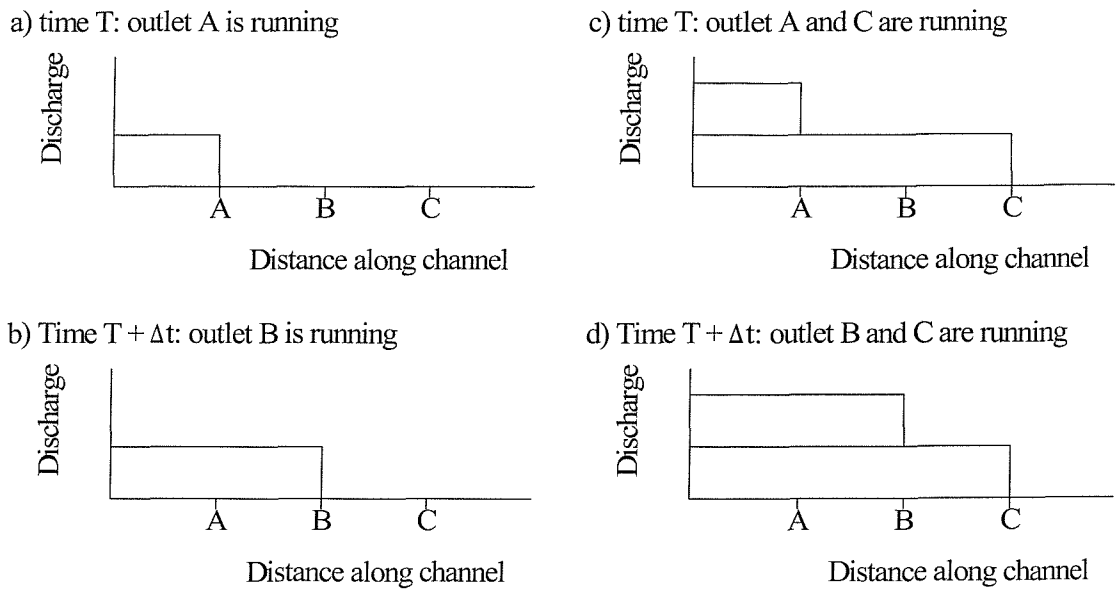


Figure 5.2 Spatial and temporal variation of discharge

In multi-machine terms this situation can be seen as one machine (stream tube) that first processes one job (outlet A) and then another (outlet B). Figure 5.2c shows the same channel, again at time T, but now both outlet A and C are running. Figure 5.2d shows that after a while, at time $T + \Delta t$, outlet A has finished running, the water has been diverted and now outlet B is running. Meanwhile the situation for outlet C remains unchanged and the outlet is still running. The sequence dependent setup time in this situation for outlet B is still the time needed for water to travel from outlet A to outlet B. However the channel is partially filled which will reduce the time needed to divert the water. In multi-machine scheduling terms this situation can be seen as two machines, the first one processing job A and B, the second processing job C. This example shows that sequence dependent setup times for multi-machine scheduling not only depends on the locations of two outlets, as is the case for single machine scheduling, but also on the discharge of the channel between these two outlets. Where in single machine scheduling the water always travels through a dry channel, in multi-machine scheduling water can travel through a dry channel or a

(partially) filled channel. Whether or not a channel is filled will have an influence on the travel time. Furthermore it is conceivable that during the time that water is diverted from one outlet to another the discharge in the channel changes. This could happen if, for instance a third outlet located in between the two outlets finishes running. This means that the sequence dependent setup time is also dependent on the change in discharge between one outlet and another. How to exactly incorporate this multi-dimensional variable into multi-machine modelling needs to be further investigated in future.

It is possible to simplify the multi-machine scheduling problem with sequence dependent setup times by assuming the channel is always dry. By assuming the channel is always empty when water is diverted from one outlet to another, sequence dependent setup time can now be calculated the same way as sequence dependent setup times for single machine scheduling. Using this approach ensures that the sequence dependent setup times in reality can be less than the calculated sequence dependent setup time, but never more. This means users may receive water earlier then scheduled, but never later.

5.4 Analysis and development non-contiguous simple multi-machine scheduling with sequence dependent setup times

The non-contiguous simple multi-machine model with sequence dependent setup times (herein referred to as Model 7) allows scheduling of a number of jobs (irrigation events) according to their target start times, whilst taking into account the sequence dependent setup times. In this model more than one machine (stream tube) is available to process jobs, i.e. users are allowed to irrigate simultaneously. A fictitious job with index 0 is used to simplify writing the constraints. Let $\boldsymbol{Q} = \{0,1,...,N\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: duration, target start time, earliness cost, tardiness cost, required discharge and sequence dependent setup times. Also let $\boldsymbol{V} = \{1,2,...,W\}$ be a set of machines (or stream tubes) available to supply water to field outlets. The discharge for each machine $w \in \boldsymbol{V}$ is specified and is equal to the discharge of the outlets.

The objective function consists of two terms. The first aims to find the sequence of jobs and scheduled start time for each job so that every job starts as close as possible to the target start time. This is achieved by minimising the penalties incurred when a job is either

early or tardy. The second terms minimizes the discharge, i.e. minimizes the number of stream tubes used. The objective function can now be written as

$$\text{Minimize } Z = \sum_{i=1}^{N} (\alpha_i E_i + \beta_i T_i) + c \sum_{w=1}^{W} \psi_w \tag{5.21}$$

As any job can directly precede any other job on a machine a variable is used to define which job precedes which other job on what machine

$$\begin{aligned} \varphi_{ijw} &= 1 \quad \text{if job } i \text{ directly precedes job } j \text{ on machine } w \\ &= 0 \quad \text{otherwise} \end{aligned} \qquad i \neq j \tag{5.22}$$

Any job can be processed by any machine, so a variable is used to define which job is processed by which machine

$$\begin{aligned} \tau_{iw} &= 1 \quad \text{if job } i \text{ is processed by machine } w \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{5.23}$$

The following variable defines whether a machine is used or not

$$\begin{aligned} \psi_w &= 1 \quad \text{if machine } w \text{ is activated} \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{5.24}$$

The scheduled start time of a job is determined from the target start time, the earliness and the tardiness

$$S_i = r_i + T_i - E_i \qquad \forall i = 1,2,...,N \tag{5.25}$$

Negative earliness or tardiness does not exist therefore for T_i and E_i the following constraints need to be satisfied

$$T_i \geq 0 \qquad \forall i = 1,2,...,N \tag{5.26}$$

$$E_i \geq 0 \qquad \forall i = 1,2,...,N \tag{5.27}$$

Each job can be processed by only one machine

$$\sum_{w=1}^{W} \tau_{iw} = 1 \qquad i = 1,2,...,N \tag{5.28}$$

Each job can at most precede one other job

$$\sum_{j=1}^{N} \varphi_{ijw} \leq \tau_{iw} \qquad \forall i = 0,1,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{5.29}$$

Each job (with the exception of job 0) must follow one other job

$$\sum_{i=0}^{N} \varphi_{ijw} = \tau_{jw} \qquad \forall j = 1,2,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{5.30}$$

No job is allowed to start before the previous job on the same machine has finished

$$S_j - S_i - M\varphi_{ijw} \geq d_i + t_{ij} - M \qquad \forall i = 0,1,...,N;\ \forall j = 1,2,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{5.31}$$

Each job should be finished within the interval

$$S_i + d_i \leq g \qquad \forall i = 1,2,...,N \tag{5.32}$$

A stream tube is activated if it supplies at least one outlet with water

$$\sum_{i=1}^{N} \tau_{iw} \leq \psi_w W \qquad \forall w = 1,2,...,W \tag{5.33}$$

The capacity of a channel may not be exceeded

$$q \sum_{w=1}^{W} \psi_w \leq Q \tag{5.34}$$

The following constraint is not essential to the model but is found to greatly reduce solution times.

$$\sum_{i=1}^{N} (\tau_{iw} d_i + \varphi_{ijw} t_{ij}) \leq g \qquad \forall j = 0,1,...,N; \forall w = 1,2,...,W \tag{5.35}$$

Model 7 is therefore defined by the objective function (5.21) and the constraints (5.22) - (5.35).

5.5 Analysis and development contiguous simple multi-machine scheduling with sequence dependent setup times

The contiguous simple multi-machine model with sequence dependent setup times, herein referred to as Model 8, is based on Model 7. The objective function and most of the constraints described for Model 7 remain valid. Some extra constraints are needed to ensure correct insertion of idle time.

5.5.1 Contiguous simple multi-machine scheduling with sequence dependent setup times: Model 8a

In Model 8a all idle time is inserted after the last job in the schedule has been completed, therefore no job should finish later than the sum of durations and sequence dependent setup times of all jobs processed on the same machine

$$S_i + d_i - M(1 - \tau_{iw}) \leq \sum_{n=0}^{N} (\tau_{nw} d_n) + \sum_{n=0}^{N} \sum_{j=0}^{N} \varphi_{njw} t_{nj} \qquad \forall i = 0,1,...,N; \ \forall w = 1,2,...,W; \ j \neq n \tag{5.36}$$

where τ_{nw} = binary variable; and φ_{njw} = binary variable. Model 8a is therefore defined by the objective function (5.21) and the constraints (5.22) - (5.36).

5.5.2 Contiguous simple multi-machine scheduling with sequence dependent setup times: Model 8b

Model 8b is similar to Model 8a with the difference that all idle time is inserted before the first job in the schedule starts. The amount of idle time inserted on each machine depends on the duration and sequence dependent setup times of the jobs processed on that machine. Therefore

$$X_{aw} = g - \sum_{i=0}^{N} d_i \tau_{iw} - \sum_{i=0}^{N}\sum_{j=0}^{N} \varphi_{ijw} t_{ij} \qquad \forall w = 1,2,...,W \tag{5.37}$$

$$S_i \geq X_{aw} + t_{0i} - M(1 - \tau_{iw}) \qquad \forall i = 1,2,...,N;\ \forall w = 1,2,...,W \tag{5.38}$$

Model 8 b is therefore defined by the objective function (5.21) and the constraints (5.22) - (5.35), (5.37) and (5.38).

5.5.3 Contiguous simple multi-machine scheduling with sequence dependent setup times: Model 8c

In model 8c idle time is allowed to be inserted both before the first job of a schedule starts and after the last job of the schedule is completed. The total amount of idle time inserted (before and after all jobs) on a machine depends on the sum of duration and sequence dependent setup time of all jobs processed by that machine

$$g = \sum_{i=0}^{N} \tau_{iw} d_i + \sum_{i=0}^{N}\sum_{j=0}^{N} (\varphi_{ijw} t_{ij}) + X_{aw} + X_{bw} \qquad \forall w = 1,2,...,N \tag{5.39}$$

Idle time can be inserted at the end of a schedule, but this may only be done after the last job of the schedule has been completed

$$S_i + d_i - M(1 - \tau_{iw}) \leq g - X_{bw} \qquad \forall i = 1,2,...,N;\ \forall w = 1,2,...,W \tag{5.40}$$

Idle time can be inserted at the beginning of a schedule, but this may only be done before the first job starts. therefore (5.38) is necessary. Model 8c is now defined by the objective function (5.21) and the constraints (5.22) - (5.35) and (5.38) - (5.40).

5.6 Timeliness

One of the most important activities of irrigation district managers is the timely delivery of water to tertiary units (Javan et al., 2002). Some irrigation districts have rules and policies regarding the late or early delivery of water, e.g water must be delivered within 24 or 48 hours of the requested time (e.g. Palmer et al., 1991; McGornick, 1993). Discharges in the channels can be increased, if the capacity of the channels and the availability of water permits this, as to allow more users to irrigate simultaneously. This increases the flexibility of a irrigation schedule and can reduce the earliness/tardiness. Thus it may be possible to observe the restrictions set on the early or late delivery of water. If the discharge can be increased sufficiently it is possible to give every user water exactly on the requested time. This could however lead to excessive waste of water or number of gate operations. The necessary increase in discharge depends on the requested start times and durations and can therefore only be determined together with the schedule. By combining the model that minimises the discharge and the models that minimize the earliness/tardiness it is possible to develop a model that allows restrictions on earliness/tardiness and minimises operational spillage at the same time.

Restrictions on maximum allowable earliness and tardiness can be formulated as follows

$$E_i \leq E_{\max} \qquad \forall i = 1,2,...,N \tag{5.41}$$

where E_{max} = maximum allowable earliness,

$$T_i \leq T_{\max} \qquad \forall i = 1,2,...,N \tag{5.42}$$

where T_{max} = maximum allowable tardiness. These constraints can be added to any of the simple multi-machine models described in previous paragraphs.

5.7 Results and discussion of simple multi-machine scheduling

Suryavanshi and Reddy (1986) described a tertiary unit in India with 8 users who are allowed to irrigate simultaneously. In their paper a schematic representation and irrigation

durations for this tertiary unit are given. Figure 5.3 shows this schematic and Table 5.2 shows the irrigation durations. Anwar and Clarke (2001) generated random target start times for the tertiary unit described by Suryavanshi and Reddy (1986). Table 5.2 also shows these target start times.

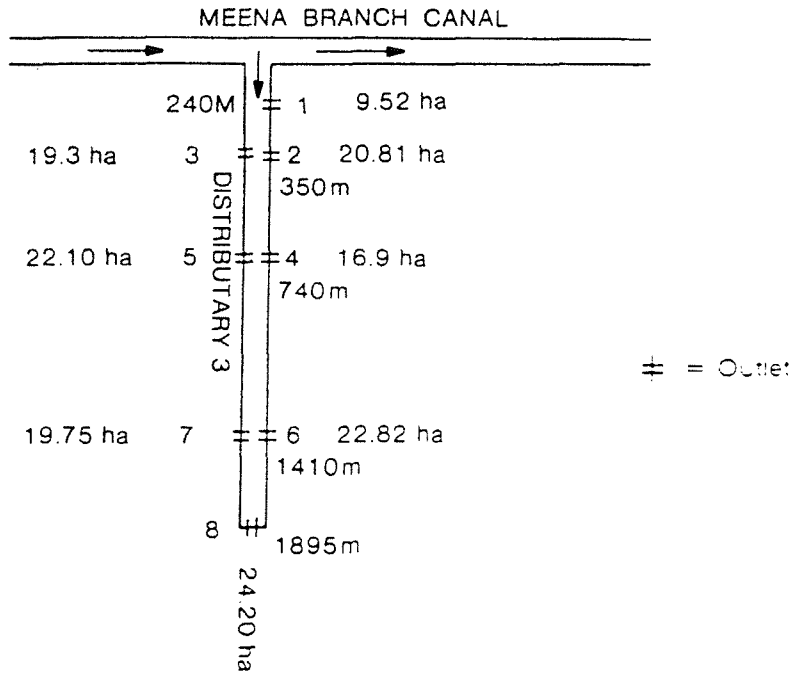


Figure 5.3 Tertiary unit, Kukadi Project, India (Suryavanshi and Reddy, 1986)

Table 5.2 Outlet data

Outlet number (1)	Irrigation duration (days) (2)	Target start time (days) (3)
1	0.80	3.55
2	2.13	0.41
3	2.40	2.16
4	1.72	1.49
5	2.05	0.61
6	2.43	0.26
7	2.05	1.60
8	2.50	3.03

Model 5, with and without earliness/tardiness restrictions, was implemented in Lingo 6.0® for Windows® using data from Table 5.2. The Lingo input files can be found in Appendix A. Figure 5.4 shows the schedule if no restrictions are imposed on the maximum allowable earliness and tardiness (this is the same result presented by Anwar and Clarke, 2001). It can be seen that 3 stream tubes (each of 30 l/s) are needed to supply the outlets with water.

The total earliness/tardiness for this schedule is 4.73 days, an average of 15.9 hours per outlet. It can also be seen that job 1 is scheduled to start 1.01 days earlier than requested and job 7 is scheduled to start 2.17 days later than requested, all other jobs have an earliness/tardiness of less than 1 day. Figure 5.5 shows the schedule if a maximum allowable earliness/tardiness of 24 hours per outlet is imposed. It can be seen that 4 stream tubes (each of 30 l/s) are needed to accommodate this constraint. The total earliness/tardiness of this schedule is 1.44 days, an average of 4.32 hours per outlet. None of the outlet have an earliness/tardiness of more than 1 day. The restrictions on maximum allowable earliness tardiness do however have disadvantages. The discharge in the channel needs to be increased from 90 l/s to 120 l/s. If the channel is operated continuously during the irrigation interval, 49% of the water is lost to operational spillage (volume of water lost as a proportion of that requested) compared to 12% operational water loss if no restrictions are imposed.

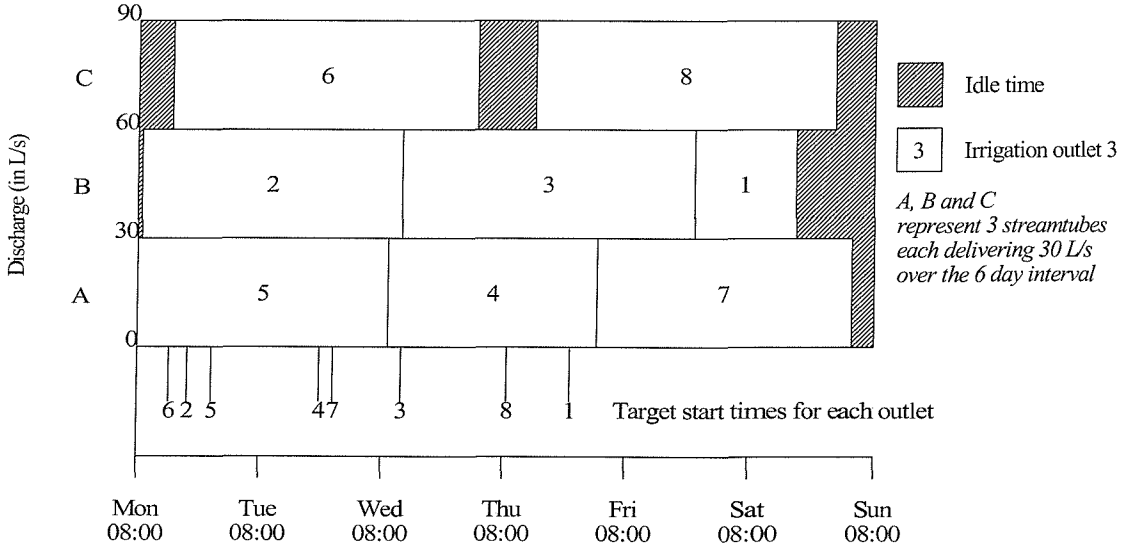


Figure 5.4 No restrictions on earliness/tardiness

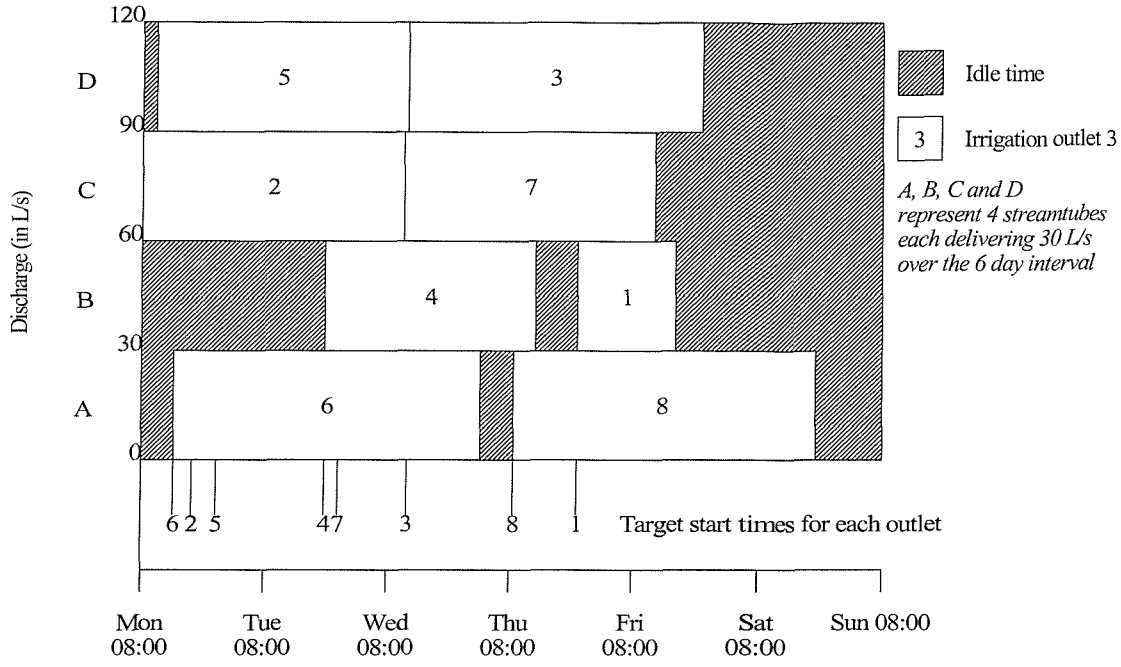


Figure 5.5 Timeliness restriction: water must be delivered within 24 hours of the target start time

As reported in section 3.5, solution times increase with the complexity of the model that is being solved. Including sequence dependent setup times into a model makes the model more complex and it was not possible to obtain solutions for Models 7, 8a, 8b or 8c within a reasonable time (<1day). Therefore no results are presented for these models.

5.8 Analysis and development of a two stage formulation for simple multi-machine scheduling

The models described in the previous paragraphs all have a dual goal objective function: not only the earliness/tardiness is minimised, but also the number of machines needed to process all jobs. It is possible to split these two goals into a model that is solved in two separate stages, the first one minimises the number of machines needed and the second minimises the earliness/tardiness. The advantage of two separate models rather than one combined model is that the number of variables and constraints is reduced and the models can be solved quicker. Separate models however do not allow simultaneous determination of number of machines and earliness/tardiness, as is needed when timeliness constraints and sequence dependent setup times apply. Therefore the use of the models described below is limited.

5.8.1 Integer programme to minimize the number of stream tubes

Wang et al. (1995) developed a model to determine the minimum number of machines (stream tubes) needed to provide all outlets with water. Let $\boldsymbol{Q} = \{1,2,...,N\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: duration and sequence independent setup time. Also let $V = \{1,2,...,W\}$ be a set of machines (stream tubes) available to supply water to outlets. As all jobs require equal discharge, the size of each machine is equal to this discharge. The aim of the model is to minimize the discharge in the channel, i.e. to minimize the number of activated machines

$$\text{minimize} \quad Z = \sum_{w=1}^{W} \psi_w \tag{5.43}$$

A variable is used to define whether a machine is used or not

$$\begin{aligned} \psi_w &= 1 \quad \text{if machine } w \text{ is activated} \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{5.44}$$

As any job can be processed by any machine a variable is used to define which job is processed by which machine

$$\begin{aligned} \tau_{iw} &= 1 \quad \text{if job } i \text{ is processed by machine } w \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{5.45}$$

No machine is allowed to run past the end of the irrigation interval

$$\sum_{i=1}^{N} d_i \tau_{iw} \leq g \qquad \forall w = 1,2,...,W \tag{5.46}$$

An outlet receives water from only one stream tube

$$\sum_{w=1}^{W} \tau_{iw} = 1 \qquad \forall i = 1,2,...,N \tag{5.47}$$

A machine is activated if it processes at least one job, i.e. it supplies at least one outlet

with water

$$\sum_{i=1}^{N} \tau_{iw} \leq \psi_w W \qquad \forall w = 1,2,...,W \tag{5.48}$$

The model to determine the minimum discharge required to supply all outlets with water is therefore defined by the objective function (5.43) and the constraints (5.45), (5.44), (5.46), (5.47) and (5.48).

5.8.2 Non-contiguous simple multi-machine scheduling

The second stage of the non-contiguous simple multi-machine model closely resembles Model 5 described previously. As the objective of the model is no longer to both minimise the number of machines and earliness/tardiness, but just to minimise earliness/tardiness, the second term of the objective function of Model 5 can be removed

$$\text{minimize } Z = \sum_{i=1}^{N} (\alpha_i E_i + \beta_i T_i) \tag{5.49}$$

As the number of stream tubes that is needed is determined during the first stage of the model (described in paragraph 5.5.1), constraints in Model 5 directly linked to this goal are unnecessary and (5.2) and (5.13) can be removed. The alternative Model 5 is therefore defined by the objective function (5.49) and the constraints (5.3), (5.4), (5.5) - (5.12), (5.14) and (5.15).

5.8.3 Contiguous simple multi-machine scheduling

The second stages of the contiguous simple multi-machine models closely resemble model 6a, 6b and 6c. The objective functions need to be modified in the same way as the alternative formulation of Model 5 and the same two constraints need to be removed. Therefore alternative Model 6a is defined by the objective function and the constraints (5.3), (5.4), (5.5) - (5.12) and (5.14) - (5.16). Model 6b is defined by the objective function

(5.49) and the constraints (5.3), (5.4), (5.5) - (5.12), (5.14), (5.15), (5.17) and (5.18). Model 6c is therefore defined by the objective function (5.49) and the constraints (5.3), (5.4), (5.5) - (5.12), (5.14), (5.15) and (5.18) - (5.20)

5.8.4 Comparison single stage and two stage formulation

The single and two stage models are essentially the same, the difference lies in the number of steps involved in solving the models. The results from both the single and two stage formulation are therefore the same. Table 5.3 shows the number of constraints the number of variables and constraints for each model for both the single and two stage formulations

Table 5.3: Number of variable and constraints for models 5, 6a, 6b and 6c[1]

Model	Formulation	Number of variables	Number of constraints
(1)	(2)	(3)	(4)
5	single stage	$N^2W + NW + 3N + W$	$N^2W + 3N + 2W$
	first stage	$NW + W$	$NW + W + N + 1$
	second stage	$N^2W + NW + 3N$	$N^2W + 3N + W$
6a	single stage	$N^2W + NW + 3N + W$	$N^2W + NW + 3N$
	first stage	$NW + W$	$NW + W + N + 1$
	second stage	$N^2W + NW + 3N$	$N^2W + NW + 3N + W$
6b	single stage	$N^2W + NW + 3N + 2W$	$N^2W + NW + 3N + 2W$
	first stage	$NW + W$	$NW + W + N + 1$
	second stage	$N^2W + NW + 3N + W$	$N^2W + NW + 3N + W$
6c	single stage	$N^2W + NW + 3N + 3W$	$N^2W + 2NW + 3N + 2W$
	first stage	$NW + W$	$NW + W + N + 1$
	second stage	$N^2W + NW + 3N + 3W$	$N^2W + 2NW + 3N$

[1] N = number of jobs; and W = number of stream tubes

At first sight both formulations have a number of variables and constraints in the order of N^2W. By applying the two stages model however it is possible to reduce the number of stream tubes and thereby the number of variables and constraints. The tertiary unit described in Section 5.7 can be used to demonstrate this reduction. The maximum number of stream tubes that could be required to supply all users with water is one for each user, in

this case 8. This means that if single stage Model 5 is used to obtain a schedule, the model has 608 variables and 552 constraints. When the two stages Model 5 is used, the first stage will minimise the number of stream tube required to supply all users with water. This stage of the model has 72 variables and 81 constraints. The outcome of this stage is that 3 stream tubes is enough to supply all users with water. The second stage of the model which minimises the earliness/tardiness now has 240 variables and 219 constraints for the two stages model the total number of variables is 312 and the total number of constraints is 300. Compared to the single stage model this is a reduction of 48% in the number of variables and a 46% reduction in the number of constraints. Table 5.4 shows the computation times for the example

Table 5.4: Computation times (in seconds) for Model 5, 6a, 6b and 6c

Formulation		Model			
		5	6a	6b	6c
(1)		(2)	(3)	(4)	(5)
single stage model		>20000	>20000	>20000	>20000
two stage model	first stage	<1	<1	<1	<1
	second stage	115	63	66	67
	total	115	63	66	67

It can be seen in Table 5.4 that there is a large difference between the computation times of the single stage models, none of which solved in 5.5 hours (20000 seconds) and the two stage models, all of which solved in less then 2 minutes.

5.9 Conclusions simple multi-machine scheduling

Several models are presented in this chapter. The models allow both non-contiguous and contiguous simple multi-machine scheduling without and with sequence dependent setup times. It is also shown how timeliness of water delivery can be incorporated into the models. By applying the models to a tertiary unit it is shown how the earliness/tardiness of a model can be reduced, however at the cost of increased operational spillage. There are several disadvantages to the models developed in this chapter.

1. The discharge of the outlets are assumed to be identical, but this is not necessarily

always true. This issue will be addressed in the next chapter, Chapter 6.

2. Sequence dependent setup times are calculated as if the channel is always empty. In reality the channel may be partially filled. This can cause sequence dependent setup times to be less than calculated. Further research into this issue is needed.
3. Solution times increase with the number of jobs to be scheduled and as a result only smaller problems can be solved. Solution techniques such as heuristics or genetic algorithms may be more appropriate to solve larger problems.
4. It is assumed that the dimensions along the length of a channel remain constant. This may be true, but in many cases channels become smaller towards the tail-end. This may cause capacity problems if several tail-end users are scheduled to irrigate simultaneously. This issue will need to be investigated in future.
5. It is assumed sufficient water resources are available to meet the irrigation demands. More research is necessary to study the effect of a less than adequate water supply. Possible solutions could be imposing an upper limit on the requested discharge or adjusting the duration.

6 Complex multi-machine scheduling

In multi-machine scheduling water is modelled as several (imaginary) stream tubes. In simple multi-machine scheduling each job is processed by one machine (stream tube) and each machine can process several jobs in sequence. The limitation of such a model is that all outlets need to have an identical discharge for simple multi-machine scheduling to work. In many irrigation systems it is possible for users not only to ask for a specific duration and start time for their irrigation turn, but also for a specific discharge. This means that the discharge can vary from user to user. In complex multi-machine scheduling two or even more machines are allowed to process the same job, thereby allowing different discharges to be delivered to each user. Figure 6.1 illustrates the difference between simple multi-machine scheduling and complex multi-machine scheduling.

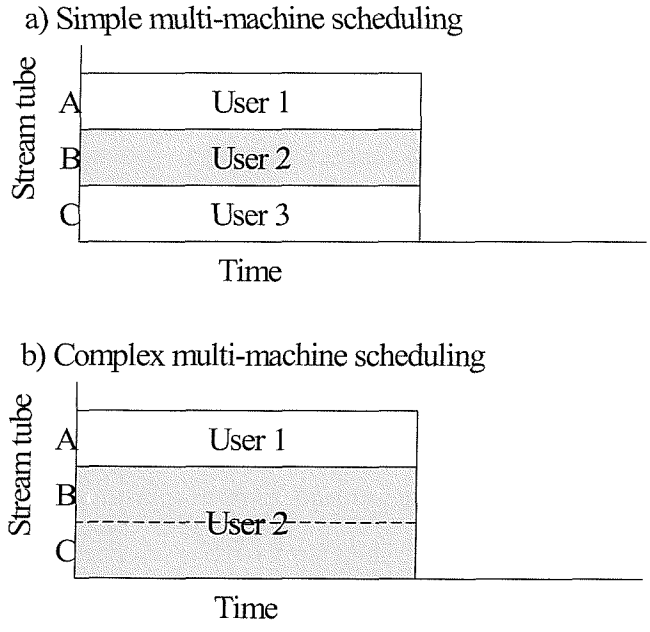


Figure 6.1 Comparison simple and complex multi-machine scheduling

Figure 6.1a shows simple multi-machine scheduling. Three stream tubes, A, B and C, each of equal discharge, are available. Stream tube A supplies outlet 1, stream tube B supplies outlet 2 and stream tube C supplies outlet 3. All outlet receive the same discharge. Figure 6.1b shows complex multi-machine scheduling. Again three stream tubes, A, B and C, each of equal size are available. Stream tube A supplies outlet 1, and stream tubes B and C

supply outlet 2. Outlet 2 receives twice the discharge of outlet 1.

In this chapter the models for complex multi-machine scheduling and complex multi-machine scheduling with sequence dependent setup times will be developed. The highlighted area of Table 6.1 shows the models to be developed in this chapter:

Table 6.1: Overview of different models and their features

(1)	Model (2)	Non- contiguous (3)	Contiguous: Idle time after all jobs (4)	Contiguous: Idle time before all (5)	Contiguous: Idle time before and/or after all (6)	Setup times (7)	Sequential irrigation (8)	Simultaneous irrigation (9)	Simple multi-machine (10)	Complex multi-machine (11)
Multi-machine model	5	✓					✓	✓	✓	
	6		✓	✓	✓		✓	✓	✓	
	7	✓				✓	✓	✓	✓	
	8		✓	✓	✓	✓	✓	✓	✓	
	9	✓					✓	✓	✓	✓
	10a		✓				✓	✓	✓	✓
	10b			✓			✓	✓	✓	✓
	10c				✓		✓	✓	✓	✓
	11	✓				✓	✓	✓	✓	✓
	12a		✓			✓	✓	✓	✓	✓
	12b			✓		✓	✓	✓	✓	✓
	12c				✓	✓	✓	✓	✓	✓

6.1 Analysis and development non-contiguous complex multi-machine scheduling model

The non-contiguous complex multi-machine model, herein referred to as Model 9, allows scheduling of a number of jobs (irrigation events) according to their target start

times. In this model more than one machine (stream tube) is available to process each of the jobs, i.e. users are allowed to irrigate simultaneously. A fictitious job with index 0 is used to simplify writing of the constraints. Let $\boldsymbol{Q} = \{0,1,2,...,N\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: duration, target start time, earliness cost, tardiness cost and required discharge. Also let $\boldsymbol{V} = \{1,2,...,W\}$ be a set of machines available to process jobs. The discharge for each machine $w \in \boldsymbol{V}$ is specified. In Model 9 there are two decisions to be made, which machine processes which job and what is the scheduled start time of each job. If the answers to these questions are known, the schedule is known too. The objective of the model is to find a schedule such that every job starts as close as possible to the target start time. This is achieved by minimising the penalties incurred when a job is either early or tardy. The objective function can be written as

$$\text{Minimize } Z = \sum_{i=1}^{N} (\alpha_i E_i + \beta_i T_i) + c \sum_{w=1}^{W} \psi_w \tag{6.1}$$

As any job can directly precede any other job on a machine a variable is used to define which job precedes which other job on what machine

$$\begin{aligned} \varphi_{ijw} &= 1 \quad \text{if job } i \text{ directly precedes job } j \text{ on machine } w \\ &= 0 \quad \text{otherwise} \end{aligned} \qquad i \neq j \tag{6.2}$$

A job can be processed by any machine, so a variable is used to define which job is processed by which machine

$$\begin{aligned} \tau_{iw} &= 1 \quad \text{if job } i \text{ is processed by machine } w \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{6.3}$$

The following variable defines whether a variable is used or not

$$\begin{aligned} \psi_w &= 1 \quad \text{if machine } w \text{ is activated} \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{6.4}$$

The scheduled start time of a job is determined from the target start time, the earliness and the tardiness

$$S_i = r_i + T_i - E_i \qquad \forall i = 1,2,...,N \tag{6.5}$$

It is not possible for a schedule to incur a negative earliness or tardiness (in fact a negative tardiness is the same as a positive earliness and vice versa), therefore the following constraints need to be satisfied

$$T_i \geq 0 \qquad \forall i = 1,2,...,N \tag{6.6}$$

$$E_i \geq 0 \qquad \forall i = 1,2,...,N \tag{6.7}$$

Each outlet must receive the required discharge, therefore each job must be processed by the right number of machines

$$q \sum_{w=1}^{W} \tau_{iw} = Q_i \qquad i = 1,2,...,N \tag{6.8}$$

where Q_i = discharge required for job i. Each job can at most precede one other job

$$\sum_{j=1}^{N} \varphi_{ijw} \leq \tau_{iw} \qquad \forall i = 0,1,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{6.9}$$

Each job (with the exception of job 0) must follow one other job

$$\sum_{i=0}^{N} \varphi_{ijw} = \tau_{jw} \qquad \forall j = 1,2,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{6.10}$$

No job is allowed to start before the previous job on the same machine has finished

$$S_j - S_i - M\varphi_{ijw} \geq d_i - M \qquad \forall i = 0,1,...,N;\ \forall j = 1,2,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{6.11}$$

Each job should be finished within the interval

$$S_i + d_i \leq g \qquad \forall i = 1,2,...,N \tag{6.12}$$

The capacity of a canal may not be exceeded

$$q \sum_{w=1}^{W} \psi_w \leq Q_c \tag{6.13}$$

Any restrictions on maximum allowable earliness and/or tardiness can be accommodated as follows

$$E_i \leq E_{\max} \qquad \forall i = 1,2,...,N \tag{6.14}$$

$$T_i \leq T_{\max} \qquad \forall i = 1,2,...,N \tag{6.15}$$

A stream tube is activated if it supplies at least one outlet with water

$$\sum_{i=1}^{N} \tau_{iw} \leq \psi_w W \qquad \forall w = 1,2,...,W \tag{6.16}$$

The following constraint is not essential to the model but is found to greatly reduce solution times. For each machine the sum of durations must be less than or equal to the length of the interval.

$$\sum_{i=1}^{N} d_i \tau_{iw} \leq g \qquad \forall w = 1,2,...,W \tag{6.17}$$

Model 9 is therefore defined by the objective function (6.1) and the constraints (6.2) - (6.17).

6.2 Analysis and development contiguous complex multi-machine scheduling model

The contiguous complex machine scheduling model, herein to be referred to as Model 10

is based on Model 9. The decision variables, objective function and constraints described for this model remain valid. Some additional constraints are needed to ensure correct insertion of idle time.

6.2.1 Contiguous complex multi-machine scheduling: Model 10a

To ensure all idle time is inserted after the last job in the schedule is completed, no job should finish later than the sum of the durations of all job processed on the same machine

$$S_i + d_i - M(1-\tau_{iw}) \leq \sum_{i=1}^{N} d_i\tau_{iw} \qquad \forall i = 0,1,...,N;\ \forall w = 1,2,...,W \tag{6.18}$$

Model 10a is therefore defined by the objective function (6.1) and the constraints (6.2) - (6.18).

6.2.2 Contiguous complex multi-machine scheduling: Model 10b

Model 10b is similar to Model 10a with the difference that all idle time is inserted before the first job is scheduled to start. The amount of idle time inserted on each machine depends on the duration of the jobs processed by that machine. Therefore

$$X_{aw} = g - \sum_{i=1}^{N} d_i\tau_{iw} \qquad \forall w = 1,2,...,W \tag{6.19}$$

Idle time on each machine is inserted before the all jobs. Therefore

$$S_i + M(1-\tau_{iw}) \geq X_{aw} \qquad \forall i = 1,2,...,N;\ \forall w = 1,2,...,N \tag{6.20}$$

Model 10b is therefore defined by the objective function (6.1) and the constraints (6.2) - (6.17), (6.19) and (6.20).

6.2.3 Contiguous complex multi-machine scheduling: Model 10c

In Model 10c idle time is allowed to be inserted both before the first job in the schedule starts and after the last job in the schedule is finished. The total amount of idle time inserted (before and after all jobs) on a machine depends on the duration of the jobs processed by that machine

$$g = \sum_{i=1}^{N} d_i \tau_{iw} + X_{aw} + X_{bw} \qquad \forall i = 0,1,...,N;\ \forall w = 1,2,...,W \tag{6.21}$$

If idle time is inserted at the end of a schedule this may only be done after all jobs have been completed

$$S_i + d_i - M(1 - \tau_{iw}) \leq g - X_{bw} \qquad \forall i = 0,1,...,N;\ \forall w = 1,2,...,W \tag{6.22}$$

If idle time is inserted at the beginning of a schedule this may only be done before the first job starts. Therefore constraint (6.20) is also necessary. Model 10c is therefore defined by the objective function (6.1) and the constraints (6.2) - (6.17) and (6.20) - (6.22).

6.3 Analysis and development of a two stage formulation for complex multi-machine scheduling

The models described in the previous paragraphs all have a dual goal objective function: not only the earliness/tardiness is minimised, but also the number of machines needed to process all jobs. As with the models described in Chapter 5, it is possible to split these two goals into a model that is solved in two separate stages, the first stage minimises the number of machines needed and the second minimises the earliness/tardiness. The advantage of two separate models rather than one combined model is that the number of variables and constraints is reduced. Separate models however do not allow simultaneous determination of number of machines and earliness/tardiness, as is needed when maximum allowable earliness/tardiness restrictions apply. Therefore the use of the models described below is limited.

6.3.1 Integer programme to minimize the number of stream tubes

Wang et al. (1995) developed a model to determine the minimum number of machines (stream tubes) needed to provide all outlets with water. This model is intended for use in simple multi-machine scheduling. With some enhancement it can also be used for complex multi-machine scheduling. Let $\boldsymbol{Q} = \{1,2,...,N\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: duration, sequence independent setup time and required discharge. Also let $\boldsymbol{V} = \{1,2,...,W\}$ be a set of machines (stream tubes) available to supply water to outlets. The discharge for each machine $w \in \boldsymbol{V}$ is specified. The aim of the model is to minimize the discharge in the channel, i.e. to minimize the number of activated machines

$$\text{minimize} \quad Z = \sum_{w=1}^{W} \psi_w \tag{6.23}$$

As any job may be processed by any machine a variable is used to define which job is processed by which machine

$$\begin{aligned} \tau_{iw} &= 1 \quad \text{if job } i \text{ is processed by machine } w \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{6.24}$$

A variable is used to define whether a machine is used or not

$$\begin{aligned} \psi_w &= 1 \quad \text{if machine } w \text{ is activated} \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{6.25}$$

No machine is allowed to run past the end of the irrigation interval

$$\sum_{i=1}^{N} d_i \tau_{iw} \leq g \qquad \forall w = 1,2,...,W \tag{6.26}$$

Each outlet must receive the required discharge, therefore each job must be processed by the right number of machines

$$q \sum_{w=1}^{W} \tau_{iw} = Q_i \qquad \forall i = 1,2,...,N \tag{6.27}$$

A machine is activated if it processes at least one job, i.e. it supplies at least one outlet with water

$$\sum_{i=1}^{N} \tau_{iw} \leq \psi_w W \qquad \forall w = 1,2,...,W \tag{6.28}$$

The first stage of the model that determines the minimum discharge required to supply all outlets with water is therefore defined by the objective function (6.23), and the constraints (6.24), (6.25), (6.26), (6.27) and (6.28).

6.3.2 Non-contiguous complex multi-machine scheduling

The separate non-contiguous simple multi-machine model closely resembles Model 9 described previously. As the objective of the model is no longer to both minimise the number of machines and earliness/tardiness, but just to minimise earliness/tardiness, the second term of the objective function of Model 9 can be removed

$$\text{minimize } Z = \sum_{i=1}^{N} (\alpha_i E_i + \beta_i T_i) \tag{6.29}$$

As the number of stream tubes that is needed is determined with the model described in paragraph 6.3.1, constraints in Model 9 directly linked to this goal are unnecessary and (6.4) and (6.16) can be removed. Maximum allowable earliness/tardiness cannot be accommodated, so constraints (6.14) and (6.15) can also be removed. The second stage of the model for non-contiguous complex multi-machine scheduling is therefore defined by the objective function (6.29) and the constraints (6.2), (6.3), (6.5) - (6.13) and (6.17).

6.3.3 Contiguous complex multi-machine scheduling

The second stages of the contiguous complex multi-machine models closely resemble model 6a, 6b and 6c. The objective functions need to be modified in the same way as the second stages of the two-stage formulation of Model 5 and the same four constraints need to be removed. Therefore the second stage of Model 6a is defined by the objective function (6.29) and the constraints (6.2), (6.3), (6.5) - (6.13), (6.17) and (6.18). The second stage of Model 6b is defined by the objective function (6.29) and the constraints (6.2), (6.3), (6.5) - (6.13), (6.17), (6.19) and (6.20). Finally the second stage of Model 6c is defined by the objective function (6.29) and the constraints (6.2), (6.3), (6.5) - (6.13), (6.17) and (6.20) - (6.22).

6.4 Results and discussion complex multi-machine scheduling

The models for complex multi-machine scheduling are computationally very demanding and only small problems can be solved. Although in the literature several tertiary units have been described that could be used for applying the models developed in this chapter, all these tertiary units are too large to be solved in a reasonable time (< 7 days). The tertiary unit described by Suryavanshi and Reddy (1986), described in the previous chapter, can be used as a basis to show a typical complex multi-machine schedule. The required discharge of two randomly chosen outlets (outlet 2 and 5) are doubled and tripled to 60 L/s and 90 L/s respectively. If one stream tube delivers 30 L/s, this means that outlet 2 needs to be processed by 2 machines and outlet 5 by 3. The alternative formulation for Model 9 will be used to find the optimal schedule. This alternative formulation consists of two models, the first minimises the discharge in the channel and the second minimises the earliness/tardiness.

The alternative formulation for Model 9 was implemented in Xpress^{MP}® for Windows®. From the first stage model follows that 5 stream tubes delivering 150 L/s are needed to supply the 8 outlets with the required discharge. Figure 6.2 shows the schedule obtained after applying the both stages of the alternative formulation of Model 9. It can be seen in Figure 6.2 that 3 stream tubes (B, C and D) each deliver 30 L/s to outlet giving 90 L/s as required. It can also be seen that 2 stream tubes (B and C) each deliver 30 L/s to

outlet 2, giving a total of 60 L/s. Finally it can be seen that the remaining 6 outlets each receive 30 L/s delivered by one stream tube.

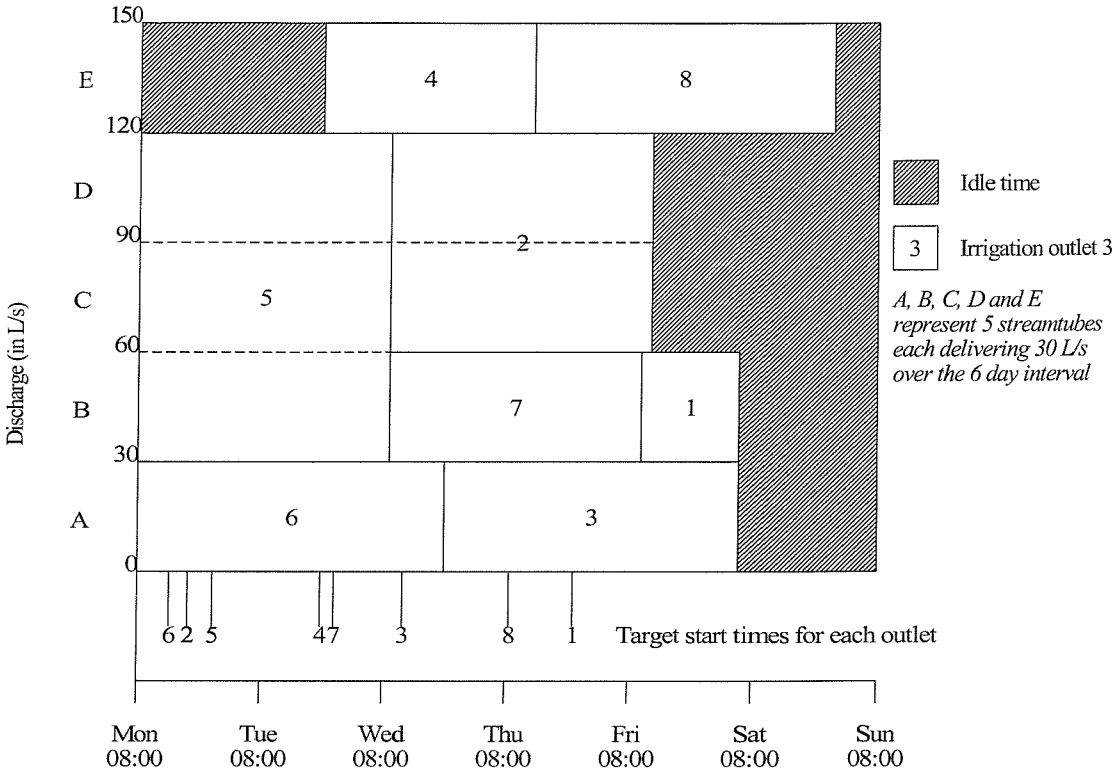


Figure 6.2 Schedule applying Model 9

As with the models developed in the previous chapters, here too the solution times increase with the complexity of the models. It was not possible to obtain solutions for any other model, but the alternative formulation of Model 9.

6.5 Non-contiguous complex multi-machine scheduling with sequence dependent setup times

The non-contiguous complex multi-machine model, herein referred to as Model 11, allows scheduling of a number of jobs (irrigation events) according to their target start times. In this model more than one machine (stream tubes) is available to process the jobs, i.e. users are allowed to irrigate simultaneously. A fictitious job with index 0 is used to simplify writing of the constraints. Let $\boldsymbol{Q} = \{0,1,2,...,N\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: duration, target start time, earliness

cost, tardiness cost, required discharge, sequence dependent setup times. Also let $V = \{1,2,...,W\}$ be a set of machines available to process jobs. The discharge for each machine $w \in V$ is specified. In Model 11 there are two decisions to be made, which machine processes which job and what is the scheduled start time of each job. If the answers to these questions are known, the schedule is known too. The objective of the model is to find a schedule such that every job starts as close as possible to the target start time. This is achieved by minimising the penalties incurred when a job is either early or tardy. The objective function can be written as

$$\text{Minimize } Z = \sum_{i=1}^{N} (\alpha_i E_i + \beta_i T_i) + c \sum_{w=1}^{W} \psi_w \tag{6.30}$$

As any job can directly precede any other job on a machine a variable is used to define which job precedes which other job on what machine

$$\begin{aligned} \varphi_{ijw} &= 1 \quad \text{if job } i \text{ directly precedes job } j \text{ on machine } w \\ &= 0 \quad \text{otherwise} \end{aligned} \qquad i \neq j \tag{6.31}$$

A job can be processed by any machine, so a variable is used to define which job is processed by which machine

$$\begin{aligned} \tau_{iw} &= 1 \quad \text{if job } i \text{ is processed by machine } w \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{6.32}$$

The following variable defines whether is machine is activated or not

$$\begin{aligned} \psi_w &= 1 \quad \text{if machine } w \text{ is activated} \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{6.33}$$

The scheduled start time of a job is determined from the target start time, the earliness and the tardiness

$$S_i = r_i + T_i - E_i \qquad \forall i = 1,2,...,N \tag{6.34}$$

It is not possible for a schedule to incur a negative earliness or tardiness, therefore the following constraints need to be satisfied

$$T_i \geq 0 \qquad \forall i = 1,2,...,N \tag{6.35}$$

$$E_i \geq 0 \qquad \forall i = 1,2,...,N \tag{6.36}$$

Each outlet must receive the required discharge, therefore each job must be processed by the right number of machines

$$q \sum_{w=1}^{W} \tau_{iw} = Q_i \qquad i = 1,2,...,N \tag{6.37}$$

Each job can at most precede one other job

$$\sum_{j=1}^{N} \varphi_{ijw} \leq \tau_{iw} \qquad \forall i = 0,1,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{6.38}$$

Each job (with the exception of job 0) must follow one other job

$$\sum_{i=0}^{N} \varphi_{ijw} = \tau_{jw} \qquad \forall j = 1,2,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{6.39}$$

No job is allowed to start before the previous job on the same machine has finished

$$S_j - S_i - M\varphi_{ijw} \geq d_i + t_{ij} - M \qquad \forall i = 0,1,...,N;\ \forall j = 1,2,...,N;\ \forall w = 1,2,...,W;\ i \neq j \tag{6.40}$$

Each job should be finished within the interval

$$S_i + d_i \leq g \qquad \forall i = 1,2,...,N \tag{6.41}$$

The capacity of a channel may not be exceeded

$$q \sum_{w=1}^{W} \psi_w \leq Q \tag{6.42}$$

A stream tube is activated if it supplies at least one outlet with water

$$\sum_{i=1}^{N} \tau_{iw} \leq \psi_w W \qquad \forall w = 1,2,...,W \tag{6.43}$$

The following constraint is not essential to the model but is found to greatly reduce solution times. For each machine the sum of durations and sequence independent setup times must be less than or equal to the length of the interval.

$$\sum_{i=1}^{N} (\tau_{iw} d_i + \varphi_{ijw} t_{ij}) \leq g \qquad \forall w = 1,2,...,W \tag{6.44}$$

Model 11 is therefore defined by the objective function (6.30) and the constraints (6.31) - (6.44).

6.6 Contiguous complex multi-machine scheduling with sequence dependent setup times

The contiguous complex machine scheduling model with sequence dependent setup times, herein referred to as Model 12 is based on Model 11. The objective function and constraints described for this model remain valid. Some extra constraints are necessary to ensure correct insertion of idle time.

6.6.1 Contiguous complex multi-machine scheduling with sequence dependent setup times: Model 12a

To ensure all idle time is inserted after the last job in the schedule is finished, no job should finish later than the sum of the durations and sequence dependent setup times of all jobs processed by the same machine.

$$S_i + d_i - M(1 - \tau_{iw}) \leq \sum_{n=1}^{N} \tau_{nw} d_n + \sum_{n=0}^{N} \sum_{j=0}^{N} \varphi_{njw} t_{nj} \qquad \forall i = 0,1,...,N;\ \forall w = 1,2,...,N \quad (6.45)$$

Model 12a is therefore defined by the objective function (6.30) and the constraints (6.31) - (6.45).

6.6.2 Contiguous complex multi-machine scheduling with sequence dependent setup times: Model 12b

Model 12b is similar to Model 12a with the difference that all idle time is inserted before the first job in the schedule starts. The amount of idle time inserted on each machine depends on the duration and sequence dependent setup times of the jobs processed by that machine.

$$X_{aw} = g - \sum_{i=0}^{N} \tau_{iw} d_i - \sum_{i=0}^{N} \sum_{j=0}^{N} \varphi_{ijw} t_{ij} \qquad \forall w = 1,2,...,W \quad (6.46)$$

Job may only start after the inserted idle time has passed

$$S_i \geq X_{aw} + t_{0i} - M(1 - \tau_{iw}) \qquad \forall i = 1,2,...,N;\ \forall w = 1,2,...,W \quad (6.47)$$

Model 12b therefore is defined by the objective function (6.30) and the constraints (6.31) - (6.44), (6.46) and (6.47).

6.6.3 Contiguous complex multi-machine scheduling with sequence dependent setup times: Model 12c

In Model 12c idle time can be inserted both before the first job in the schedule has started and after the last job in the scheduled is completed. The total amount of idle time inserted (before and after all jobs) on a machine depends on the sum of durations and sequence dependent setup times of all jobs processed by that machine

$$g = \sum_{i=0}^{N} \tau_{iw} d_i + \sum_{i=0}^{N} \sum_{j=0}^{N} \varphi_{ijw} t_{ij} + X_{bw} + X_{bw} \qquad \forall w = 1,2,...,W \tag{6.48}$$

Idle time can be inserted at the end of a schedule but this may only be done after the last job of the schedule has been completed.

$$S_i + d_i - M(1 - \tau_{iw}) \leq g - X_{bw} \qquad \forall i = 1,2,...,N;\ \forall w = 1,2,...,W \tag{6.49}$$

If idle time is inserted at the beginning of a schedule this may only be done before the first job in the schedule starts. Therefore constraint (6.47) is also necessary. Model 12c is therefore defined by the objective function (6.30) and the constraints (6.31) - (6.44) and (6.47) - (6.49).

6.7 Conclusions complex multi-machine scheduling

It has been said before that an irrigation department is no different from any other organisation in that limited resources have to be allocated taking into account certain demands and constraints. There is however, one big difference, where in other machine scheduling situations the machines are real and limited in number, water can be divided into any number of imaginary stream tubes. By allowing each job (outlet) to be processed by more than one machine (stream tube) at the same time, it is possible to deliver virtually any discharge to an outlet. This unique feature enables scheduling of outlets that have varying discharges. The complex multi-machine models presented in this chapter take advantage of this concept. The models allow both non-contiguous and contiguous complex multi-machine scheduling. The models can also take into account managements policies on maximum allowable earliness and/or tardiness. Sequence dependent setup times are incorporated into one set of models. By applying one of the models to a tertiary unit the principle of complex multi-machine scheduling is shown. There are some disadvantages to the models developed in this chapter.

1. Sequence dependent setup times are calculated as if the channel is always empty. In reality the channel may be partially filled. This can cause sequence dependent setup times to be less than calculated. Further research into this issue is needed.

2. Solution times increase with the number of jobs to be scheduled and as a result only smaller problems can be solved. Solution techniques such as heuristics or genetic algorithms may be more appropriate to solve larger problems.
3. It is assumed that the dimensions along the length of a channel remain constant. This may be true, but in many cases channels become smaller towards the tail-end. This may cause capacity problems if several tail-end users are scheduled to irrigate simultaneously. This issue will need to be investigated in future.

7 Multi-interval scheduling

In multi-interval (or multi-period) scheduling decisions made in a previous interval can be used to influence the decisions that will be made for the current interval. In irrigation scheduling the earliness and tardiness incurred in earlier intervals can be used to give priority to those users who have been most disadvantaged. Any of the models developed in the previous chapters can be used for both single and multi-interval scheduling, with the exception of the alternative formulation of Models 1, 2a, 2b and 2c presented in section 3.4, as they do not allow individual earliness/tardiness costs for each job .

Anwar and Clarke (2001) described a method to determine the weighting factor for multi-interval scheduling, based on the lead/lag times. Lead/lag time was defined as the absolute difference between scheduled and target start time and is therefore equal to the sum of earliness and tardiness. If earliness is equally as undesirable as tardiness, the earliness cost per unit of time of job j can be set equal to the tardiness cost per unit of time of job j. The equation for earliness/tardiness costs given by Anwar and Clarke (2001) is expressed in earliness/tardiness form as

$$\alpha_{iP} = \beta_{iP} = \frac{\sum_{p=1}^{P-1} (E_{ip} + T_{ip})}{\sum_{i=1}^{N} \left(\sum_{p=1}^{P-1} (E_{ip} + T_{ip}) \right)} \times N \qquad \forall i = 1,2,...,N \tag{7.1}$$

Where $\alpha_{i,p}$ = earliness cost of job of job i in interval p; $\beta_{i,p}$ = tardiness cost of job i in interval p; E_{ip} = earliness of job i in interval p; T_{ip} = tardiness of job i in interval p; p = index representing interval 1,...,P; and P = current interval. If either earliness or tardiness is more undesirable than the other it is possible to adjust the earliness and tardiness costs accordingly. This could be done for instance by multiplying the more undesirable costs by a certain factor that reflects the undesirability. It is believed however that in irrigation scheduling the costs earliness and tardiness are equal and that there is no need for such adjustment.

The method of determining the costs of earliness/tardiness for each job described by Anwar and Clarke (2001) works well in most situations, but a problem arises when one or more jobs in the first interval is scheduled to start at exactly it's target start time (i.e. scheduled with no earliness/tardiness). For a job with no earliness or tardiness, the numerator in (7.1) becomes zero. This sets the costs of earliness/tardiness for this job to

zero. With a cost of earliness/tardiness of zero, this would allow the job to be scheduled anywhere since such a job does not contribute to the objective value.

Anwar and Clarke (2001) used the standard deviation as an index of the quality of a schedule. This is no problem if all of the users irrigate all of the time, but this does not recognise that users may abstain during one or more intervals. This may give misleading results.

In this chapter a improved method for determining the earliness/tardiness cost will be developed. A new way of measuring the quality of a schedule will be presented, which takes into account that sometimes, especially early or late in a season, users may wish to skip one or more irrigation turns.

7.1 Analysis and development improved method of determining earliness/tardiness costs

The method of determining the costs of earliness/tardiness for each job described by Anwar and Clarke (2001) works well in most situations, but a problem arises when one or more jobs in the first interval is scheduled to start at exactly its target start time (i.e. scheduled with no earliness/tardiness). For a job with no earliness or tardiness, the numerator in (7.1) becomes zero. This sets the costs of earliness/tardiness for this job to zero. With a cost of earliness/tardiness of zero, this would allow the job to be scheduled anywhere since such a job does not contribute to the objective value. Replacing (7.1) by an exponential function will set the costs of earliness and tardiness to 1 for all jobs with zero earliness/tardiness. This ensures that all jobs contribute to the objective value. Therefore (7.1) becomes

$$\alpha_{iP} = \beta_{iP} = e^{\left(\frac{\sum_{p=1}^{P-1} (E_{ip} + T_{ip})}{\sum_{i=1}^{N} \left(\sum_{p=1}^{P-1} (E_{ip} + T_{ip}) \right)} \times N \right)} \qquad \forall i = 1,2,...,N \tag{7.2}$$

7.2 Results and discussion improved method for determining earliness/tardiness costs

Anwar and Clarke (2001) used the interval- averaged standard deviation (standard

deviation index) to investigate the usefulness of earliness/tardiness costs. A value of 0 indicates that all users have an equal amount of earliness/tardiness, i.e. the schedule is fair. The higher the value of the standard deviation index the less fair the schedule. To investigate the differences between using costs of earliness/tardiness estimated by (7.1) and (7.2) the standard deviation index of the earliness and tardiness can be used. Figure 7.1 compares the effect of using (7.1) and (7.2) on the standard deviation index using Model 1. Target start times and duration are randomly generated but remain constant for all intervals and in the first interval at least one job is scheduled to start at the target start time. Although in reality it is highly unlikely that all users always ask for the same duration and target start times, this example clearly shows the need for using earliness/tardiness costs. Figure 7.1 also shows the behaviour of the standard deviation index if the costs of earliness/tardiness are set to 1 for all intervals (single interval model). It is clear that if (7.1) is used to calculate the costs of earliness and tardiness the standard deviation index increases initially. Since the input data remains constant the standard deviation index for the single interval model also remains constant. In contrast the standard deviation index using (7.2) consistently decreases with interval.

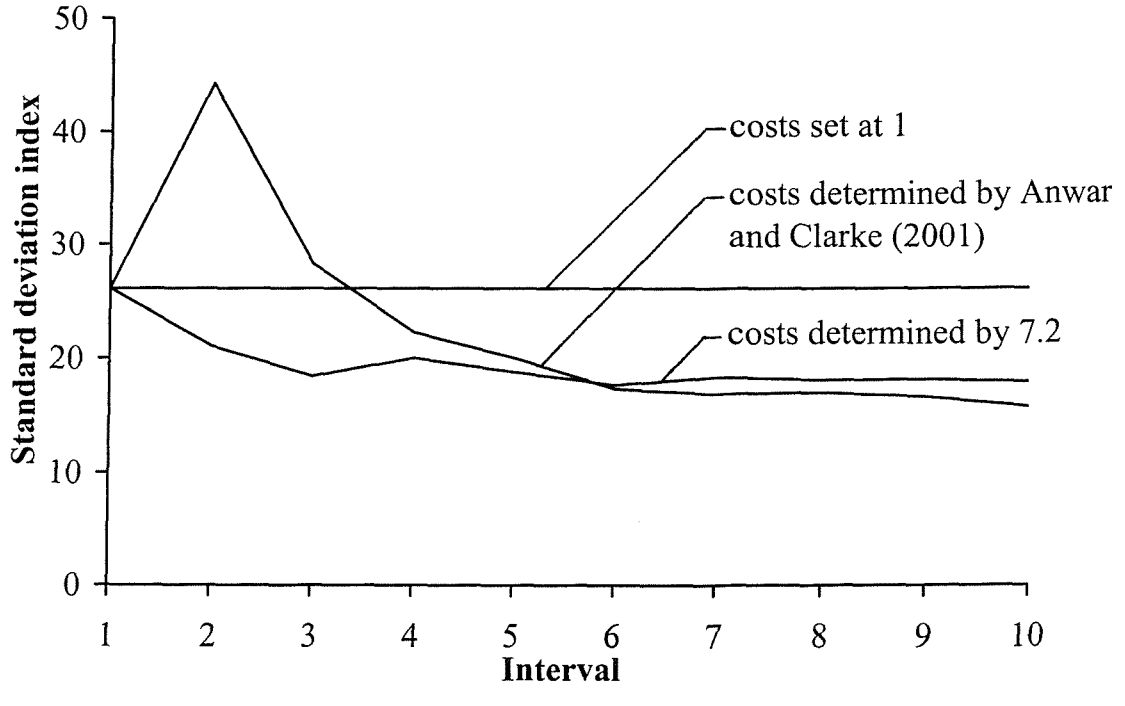


Figure 7.1 Standard deviation index, one or more jobs with no earliness/tardiness, duration and target start times constant over all intervals

In practice the input data may not remain constant. Figure 7.2 shows how the standard

deviation index behaves for Model 1, with target start times and duration randomly generated and changing with each interval, such that again at least one job is scheduled to start on time. Due to the changing input data the single interval model shows a decrease in standard deviation index. The rise in standard deviation index, when determining costs of earliness/tardiness with (7.1) is even more pronounced than in Figure 7.2. Using (7.2) again shows a decrease in standard deviation index with interval.

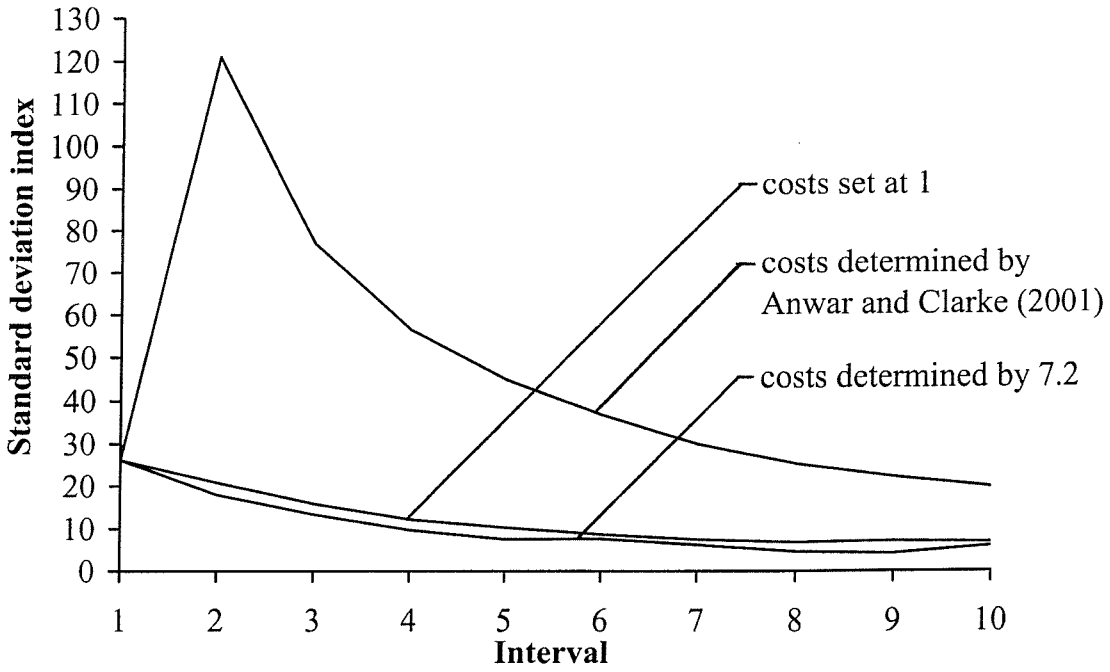


Figure 7.2 Standard deviation index, one or more jobs with no earliness/tardiness, duration and target start times vary over all intervals

It is possible that in the first interval none of the jobs are scheduled to start on time. Figure 7.3 shows how using (7.2) improves the standard deviation index compared to (7.1). Target start times and duration are again randomly generated but remain constant for all intervals.

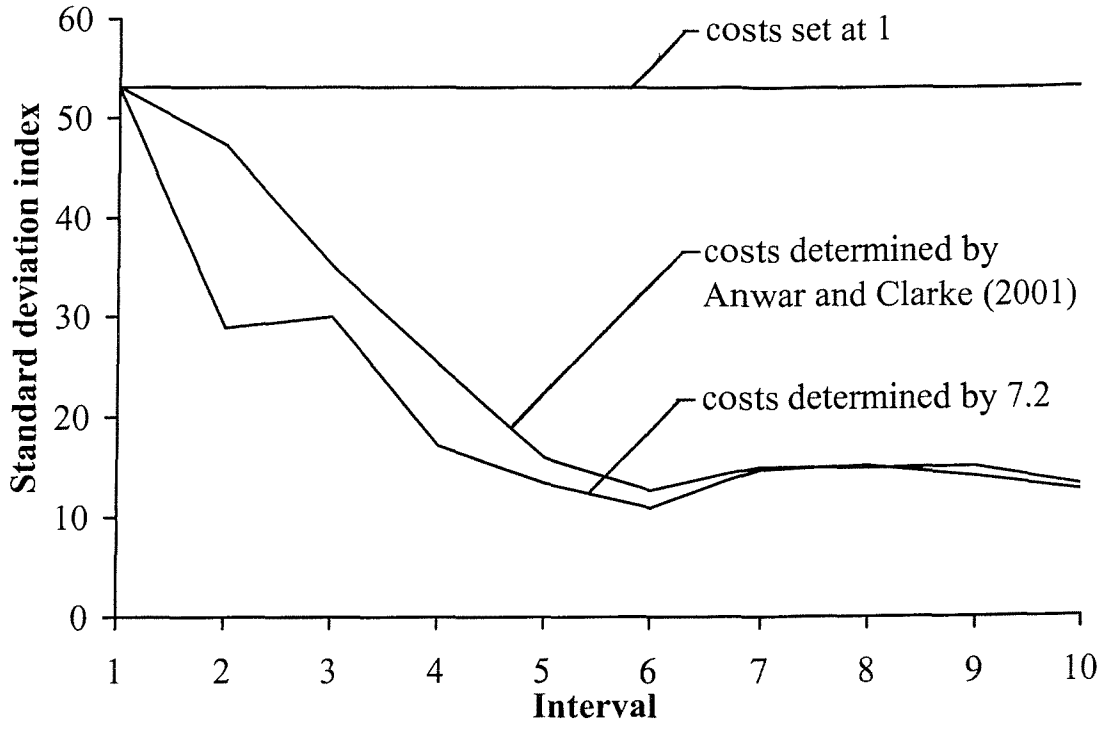


Figure 7.3 Standard deviation index, all jobs with earliness/tardiness, duartion and target start times vary over all intervals

Figure 7.4 shows that (7.1) and (7.2) perform equally well if the input data is random for each interval and no jobs start on time during the first interval. Both multi-interval models using (7.1) and (7.2) to calculate the costs of earliness and tardiness perform better than the single interval model.

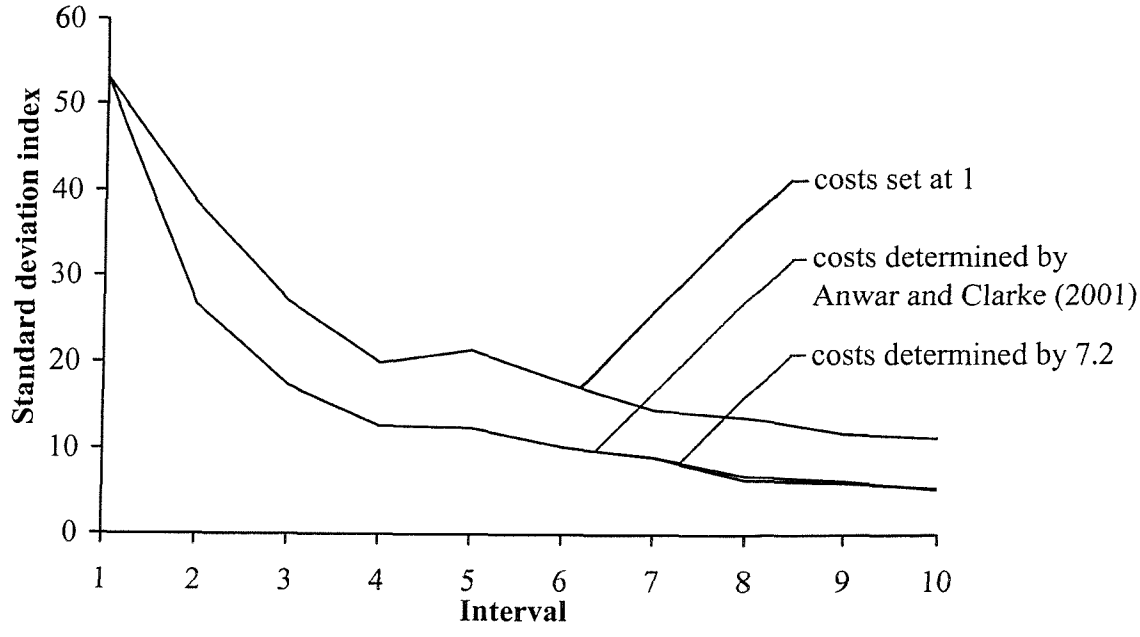


Figure 7.4 Standard deviation index, all jobs with earliness/tardiness, duration and target start times vary over all intervals

7.3 Development and analysis earliness/tardiness index

The purpose of using earliness and tardiness costs is to distribute earliness and tardiness over all users so that no one is disadvantaged or receives preferential treatment. In certain irrigation intervals, like early or late in a growing season, not all users may wish to irrigate. Comparing the two different methods to determine these costs by taking the standard deviation index works well if users always receive water, but not if a user skips one or more turns. A simple hypothetical example shows the effect of skipping turns. User A has received water every interval over a 10-interval period. User B skipped a number of turns and received water every other interval over the same period. If both always receive their water 10 hours late user A will have a total earliness/tardiness of 100 hours whereas user B will have a total earliness/tardiness of 50 hours. From this it seems user B is better off, yet every time water is delivered 10 hours late just like for user A.

The problem with using the standard deviation index is that it does not distinguish between an earliness and tardiness of zero (a job is scheduled to start on time) or null (user abstains from irrigating). A more appropriate way to determine the distribution of earliness and tardiness amongst users is to divide total earliness/tardiness by the number of periods a user has actually received water, i.e. when the earliness and tardiness are not null. This method can then be used to investigate the effect that skipping a turn has on the earliness/ tardiness cost determination, by introducing the earliness/tardiness index.

$$\varepsilon_p = \sigma_n \left(\sum_{i=1}^{N} \frac{\sum_{p=1}^{P} (E_{ip} + T_{ip})}{\sum_{p=1}^{P} H_{ip}} \right) \qquad \forall i = 1,2,...,N;\ E_{ip} \geq 0;\ T_{ip} \geq 0 \qquad (7.3)$$

where ε_p = earliness-tardiness index; and $H_{ip} = 1$ when user i irrigates during period p, 0 otherwise. If no null values occur, the earliness/tradiness index is identical to the standard deviation index.

Table 7.1 shows a comparison between using the standard deviation index and the earliness/tardiness index for a case of two users over a 10-interval period. During this period user A irrigates five times and user B ten times. After the 10 intervals both users have incurred a total earliness/tardiness of 25 hours. The standard deviation index during the first interval is 0 and remains 0 over the next 9 intervals, suggesting that there is no

difference between the users and both are treated equally. The earliness/tardiness index during the first interval is also 0, but during the next 9 intervals it can be seen that the earliness/tardiness index goes up, meaning that there is a difference between the two users. This shows that where the standard deviation index does not discriminate between 0 (no earliness/tardiness incurred) and null values (irrigation turn skipped), the earliness/tardiness index does. Therefore the earliness/tardiness index is a better measure than the standard deviation index.

Table 7.1: Comparison standard deviation index and earliness/tardiness index

interval	earliness/tardiness user A (hours)	earliness/tardiness user B (hours)	standard deviation index	earliness/tardiness index
(1)	(2)	(3)	(4)	(5)
1	5	5	0	0
2	null	0	0	1.25
3	5	5	0	0.83
4	null	0	0	1.25
5	5	5	0	1.00
6	null	0	0	1.25
7	5	5	0	1.07
8	null	0	0	1.25
9	5	5	0	1.11
10	null	0	0	1.25

7.4 Results and discussion earliness/tardiness index

Figure 7.5 shows the effect of having earliness and tardiness with null values on the standard deviation index and the earliness/tardiness index. Random target start times and durations were generated for 8 users. One user was randomly selected to skip every other turn. Model 1 was used to simulate a 10-interval period and weights were determined by (7.2).

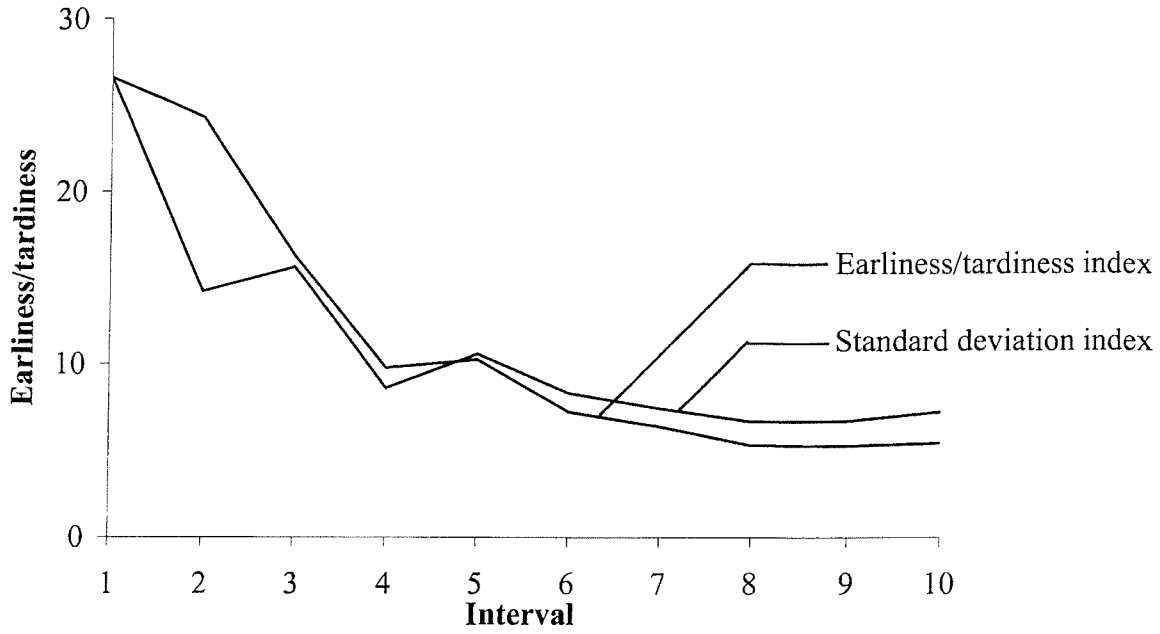


Figure 7.5 Earliness/tardiness index vs. standard deviation index

It can be shown that if there are no null values in a schedule the earliness/tardiness index is equal to the standard deviation index, Figure 7.5 shows that if one or more users skip at least one turn the use of the standard deviation index can give misleading results, sometimes appearing to perform better, sometimes appearing to perform worse.

7.5 Conclusions multi-interval scheduling

In this chapter an improved method for determining the earliness/tardiness costs is developed. A new method of measuring the quality of a schedule is also presented. The earliness/tardiness costs can be used to influence the decisions to be made in the current interval so that those users who have been disadvantaged during previous intervals get priority. This can help to ensure that after an entire irrigation season all users have incurred an equal amount of earliness/tardiness and no user has received preferential treatment.

The earliness/tardiness index can be used as a measurement for the fairness of scheduling within a tertiary unit. It can also be used as a tool to determine the effect of different scheduling policies or to compare scheduling in the different tertiary units of an irrigation system. This could help in the identification of problems within the system. It

may also be possible to use the earliness/tardiness index to compare across different irrigation systems, but more research would be needed to determine if the length of an irrigation interval has an effect on the earliness/tardiness index and if it needs to be included as a parameter in expression (7.3).

Both the improved method of determining the earliness/tardiness costs and the new measurement of the quality of a schedule work well. It must be brought to mind however that both are tested with randomly generated data as no field data is available. There may weaknesses and shortcomings in either method that would only become apparent if tested with real data. This needs to be investigated further in future.

8 Conclusions

Water delivery arrangements in irrigation systems range from completely fixed where duration, discharge and start time are set once by the irrigation district and seldom or never change to on-demand where users can take water whenever needed, in whatever quantity desired. In between these two extremes lies a range of arranged-demand systems where duration, discharge and/or start time are flexible to a certain degree but are agreed upon before the irrigation event takes place. In these systems it is the task of the irrigation district to deliver water so as to best meet the requirements of the users within the physical constraints of the system. Developing schedules for water deliveries can be a complicated process, especially when a large number of users is involved. It is therefore important to have a tool that can determine the optimal schedule given the requirements of the users and the constraints of the irrigation system. In Operations Research optimisation is used to determine schedules in a large variety of situations, ranging from product assembly to aircraft landings. The work in this thesis draws an analogy between the generic machine scheduling problem studied in Operations Research and the scheduling of water deliveries in irrigation systems. Although a number of models for irrigation purposes have been developed in earlier research, none seem to have made the connection with Operations Research and the wealth of information and techniques available in that field of study. For the first time this large source of insight is used specifically for irrigation scheduling, opening a whole new area of study.

When users irrigate sequentially the water in a channel can be seen as a single machine that processes a number of jobs (users). Although an analogy can be drawn between machine scheduling and irrigation scheduling, machine scheduling models are not directly applicable. Irrigation scheduling models have to take into account some unique features that are inherent to many irrigation situations. First of all it is customary in Operations Research to work with due dates and completion times, whereas in irrigation scheduling target start times and scheduled start times are more appropriate. Secondly, in Operation Research a job either has a due date or a deadline. If there is a due date earliness and tardiness are allowed and have no limit. If there is a dead line a job is not allowed to be tardy at all. In irrigation scheduling an intermediate situation occurs where a job is allowed to be early and tardy, but there is a fixed period, the irrigation interval in which all jobs must be completed.

A set of mixed integer linear programming models is developed in this work, that can aid the scheduling of water deliveries. Different management options such as contiguous and non-contiguous scheduling can be chosen to reduce the costs associated with operational spillage and gate operations or to better match target start times and scheduled start times. Mixed integer programming can be computationally very demanding. Computation times increase with the size and complexity of the problem. For example Model 1 with 20 users on average solves in around 1.2 hours, for 25 users the average solution time increases to 6.5 hours.

Where in pressurised irrigation systems travel times are non-existent, in open channels travel times play an important role when determining schedules. The position of two users relative to each other influence the travel time between them. This travel time can vary from negligible to considerable depending on the order in which they irrigate. Not taking account of travel times can result in early and/or late deliveries of water and also in unintended reduction or increase of irrigation duration, which can lead to under or over irrigation.

When users irrigate simultaneously water can be seen as a series of machines (stream tubes) that each process a job. Each stream tube can only supply one outlet at a time, but can supply a number of outlets in sequence. Using multi-machine scheduling it is possible to minimize the discharge required to supply all users with water. This could free up water for other (non-irrigation) uses. Minimizing the discharge can mean increased earliness/tardiness. Alternatively it is possible to minimize earliness/tardiness by allowing an increased discharge in the channel, to match irrigation demand and water delivery optimally. Irrigation scheduling is unique in that a job can be processed by more than one machine at the same time. By varying the number of stream tubes supplying a outlet with water, it becomes possible to schedule water deliveries in irrigation systems where discharges vary between users.

The irrigation industry is increasingly becoming service-oriented. Users have their wishes and requirements regarding the delivery of water to their fields. Some irrigation district already have policies regarding the timely delivery of water. As it is almost inevitable that there will be some difference between the requested duration/discharge/start time of an irrigation turn and the actual duration/discharge/start time, it is important to balance the inequity between users that is a result of these discrepancies. It is vital that to consider the fairness of schedules over a longer period than a single interval, as what may

seem fair in one interval, may not be fair in the long run. Earliness/tardiness costs can be used to influence the decisions to be made during the current interval by giving priority to those who have been disadvantaged most during the previous intervals. All models presented in this thesis accommodate the use of earliness tardiness costs. To measure the fairness of a schedule the earliness/tardiness index can be used. This index can also be used to determine the effect of different scheduling policies or to compare scheduling in different tertiary units of an irrigation systems. This could help in the identification of problems in the irrigation system.

The work done in this thesis has made a number of models available for use in irrigation scheduling. It is possible to use the models for scheduling in pressured systems where travel times are non-existent as well as in open channel systems where travel times can be considerable. Both sequential and simultaneous irrigation are accommodated and it is possible to schedule irrigation turns for systems where discharges vary between users. The models allow timeliness of delivery to be considered. Two new methods accompany the models, the first determines the costs of untimely deliveries so the models can be used to schedule over longer periods of time then just one irrigation interval. The second method can be used to measure the fairness of a schedule.

8.1 Future research

The work in this thesis covers a large variety of arranged-demand irrigation situations. The models presented in this thesis can be used as a basis to further develop and enhance arranged-demand irrigation scheduling. New solution techniques such as heuristics and genetic algorithms can be developed to bring down computation times, especially for larger problems. Although these techniques can give quick solutions to large problems, there is no guarantee that those solutions are optimal or even near it. The mixed integer linear programming models presented here play an important role in providing the bench mark solutions against which the performance of heuristics and/or genetic algorithms can be tested.

Further research can be done on the issue of travel times. Hydro-dynamic modelling could be incorporated into the models as to give accurate travel times, even for complex multi-machine scheduling. Changes in channel dimensions, which are common in the tail-

end of many systems, are another interesting area of future research. These changes will not only influence travel times, but could also decrease the capacity and limit the number of users that can irrigate simultaneously. Perhaps another area of research where arranged-demand irrigation scheduling can be of use is the service levels irrigation districts provide. One question that can be studied is what level of service can be provided given the available capacity. Turning this question around into: what capacity is needed to provide a certain level of service, is a problem that can be studied during the planning phase of new irrigation systems. As shown many opportunities for future research exist, and even though this work has now come to an end, it is the start of a whole new area of research.

Appendix A: Lingo input files

Model 1

```
SETS:
      job/1..16/: alpha, beta, earliness, tardiness, duration, scheduled,
target;
      link( job, job): delta;
ENDSETS

DATA:
      duration    =     461   451   537   387   537   537   202   193   360
                        193   537   193   537   451   572   572;
      target      =     2140  9783  8009  8889  4841  771   3362  5635  5408
                        5699  1365  1503  19    9515  1206  990;
      alpha       =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1;
      interval    =     10080;
      M           =     100000;
ENDDATA

!OBJECTIVE FUNCTION 3.1;
min = @sum( job(i): earliness(i) * alpha(i) + tardiness(i) * beta(i));

!CONSTRAINT 3.2;
@for( link(i,j): @bin( delta(i,j)));

!CONSTRAINT 3.6;
@for( job(i): scheduled(i) = target(i) - earliness(i) + tardiness(i));

!CONSTRAINT 3.9;
@for( job(i): @for( job(j)| j #ne# i: scheduled(j) - scheduled(i) + M *
(1-delta(i,j)) >= duration(i)));

!CONSTRAINT 3.10;
@for( job(i): @for( job(j)| j #ne# i: scheduled(i) - scheduled(j) + M *
delta(i,j) >= duration(j)));

!CONSTRAINT 3.11;
@for( job(i): scheduled(i) + duration(i) <= interval);
```

Model 1 alternative formulation

```
SETS:
      job/1..16/: alpha, beta, earliness, tardiness, duration, scheduled,
      target;
      link( job, job): delta;
ENDSETS

DATA:
      duration    =     461   451   537   387   537   537   202   193   360
                        193   537   193   537   451   572   572;
      target      =     2140  9783  8009  8889  4841  771   3362  5635  5408
                        5699  1365  1503  19    9515  1206  990;
      alpha       =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1;
      interval    =     10080;
      M           =     100000;
ENDDATA

!OBJECTIVE FUNCTION 3.1;
```

```
min = @sum( job(i): earliness(i) * alpha(i) + tardiness(i) * beta(i));

!CONSTRAINT 3.2;
@for( link(i,j): @bin( delta(i,j)));

!CONSTRAINT 3.6;
@for( job(i): scheduled(i) = target(i) - earliness(i) + tardiness(i));

!CONSTRAINT 3.9;
@for( job(i): @for( job(j)| j #ne# i: scheduled(j) - scheduled(i) + M *
(1-delta(i,j)) >= duration(i)));

!CONSTRAINT 3.10;
@for( job(i): @for( job(j)| j #ne# i: scheduled(i) - scheduled(j) + M *
delta(i,j) >= duration(j)));

!CONSTRAINT 3.11;
@for( job(i): scheduled(i) + duration(i) <= interval);
```

Model 1 modified for sequence independent setup times

```
SETS:
      job/1..16/: alpha, beta, earliness, tardiness, duration, scheduled,
      target, setup;
      link( job, job): delta;
ENDSETS

DATA:
      duration    =     449   439   525   575   525   525   189   180   348
                        180   525   180   525   439   559   559;
      target      =     2140  9783  8009  8889  4841  771   3362  5635  5408
                        5699  1365  1503  19    9515  1206  990;
      alpha       =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1;
      setup       =     13    13    13    13    13    13    13    13    13
                        13    13    13    13    13    14;
      interval    =     7;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION D.1;
min = @sum( job(i): alpha(i) * earliness(i) + beta(i) * tardiness(i));

!CONSTRAINT D.2;
@for( link(i,j): @bin( delta(i,j)));

!CONSTRAINT D.5;
@for( job(i): scheduled(i) - tardiness(i) + earliness(i) = target(i));

!CONSTRAINT D.6;
@for( link(i,j)| j #ne# i: scheduled(j) - scheduled(i) + M * (1-
delta(i,j)) >= duration(i) + setup(j) );

!CONSTRAINT D.7;
@for( link(i,j)| j #ne# i: scheduled(i) - scheduled(j) + M * delta(i,j)
>= duration(j) + setup(i));

!CONSTRAINT D.8;
@for( job(i): scheduled(i) + duration(i) <= interval);
```

Model 2a

```
SETS:
      job/1..16/: alpha, beta, earliness, tardiness, duration, scheduled,
      target, setup;
      link( job, job): delta;
ENDSETS

DATA:
      duration    =     461   451   537   387   537   537   202   193   360
                        193   537   193   537   451   572   572   ;
      target      =     2140  9783  8009  8889  4841  771   3362  5635  5408
                        5699  1365  1503  19    9515  1206  990   ;
      alpha       =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1     ;
      beta        =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1     ;
      M           =     100000;
ENDDATA

!OBJECTIVE FUNCTION 3.1;
min = @sum( job(i): earliness(i) * alpha(i) + tardiness(i) * beta(i));

!CONSTRAINT 3.2;
@for( link(i,j): @bin( delta(i,j)));

!CONSTRAINT 3.6;
@for( job(i): scheduled(i) - tardiness(i) + earliness(i) = target(i));

!CONSTRAINT 3.9;
@for( link(i,j)| j #ne# i: scheduled(j) - scheduled(i) + 100000 * (1-
delta(i,j)) >= duration(i));

!CONSTRAINT 3.10;
@for( link(i,j)| j #ne# i: scheduled(i) - scheduled(j) + 100000 *
delta(i,j) >= duration(j));

!CONSTRAINT 3.12;
@for( job(i): scheduled(i) + duration(i) <= @sum( job(n): duration(n)));
```

Model 2a alternative formulation

```
SETS:
      job/1..16/: earliness, tardiness, duration, target;
      link( job, job): delta;
ENDSETS

DATA:
      duration    =     461   451   537   387   537   537   202   193   360
                        193   537   193   537   451   572   572;
      target      =     2140  9783  8009  8889  4841  771   3362  5635  5408
                        5699  1365  1503  19    9515  1206  990;
      alpha       =     1;
      beta        =     1;
      interval    =     10080;
      M           =     100000;
ENDDATA

!OBJECTIVE FUNCTION 3.17;
min = @sum( job(i): alpha * earliness(i) + beta * tardiness(i));

!CONSTRAINT 3.18;
@for( link(i,j): @bin(delta(i,j)));
```

```
!CONSTRAINT 3.19;
@for( job(i): @sum( link(j,i): delta(j,i)) = 1);

!CONSTRAINT 3.20;
@for( job(j): @sum( link(j,i): delta(j,i)) = 1);

!CONSTRAINT 3.33;
@for( job(i): @sum( job(j): @sum( job(k)|k #le# i-1: delta(j,k) *
duration(j))) + earliness(i) - tardiness(i) = @sum( job(j): delta(j,i) *
target(j)));
```

Model 2b

```
SETS:
      job/1..16/: alpha, beta, earliness, tardiness, duration, scheduled,
      target;
      link( job, job): delta;
ENDSETS

DATA:
      duration    =     461   451   537   387   537   537   202   193   360
                        193   537   193   537   451   572   572;
      target      =     2140  9783  8009  8889  4841  771   3362  5635  5408
                        5699  1365  1503  19    9515  1206  990   ;
      alpha       =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1     ;
      beta        =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1     ;
      interval    =     10080;
      M           =     100000;
ENDDATA

!OBJECTIVE FUNCTION 3.1;
min = @sum( job(i): earliness(i) * alpha(i) + tardiness(i) * beta(i));

!CONSTRAINT 3.2;
@for( link(i,j): @bin( delta(i,j)));

!CONSTRAINT 3.6;
@for( job(i): scheduled(i) = target(i) - earliness(i) + tardiness(i));

!CONSTRAINT 3.9;
@for( link(i,j)| j #ne# i: scheduled(j) - scheduled(i) + M * (1-
delta(i,j)) >= duration(i) );

!CONSTRAINT 3.10;
@for( link(i,j)| j #ne# i: scheduled(i) - scheduled(j) + M * delta(i,j)
>= duration(j));

!CONSTRAINT 3.11;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 3.13;
@for( job(i): scheduled(i) >= interval - @sum( job(n): duration(n)));
```

Model 2b alternative formulation

```
SETS:
      job/1..16/: earliness, tardiness, duration, target;
      link( job, job): delta;
ENDSETS
```

```
DATA:
      duration    =     461   451   537   387   537   537   202   193   360
                        193   537   193   537   451   572   572;
      target      =     2140  9783  8009  8889  4841  771   3362  5635  5408
                        5699  1365  1503  19    9515  1206  990;
      alpha       =     1;
      beta        =     1;
      interval    =     10080;
      M           =     100000;
ENDDATA

!OBJECTIVE FUNCTION 3.17;
min = @sum( job(i): alpha * earliness(i) + beta * tardiness(i));

!CONSTRAINT 3.18;
@for( link(i,j): @bin(delta(i,j)));

!CONSTRAINT 3.19;
@for( job(i): @sum( link(j,i): delta(j,i)) = 1);

!CONSTRAINT 3.20;
@for( job(j): @sum( link(j,i): delta(j,i)) = 1);

!CONSTRAINT 3.30;
@for( job(i): @sum( job(j): @sum( job(k)|k #le# i-1: delta(j,k) *
duration(j))) + earliness(i) - tardiness(i) = @sum( job(j): delta(j,i) *
target(j)) - Xa);

!CONSTRAINT 3.31;
@sum( job(k): duration(k)) = interval - Xa;
```

Model 2c

```
SETS:
      job/1..16/: alpha, beta, earliness, tardiness, duration, scheduled,
      target;
      link( job, job):delta;
ENDSETS

DATA:
      duration    =     461   451   537   387   537   537   202   193   360
                        193   537   193   537   451   572   572   ;
      target      =     2140  9783  8009  8889  4841  771   3362  5635  5408
                        5699  1365  1503  19    9515  1206  990   ;
      alpha       =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1     ;
      beta        =     1     1     1     1     1     1     1     1     1
                        1     1     1     1     1     1     1     ;
      interval    =     10080;
      M           =     100000;
ENDDATA

!OBJECTIVE FUNCTION 3.1;
min = @sum( job(i): earliness(i) * alpha(i) + tardiness(i) * beta(i));

!CONSTRAINT 3.2;
@for( link(i,j): @bin(delta(i,j)));

!CONSTRAINT 3.6;
@for( job(i): scheduled(i) = target(i) - earliness(i) + tardiness(i));

!CONSTRAINT 3.9;
@for( link(i,j)| j #ne# i: scheduled(j) - scheduled(i) + M * (1-
delta(i,j)) >= duration(i) );
```

```
!CONSTRAINT 3.10;
@for( link(i,j)| j #ne# i: scheduled(i) - scheduled(j) + M *delta(i,j)
>= duration(j));

!CONSTRAINT 3.4;
irrigation_interval = @sum( job(i): duration(i)) + Xa + Xb;

!CONSTRAINT 3.15;
@for( job(i): scheduled(i) + duration(i) <= interval - Xb);

!CONSTRAINT 3.16;
@for( job(i): scheduled(i) >= Xa);
```

Model 2c alternative formulation

```
SETS:
      job/1..16/: earliness, tardiness, duration, target, X;
      link( job, job): delta;
ENDSETS

DATA:
      duration    =     461   451   537   387   537   537   202   193   360
                        193   537   193   537   451   572   572;
      target      =     2140  9783  8009  8889  4841  771   3362  5635  5408
                        5699  1365  1503  19    9515  1206  990;
      alpha       =     1;
      beta        =     1;
      interval    =     10080;
      M           =     100000;
ENDDATA

!OBJECTIVE FUNCTION 3.17;
min = @sum( job(i): alpha * earliness(i) + beta * tardiness(i));

!CONSTRAINT 3.18;
@for( link(i,j): @bin(delta(i,j)));

!CONSTRAINT 3.19;
@for( job(i): @sum( link(j,i): delta(j,i)) = 1);

!CONSTRAINT 3.20;
@for( job(j): @sum( link(j,i): delta(j,i)) = 1);

!CONSTRAINT 3.34;
@for( job(i): @sum( job(j): @sum( job(k)|k #le# i-1: delta(j,k) *
duration(j))) + earliness(i) - tardiness(i) = @sum( job(j): delta(j,i) *
target(j))- Xa);
```

Model 3

```
SETS:
      job/1..16/: alpha, beta, earliness, tardiness, scheduled, target,
      duration, initial;
      link( job, job): delta, setup;
ENDSETS

DATA:
      duration    =     453   443   529   379   529   529   194   185   352
                        185   529   185   530   443   564   564   ;
      target      =     2140  9783  8009  8889  4841  771   3362  5635  5408
                        5699  1365  1503  19    9515  1206  990   ;
```

```
alpha       =    1    1    1    1    1    1    1    1    1
                 1    1    1    1    1    1    1    ;
beta        =    1    1    1    1    1    1    1    1    1
                 1    1    1    1    1    1    1    ;
irrigation  =    10080;
M           =    100000;
initial     =    108  91   70   80   59   39   39   37   33
                 31   30   29   23   23   15   7    ;
setup       =    0    0    0    42   21   0    16   14   10
                 8    7    6    0    0    7    0
                 17   0    0    42   21   0    16   14   10
                 8    7    6    0    0    7    0
                 38   21   0    42   21   0    16   14   10
                 8    7    6    0    0    7    0
                 69   52   31   0    0    0    16   14   10
                 8    7    6    0    0    7    0
                 69   52   31   21   0    0    16   14   10
                 8    7    6    0    0    7    0
                 69   52   31   42   21   0    16   14   10
                 8    7    6    0    0    7    0
                 85   68   47   57   36   16   0    0    0
                 0    0    0    0    0    7    0
                 85   68   47   57   36   16   2    0    0
                 0    0    0    0    0    7    0
                 85   68   47   57   36   16   6    3    0
                 0    0    0    0    0    7    0
                 85   68   47   57   36   16   8    6    2
                 0    0    0    0    0    7    0
                 85   68   47   57   36   16   9    7    4
                 1    0    0    0    0    7    0
                 85   68   47   57   36   16   10   8    5
                 2    1    0    0    0    7    0
                 85   68   47   57   36   16   16   14   10
                 8    7    6    0    0    7    0
                 85   68   47   57   36   16   16   14   10
                 8    7    6    0    0    7    0
                 101  83   63   73   52   31   32   30   26
                 24   23   21   16   16   0    0
                 101  83   63   73   52   31   32   30   26
                 24   23   21   16   16   7    0    ;
ENDDATA

!OBJECTIVE FUNCTION 4.1;
min = @sum( job(i): earliness(i) * alpha(i) + tardiness(i) * beta(i));

!CONSTRAINT 4.2;
@for( link(i,j): @bin( delta(i,j)));

!CONSTRAINT 4.3;
@for( job(i): scheduled(i) = target(i) - earliness(i) + tardiness(i));

!CONSTRAINT 4.6;
@for( link(i,j)| j #ne# i: scheduled(j) - scheduled(i) + M * (1-
delta(i,j)) >= duration(i) + setup(i,j) );

!CONSTRAINT 4.7;
@for( link(i,j)| j #ne# i: scheduled(i) - scheduled(j) + M * delta(i,j)
>= duration(j)+ setup(j,i));

!CONSTRAINT 4.8;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 4.9;
@for( job(i): scheduled(i) >= initial(i));
```

Model 4a

```
SETS:
      job/1..11/: alpha, beta, earliness, tardiness, scheduled, target;
      link( job, job): gamma, sequence_duration;
ENDSETS


DATA:
      target                  =     0     2800  584   251   2829  267   4914
                                    5525  2197  5419  2000;
      sequence_duration       =     0     39    37    33    31    30    29
                                    23    23    15    7
                                    194   0     194   194   194   194   194
                                    194   194   201   194
                                    185   187   0     185   185   185   185
                                    185   185   192   185
                                    352   358   355   0     352   352   352
                                    352   352   359   352
                                    185   193   191   187   0     185   185
                                    185   185   192   185
                                    592   538   536   533   530   0     529
                                    529   529   536   529
                                    185   195   193   190   187   186   0
                                    185   185   192   185
                                    530   546   544   540   538   537   536
                                    0     530   537   530
                                    443   459   457   453   451   450   449
                                    443   0     450   443
                                    563   595   593   589   587   586   584
                                    579   579   0     563
                                    563   595   593   589   587   586   584
                                    579   579   570   0;
      M                       =     100000;
      interval                =     7920  ;
      alpha                   =     1     1     1     1     1     1     1
                                    1     1     1     1;
      beta                    =     1     1     1     1     1     1     1
                                    1     1     1     1;
ENDDATA

!OBJECTIVE FUNCTION 4.10;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i));

!CONSTRAINT 4.11;
@for( link(i,j): @bin( gamma(i,j)));

!CONSTRAINT 4.12;
@for( job(i): scheduled(i) = target(i) - earliness(i) + tardiness(i));

!CONSTRAINT 4.15;
@for( job(i): @sum( job(j): gamma(i,j)) = 1);

!CONSTRAINT 4.16;
@for( job(j): @sum( job(i): gamma(i,j)) = 1);

!CONSTRAINT 4.17;
@for ( job(i): @for( job(j)| j #gt# 1 #and# i #ne# j: scheduled(i) +
sequence_duration(i,j) * gamma(i,j) + M*(gamma(i,j)-1) <=
scheduled(j)));

!CONSTRAINT 4.18;
@for( job(i): scheduled(i) + @sum( job(j): sequence_duration(i,j) *
gamma(i,j)) <= @sum( job(k): @sum(job(j): sequence_duration(k,j) *
gamma(k,j))));
```

Model 4b

```
SETS:
      job/1..11/: alpha, beta, earliness, tardiness, scheduled, target;
      link( job, job): gamma, sequence_duration;
ENDSETS

DATA:
      target                  =     0     2800  584   251   2829  267   4914
                                    5525  2197  5419  2000;
      sequence_duration       =     0     39    37    33    31    30    29
                                    23    23    15    7
                                    194   0     194   194   194   194   194
                                    194   194   201   194
                                    185   187   0     185   185   185   185
                                    185   185   192   185
                                    352   358   355   0     352   352   352
                                    352   352   359   352
                                    185   193   191   187   0     185   185
                                    185   185   192   185
                                    592   538   536   533   530   0     529
                                    529   529   536   529
                                    185   195   193   190   187   186   0
                                    185   185   192   185
                                    530   546   544   540   538   537   536
                                    0     530   537   530
                                    443   459   457   453   451   450   449
                                    443   0     450   443
                                    563   595   593   589   587   586   584
                                    579   579   0     563
                                    563   595   593   589   587   586   584
                                    579   579   570   0;
      M                       =     100000;
      interval                =     7920  ;
      alpha                   =     1     1     1     1     1     1     1
                                    1     1     1     1;
      beta                    =     1     1     1     1     1     1     1
                                    1     1     1     1;
ENDDATA

!OBJECTIVE FUNCTION 4.10;
min = @sum( job(i): earliness(i) * alpha(i) + tardiness(i) * beta(i));

!DECISION VARIABLE 4.11;
@for( link(i,j): @bin( gamma(i,j)));

!CONSTRAINT 4.12;
@for( job(i): scheduled(i) - tardiness(i) + earliness(i) = target(i));

!CONSTRAINT 4.15;
@for( job(i): @sum( job(j)| i #ne# j: gamma(i,j)) = 1);

!CONSTRAINT 4.16;
@for( job(j): @sum( job(i)| i #ne# j: gamma(i,j)) = 1);

!CONSTRAINT 4.17;
@for ( link(i,j)| j #gt# 1: scheduled(i) + sequence_duration(i,j) *
gamma(i,j) + M*(gamma(i,j)-1) <= scheduled(j));

!CONSTRAINT 4.19;
Xa= interval- @sum( job(i): @sum(job(j): gamma(i,j) *
sequence_duration(i,j)));

!CONSTRAINT 4.20;
@for( job(i): scheduled(i) >= Xa);
```

Model 4c

```
SETS:
      job/1..11/: alpha, beta, earliness, tardiness, scheduled, target;
      link( job, job): gamma, sequence_duration;
ENDSETS

DATA:
      target                  =     0     2800  584   251   2829  267   4914
                                    5525  2197  5419
      2000;
      sequence_duration       =     0     39    37    33    31    30    29
                                    23    23    15    7
                                    194   0     194   194   194   194   194
                                    194   194   201   194
                                    185   187   0     185   185   185   185
                                    185   185   192   185
                                    352   358   355   0     352   352   352
                                    352   352   359   352
                                    185   193   191   187   0     185   185
                                    185   185   192   185
                                    592   538   536   533   530   0     529
                                    529   529   536   529
                                    185   195   193   190   187   186   0
                                    185   185   192   185
                                    530   546   544   540   538   537   536
                                    0     530   537   530
                                    443   459   457   453   451   450   449
                                    443   0     450   443
                                    563   595   593   589   587   586   584
                                    579   579   0     563
                                    563   595   593   589   587   586   584
                                    579   579   570   0;
      M                       =     100000;
      interval                =     7920  ;
      alpha                   =     1     1     1     1     1     1     1
                                    1     1     1     1;
      beta                    =     1     1     1     1     1     1     1
                                    1     1     1     1;
ENDDATA

!OBJECTIVE FUNCTION 4.10;
min = @sum( job(i)|i #gt# 1: earliness(i) * alpha(i) + tardiness(i) *
beta(i));

!DECISION VARIABLE 4.11;
@for( link(i,j): @bin( gamma(i,j)));

!CONSTRAINT 4.12;
@for( job(i): scheduled(i) - tardiness(i) + earliness(i) = target(i));

!CONSTRAINT 4.15;
@for( job(i): @sum( job(j)| i #ne# j: gamma(i,j)) = 1);

!CONSTRAINT 4.16;
@for( job(j): @sum( job(i)| i #ne# j: gamma(i,j)) = 1);

!CONSTRAINT 4.17;
@for ( link(i,j)| j #gt# 1: scheduled(i) + sequence_duration(i,j) *
gamma(i,j) + M*(gamma(i,j)-1) <= scheduled(j));

!CONSTRAINT 5.20;
@for( job(i): scheduled(i) >= Xa);
```

```
!CONSTRAINT 5.21;
@for( job(i): scheduled(i) + @sum( job(j): sequence_duration(i,j) *
gamma(i,j))- @sum( job(k): @sum(job(j): sequence_duration(k,j) *
gamma(k,j))) - Xa <= 0);
```

Model 5 single stage

```
SETS:
      job /1..9/: alpha, beta, duration, target, earliness, tardiness,
      scheduled;
      tube /1..8/: psi;
      combi (job, tube): tau;
      combi2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 5.1;
min = @sum( job(i): alpha(i) * earliness(i) + beta(i) * tardiness(i)) +
100 * @sum( tube(w): psi(w));

!CONSTRAINT 5.2;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 5.3;
@for( combi2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.4;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.8;
@for( job(i)|i #gt# 1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.11;
@for( job(i): @for( job(j)| j #gt# 1 #and# j #ne# i: @for( tube(w):
scheduled(j) - scheduled(i) - M*phi(i,j,w) >= duration(i) - M)));

!CONSTRAINT 5.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.13;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) *
@size(job));
```

```
!CONSTRAINT 5.15;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);
```

Model 5 single stage with maximum allowable earliness/tardiness

```
SETS:
      job /1..11/: duration, target, earliness, tardiness, scheduled,
alpha, beta;
      tube /1..3/: psi;
      combi (job, tube): tau;
      combi2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 5.1;
min = @sum( job(i): alpha(i) * earliness(i) + beta(i) * tardiness(i)) +
100 * @sum( tube(w): psi(w));

!CONSTRAINT 5.2;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 5.3;
@for( combi2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.4;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.8;
@for( job(i)|i #gt# 1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.11;
@for( job(i): @for( job(j)| j #gt# 1 #and# j #ne# i: @for( tube(w):
scheduled(j) - scheduled(i) - M*phi(i,j,w) >= duration(i) - M)));

!CONSTRAINT 5.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.13;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) *
@size(job));

!CONSTRAINT 5.15;
```

```
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);

!CONSTRAINT 5.41;
@for( job(i)| i #gt# 1: earliness(i) <= 24);

!CONSTRAINT 5.42;
@for( job(i)| i #gt# 1: tardiness(i) <= 24);
```

Model 5 6a 6b 6c first stage

```
SETS:
      tube /1..8/: phi;
      job /1..8/: duration;
      combi(job, tube): tau;
ENDSETS

DATA:
      duration    =     0.8   2.13  2.40  1.72  2.05  2.43  2.05  2.50;
      interval    =     6;
ENDDATA


!OBJECTIVE FUNCTION 5.43;
min = @sum( tube(w): phi(w));

!CONSTRAINT 5.44;
@for( tube(w): @bin( phi(w)));

!CONSTRAINT 5.45;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.46;
@for( tube(w): @sum( job(i): tau(i,w) * duration(i)) <= interval);

!CONSTRAINT 5.47;
@for( job(i): @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.48;
@for( combi(i,w): tau(i,w) <= phi(w) * @size(job));
```

Model 5 second stage

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta;
      tube /1..3/:;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA


!OBJECTIVE FUNCTION 5.49;
```

```
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) ;

!CONSTRAINT 5.3;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.4 ;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.8;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.11;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) -M)));

!CONSTRAINT 5.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.15;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);
```

Model 6a single stage

```
SETS:
      job /1..9/: alpha, beta, duration, target, earliness, tardiness,
      scheduled;
      tube /1..8/: psi;
      combi (job, tube): tau;
      combi2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 5.1;
min = @sum( job(i): alpha(i) * earliness(i) + beta(i) * tardiness(i)) +
100 * @sum( tube(w): psi(w));

!CONSTRAINT 5.2;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 5.3;
@for( combi2(i,j,w): @bin(phi(i,j,w)));
```

```
!CONSTRAINT 5.4;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.8;
@for( job(i)|i #gt# 1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.11;
@for( job(i): @for( job(j)| j #gt# 1 #and# j #ne# i: @for( tube(w):
scheduled(j) - scheduled(i) - M*phi(i,j,w) >= duration(i) - M)));

!CONSTRAINT 5.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.13;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) *
@size(job));

!CONSTRAINT 5.15;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);

!CONSTRAINT 5.16;
@for( job(i)| i #gt# 1: @for( tube(w): scheduled(i) + duration(i) -
M*(1-tau(i,w)) <= @sum( job(n): duration(n) * tau(n,w))));
```

Model 6a second stage

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta;
      tube /1..5/:;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 5.49;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) ;

!CONSTRAINT 5.3;
@for( link2(i,j,w): @bin(phi(i,j,w)));
```

```
!CONSTRAINT 5.4;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.8;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 5.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.11;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) -M)));

!CONSTRAINT 5.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.15;
@for( tube(w): @sum( job(i): duration(i) * tau(i,w)) <= interval);

!CONSTRAINT 5.16;
@for( job(i): @for( tube(w): scheduled(i) + duration(i) - M*(1-tau(i,w))
<= @sum( job(k): duration(k) * tau(k,w))));
```

Model 6b single stage

```
SETS:
      job /1..9/: alpha, beta, duration, target, earliness, tardiness,
      scheduled;
      tube /1..8/: psi, Xa;
      combi (job, tube): tau;
      combi2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 5.1;
min = @sum( job(i): alpha(i) * earliness(i) + beta(i) * tardiness(i)) +
100 * @sum( tube(w): psi(w));

!CONSTRAINT 5.2;
@for( tube(w): @bin(psi(w)));


!CONSTRAINT 5.3;
@for( combi2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.4;
```

```
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.8;
@for( job(i)|i #gt# 1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.11;
@for( job(i): @for( job(j)| j #gt# 1 #and# j #ne# i: @for( tube(w):
scheduled(j) - scheduled(i) - M*phi(i,j,w) >= duration(i) - M)));

!CONSTRAINT 5.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.13;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) *
@size(job));

!CONSTRAINT 5.15;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);

!CONSTRAINT 5.17;
@for( tube(w): Xa(w) = interval - @sum( job(i): duration(i) *
tau(i,w)));

!CONSTRAINT 5.18;
@for( job(i)| i #gt# 1: @for( tube(w): scheduled(i) + M*(1-tau(i,w)) >=
Xa(w)));
```

Model 6b second stage

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta;
      tube /1..3/: Xa;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 5.49;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) ;
```

```
!CONSTRAINT 5.3;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.4;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.8;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.11;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) -M)));

!CONSTRAINT 5.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.15;
@for(tube(w): @sum( job(i): tau(i,w) * duration(i)) <= interval);

!CONSTRAINT 5.17;
@for( tube(w): Xa(w) = interval - @sum( job(i): duration(i)*tau(i,w)));

!CONSTRAINT 5.18;
@for( tube(w): @for( job(i)|i #gt# 1: scheduled(i) + M*(1-tau(i,w)) >=
Xa(w)));
```

Model 6c single stage

```
SETS:
      job /1..9/: alpha, beta, duration, target, earliness, tardiness,
      scheduled;
      tube /1..8/: psi, Xa, Xb;
      combi (job, tube): tau;
      combi2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA


!OBJECTIVE FUNCTION 5.1;
min = @sum( job(i): alpha(i) * earliness(i) + beta(i) * tardiness(i)) +
100 * @sum( tube(w): psi(w));

!CONSTRAINT 5.2;
```

```
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 5.3;
@for( combi2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.4;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.8;
@for( job(i)|i #gt# 1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.11;
@for( job(i): @for( job(j)| j #gt# 1 #and# j #ne# i: @for( tube(w):
scheduled(j) - scheduled(i) - M*phi(i,j,w) >= duration(i) - M)));

!CONSTRAINT 5.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.13;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) *
@size(job));

!CONSTRAINT 5.15;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);

!CONSTRAINT 5.18;
@for( job(i)|i #gt# 1: @for( tube(w): scheduled(i) + M*(1-tau(i,w)) >=
Xa(w)));

!CONSTRAINT 5.19;
@for( tube(w): interval = @sum( job(i): duration(i) * tau(i,w)) + Xa(w)
+ Xb(w));

!CONSTRAINT 5.20;
@for( job(i): @for( tube(w): scheduled(i) + duration(i) - M*(1-tau(i,w))
<= interval - Xb(w)));
```

Model 6c second stage

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta;
      tube /1..3/: Xa, Xb;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
```

```
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      g           =     6;
      M           =     10000;
ENDDATA


!OBJECTIVE FUNCTION 5.49;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) ;

!CONSTRAINT 5.3;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.4;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.8;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.11;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) -M)));

!CONSTRAINT 5.12;
@for( job(i): scheduled(i) + duration(i) <= g);

!CONSTRAINT 5.15;
@for(tube(w): @sum( job(i): tau(i,w) * duration(i)) <= g);

!CONSTRAINT 5.18;
@for( tube(w): @for( job(i)|i #gt# 1: scheduled(i) + M*(1-tau(i,w)) >=
Xa(w)));

!CONSTRAINT 5.19;
@for( tube(w): g = @sum( job(i): duration(i) * tau(i,w)) + Xa(w) +
Xb(w));

!CONSTRAINT 5.20;
@for( tube(w):@for( job(i)| i #gt# 1: scheduled(i) + duration(i) -M*(1-
tau(i,w)) <= g - xb(w)));
```

Model 7

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta;
      tube /1..3/:xa, psi;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
      link3 (job,job): setup;
ENDSETS
```

```
DATA:
duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05  2.50;
target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60  3.03;
alpha       =     1     1     1     1     1     1     1     1     1;
beta        =     1     1     1     1     1     1     1     1     1;
interval    =     6;
M           =     10000;
setup       =     0.00  0.02  0.03  0.03  0.07  0.07  0.13  0.13  0.17
                  0.00  0.00  0.01  0.01  0.05  0.05  0.11  0.11  0.15
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00;
ENDDATA

!OBJECTIVE FUNCTION 5.21;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) + 100 * @sum( tube(w): psi(w)) ;

!CONSTRAINT 5.22;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.23;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.24;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 5.25;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.28;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.29;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.30;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.31;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) + setup(i,j)
-M)));

!CONSTRAINT 5.32;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.33;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) * @size(
tube));

!CONSTRAINT 5.35;
@for(tube(w): @for( job(j): @sum( job(i): tau(i,w) * duration(i) +
setup(i,j) * phi(i,j,w)) <= interval))
```

Model 8a

```
SETS:
     job /1..9/: duration, target, earliness, tardiness,scheduled,
```

```
      alpha, beta;
      tube /1..3/: psi, Xa;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
      link3 (job,job): setup;
ENDSETS

DATA:
duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05  2.50;
target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60  3.03;
alpha       =     1     1     1     1     1     1     1     1     1;
beta        =     1     1     1     1     1     1     1     1     1;
interval    =     6;
M           =     10000;
setup       =     0.00  0.02  0.03  0.03  0.07  0.07  0.13  0.13  0.17
                  0.00  0.00  0.01  0.01  0.05  0.05  0.11  0.11  0.15
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00;
ENDDATA

!OBJECTIVE FUNCTION 5.21;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) + 100 * @sum( tube(w): psi(w)) ;

!CONSTRAINT 5.22;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.23;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.24;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 5.25;
@for( job(i):scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.28;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.29;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.30;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.31;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):scheduled(j)
-scheduled(i) - M* phi(i,j,w) >= duration(i) + setup(i,j) -M)));

!CONSTRAINT 5.32;
@for( job(i):scheduled(i) + duration(i) <= g);

!CONSTRAINT 5.33;
@for( tube(w): @sum( job(i): tau(i,w)) <= psi(w) * @size(tube));
!CONSTRAINT 5.35;
@for(tube(w): @sum( job(i): tau(i,w) * duration(i)) <= g);

!CONSTRAINT 5.36;
@for( job(i): @for( tube(w):scheduled(i) + duration(i) - M * (1-
```

```
tau(i,w)) <= @sum( job(k): tau(k,w)* duration(k)) + @sum( job(k):
@sum(job( j)|j #ne# k: phi(k,j,w) * setup(k,j))));
```

Model 8b

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta;
      tube /1..3/: psi, Xa;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
      link3 (job,job): setup;
ENDSETS

DATA:
duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05  2.50;
target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60  3.03;
alpha       =     1     1     1     1     1     1     1     1     1;
beta        =     1     1     1     1     1     1     1     1     1;
interval    =     6;
M           =     10000;
setup       =     0.00  0.02  0.03  0.03  0.07  0.07  0.13  0.13  0.17
                  0.00  0.00  0.01  0.01  0.05  0.05  0.11  0.11  0.15
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00;
ENDDATA

!OBJECTIVE FUNCTION 5.21;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) + 100 * @sum( tube(w): psi(w)) ;

!CONSTRAINT 5.22;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.23;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.24;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 5.25;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.28;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.29;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.30;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.31;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) + setup(i,j)
-M)));
```

```
!CONSTRAINT 5.32;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.33;
@for( tube(w): @sum( job(i): tau(i,w)) <= psi(w) * @size( tube));

!CONSTRAINT 5.35;
@for(tube(w): @sum( job(i): tau(i,w) * duration(i)) <= interval);

!CONSTRAINT 5.37;
@for( tube(w): Xa(w) = interval - @sum( job(i): duration(i)*tau(i,w)) -
@sum( job(i): @sum( job(j): phi(i,j,w) * setup(i,j))));

!CONSTRAINT 5.38;
@for( tube(w): @for( job(i)|i #gt# 1: scheduled(i) >= Xa(w) + setup(1,i)
- M*(1-tau(i,w))));
```

Model 8c

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta;
      tube /1..3/: psi, Xa, Xb;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
      link3 (job,job): setup;
ENDSETS

DATA:
duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05  2.50;
target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60  3.03;
alpha       =     1     1     1     1     1     1     1     1     1;
beta        =     1     1     1     1     1     1     1     1     1;
interval    =     6;
M           =     10000;
setup       =     0.00  0.02  0.03  0.03  0.07  0.07  0.13  0.13  0.17
                  0.00  0.00  0.01  0.01  0.05  0.05  0.11  0.11  0.15
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00;
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00;
ENDDATA

!OBJECTIVE FUNCTION 5.21;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) + 100 * @sum( tube(w): psi(w));

!CONSTRAINT 5.22;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.23;
@for( link(i,w): @bin(tau(i,w)));


!CONSTRAINT 5.24;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 5.25;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));
```

```
!CONSTRAINT 5.28;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = 1);

!CONSTRAINT 5.29;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.30;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.31;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) + setup(i,j)
-M)));

!CONSTRAINT 5.32;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.33;
@for( tube(w): @sum( job(i): tau(i,w)) <= psi(w) * @size(tube));

!CONSTRAINT 5.35;
@for(tube(w): @sum( job(i): tau(i,w) * duration(i)) <= interval);

!CONSTRAINT 5.38;
@for( tube(w): @for( job(i)|i #gt# 1: scheduled(i) >= Xa(w) - M*(1-
tau(i,w)) + setup(1,i)));

!CONSTRAINT 5.39;
@for( tube(w): @sum( job(i): duration(i)*tau(i,w)) + @sum( job(i): @sum(
job(j): phi(i,j,w) * setup(i,j))) + Xa(w) + Xb(w)= interval);

!CONSTRAINT 5.40;
@for( tube(w):@for( job(i)| i #gt# 1: scheduled(i) + duration(i) -M*(1-
tau(i,w)) <= interval - Xb(w)));
```

Model 9 single stage

```
SETS:
      job /1..9/: alpha, beta, duration, target, earliness, tardiness,
      scheduled, q;
      tube /1..8/: psi;
      combi (job, tube): tau;
      combi2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      q           =     1     1     2     1     1     3     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA


!OBJECTIVE FUNCTION 6.1;
min = @sum( job(i): alpha(i) * earliness(i) + beta(i) * tardiness(i)) +
100 * @sum( tube(w): psi(w));

!CONSTRAINT 6.2;
@for( combi2(i,j,w): @bin(phi(i,j,w)));
```

```
!CONSTRAINT 6.3;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.4;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 6.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.8;
@for( job(i)|i #gt# 1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 6.11;
@for( job(i): @for( job(j)| j #gt# 1 #and# j #ne# i: @for( tube(w):
scheduled(j) - scheduled(i) - M*phi(i,j,w) >= duration(i) - M)));

!CONSTRAINT 6.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 6.16;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) *
@size(job));

!CONSTRAINT 6.17;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);
```

Model 9 single stage with time restrictions

```
SETS:
      job /1..11/: duration, target, earliness, tardiness, scheduled,
      alpha, beta;
      tube /1..3/: psi;
      combi (job, tube): tau;
      combi2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      q           =     1     1     2     1     1     3     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 6.1;
min = @sum( job(i): alpa(i) * earliness(i) + beta(i) * tardiness(i)) +
100 * @sum( tube(w): psi(w));

!CONSTRAINT 6.2;
@for( combi2(i,j,w): @bin(phi(i,j,w)));
```

```
!CONSTRAINT 6.3;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.4;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 6.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.8;
@for( job(i)|i #gt# 1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 6.11;
@for( job(i): @for( job(j)| j #gt# 1 #and# j #ne# i: @for( tube(w):
scheduled(j) - scheduled(i) - M*phi(i,j,w) >= duration(i) - M)));

!CONSTRAINT 6.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 6.14;
@for( job(i)| i #gt# 1: earliness(i) <= 24);

!CONSTRAINT 6.15;
@for( job(i)| i #gt# 1: tardiness(i) <= 24);

!CONSTRAINT 6.16;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) *
@size(job));

!CONSTRAINT 6.17;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);
```

Model 9 10a 10b 10c first stage

```
SETS:
     tube /1..8/: phi;
     job /1..8/: duration, q;
     combi (job, tube): tau;
ENDSETS

DATA:
     duration   =    0.8   2.13  2.40  1.72  2.05  2.43  2.05  2.50;
     q          =    1     2     1     1     3     1     1     1     ;
     interval   =    6;
ENDDATA

!OBJECTIVE FUNCTION 5.43;
min = @sum( tube(w): phi(w));

!CONSTRAINT 5.44;
@for( tube(w): @bin( phi(w)));

!CONSTRAINT 5.45;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.46;
@for( tube(w): @sum( job(i): tau(i,w) * duration(i)) <= interval);
```

```
!CONSTRAINT 5.47;
@for( job(i): @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 5.48;
@for( combi(i,w): tau(i,w) <= phi(w) * @size(job));
```

Model 9 second stage

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta, q;
      tube /1..5/:;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      q           =     1     1     2     1     1     3     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 6.29;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) ;

!CONSTRAINT 6.2;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 6.3;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.8;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));



!CONSTRAINT 6.11;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) -M)));

!CONSTRAINT 6.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 6.17;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);
```

Model 10a single stage

```
SETS:
      job /1..9/: alpha, beta, duration, target, earliness, tardiness,
      scheduled, q;
      tube /1..8/: psi;
      combi (job, tube): tau;
      combi2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      q           =     1     1     2     1     1     3     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 6.1;
min = @sum( job(i): alpha(i) * earliness(i) + beta(i) * tardiness(i)) +
100 * @sum( tube(w): psi(w));

!CONSTRAINT 6.2;
@for( combi2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 6.3;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.4;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 6.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.8;
@for( job(i)|i #gt# 1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 6.11;
@for( job(i): @for( job(j)| j #gt# 1 #and# j #ne# i: @for( tube(w):
scheduled(j) - scheduled(i) - M*phi(i,j,w) >= duration(i) - M)));

!CONSTRAINT 6.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 6.16;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) *
@size(job));

!CONSTRAINT 6.17;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);
```

```
!CONSTRAINT 6.18;
@for( job(i)| i #gt# 1: @for( tube(w): scheduled(i) + duration(i) -
M*(1-tau(i,w)) <= @sum( job(n): duration(n) * tau(n,w))));
```

Model 10a second stage

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta, q;
      tube /1..5/:;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      q           =     1     1     2     1     1     3     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 6.29;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) ;

!CONSTRAINT 6.2;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 6.3;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.8;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));



!CONSTRAINT 6.11;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) -M)));

!CONSTRAINT 6.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 6.17;
@for( tube(w): @sum( job(i): duration(i) * tau(i,w)) <= interval);

!CONSTRAINT 6.18;
@for( job(i): @for( tube(w): scheduled(i) + duration(i) - M*(1-tau(i,w))
```

```
<= @sum( job(k): duration(k) * tau(k,w))));
```

Model 10b single stage

```
SETS:
      job /1..9/: alpha, beta, duration, target, earliness, tardiness,
      scheduled, q;
      tube /1..8/: psi, Xa;
      combi (job, tube): tau;
      combi2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      q           =     1     1     2     1     1     3     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 5.1;
min = @sum( job(i): alpha(i) * earliness(i) + beta(i) * tardiness(i)) +
100 * @sum( tube(w): psi(w));

!CONSTRAINT 5.2;
@for( combi2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 5.3;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 5.4;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 5.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 5.8;
@for( job(i)|i #gt# 1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 5.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 5.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 5.11;
@for( job(i): @for( job(j)| j #gt# 1 #and# j #ne# i: @for( tube(w):
scheduled(j) - scheduled(i) - M*phi(i,j,w) >= duration(i) - M)));

!CONSTRAINT 5.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 5.16;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) *
@size(job));

!CONSTRAINT 5.17;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
```

```
interval);

!CONSTRAINT 5.19;
@for( tube(w): Xa(w) = interval - @sum( job(i): duration(i) *
tau(i,w)));

!CONSTRAINT 5.20;
@for( job(i)| i #gt# 1: @for( tube(w): scheduled(i) + M*(1-tau(i,w)) >=
Xa(w)));
```

Model 10b second stage

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta, q;
      tube /1..5/: Xa;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      q           =     1     1     2     1     1     3     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 6.23;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) ;

!CONSTRAINT 6.2;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 6.3;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.8;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));


!CONSTRAINT 6.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 6.11;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) -M)));

!CONSTRAINT 6.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 6.17;
@for(tube(w): @sum( job(i): tau(i,w) * duration(i)) <= interval);
```

```
!CONSTRAINT 6.19;
@for( tube(w): Xa(w) = interval - @sum( job(i): duration(i)*tau(i,w)));

!CONSTRAINT 6.20;
@for( tube(w): @for( job(i)|i #gt# 1: scheduled(i) + M*(1-tau(i,w)) >=
Xa(w)));
```

Model 10c single stage

```
SETS:
      job /1..9/: alpha, beta, duration, target, earliness, tardiness,
      scheduled, q;
      tube /1..8/: psi, Xa, Xb;
      combi (job, tube): tau;
      combi2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      q           =     1     1     2     1     1     3     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 6.1;
min = @sum( job(i): alpha(i) * earliness(i) + beta(i) * tardiness(i)) +
100 * @sum( tube(w): psi(w));

!CONSTRAINT 6.2;
@for( combi2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 6.3;
@for( combi(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.4;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 6.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.8;
@for( job(i)|i #gt# 1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 6.11;
@for( job(i): @for( job(j)| j #gt# 1 #and# j #ne# i: @for( tube(w):
scheduled(j) - scheduled(i) - M*phi(i,j,w) >= duration(i) - M)));

!CONSTRAINT 6.12;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 6.16;
```

```
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) *
@size(job));

!CONSTRAINT 6.17;
@for( tube(w): @sum( job(i)| i #gt# 1: duration(i) * tau(i,w)) <=
interval);

!CONSTRAINT 6.20;
@for( job(i)|i #gt# 1: @for( tube(w): scheduled(i) + M*(1-tau(i,w)) >=
Xa(w)));

!CONSTRAINT 6.21;
@for( tube(w): interval = @sum( job(i): duration(i) * tau(i,w)) + Xa(w)
+ Xb(w));

!CONSTRAINT 6.22;
@for( job(i): @for( tube(w): scheduled(i) + duration(i) - M*(1-tau(i,w))
<= interval - Xb(w)));
```

Model 10c second stage

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta, q;
      tube /1..5/: Xa, Xb;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
ENDSETS

DATA:
      duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05
                        2.50;
      target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60
                        3.03;
      alpha       =     1     1     1     1     1     1     1     1     1;
      beta        =     1     1     1     1     1     1     1     1     1;
      q           =     1     1     2     1     1     3     1     1     1;
      interval    =     6;
      M           =     10000;
ENDDATA

!OBJECTIVE FUNCTION 6.23;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) ;

!CONSTRAINT 6.2;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 6.3;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.5;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.8;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.9;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.10;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));
```

```
!CONSTRAINT 6.11;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) -M)));

!CONSTRAINT 6.12;
@for( job(i): scheduled(i) + duration(i) <= g);

!CONSTRAINT 6.17;
@for(tube(w): @sum( job(i): tau(i,w) * duration(i)) <= g);

!CONSTRAINT 6.20;
@for( tube(w): @for( job(i)|i #gt# 1: scheduled(i) + M*(1-tau(i,w)) >=
Xa(w)));

!CONSTRAINT 6.21;
@for( tube(w): g = @sum( job(i): duration(i) * tau(i,w)) + Xa(w) +
Xb(w));

!CONSTRAINT 6.22;
@for( tube(w):@for( job(i)| i #gt# 1: scheduled(i) + duration(i) -M*(1-
tau(i,w)) <= g - xb(w)));
```

Model 11

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta, q;
      tube /1..8/:xa, psi;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
      link3 (job,job): setup;
ENDSETS

DATA:
duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05  2.50;
target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60  3.03;
q           =     1     1     2     1     1     3     1     1     1;
alpha       =     1     1     1     1     1     1     1     1     1;
beta        =     1     1     1     1     1     1     1     1     1;
interval    =     6;
M           =     10000;
setup       =     0.00  0.02  0.03  0.03  0.07  0.07  0.13  0.13  0.17
                  0.00  0.00  0.01  0.01  0.05  0.05  0.11  0.11  0.15
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00;
ENDDATA


!OBJECTIVE FUNCTION 6.30;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) + 100 * @sum( tube(w): psi(w)) ;

!CONSTRAINT 6.31;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 6.32;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.33;
```

```
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 6.34;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.37;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.38;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.39;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 6.40;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) + setup(i,j)
-M)));

!CONSTRAINT 6.41;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 6.43;
@for( tube(w): @sum( job(i)| i #gt# 1: tau(i,w)) <= psi(w) * @size(
tube));

!CONSTRAINT 6.44;
@for(tube(w): @for( job(j): @sum( job(i): tau(i,w) * duration(i) +
setup(i,j) * phi(i,j,w)) <= interval));
```

Model 12a

```
SETS:
      job /1..9/: duration, target, earliness, tardiness,scheduled,
      alpha, beta, q;
      tube /1..8/: psi, Xa;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
      link3 (job,job): setup;
ENDSETS

DATA:
duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05  2.50;
target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60  3.03;
alpha       =     1     1     1     1     1     1     1     1     1;
beta        =     1     1     1     1     1     1     1     1     1;
q           =     1     1     2     1     1     3     1     1     1;
interval    =     6;
M           =     10000;
setup       =     0.00  0.02  0.03  0.03  0.07  0.07  0.13  0.13  0.17
                  0.00  0.00  0.01  0.01  0.05  0.05  0.11  0.11  0.15
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00;
ENDDATA

!OBJECTIVE FUNCTION 6.30;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
```

```
tardiness(i)) + 100 * @sum( tube(w): psi(w)) ;

!CONSTRAINT 6.31;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 6.32;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.33;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 6.34;
@for( job(i):scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.37;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.38;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.39;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 6.40;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):scheduled(j)
-scheduled(i) - M* phi(i,j,w) >= duration(i) + setup(i,j) -M)));

!CONSTRAINT 6.41;
@for( job(i):scheduled(i) + duration(i) <= g);

!CONSTRAINT 6.43;
@for( tube(w): @sum( job(i): tau(i,w)) <= psi(w) * @size(tube));

!CONSTRAINT 6.44;
@for(tube(w): @sum( job(i): tau(i,w) * duration(i)) <= g);

!CONSTRAINT 6.45;
@for( job(i): @for( tube(w):scheduled(i) + duration(i) - M * (1-
tau(i,w)) <= @sum( job(k): tau(k,w)* duration(k)) + @sum( job(k):
@sum(job( j)|j #ne# k: phi(k,j,w) * setup(k,j))));
```

Model 12b

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
alpha, beta;
      tube /1..8/: psi, Xa;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
      link3 (job,job): setup;
ENDSETS

DATA:
duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05  2.50;
target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60  3.03;
alpha             =     1     1     1     1     1     1     1     1     1;
beta        =     1     1     1     1     1     1     1     1     1;
q           =     1     1     2     1     1     3     1     1     1;
interval    =     6;
M           =     10000;
setup             =     0.00  0.02  0.03  0.03  0.07  0.07  0.13  0.13  0.17
                  0.00  0.00  0.01  0.01  0.05  0.05  0.11  0.11  0.15
```

```
                0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00;
ENDDATA


!OBJECTIVE FUNCTION 6.30;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) + 100 * @sum( tube(w): psi(w)) ;

!CONSTRAINT 6.31;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 6.32;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.33;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 6.34;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.37;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.38;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.39;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 6.40;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) + setup(i,j)
-M)));

!CONSTRAINT 6.41;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 6.43;
@for( tube(w): @sum( job(i): tau(i,w)) <= psi(w) * @size( tube));

!CONSTRAINT 6.44;
@for(tube(w): @sum( job(i): tau(i,w) * duration(i)) <= interval);

!CONSTRAINT 6.46;
@for( tube(w): Xa(w) = interval - @sum( job(i): duration(i)*tau(i,w)) -
@sum( job(i): @sum( job(j): phi(i,j,w) * setup(i,j))));

!CONSTRAINT 6.47;
@for( tube(w): @for( job(i)|i #gt# 1: scheduled(i) >= Xa(w) + setup(1,i)
- M*(1-tau(i,w))));
```

Model 12c

```
SETS:
      job /1..9/: duration, target, earliness, tardiness, scheduled,
      alpha, beta, q;
```

```
      tube /1..8/: psi, Xa, Xb;
      link (job, tube): tau;
      link2 (job, job, tube): phi;
      link3 (job,job): setup;
ENDSETS

DATA:
duration    =     0     0.80  2.13  2.40  1.72  2.05  2.43  2.05  2.50;
target      =     0     3.55  0.41  2.16  1.49  0.61  0.26  1.60  3.03;
alpha       =     1     1     1     1     1     1     1     1     1;
beta        =     1     1     1     1     1     1     1     1     1;
q           =     1     1     2     1     1     3     1     1     1;
interval    =     6;
M           =     10000;
setup       =     0.00  0.02  0.03  0.03  0.07  0.07  0.13  0.13  0.17
                  0.00  0.00  0.01  0.01  0.05  0.05  0.11  0.11  0.15
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.04  0.04  0.08  0.08  0.14
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.06  0.06  0.11
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.04
                  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00;
ENDDATA

!OBJECTIVE FUNCTION 6.30;
min = @sum( job(i)| i #gt# 1: alpha(i) * earliness(i) + beta(i) *
tardiness(i)) + 100 * @sum( tube(w): psi(w));

!CONSTRAINT 6.31;
@for( link2(i,j,w): @bin(phi(i,j,w)));

!CONSTRAINT 6.32;
@for( link(i,w): @bin(tau(i,w)));

!CONSTRAINT 6.33;
@for( tube(w): @bin(psi(w)));

!CONSTRAINT 6.34;
@for( job(i): scheduled(i) = target(i) + tardiness(i) - earliness(i));

!CONSTRAINT 6.37;
@for( job(i)| i #gt#1: @sum( tube(w): tau(i,w)) = q(i));

!CONSTRAINT 6.38;
@for( job(i): @for( tube(w): @sum( job(j)|j #gt# 1 #and# j #ne# i:
phi(i,j,w)) <= tau(i,w)));

!CONSTRAINT 6.39;
@for( job(j)| j #gt# 1: @for( tube(w): @sum( job(i)| i #ne# j:
phi(i,j,w)) = tau(j,w)));

!CONSTRAINT 6.40;
@for( job(j)| j #gt# 1: @for( job(i)| i #ne# j:@for(tube(w):
scheduled(j) - scheduled(i) - M* phi(i,j,w) >= duration(i) + setup(i,j)
-M)));

!CONSTRAINT 6.41;
@for( job(i): scheduled(i) + duration(i) <= interval);

!CONSTRAINT 6.43;
@for( tube(w): @sum( job(i): tau(i,w)) <= psi(w) * @size(tube));

!CONSTRAINT 6.44;
@for(tube(w): @sum( job(i): tau(i,w) * duration(i)) <= interval);
```

```
!CONSTRAINT 6.47;
@for( tube(w): @for( job(i)|i #gt# 1: scheduled(i) >= Xa(w) - M*(1-
tau(i,w)) + setup(1,i)));

!CONSTRAINT 6.48;
@for( tube(w): @sum( job(i): duration(i)*tau(i,w)) + @sum( job(i): @sum(
job(j): phi(i,j,w) * setup(i,j))) + Xa(w) + Xb(w)= interval);

!CONSTRAINT 6.49;
@for( tube(w):@for( job(i)| i #gt# 1: scheduled(i) + duration(i) -M*(1-
tau(i,w)) <= interval - Xb(w)));
```

Appendix B: Irrigation scheduling I: an integer programming approach

The following paper has been accepted for publication by the Journal of Irrigation and Drainage Engineering (American Society of Civil Engineers).

IRRIGATION SCHEDULING I: AN INTEGER PROGRAMMING APPROACH

by Tonny T. de Vries[2] and Arif A. Anwar[3]

Abstract

This paper shows how a sequential irrigation schedule for a tertiary unit can be interpreted as a single machine scheduling problem with earliness, tardiness and a common deadline. An integer program solution is presented for this irrigation scheduling problem. Two different models are presented to reflect different management options at the tertiary level. The first model allows jobs to be scheduled noncontiguously. In the second model only contiguous jobs are allowed. The second model has three sub models reflecting the various ways in which contiguous jobs can be scheduled over a fixed interval. Earlier work in determining unit costs of earliness/tardiness is reviewed and an alternative improved method is suggested. The models presented in this paper are applied to a tertiary unit with 16 users, both as a single interval and multi interval irrigation scheduling problem. An alternative integer program is also presented which although computationally more efficient can only be used for single period scheduling problems. The models developed in this paper can be used to solve small scheduling problems and also to calibrate the heuristics as presented in the companion paper.

Introduction

Many irrigation systems throughout the world are operated by distributing water sequentially amongst a group of users, normally within a tertiary unit. Such rotational irrigation is typical of small holdings within large irrigation systems and has received considerable attention, particularly with regards to improving equity (e.g. Latif and Sarwar, 1994; Shah and Willardson, 1993). Bishop and Long (1983) developed a rotational schedule taking travel time into account to improve equity. Khepar et al (2000) in their study of the Kotkapura Distributary of the Sirhind System in India, modified the duration allocated to each user to account for seepage losses in the field channel. In both these cited studies, water is essentially allocated pro-rata with area and a sequential schedule prepared which is subsequently adjusted for travel time and/or seepage losses. In such rigid schedules, there is no opportunity for a user to specify how much water is required (duration, flow rate) or the time at which the water is required (start time of irrigation). Wang et al (1995) used integer programming to develop a schedule for a canal whereby the duration of flow at an outlet could be specified by a user. Anwar and Clarke (2001) expanded the work by Wang at al. (1995) and incorporated both duration and start time into the model. Santhi and Pundarikanthan (2000) have criticised the work by Wang et al. (1995) as hypothetical because of the assumption that all outlets have equal discharge. Reddy et al. (1999) showed that a time-block model can be used to eliminate the hypothetical constraint of all outlets having equal discharge. Santhi and Pundarikanthan (2000) also suggested that integer programmes do not take into account

---

[2] Research Student, Dept. Of Civ. and Envir.. Engrg., Univ. of Southampton, Highfield, Southampton, U.K. SO17 1BJ. E-mail: T.T.de-Vries@soton.ac.uk .

[3] Lect., Inst. of Irrigation and Devel. Studies, Dept. of Civ. and Envir. Engrg., Univ. of Southampton, Highfield, Southampton, U.K. SO17 1BJ. Email: A.A.Anwar@soton.ac.uk .

practical management problems.

The machine scheduling problem is a generic problem and has found applications ranging from; the packaging industry, Alder et al (1993) to aircraft landing problems, Beasley et al (2000). In this paper integer programming is used to develop schedules for irrigation systems. The paper shows how the irrigation scheduling problem can be interpreted as the single-machine scheduling problem, hence utilizing some of the wealth of research and literature available on the machine scheduling problem.

The single machine problem and the irrigation scheduling problem

The single-machine problem is one of scheduling any given number of jobs on a single machine. Some characteristics of a typical single machine problem are

- jobs cannot be preempted.
- jobs cannot be serviced simultaneously.
- idle time between jobs may or may not be permitted.
- jobs are processed only once.

In Operations Research over the past decade or so, interest has grown in just-in-time scheduling. Such scheduling, often referred to as the Earliness/Tardiness or E/T problem (Mannur and Addagatla, 1993), encapsulates the notion that a job should finish neither early nor tardy. The earliness costs can be considered as holding costs for finished goods, deterioration of perishable goods and opportunity costs, whereas tardiness costs can be regarded as the costs of back logs, lost sales and loss of goodwill (Liaw, 1999). This dimension of scheduling is captured by penalizing both earliness and tardiness. Baker and Scudder (1990) have presented an authoritative review on the subject.

An analogy can be drawn between a sequential irrigation schedule and the single machine problem. The resource, i.e. the water in the channel is comparable to the machine. The duration of water required by any user is comparable to the processing time of any job. In an irrigation schedule, a user may specify when water is required, i.e. a target start time, likewise, for just-in-time scheduling, the due (completion) time of every job is specified. As with job scheduling, when a user starts to irrigate, the user must be permitted to irrigate without interruption, i.e. preemption is not allowed. Since the schedule is sequential, no two users abstract water from the channel simultaneously - jobs cannot be processed simultaneously on the machine. It would be desirable to provide a user with water exactly when requested, neither early nor tardy, i.e. just-in-time. Finally for a given irrigation interval, a user is provided water only once - jobs are processed only once.

More recent developments in the single machine scheduling incorporate sequence dependent job setup times (Allahverdi et al, 1999; Yang and Liao, 1999). Sequence dependent set up times captures the notion that when one job is complete, the next job can only be started after the machine is appropriately set up. The duration of this setup time will depend upon which job has been completed and which is about to be executed - hence sequence dependent setup time. An analogy also exists between the single machine problem with sequence dependent setup times and the rotational irrigation schedule. When an upstream user completes irrigating, there is a certain travel time before the next user can start to irrigate. In fact this issue of travel time causes the inequity as discussed by Bishop and Long (1983). Therefore strictly speaking the rotational irrigation schedule is a single machine problem with sequence dependent setup times. Where the setup times are equal and/or small relative to processing times, these can be approximated by inclusion in the processing time (Bianco et al., 1988). This simplifies the scheduling problem, and for the purpose of this paper, setup times are ignored.

Irrigation schedules operate over an irrigation interval typically one week or ten days. Therefore, all jobs or irrigation events must be completed by this time. Such a constraint is, as will be demonstrated not an insignificant consideration. Chand and Schneeberger(1988) introduced a single machine scheduling problem where no job is allowed to be tardy. Bagchi and Ahmadi (1992) use the term deadlines to describe the situation where each job has a specific deadline before which it must to be completed. Both these problems have relevance to the irrigation scheduling problem. In irrigation scheduling jobs are allowed to be tardy and do not have job specific deadlines, however all jobs have a common deadline, represented by the irrigation interval. All jobs must be completed before the end of the irrigation interval.

In single machine scheduling problems idle time insertion can lead to significant improvements in the solution. Baker and Scudder (1990) considered the assumption of no inserted idle time inconsistent with the E/T criterion. As described below in irrigation scheduling there is however a legitimate reason for not always allowing inserted idle time. The method of inserting idle time in any irrigation scheduled depends on the manner in which the channel is operated and hence two models are presented. Contrary to the suggestion by Santhi and Pundarikanthan (2000), constraints can be imposed on models to reflect practical management issues as presented. Two models (the second with three further variations) are presented to reflect differing management practices.

Model 1

One alternative is to allow the channel to flow continuously for the entire duration of the irrigation interval. Users abstract water as scheduled and when water is not being used, it spills into a drainage system and/or if possible is reused further downstream. In this form of management the jobs need not be contiguous. This offers greater opportunity for scheduled start time to match target start time, however at the expense of wastage. One could suggest that the channel be shut after every contiguous block of jobs is complete and reopened when the next job is scheduled. However, this would require an excessive number of gate operations.

Model 2

This model requires the jobs to be contiguous, i.e. idle time between jobs is not permitted. Hence the gate can be opened before the first job is scheduled and closed once the last job is completed and therefore avoids spillage. There is less opportunity to match scheduled start time with target start time and therefore this model is likely to give a lower level of service than Model 1. However even in keeping the jobs contiguous there is a choice of the idle time proceeding all jobs, preceding all jobs, or both proceeding and preceding all jobs. Hence Model 2 has three variations, namely

Model 2a all jobs are contiguous, idle time proceeds all jobs.
Model 2b all jobs are contiguous, idle time precedes all jobs.
Model 2c all jobs are contiguous, idle time precedes and/or proceeds all jobs.

For the purpose of this paper a distinction is made between duration and time; duration is used to describe an interval along a time-line eg. irrigation duration. In contrast time refers to an instantaneous point along a time-line eg. scheduled start time. Time along the time line increases from left to right.

Integer programme

Model 1 can be formulated as a mixed integer programme based on that by Coleman (1992). Let $\boldsymbol{Q} = \{J_1, J_2,...,J_n\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $J_i \in \boldsymbol{Q}$: duration, target start time, earliness cost and tardiness cost. A detailed description of the decision variable, objective function and the constraints on each variable follows.

Decision variable

Any job (irrigation event) can precede any other job in the schedule. To define which job precedes which, a decision variable is introduced. This dimensionless variable is a binary integer with indices i and j representing a job. Then

$$\begin{aligned} \delta_{ij} &= 1 \quad \text{if job } i \text{ precedes job } j \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{1}$$

where δ_{ij} = decision variable, i = index representing job 1,...,N; j = index representing job 1,...,N; and N = number of jobs to be scheduled.

Objective function

The goal of the objective function is to find the sequence of jobs and scheduled start time for each job so that every job starts as close as possible to the target start time. This is achieved by minimising the penalties incurred when a job is either early or tardy. The objective function can be written as

$$\text{Minimize } Z = \sum_{i=1}^{N} (E_i\alpha_i + T_i\beta_i) \quad \forall J_i, \quad J_i \in Q \tag{2}$$

where Z = objective function variable; E_i = earliness of job i (appropriate time units); α_i = cost of earliness per unit of time for job i; T_i = tardiness of job i (appropriate time units); and β_i = cost of tardiness per unit of time for job i; and J_i = any job to be scheduled.

Constraints

The scheduled start time of a job is determined from the target start time, the earliness and the tardiness

$$S_i = r_i + T_i - E_i \qquad i = 1,...,N \tag{3}$$

where S_i = scheduled start time of job i; and r_i = target start time for job i. Any two jobs cannot be serviced simultaneously. This implies that either job i precedes job j or vice versa. This can be enforced by using the following two disjunctive constraints

$$S_j - S_i + M(1 - \delta_{ij}) \geq d_i \qquad i = 1,...,N;\ j = 1,...,N;\ i \neq j \tag{4}$$

$$S_i - S_j + M\delta_{ij} \geq d_j \qquad i = 1,..,N;\ j = 1,...,N;\ i \neq j \tag{5}$$

where S_j = scheduled start time of job j; M = a large positive number; d_i = duration of job i; and d_j = duration of job j. Every job must be completed within the irrigation interval (before the common deadline).

$$S_i + d_i \leq g \tag{6}$$

where g = irrigation interval over which all jobs must be completed (common deadline).

Model 2a, 2b and 2c

In Model 2a all jobs are scheduled contiguously with idle time inserted at the end of the last job. Therefore (6) is replaced by

$$S_i + d_i \leq \sum_{i=1}^{N} d_i \qquad i = 1,..,N \tag{7}$$

In Model 2b all jobs are scheduled contiguously and idle time is inserted prior to the start of the first job. Hence constraint (6) becomes

$$S_i \geq g - \sum_{i=1}^{N} d_i \qquad i = 1,...,N \tag{8}$$

In Model 2c, all jobs are scheduled contiguously with idle time preceding the start of the first job and/or proceeding the end of the last job, therefore (6) is altered to

$$g = \sum_{i=1}^{N} d_i + X_a + X_b \qquad i = 1,...,N \tag{9}$$

where X_a = idle time preceding the start of the first job in the schedule; and X_b = idle time proceeding the end of the last job in the schedule. The following additional constraints also need to be included for Model 2c.

$$S_i + d_i \leq g - X_b \qquad i = 1,...,N \tag{10}$$

and

$$S_i \geq X_a \qquad i = 1,...,N \tag{11}$$

Multi interval scheduling

In multi-interval (or multi-period) scheduling decisions made in a previous interval can be used to influence the decisions that will be made for the current interval. In irrigation scheduling the earliness and tardiness incurred in earlier intervals can be used to give priority to those users who have been most affected. To do so earliness and tardiness may be weighted to give preference to one job being processed on time over another. Appropriate costs of earliness/tardiness are difficult to ascertain. Very often costs of earliness/tardiness are set arbitrarily, in many cases by setting all costs of earliness/tardiness equal to 1 (Arkin and Roundy, 1991), which is the same as using a single interval model. Anwar and Clarke (2001) described a method to determine the weighting factor for multi-interval scheduling, based on the lead/lag times. Lead/lag time was defined as the absolute difference between scheduled and target start time and is therefore equal to the sum of earliness and tardiness. Earliness is equally as undesirable as tardiness and the earliness cost per unit of time of job j can be set equal to the tardiness cost per unit of time of job j. The equation for earliness/tardiness costs given by Anwar and Clarke (2001) can be expressed in earliness/tardiness form as

$$\alpha_{i,P} = \beta_{i,P} = \frac{\sum_{p=1}^{P-1} (E_{i,p} + T_{i,p})}{\sum_{i=1}^{N} \left(\sum_{p=1}^{P-1} (E_{i,p} + T_{i,p}) \right)} \times N \qquad \text{for all jobs} \tag{12}$$

Where $\alpha_{j,p}$ = earliness cost of job of job j in interval p; $\beta_{j,p}$ = tardiness cost of job j in interval p; p = index representing interval 1,...,P; and P = current interval.

This method of determining the costs of earliness/tardiness for each job works well in most situations, but a problem arises when one or more jobs in the first interval are scheduled to start at exactly its target start time (i.e. scheduled with no earliness/tardiness). For a job with no earliness or tardiness, the numerator in (12) becomes zero. This sets the costs of earliness/tardiness for this jobs to zero. With a cost of earliness/tardiness of zero, this would allow the job to be scheduled anywhere since from (2) such a job does not contribute to the objective value. Replacing (12) by an exponential function will set the costs of earliness and tardiness to 1 for all jobs with zero earliness/tardiness. This ensures that all jobs contribute to the objective value. Therefore (12) becomes

$$\alpha_{i,P} = \beta_{i,P} = e^{\left(\frac{\sum_{p=1}^{P-1} (E_{i,p} + T_{i,p})}{\sum_{i=1}^{N} \left(\sum_{p=1}^{P-1} (E_{i,p} + T_{i,p}) \right)} \times N \right)} \quad \text{for all jobs} \tag{13}$$

To investigate the differences between using costs of earliness/tardiness estimated by (12) and (13) the interval-averaged standard deviation of the earliness and tardiness can be used. Figure 1a compares the effect of both methods on the standard deviation using Model 1. Target start times and duration are randomly generated but remain constant for all intervals and in the first interval at least one job is scheduled to start at the target start time. Figure 1a also shows the behaviour of the standard deviation if the costs of earliness/tardiness are set to 1 for all intervals (single interval model). It is clear that the standard deviation increases initially if (12) is used to calculate the costs of earliness and tardiness. Since the input data remains constant the standard deviation for the single interval model also remains constant. In contrast the standard deviation using (13) decreases with interval.

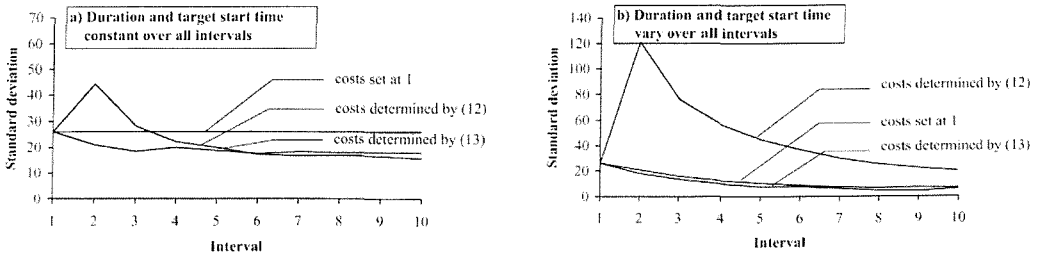


Figure 1 Standard deviation, one or more jobs with no earliness/tardiness

In practice the input data may not remain constant. Figure 1b show how the standard deviation behaves for Model 1, with target start times and duration randomly generated and changing with each interval, such that again at least one job is scheduled to start on time. Due to the changing input data the single interval model shows a decrease in standard deviation. The rise in standard deviation, when determining costs of earliness/tardiness with (12) is even more pronounced then in Figure 1a. Using (13) again shows a decrease in standard deviation with interval.

It is possible that in the first interval none of the jobs are scheduled to start on time. Figure 2a shows how using (13) improves the standard deviation compared to (12). Target

start times and duration are again randomly generated but remain constant for all intervals. Figure 2b shows that (12) and (13) perform equally well if the input data is random for each interval and no jobs start on time during the first interval. Both multi-interval models using (12) and (13) to calculate the costs of earliness and tardiness perform better than the single interval model.

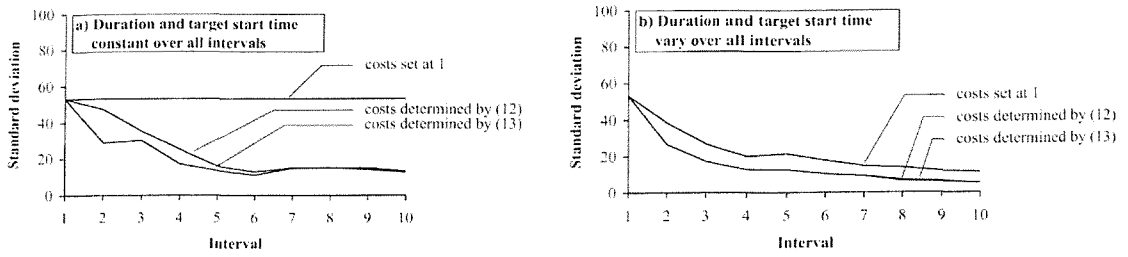


Figure 2 Standard deviation, all jobs with earliness/tardiness

The models presented in this section can be computationally demanding and relatively small problems may not solve in reasonable time. In Appendix I, an alternative formulation for each of Model 1, 2a, 2b and 2c is presented. The alternative formulation is computationally more efficient and allows larger problems to be solved but does not allow for different costs of earliness and tardiness for each job as needed for multi-interval scheduling. Table 1 shows the number of constraints and variables for each model with both formulations.

TABLE 1: Number of variables and constraints for all models

Model	Formulation	Number of variables	Number of constraints
(1)	(2)	(3)	(4)
1	First	N^2+3N	$2N^2+1$
	Alternative	N^2+3N	$3N+2$
2a	First	N^2+3N	$2N^2+1$
	Alternative	N^2+2N	$3N+1$
2b	First	N^2+3N+1	$2N^2+N+2$
	Alternative	N^2+2N	$3N+1$
2c	First	N^2+3N+3	N^2+2N+1
	Alternative	N^2+2N+1	$3N+1$

Table 1 shows the number of variables for all models is $O(N^2)$. The difference in number of constraints between the two formulations explains why the alternative formulation is less computationally demanding: for the first formulation the number of constraints is $O(N^2)$, but for the alternative it is only $O(N)$. However this is not the only influence on computation time. Table 2 shows computation times for a problem with 16 jobs.

TABLE 2: Computation times (s) for Model 1, 2a, 2b, 2c

Formulation	Model

	1 (1)	2a (2)	2b (3)	2c (4)
First	185	>20000	>20000	>20000
Alternative	88	12074	19	8286

It can be seen that there is a considerable variation in computation time between the four models. Computation time is particularly data sensitive, and cannot be predicted a priori.

Practical application

Models 1, 2a, 2b and 2c were implemented in Lingo 6.0® for Windows®, using data from Bishop and Long (1983). Bishop and Long (1983) presented a procedure for setting up a rotation delivery schedule and applied it to a tertiary unit of 37.06 ha with 16 water users. Water in this tertiary unit is allocated pro rata with area at 172 min/ha and 349 min are available for management/canal filling. Bishop and Long (1983) suggested the irrigation to be scheduled in turn from downstream upwards, target start times were not included in their data. For the purpose of this practical application the irrigation interval is set at 7 days and the target start times are randomly generated over this interval. The duration of each job is 172 min/ha as suggested by Bishop and Long (1983). Management/fill time is equally divided amongst all users to account for setup times and is added to the duration.

Figure 3 shows the four different schedules obtained when applying Model 1, 2a, 2b and 2c. It can be seen in Model1 that idle time is inserted in three places in the schedule; after lot 24.4 is irrigated, after lot 24.3 is irrigated and after lot 26.1 is irrigated. The schedule from Model 1a results in an average earliness/tardiness of 4.0 hrs/user. For Model 2a all idle time is inserted after all users have finished irrigating. This schedule results in an average earliness/tardiness of 20.5 hrs/user. Figure 3 also shows that for Model 2b all idle time is inserted before any of the users starts irrigation. The average earliness/tardiness of this schedule is 33.3 hrs/user. Figure 3 shows how for Model 2c 7.7 hours of idle time is inserted before the first user (lot 24.1) and another 48.3 hours is inserted after the last user (lot 25.3) has finished irrigation. This schedule results in an average earliness/tardiness of 19.8 hrs/user.

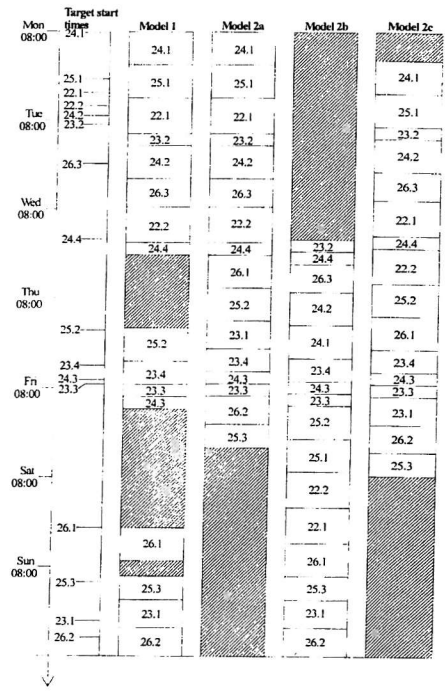


Figure 3 Single-interval schedules applying Models 1, 2a, 2b and 2c

It is clear that Model 1 gives the best results in terms of average earliness/tardiness, however this model leads either to operational spillage or an excessive number of gate operations. Model 2b gives the highest average earliness/tardiness. This is due to the concentration of target start times early in the interval. Model 2a and 2c perform better than Model 2b, with only a small difference between them. Their results are not as good as those of Model 1, however gate operations are limited to opening once and closing once and spillage can be easily avoided.

Table 3 is a multi-interval schedule obtained by applying Model 1 over 3 intervals.

TABLE 3: Irrigation schedule for three intervals

Interval 1			Interval 2			Interval 3		
Lot (1)	On (2)	Off (3)	Lot (4)	On (5)	Off (6)	Lot (7)	On (8)	Off (9)
24.1	Mo 08:00	Mo 16:57	26.2	Mo 10:22	Mo 19:23	23.1	Mo 08:00	Mo 18:49
25.1	Mo 16:57	Tu 01:54	23.3	Mo 19:23	Mo 23:15	25.1	Mo 18:49	Tu 07:42
22.1	Tu 01:54	Tu 11:26	26.3	Mo 23:15	Tu 08:28	24.3	Tu 07:42	Tu 12:20
23.2	Tu 11:26	Tu 14:39	22.2	Tu 08:28	Tu 19:54	24.1	Tu 12:20	We 01:13
24.2	Tu 14:39	Tu 23:36	24.2	Tu 20:15	We 06:59	26.3	We 01:13	We 12:17
26.3	Tu 23:36	We 07:17	23.4	We 06:59	We 14:11	22.2	We 12:17	Th 02:00
22.2	We 07:17	We 16:49	24.4	We 14:11	We 18:13	26.1	Th 02:00	Th 14:53
24.4	We 16:49	We 20:11	26.1	We 18:13	Th 04:57	25.3	Th 14:53	Fr 00:10
25.2	Th 16:02	Fr 00:59	23.2	Th 04:57	Th 08:49	23.2	Fr 06:58	Fr 11:36
23.4	Fr 00:59	Fr 06:59	25.1	Th 08:49	Th 19:33	24.2	Fr 11:36	Sa 00:29
23.3	Fr 06:59	Fr 10:12	25.3	Th 19:33	Fr 03:17	24.4	Sa 00:29	Sa 05:19
24.3	Fr 10:12	Fr 13:25	25.2	Fr 03:17	Fr 14:01	25.2	Sa 05:19	Sa 18:12
26.1	Sa 21:29	Su 06:26	23.1	Fr 16:42	Sa 01:43	23.3	Sa 18:12	Sa 22:50

25.3	Su 10:31	Su 16:58	22.1	Sa 01:43	Sa 13:09	26.2	Su 22:50	Su 09:39
23.1	Su 16:58	Mo 00:29	24.1	Sa 13:09	Sa 23:53	23.4	Su 09:39	Su 18:17
26.2	Mo 00:29	Mo 08:00	24.3	Mo 04:08	Mo 08:00	22.1	Su 18:17	Mo 08:00

The data for the first interval is the same as used for Figure 3. For the second and third interval new target start times are generated randomly. To simulate an increase in water demand the duration of each job has been increased by 20% for the second interval and by another 20% for the third interval. Earliness and tardiness costs have been calculated using (13). The average earliness/tardiness for the first interval is 4.0 hrs/user. Due to the increased duration and different start times the schedules for the second and third intervals are distinct with an average earliness/tardiness of 4.1 hrs/user and 4.9 hrs/user respectively.

The technique used to determine the schedules presented in Table 3 is more sophisticated than that presented by Bishop and Long (1983), but the output is a simple, easy to understand schedule. The difference lies in the level of service that can be provided with this new technique. Whereas Bishop and Long (1983) promoted a rigid schedule that is fixed and cannot be changed, the proposed method allows users to request water when it is needed most. The method can also be adapted so that both target start times and duration can be requested and a schedule is generated accordingly.

Conclusion

Integer programmes provide optimum solutions, however for combinatorial problems such as scheduling, they are computationally demanding and often cannot solve large problems within reasonable time. Although other methods are needed to be able to use single machine scheduling in practical irrigation systems, integer programming also has it uses. Methods such as the heuristics proposed in the companion paper are computationally efficient and can handle any practical size of problems but do not necessarily provide an optimum solution. As the companion paper shows integer programming can be used to test the quality of heuristics in addition to solving problems with a small number of jobs.

Sequence dependent setup times capture the notion that a certain amount of time is required to prepare for the next job to be processed. In an irrigation environment this is the time needed for water to travel from one user to the next. Setup times have currently not been considered. This is a valid assumption if the setup times required are small compared to the duration of jobs. However, if this is not the case sequence dependent setup times need to be included.

In certain irrigation intervals, like early or late in a growing season, not all users may wish to irrigate. The proposed method of determining the costs of earliness and tardiness do not take this possibility into account. Setting the duration of the job that is to be skipped to zero would mean that this job will be considered as scheduled to be on time, allowing it to be more early or tardy in a consequent interval. This effectively penalises a user for not irrigating. This issue needs to be addressed in further work

Authors' note

Trade names and company names are used in this paper solely for the purpose of providing specific information. Their mention does not constitute a guarantee or warranty or endorsement of the company or product by the authors or by the Department of Civil & Environmental Engineering, University of Southampton, UK.

Appendix I. Alternative single interval mixed integer linear programme

Model 1 can be formulated as a mixed integer linear programme based on that presented by Fry and Leong (1987). A detailed description of the decision variable, objective function and the constraints on each variable follows.

Decision variable

Any job (irrigation event) can be assigned to any position in the schedule. To define which job is assigned to which position, a decision variable is introduced. This dimensionless variable is a binary integer with index i representing job J_i and index k representing a position in a schedule. Then

$$\lambda_{ik} = \begin{cases} 1 & \text{if job } i \text{ is assigned to position } k \text{ in the schedule} \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

where λ_{ik} = decision variable; and k = index representing position 1,..,N;

Objective function

The objective is to minimize the cost of a job being scheduled early or tardy becomes

$$\text{minimize} \quad Z = \alpha \sum_{k=1}^{N} E_k + \beta \sum_{k=1}^{N} T_k \tag{15}$$

where Z = objective function variable; α = earliness penalty cost per unit of time; E_k = earliness of the job in position k; β = tardiness cost per unit of time; and T_k = tardiness of the job in position k.

Constraints

Each job can only once be assigned to a position

$$\sum_{i=1}^{N} \lambda_{ik} = 1 \qquad k = 1,...,N \tag{16}$$

Each position can only be assigned once

$$\sum_{k=1}^{N} \lambda_{ik} = 1 \qquad i = 1,...,N \tag{17}$$

The scheduled start time for the job in position k is equal to the target start time and the earliness/tardiness of that job

$$S_k = r_k + T_k - E_k \tag{18}$$

where S_k = scheduled start time of the job in position k; and r_k = target start time of the job in position k.

Since

$$S_k = \sum_{m=1}^{k} Y_m + \sum_{m=1}^{k-1} d_m \tag{19}$$

where m = an index representing a position in the schedule; Y_m = idle time inserted directly before the job in position m; and with

$$d_m = \sum_{i=1}^{N} \lambda_{im} d_i \tag{20}$$

and

$$r_k = \sum_{i=1}^{N} \lambda_{ik} r_i \tag{21}$$

Using (19), (20) and (21), (18) can be rewritten as

$$\sum_{m=1}^{k} Y_m + \sum_{m=1}^{k-1} \sum_{i=1}^{N} \lambda_{im} d_i = \sum_{i=1}^{N} \lambda_{ik} r_i - E_k + T_k \tag{22}$$

Model 2

In Model 2a all jobs are scheduled contiguously with idle time inserted at the end of the interval, and (19) is replaced by

$$S_k = \sum_{m=1}^{k-1} d_m \tag{23}$$

from (20), (21) and (23), (18) becomes

$$\sum_{m=1}^{k-1} \sum_{i=1}^{N} \lambda_{im} d_i = \sum_{i=1}^{N} \lambda_{ik} r_i - E_k + T_k \tag{24}$$

For Model 2b, (24) is replaced by

$$\sum_{m=1}^{k-1} \sum_{i=1}^{N} \lambda_{im} d_i = \sum_{i=1}^{N} \lambda_{ik} r_i - E_k + T_k - X_a \tag{25}$$

The following additional constraint is necessary for Model 2b.

$$\sum_{k=1}^{N} d_k = g - X_a \tag{26}$$

To schedule a number of jobs so that the first job is scheduled to start some time after the start of the interval and the last job is scheduled to finish sometime before the end of the interval (Model 2c), the constraints for Model 2b are used with the equality given by (26) omitted.

Appendix II. References

Alder L., Fraiman N.M., Kobacker, E. Pinendo, M., Plotnitcoff J.C. and Wu T.P. (1993). " BPSS: a scheduling system for the packaging industry". *Operations Research*, Operations Research Society of America, 41:641-48.

Allahverdi A., Gupta J.N.D. and Aldowaisan T. (1999). "A Review of Scheduling Research Involving Setup Considerations." *Omega International Journal Management Science* Pergamon 27:219-239.

Anwar A.A. and Clarke D. (2001). "Irrigation Scheduling Using Mixed-Integer Linear Programming." *Journal of Irrigation and Drainage Engineering*, ASCE, 127(2): 63-69.

Arkin, E.M. and Roundy, R.O. (1991). "Weighted-tardiness scheduling on parallel machines with proportional weights." *Operation Research*, Operations Research

Society of America, 39(1), 64-81.
Bagchi, U. and Ahmadi, R.H. (1992). "Minimizing job idleness in a deadline constrained environments." *Operations Research,* Operations Research Society of America, 40(5), 972-985.
Baker K.R. and Scudder G.D. (1990) "Sequencing with earliness and tardiness penalties: A review." *Operations Research,* Operations Research Society of America, 38(1), 22-36.
Beasley J.E., Krishnamoorthy M., Sharaiha Y.M. and Abramson D. (2000). "Scheduling Aircraft Landings - The Static Case." *Transportation Science*, Informs, 34(2), 180-197.
Bianco, L., Ricciardelli, S., Rinaldi, G. and Sassano, A. (1988). "Scheduling tasks with sequence-dependent processing times." *Naval Research Logistics*, Wiley, 35(2), 177-184.
Bishop A.A. and Long A.K. (1983). "Irrigation Water Delivery for Equity between Users." *Journal of Irrigation and Drainage Engineering*, ASCE 109(4), 349-356.
Chand, S. and Schneeberger, H. (1988). "Single machine scheduling to minimize weighted earliness subject to no tardy jobs." *European Journal of Operational Research*, Elsevier, 34(2), 221-230.
Coleman, B.J. (1992). "Technical note: a simple model for optimizing the single machine early/tardy problem with sequence dependent setups." *Production and Operations Management*, POMS, 1(2), 225-228.
Fry T.D. and Leong, G.K. (1987). "A Bi-Criterion Approach to Minimizing Inventory Costs On a Single Machine When Early Shipments Are Forbidden." *Computers and Operations Research*, Pergamon, 14(5), 363-368.
Khepar S.D., Gulati H.S., Yadav A.K. and Brar T.P.S. (2000). "A Model for Equitable Distribution of Canal Water." *Irrigation Science*, Springer Verlag, 19:191-197.
Latif, M. and Sarwar, S. (1994). "Proposal for equitable water allocation for rotational irrigation in Pakistan." *Irrigation and Drainage Systems*, Kluwer, 8(1), 35-48.
Liaw, C.-F. (1999). "A branch-and-bound algorithm for the single machine earliness and tardiness scheduling problem." *Computers and Operations Research*, Pergamon Elsevier, 26(7), 679-693.
Mannur N.R. and Addagatla J.B. (1993). "Heuristic Algorithms for Solving Earliness-Tardiness Scheduling Problem with Machine Vacations." *Computers and Industrial Engineering*, Pergamon Press 25(1-4): 255-258.
Reddy, J.M., Wilamowski, B., and Sharmasarkar, F.C. (1999). "Optimal scheduling of irrigation for lateral canals." *ICID Journal*, ICID, 48(3), 1-12.
Santhi C. and Pundarikanthan N. (2000) "A New Planning Model for Canal Scheduling of Rotational Irrigation." *Agricultural Water Management,* Elsevier 43:327-343.
Shah, M.H. and Willardson, L.S. (1993). "Equitable distribution in a rotation irrigation system." *Management of Irrigation and Drainage Systems: Integrated Perspectives*. Conference, Jul 21-23 1993, Park City, Utah, USA, ASCE, 24-31.
Wang, Z., Reddy, M.J. and Feyen J. (1995). "Improved 0-1 programming model for optimal flow scheduling in irrigation canals." *Irrigation and Drainage Systems*, Kluwer, 9, 105-116.
Yang, W. and Liao, C. (1999). "Survey of scheduling research involving setup times." *International Journal of Systems Science*, Taylor & Francis, 30(2), 143-155.

Appendix III. Notation

the following notation was used in this paper

d_i = duration of job i;
d_j = duration of job j;
E_i = number of time units job i is early;
E_k = number of time units job in position k is early;
g = total irrigation (duration) interval;
i = integer representing the index of a job;
j = integer representing the index of a job;
J_i = job to be scheduled;
k = integer representing the index of a position;
m = integer representing the index of a position;
M = a large positive number;
N = number of jobs to be scheduled;
P = current interval;
p = integer representing the index of an interval;
$\boldsymbol{Q}$ = set of jobs to be scheduled;
r_i = target start time for job i;
r_k = target start time for job in position k;
S_i = scheduled start time of job i;
S_j = scheduled start time of job j;
S_k = scheduled start time of job in position k;
T_i = number of time units job i is tardy;
T_k = number of time units job in position k is tardy;
X_a = idle time preceding the start of the first job in the schedule
X_b = idle time proceeding the end of the last job in the schedule.
Y_m = idle time inserted before job in position m;
Z = objective function variable;
α = cost of earliness per unit of time;
α_i = cost of earliness per unit of time for job i;
$\alpha_{i,p}$ = cost of earliness per unit of time for job i in interval p;
β = cost of tardiness per unit of time;
β_i = cost of tardiness per unit of time for job i;
β_i = cost of tardiness per unit of time for job i in interval p;
δ_{ij} = decision variable; and,
λ_{ki} = decision variable;

Appendix C: Irrigation scheduling II: an heuristic approach

The following paper has been accepted for publication by the Journal of Irrigation and Drainage Engineering (American Society of Civil Engineers).

IRRIGATION SCHEDULING II: A HEURISTICS APPROACH

by Arif A. Anwar[4] Tonny T. de Vries[5]

Abstract

A sequential irrigation schedule that honours user demands of duration and minimizes earliness and tardiness is interpreted as a single machine schedule with earliness and tardiness costs and a common deadline (or fixed interval). A heuristic solution is presented for this irrigation scheduling problem. Four models are presented to reflect the different methods in which an irrigation system at the tertiary unit level may be operated, the first model permits jobs to be non-contiguous i.e. idle time between jobs is permitted, whereas the others permit contiguous jobs only. The heuristic is tested extensively and the solution quality is compared with, either an optimum solution from an integer programme, or the best available solution obtained from an integer programme within allocated computation time. The heuristic is computationally efficient for all models presented, however for schedules with non-contiguous jobs, or where idle time is inserted before and after a contiguous set of jobs, solution quality deteriorates. The work brings the science of single scheduling from operations research into irrigation scheduling and suggests areas for further development.

Introduction

The companion paper demonstrated that the irrigation scheduling problem which honours user demands of duration and minimizes earliness and tardiness can be analysed as a machine scheduling problem. The scheduling of water within a tertiary unit over a given irrigation interval was shown to be similar to a single machine scheduling problem. The companion paper used integer programming to solve the single machine scheduling problem. Although integer programming does give optimum solutions, it is computationally demanding and when the number of water users (jobs) exceeds 20, it is no longer a very practical tool. In this paper heuristics are used as an alternative tool to solve the irrigation scheduling problem. Heuristics are developed for all four models introduced in the companion paper. All models are for sequential irrigation i.e. only one user may utilize water at any one time. Model 1 reflects a practice where one user does not necessarily use water immediately after the previous user finishes using water i.e. idle time is permitted. In Model 2, idle time between users is not permitted, rather the idle time is aggregated either after all users have finished using water - Model 2a, or aggregated before all users start using water - Model 2b, or both before and after all users have been scheduled water - Model 2c.

The heuristic is tested extensively tested for solution quality against hypothetically generated data to evaluate solution quality against various parameters. The heuristic is

---

[4] Lect., Inst. of Irrigation and Devel. Studies, Dept. of Civ. and Envir. Engrg., Univ. of Southampton, Highfield, Southampton, U.K. SO17 1BJ. Email: A.A.Anwar@soton.ac.uk.

[5] Research Student., Dept. Of Civ. and Envir.. Engrg., Univ. of Southampton, Highfield, Southampton, U.K. SO17 1BJ. E-mail: T.T.de-Vries@soton.ac.uk .

Keywords: Heuristics, Irrigation Scheduling, Optimization, Water Allocation Policy

then applied to a practical irrigation scheduling problem for conditions under which it provides reasonable solution quality.

Heuristics

Heuristics for the machine scheduling problem typically consist of a 'dispatching rule' which is essentially a of rule that decides which job to 'dispatch' first and which to dispatch second. This is then followed by an interchange procedure and then (if idle time is permitted), idle times are inserted between jobs where it improves the schedule. The heuristic presented here uses the four dispatching rules suggested by Kim and Yano (1994) but modified to accommodate job target start time rather than job due (completion) time.

1. EST (Earliest Start Time): In this rule the jobs are sorted in ascending order of target start time. If any two or more jobs have identical earliest start times, these are sorted using the Earliest Completion Time (ECT) rule below. The job ranked first is dispatched first.
2. ECT (Earliest Completion Time): In this rule the jobs are sorted in ascending order of completion time (sum of start time and duration). If any two or more jobs have identical earliest completion times, these are ranked using the EST rule above.
3. PRIO (Priority): In this precedence rule, every job is assigned a ranking integer initially 0. Every job *i* is then compared with every other job *j* using the EST rule. If job *i* is to precede job *j* then the ranking integer for job *i* is updated by adding 1 to it and the ranking integer for job *j* is updated by subtracting 1 from it. Conversely if using the EST rule job *j* is to precede job *i*, the ranking integers are updated by adding 1 to that of job *j* and subtracting 1 from that of job *i*. After comparing all jobs, the jobs are sorted in decreasing order of the ranking integer.
4. NEH (Nawaz, Enscore and Ham rule): This rule was developed by Nawaz et al (1983) for multi machine problems and was adapted by Kim and Yano (1994) for the single machine problem. This rule first requires jobs to be sorted by a specified rule. Kim and Yano (1994) specified four alternative sorting rules which are adapted for target start times as: EST (earliest start time), LST (latest start time), SPT (shortest processing time), and LPT (longest processing time). The dispatching rules are denoted NEHest, NEHlst, NEHspt and NEHlpt respectively. The schedule is built up by taking the first job from the sorted set directly. The next job is placed either before or after the first job depending on which insertion increases the objective function least. This procedure is repeated for each of the jobs in the sorted set to produce a sequence of jobs.

Once a schedule is prepared using the dispatching rules, the best schedule (that with the smallest objective function as defined by equation (2) in the companion paper) is improved by a pairwise interchange (PI). In this interchange every job is exchanged with every other in an attempt to improve (reduce) the objective function. If the objective function improves the new schedule is retained, otherwise it is discarded. Clearly there is little benefit in exchanging the first job with one considerably further along the time line, since the first job will have a much higher ranking than any one further along. Often pairwise interchange routines are restricted to exchange within an arbitrary number of jobs. Lee et al (1997) restricted pairwise interchange of a given job to within the next 20 jobs only. An extreme of this restriction would be to only attempt a pairwise interchange between adjacent jobs - called adjacent pairwise interchange (API). The only advantage to any such restrictions on pairwise interchange is to improve computational efficiency of the algorithm albeit possibly at the expense of solution quality. Kim and Yano (1994)

indicated that the pairwise interchange procedure is sufficiently fast that it does not merit reducing the number of exchanges. Mazzini and Armantano (2001) indicated that for their heuristic the full pairwise interchange increases computational time significantly whereas improvement in solution quality is minimal and hence advocated against the pairwise interchange for their heuristic. The heuristic presented here incorporates an adjacent pairwise interchange.

The third phase in the development of the heuristic is to take the schedule after the interchange procedure and insert idle time. This in turn depends on whether or not idle time is permitted between jobs i.e. can jobs be non-contiguous.

Model 1

This model allows idle duration to be inserted between jobs. The procedure is based on the Optimal Adjacency Theorem (Ow and Morton, 1989). It is modified here to include scheduling over a fixed interval. A block is defined as a group of contiguous jobs (no idle duration between jobs). Blocks are numbered in ascending order along the time line. For any block

$$S_{i-1} + d_{i-1} = S_i \qquad \forall\, i = 2...n_b, \quad \forall b = 1..t \tag{27}$$

where S_{i-1} = scheduled start time of job i-1, d_{i-1} = duration of job i-1; S_i = scheduled start time of job i; i = integer representing index of a job; n_b = number of jobs in block b ; b = order of block; and t = total number of blocks.

The idle duration insertion procedure consists of left displacement and right displacement procedures. The left displacement procedure attempts to improve the solution by adjusting the idle duration to the left of any block. The right displacement procedure attempts to improve the solution by adjusting idle duration to the right of any block. Figure 1 is a schematic of the left displacement procedure.

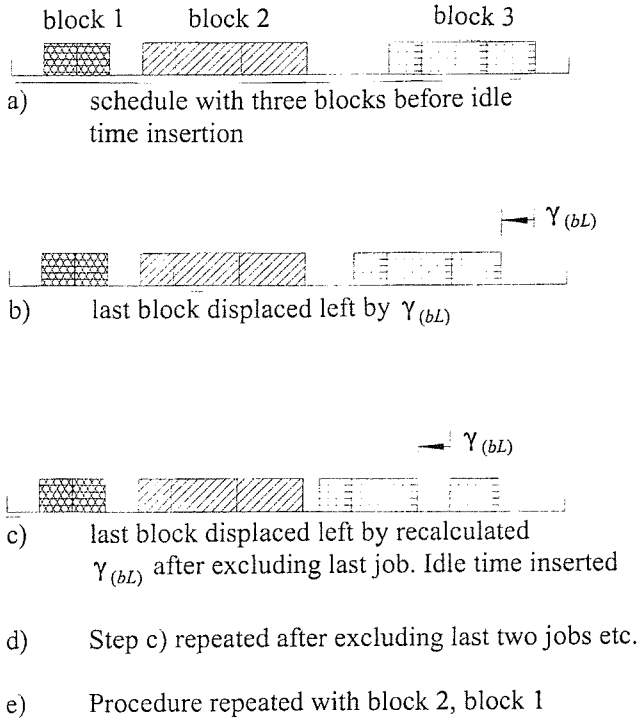


Figure 4 Schematic of left displacement procedure

It starts with the last block along the time line. Displacing a block left will improve tardiness of all jobs that are tardy, although it will exacerbate earliness. The maximum duration a block can be displaced to the left is given by

$$\gamma_{bL} = - \text{ minimum of } \begin{bmatrix} T_{\min_b} \\ S_{1_b} - \left[S_{n_{b-1}} + d_{n_{b-1}} \right] \end{bmatrix} \qquad \forall\ b = t-1..2 \qquad (28)$$

where γ_{bL} = block displacement left; $T_{\min_b}$ = minimum positive tardiness of all jobs in the block; S_{1_b} = scheduled start time of the first job in the block ; $S_{n_{b-1}}$ = scheduled start time of last job in the preceding block; and $d_{n_{b-1}}$ = duration of the last job in the preceding block. For the first block (b=1), (2) becomes

$$\gamma_{1L} = - \text{ minimum of } \begin{bmatrix} T_{\min_1} \\ S_{1_1} - 0 \end{bmatrix} \qquad (34)$$

where γ_{1L} = block displacement left for the first block; $T_{\min_1}$ = minimum positive tardiness of all jobs in the first block;. S_{1_1} = scheduled start time of the first job in the first block.. The procedure involves displacing the block by the block displacement left as calculated from (2) or (3). If this displacement decreases the objective function, the displacement is retained otherwise it is discarded. The procedure now recalculates the block displacement left from (2) or (3) excluding the last job in the block under consideration. This procedure is repeated for all the jobs in the block with each iteration excluding additional jobs on the right of the block. Once all jobs in a block have been processed in this way, the preceding block is processed in a similar manner ($b = t, t-1, t-2,....1$).

The procedure continues with inserting idle duration by displacing each block right in turn starting with the first block. For right displacement (2) and (3) become

$$\gamma_{bR} = + \text{ minimum of } \begin{bmatrix} E_{\min_b} \\ S_{1_{b+1}} - \left(S_{n_b} + d_{n_b} \right) \end{bmatrix} \qquad \forall\ b = 1..t-1 \qquad (38)$$

where γ_{bR} = block displacement right; and, $E_{\min_b}$ = minimum positive earliness of all jobs in the block; $S_{1_{b+1}}$ = scheduled start time of the first job in the proceeding block; S_{n_b} = scheduled start time of last job in the block; and d_{n_b} = duration of the last job in the block, and

$$\gamma_{tR} = + \text{ minimum of } \begin{bmatrix} \left(E_{\min} \right)_t \\ g - \left(S_{n_t} + d_{n_t} \right) \end{bmatrix} \qquad (44)$$

where γ_{tR} = block displacement right for the last block; and, $E_{\min_t}$ = minimum positive earliness of all jobs in the last block; g = irrigation interval (duration); S_{n_t} = scheduled start time of last job in the last block; and d_{n_t} = duration of the last job in the last block . The procedure is identical to that described for the left displacement except; the routine starts with the first rather than last block; all displacements are to the right; and, the first

job, second job etc. are excluded in each iteration.

After each idle duration insertion, the blocks within the feasible ordered set need to be re-identified. By inserting idle duration between two jobs, jobs may become contiguous with the preceding or proceeding block and therefore needs to be considered as jobs within that block for any further processing.

Model 2a

Model 2a is simply a sequencing problem and there is no insertion of idle time.

Model 2b

Model 2b is again a sequencing problem without any idle time and is a mirror image of Model 2a. By measuring target start time from the end of the interval, rather than the beginning of the interval, Model 2b becomes identical to Model 2a. Scheduled times will also be relative to the end of the interval rather than beginning of the interval.

Model 2c

Model 2c only inserts idle time before and/or after a contiguous block of jobs. Therefore the left displacement and right displacement procedures described for Model 1 are executed on the entire block only. The procedure then terminates without excluding any jobs.

Computational Experiments

Rardin and Uzsoy (2001) presented a comprehensive tutorial on the evaluation of heuristics and suggested the best test instances are those taken from real application. However it is rare to obtain more than a few real data sets which would be insufficient to test any heuristic comprehensively. Alternative sources of data include; random variation of real data sets; published on line libraries eg. Beasley, (1996); and, randomly generated instances. Since real data sets in sufficient quantity are unavailable, using real data sets and/or random variation of real data sets is not possible. Published online libraries for the irrigation scheduling problem do not exist. Although numerous libraries for the single machine problem do exist, none include the constraint of completing all jobs over a given interval. For this work randomly generated data are used to test the heuristic.

Rardin and Uzsoy (2001) cautioned against testing heursitics against integer programmes if the size of problems solved by integer programmes is considerably smaller than practical problems. For larger problems it may not be possible to solve the scheduling problem within reasonable computation time using integer programme. Therefore, any new heuristic needs to be tested against other heuristics, and that which consistently gives the lower objective function is considered to be the better. For the irrigation scheduling problem, results from earlier heuristics are not available. Hence the heuristic can only be tested against exact solutions obtained from the integer programmes in the companion paper. To manage computation time a limit of 10^4 seconds (2.77 hours) is set for the integer programme. In test instances where the integer programme is interrupted it reaches a feasible but not necessarily global optimum solution. Hence it is possible that the heuristic may reach a better solution when compared against the integer programme for these instances i.e. the heuristic is being compared against the best known solution.

Durations, target start times and cost of earliness/tardiness for each of the jobs are generated following the method suggested by Potts and Van Wassenhove (1982). This technique is widely used in testing of heuristics eg. Kim and Yano (1994), Mazzini and Armentano (2001). Job duration (duration a user requires water) is generated as a uniformly distributed random integer from the range [1,100].

Potts and Van Wassenhove (1982) presented the following expressions for estimating the lower and upper bounds for randomly generated target start times

$$B_L = \left(1 - F - \frac{R}{2}\right) \tag{49}$$

$$B_U = \left(1 - F + \frac{R}{2}\right) \tag{50}$$

where B_L = lower bound for target start time; B_U = upper bound for target start time; F = tardiness factor; and, R = start time range factor. A high tardiness factor implies that all jobs have a target start time nearer the beginning of the interval eg. all users requested water nearer the beginning of an interval, therefore any schedule will have a high number of jobs that are tardy - high tardiness factor. For a given data set, the tardiness factor is simply

$$F = 1 - \frac{\bar{r}}{g} \tag{8}$$

where $\bar{r}$ = average target start time of all jobs. The start time range factor is a measure of the range of target start times over the interval. A low start time range factor implies all target start times will lie within a narrow range of the interval. A start time range factor of 1.0 implies that target start time may lie anywhere along the interval. For a given data set the start time range factor is

$$R = \frac{r_{\max} - r_{\min}}{g} \tag{9}$$

where $r_{\max}$ = maximum target start time; and, $r_{\min}$ = minimum target start time. In practice users can request water for any time during an interval i.e. target start time range = 1.0. The target start time for each job is a uniformly distributed random integer generated from the range $[B_L g, B_U g]$, where g = interval. This is in slight contrast to the method suggested by Potts and Van Wassenhove (1982) in which the range is the product of lower bound/upper bound factor and makespan (sum of all job durations).

Cost of earliness/tardiness per unit of time is a uniformly distributed randomly generated integer from a range [0,5]. If cost of earliness is set equal to cost of tardiness, this implies if a user is scheduled water one unit of time early, this is equally undesirable to the case where the user is scheduled to receive water a unit of time tardy. If one user has higher cost of earliness/tardiness than a second user, this implies the first is to receive priority over the second.

The interval is a constant of 800 time units - the product of the number of jobs and the maximum duration of any job. The ratio of interval to makespan (sum of processing duration of all jobs) is defined by

$$r_{IM} = \frac{g}{\sum_{i=1}^{N} d_i} \qquad \forall i = 1...N \tag{10}$$

where r_{IM} = interval to makespan ratio; d_i = duration of job i and; N = number of jobs, a range of values 1.00 - 2.50 for the interval to makespan ratio (IM ratio) are selected. A ratio of 1.0 reflects a high demand for water, whereas a higher ratio indicates lower demand.

For each of the parameters defined above, a range of values is selected and instances generated. Each instance is checked to ensure that the target completion time

does not exceed the end of the interval, otherwise the instance is rejected and is regenerated. This reflects rejecting a request for water that is impossible to meet (completion time exceeds interval).

Computational Experiment 1

The first computational experiment is designed to test the quality of the solution from the heuristic for Model 1. For this experiment the number of jobs selected is 8 i.e. 8 water users. The tardiness factors considered are 0.9, 0.7, 0.5, 0.3 and 0.1. The target start time range factor is kept constant at 0.2.to reduce the number of variables in this experiment. Only those combinations of tardiness factors and target start time range factors yielding non-negative values of upper and lower bounds in (6) and (7) are considered. Cost of earliness/tardiness are generated randomly and, the cost of earliness is set equal to cost of tardiness. The parameters for generating random test instances are summarized in Table 1. For each set of parameters, 10 instances were generated.

TABLE 1: Parameters for Experiment 1

Parameter (1)	Values selected (2)
Number of jobs	8
Processing duration	Uniformly distributed random integers from the range [1,100]
Tardiness factor	0.9, 0.7, 0.5, 0.3, 0.1
Target start time range	0.2
Target start time	Uniformly distributed random integer $[(1-F-R/2)g, (1-F+R/2)g]$
Cost of earliness/ tardiness per unit of time	Uniformly distributed random integer from the range [0,5]
Irrigation interval	800
No. of instances	140 for each tardiness factor - total instances 700

Computational Experiment 2

Experiment 2 examines the effect of the interval to makespan ratio (IM ratio) on the quality of the solution obtained from the heuristic. Table 2 summarizes the parameters used to generate data for Experiment 2.

TABLE 2: Parameters for Experiment 2

Parameter (1)	Values selected (2)
Number of jobs	8
Processing duration	Uniformly distributed random integers from the range [1,100]
Tardiness factor	0.9, 0.7, 0.5, 0.3, 0.1
interval: makespan ratio	1.00, 1.25, 1.50, 1.75, 2.00, 2.25, 2.50
Target start time range	0.2

Target start time	Uniformly distributed random integer $[(1-F-R/2)g, (1-F+R/2)g]$
Cost of earliness/ tardiness per unit of time	Uniformly distributed random integer from the range [0,5]
No. of instances	20 for each tardiness factor - total instances 700

Computational Experiment 3

This experiment examines the effect of the number of jobs on the solution obtained from the heuristic. Although it is instructive to limit the target start time range, this arbitrarily restricts when a user might be able to wish to start irrigating. In practical terms a water user would be allowed to select a target start time at any time within the interval provided the target completion time did not exceed the interval. For Experiment 3 the target start time range is set to 1.0 and the tardiness factor is set at 0.5. Table 3 summarizes the parameters for this experiment.

TABLE 3: Parameters for Experiment 3

Parameter (1)	Values selected (2)
Number of jobs	8,10,12,15,20, 25
Processing duration	Uniformly distributed random integers from the range [1,100]
Tardiness factor	0.5
Target start time range	1.0
Target start time	Uniformly distributed random integer $[(1-F-R/2)g, (1-F+R/2)g]$
Cost of earliness/ tardiness per unit of time	Uniformly distributed random integer from the range [0,5]
Irrigation interval	No of jobs × 100
No. of instances	25 for each number of jobs - total instances 150

Computational Experiment 4

The fourth experiment examines the quality of solution obtained from the heuristic against that obtained from the integer programme for Models 2a, 2b and 2c. This reflects operation of a tertiary unit where every user starts using water directly after the previous user. The data used in Experiment 4 is reused for this experiment.

Results

Results from Experiment 1

Figure 2 compares the objective value obtained from the heuristic with that from the integer programme. In Figure 2a the tardiness factor is high. The mean error in the solution obtained from the heuristic relative to that from the integer programme is 4.9% over the 140 instances tested. The maximum error is 79%, and in 63 of the 140 (45%) instances tested, the objective value from heuristic and integer programme were identical.

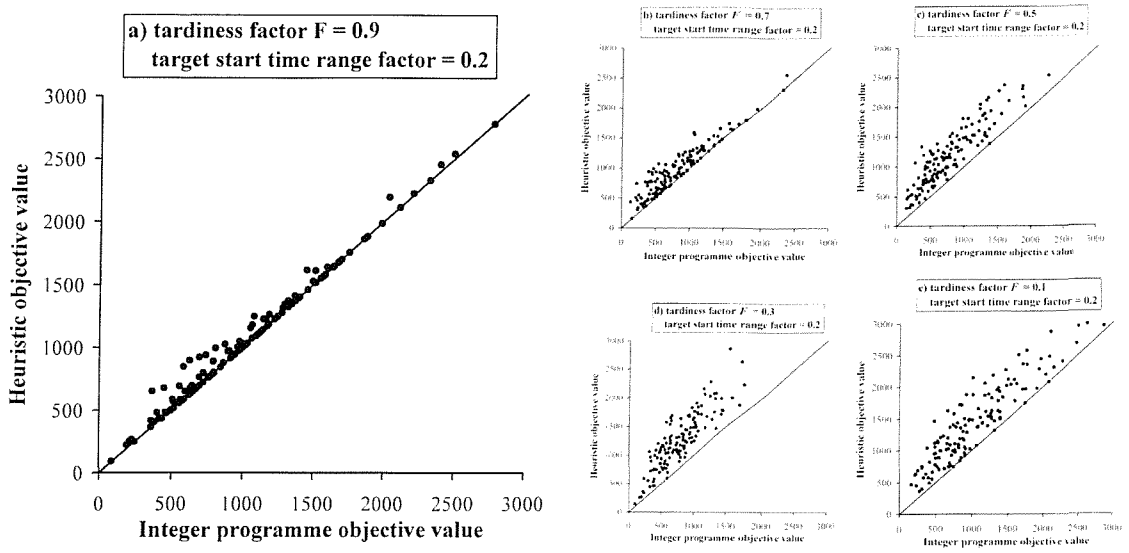


Figure 5 Solution quality of heuristic for various tardiness factors and due date range = 0.2

Figure 2b, 2c, 2d and 2e show that as the tardiness factor is decreases, the quality of the solution obtained by the heuristic is generally poorer. These results are summarized in Table 4 which shows with increasing tardiness factor, the error increases and the number of instances where the heuristic solves to the exact value (as obtained from the integer programme) decreases.

TABLE 4: Statistics for results from Experiment 1

Statistic	Tardiness factor				
	0.9	0.7	0.5	0.3	0.1
(1)	(2)	(3)	(4)	(5)	(6)
Mean error (%)	4.9	31.7	69.7	69.5	51.2
Minimum error (%)	0	0	0.47	2.41	0
Maximum error (%)	78.7	244.4	298.5	240.1	230.2
Standard deviation	0.11	0.41	0.53	0.42	0.45
No. of exact solutions	63	11	0	0	4

The heuristic is developed from a machine scheduling problem without a fixed interval, alternatively this can be expressed as a schedule with an infinite interval. If the target start times are early in the interval i.e. a high tardiness factor, all jobs will have a target start time near the beginning of the interval. This case approaches that of a machine scheduling problem with an infinite interval. As the tardiness factor decreases, the target start times of jobs move towards the end of the interval. The heuristic commits itself to a certain sequence based on the dispatching rules early in the algorithm which is followed by an interchange routine. Idle times are then inserted without reconsidering the sequence and the final schedule is produced. However the error in the solution from the heuristic clearly indicates that there is merit in re-examining the sequence of jobs. As the algorithm of the current heuristic does not perform such an analysis the solution quality deteriorates. The performance of the heuristic does appear to improve slightly at

the extreme value of tardiness factor tested. This anomaly merits further investigation.

Results from Experiment 2

Figure 3 shows the average error for various interval to makespan ratios (IM ratio).

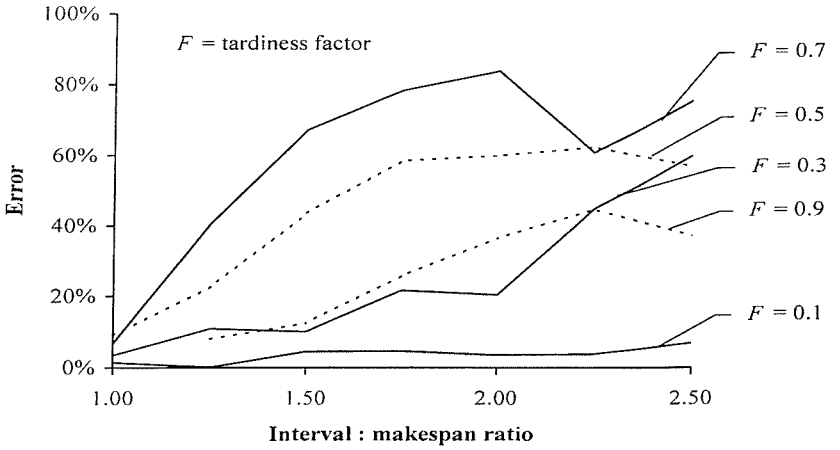


Figure 6 Solution quality of heuristic for various interval:makespan ratios

At low IM ratio values, the error is relatively small (of the order of 5%) irrespective of the tardiness factor. Furthermore, for a very low tardiness factor, the error remains relatively low across the entire range of the IM ratio. This appears to contradict Figure 1 which suggests that the best results are for the highest tardiness factor. This difficulty arises because in Experiment 1 the interval to makespan is a variable which, by virtue of the data generation, can not be controlled. In Experiment 2 the target start time range factor can not be controlled and hence is a variable. Between Experiments 1 and 2 it appears that the heuristic performs relatively better at extremes of tardiness factor.

Results from Experiment 3

Table 5 presents various statistics for Experiment 3.

TABLE 5: Statistics for results from Experiment 3

Model	Statistic	Number of jobs					
		8	10	12	15	20	25
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
Model 1	Mean error (%)	2320	4376	2077	2927	1765	1895
	Minimum error (%)	70	72	125	211	288	164
	Maximum error (%)	22000	43700	21231	47251	7494	7277
	Standard deviation	48.84	102.77	42.60	93.20	19.53	14.85
	No. of exact solutions	0	0	0	0	0	0
	Instances unsolved	0	0	0	0	2	9

The results do not show any definite trend in the quality of the solution obtained from the heuristic with the number of jobs. Comparing column (3) of Table 5 with column

(4) of Table 2, the error increases by many orders of magnitudes. This indicates that if the lower and upper bound factors are set at 0 and 1 respectively, the quality of the solution obtained from this heuristic is poor.

Figure 4 compares the solution time for the integer programme and the heuristic with number of jobs.

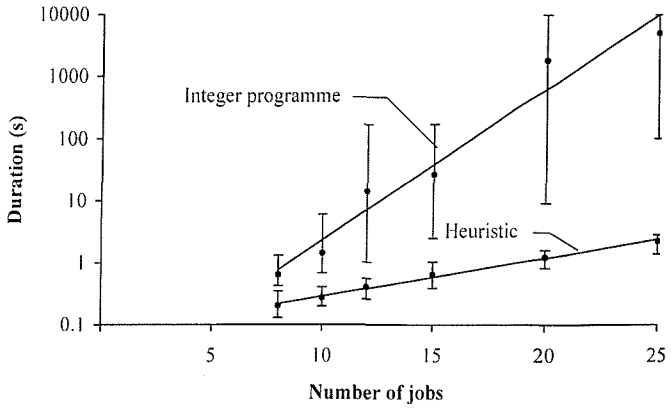


Figure 7 Solution time for heuristic and integer programme

The solution time for the integer programme not only increases by several order of magnitude but also the variation in solution time also increases considerably. The integer programme, did not solve within the allocated time of 10^4s for 2 of the 25 instances with 20 jobs. Similarly 9 of the 25 instances with 25 jobs did not reach a global optimum within the allocated time limit. In contrast the maximum solution time for the heuristic is only 2.82s.

Results from Experiment 4

Table 6 presents statistics for Experiment 4.

TABLE 6: Statistics for results from Experiment 4

Model	Statistic	Number of jobs					
		8	10	12	15	20	25
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
Model 2a	Mean error (%)	8	1.1	3.7	-7.7	-10.8	-12.1
	Minimum error (%)	0.0	0.0	7.50	-31.50	-36.0	-29.8
	Maximum error (%)	20.9	10.5	34.2	2.3	8.1	-0.4
	Standard deviation	4.2	2.6	9.9	8.2	11.6	7.5
	No. of exact solutions	19	18	n/a	n/a	n/a	n/a
	Instances unsolved	0	0	8	24	24	25
Model 2b	Mean error (%)	0.4	1.4	-3.1	-25.2	-30.1	-33.0
	Minimum error (%)	0	0	-40.9	-54.9	-53.0	-51.0
	Maximum error (%)	2.6	6.2	13.9	2.4	-4.1	-14.3
	Standard deviation	0.7	2.2	11.3	15.2	11.3	10.0
	No. of exact solutions	17	14	n/a	n/a	n/a	n/a

	Instances unsolved	0	0	13	25	25	25
Model 2c	Mean error (%)	155.5	285.2	258.4	243.5	164.1	135.3
	Minimum error (%)	42.0	80.2	85.6	129.0	114.5	45.6
	Maximum error (%)	352.9	525.7	892.3	730.2	249.6	271.0
	Standard deviation	131.8	192.4	288.0	216.0	45.8	73.6
	No. of exact solutions	0	0	0	0	0	0
	Instances unsolved	0	0	0	12	25	0

n/a: not applicable

For Model 2a the quality of the solutions obtained from the heuristic are a significant improvement over Model 1 (Table 5). For the 8 jobs problem, the solution obtained from the heuristic matched that obtained from the integer programme in 19 of the 25 instances tested. For the 10 jobs problem, the number of exact matches was 18. For the 12 jobs problem, the integer programme failed to solve within the time limit for 8 of the 25 instances tested. Coincidentally for every instance where the integer programme failed to solve, the heuristic was able to find a better solution than the integer programme. For such circumstances, the count of exact solutions is no longer applicable as a measure of solution quality from the heuristic. For the 15 jobs problem all but one instance failed to solve within the allocated time using the integer programme. For 24 of the 25 instances tested, the heuristic was able to obtain a better solution than the integer programme, hence the negative mean error in column (6). It is worth noting that the integer programme would have obtained a solution equal to or better than that from the heuristic if it were allowed to continue running beyond the allocated time. For the 20 and 25 jobs problem, all instances failed, within the time allocated, to reach a global optimum using the integer programme. Interestingly, the increased number of instances failing to reach a global optimum using the integer programme for Model 2a (as compared to Model 1) indicates that Model 2a is computationally more demanding than Model 1. The solution time using the heuristic for Model 2a is of the same order as that for Model 1a.

The results for Model 2b are similar to that for Model 2a. This is to be expected since Model 2b is in a sense an inversion of Model 2a. As with Model 2a the solution obtained from the heuristic is of reasonable quality even for problems with a small number of jobs. For the 8 jobs problem the heuristic was able to obtain the exact solution for 17 of the 25 instances tested. For the 10 jobs problem in 14 of the 25 instances, the heuristic was able to obtain the exact solution. For the 12 jobs problem and larger, the integer programme was not able to obtain an optimum solution in the allocated time and therefore, the heuristic was able to obtain a better solution than the integer programme. The solution time for the heuristic for Model 2b is of a similar order of magnitude as for Model 2a and a very small fraction of the solution time for the integer programme for Model 2b.

For Model 2c the heuristic no longer performs well, although not quite as poorly as Model 1. This is to be expected since in terms of complexity of the solution, Model 2c lies between Model 1 as that with the most complex solution, and Model 2a and 2b as those with the simplest solution. For Model 2c the heuristic was unable to obtain a better solution than the integer programme even where the integer programme did not solve to a global optimum .

Practical Application

The heuristic developed in this paper is applied to a tertiary unit taken from Bishop and Long (1983) where a schedule was developed for a tertiary unit of 37.06 ha with 16 water users. In the original schedule developed by Bishop and Long (1983) water is allocated pro rata with area at 172 min/ha, and a management/canal fill time of 349 min. Bishop and Long (1983) advocated a simple "irrigate fields starting from the downstream end upwards".

For the purpose of this example, the interval is assumed to have a duration of 1 week starting from Monday 0800hrs. The duration of each job is estimated using the allocation of 172 min/ha as suggested by Bishop and Long (1983). The management/ canal fill time (349 min) is divided equally amongst all users to account for setup time (time required when one user completes irrigation and the second one is setting up irrigating) and is added to the duration. Target start times are uniformly generated random numbers over the irrigation interval (Monday 08:00hrs to Monday 07:59hrs) with tardiness factor set at 0.50 and start time range factor set at 1.0. The actual tardiness factor for the data estimated from (7) is 0.57, and the actual start time range factor for the data from (8) is 0.97.

Table 7 presents the value of the objective function when the heuristic for each model is applied.

TABLE 7 Objective value and error for Practical Application

	Model 1	Model 2		
		2a	2b	2c
(1)	(2)	(3)	(4)	(5)
Heuristic	15, 830	19,987	33,821	25,635
Optimum value (IP)	3,822	19,667	31,952	19,001
Error (%)	314.2	1.6	5.9	34.9

Again for Models 2a and 2b the heuristics obtains reasonable results, verifying the results from the extensive computational testing. The heuristic solved in around 6 seconds as compared to approximately 3:00 hrs required by the integer programme. Therefore the schedule obtained for either of these models could be used. Table 8 presents the schedule for the first irrigation interval from the heuristic for Model 2a.

TABLE 8: Detailed schedule for irrigation interval from Model 2a

Lot	Area (ha)	Duration (hrs:min)	Target start	Scheduled start
(1)	(2)	(3)	(4)	(5)
24.1	1.05	8:57	Mo 08:19	Mo 08:00
25.1	3.00	8:57	Mo 20:51	Mo 16:57
22.1	3.20	9:32	Tu 00:30	Tu 01:54
23.2	1.00	3:13	Tu 09:03	Tu 11:26
24.2	3.00	8:57	Tu 06:45	Tu 14:39
26.3	2.56	7:41	Tu 19:40	Tu 22:36
22.2	3.20	9:32	Tu 04:06	We 07:17
24.4	1.05	3.22	We 16:02	We 16:49

25.2	3.00	8:57	Th 16:41	We 20:11
26.1	3.00	8:57	Sa 21:29	Th 05:08
25.3	2.13	6:27	Su 12:09	Th 14:05
23.4	1.97	6:00	Fr 02:08	Th 20:32
24.3	1.00	3:13	Fr 05:55	Fr 02:32
23.3	1.00	3:13	Fr 06:59	Fr 05:45
23.1	2.50	7:31	Su 22:35	Fr 08:58
26.2	2.50	7:31	Mo 03:03	Fr 16:29

For subsequent intervals, the heuristics developed can be used as a multi interval model in which case the cost of earliness/tardiness per unit of time for job for the second and subsequent periods can be calculated using the method outlined in the companion paper.

Conclusion

Heuristics present a computationally efficient method of solving scheduling problems irrespective of the number of jobs. The sequential irrigation scheduling problem is described in operations research literature as a single machine E/T (earliness/tardiness) model with deadlines, or a single machine E/T problem with deadlines. For the sequential irrigation scheduling problem if jobs are to be scheduled contiguously with idle time inserted as a contiguous block either only before or only after all jobs, then the quality of the solution obtained by the heuristics presented is very high. If jobs are scheduled non-contiguously, or if idle time is permitted to be inserted both before and after the contiguous jobs, the solution quality of the heuristic deteriorates and better heuristics need to be developed. The heuristic presented does not consider sequence dependent setup time independent of job duration. This assumption is reasonable if the setup times are small relative to job duration. However if job duration is large or varies, then sequent dependent setup times needs to be considered.

This set of companion papers draws in some of the wealth available in Operations Research on scheduling and shows how it can be applied to irrigation scheduling at an operational level. Exact solutions are developed but not recommended for large problems due to excessive computation time. The exact solutions do present useful benchmarks for the approximate but computationally efficient procedures presented. It is anticipated that these papers will provide a starting point for further development in this area of irrigation scheduling.

Perhaps beyond the more technological issues discussed in these papers is the broader issue; Should irrigation managers be providing a service i.e. water to their customers? If so then like other service industries - potable water, telecommunications, electricity etc. the customers should specify their demand (duration), and when they require the service.

The irrigation manager should provide this service. Many irrigation systems would not have the capacity to operate on an absolute demand based system - hence the alternative is a scheduled demand system - prepare a schedule that manages demand within the capacity constraints of the system. This set of companion papers presents tools for preparing such schedules.

Appendix I. References

Beasley J.E. (1996) "Obtaining Test Problems via Internet" *J. Global Optimization*, Kluwer Academic, 8, 429-433.

Bishop A.A., Long A.K. (1983) "Irrigation Water Delivery for Equity between Users." *Journal of Irrigation and Drainage Engineering*, ASCE 109(4):349-356.

Kim Y. and Yano C.A. (1994) "Minimizing mean tardiness and earliness in single-machine scheduling problems with unequal due dates." *Naval Research Logisitics*, John Wiley, 41:913-44.

Lee Y.H., Bhaskaran K., Pinedo M. (1997) "A Heuristic to Minimize the Total Weighted Sequence-dependent Setups." *IIE Transactions*, IIE 29: 45-52.

Mazzini R. and Armentano V.A. (2001) "A Heuristic for Single Machine Scheduling with Early and Tardy Costs." *European Journal of Operations Research*, Elsevier, 128:129-146.

Nawaz M., Enscore E.E. Jr. and Ham I (1983) " A Heuristic Algorithm for the m-Machine, n-Job Flow-shop Sequencing Problem" *OMEGA The International Journal of Management Science*, Pergamon, 11(1):91-95.

Ow P.S. and Morton (1989) "The single machine early/tardy problem" *Management Science,* INFORMS, 35(2): 177-191

Potts, C.N., and Van Wassenhove, L.N. (1982) "A Decomposition Algorithm for the Single Machine Total Tardiness Problem," *Operations Research Letters,* 1(5):177-81.

Rardin R.L., Uzsoy R. (2001) "Experimental Evaluation of Heuristic Optimization Algorithms: A Tutorial." *Journal of Heuristics*, Kluwer Academic, 7:261-304.

Appendix II. Notation

the following notation was used in this paper

B_L = lower bound for target start time;

B_U = upper bound factor for target start time;

d_i = duration of job i;

d_{i-1} = duration of job i-1;

$d_{n_{b-1}}$ = duration of the last job in block b-1;

d_{n_b} = duration of the last job in the last block;

d_{n_t} = duration of the last job in block b=t;

$E_{\min_b}$ = minimum positive earliness of all jobs in block b;

$E_{\min_t}$ = minimum positive earliness of all jobs in the last block;

F = tardiness factor;

g = interval;

i = index representing index of a job;

N = number of jobs.

n_b = number of jobs in block b ;

S_{1_1} = scheduled start time of the first job in the first block;

S_{1_b} = scheduled start time of the first job in block b;

$S_{1_{b+1}}$ = scheduled start time of the first job in block $b+1$;

S_i = scheduled start time of job i;

S_{i-1} = scheduled start time of job i-1,

S_{n_b} = scheduled start time of last job in block b;

$s_{n_{b-1}}$ = scheduled start time of last job in block *b-1*;

S_{n_t} = scheduled start time of last job in the last block;

R = target start time range factor;

$\bar{r}$ = average target start time;

r_{IM} = interval to makespan ratio;

$r_{\max}$ = maximum target start time;

$r_{\min}$ = minimum target start time;

$T_{\min_1}$ = minimum positive tardiness of all jobs in the first block;

$T_{\min_b}$ = minimum positive tardiness of all jobs in the last block;

t = total number of blocks in the ordered set of blocks;

γ_{1L} = block displacement left for the first block;

γ_{bL} = block displacement left;

γ_{bR} = block right displacement; and,

γ_{tR} = block right displacement for the last block.

Appendix D: setup times in irrigation scheduling

Setup times capture the notion that when one job is completed a certain amount of setup time may be required before the next job can start. In irrigation this setup time is the time needed for water to be diverted from one outlet to another.

Baker (1974), amongst others, stated that if setup times are independent of the order in which jobs are processed, they can be included in the duration of the jobs. Figure D.1 shows that because in irrigation scheduling target start times are used rather than due dates, setup times cannot be included in the duration.

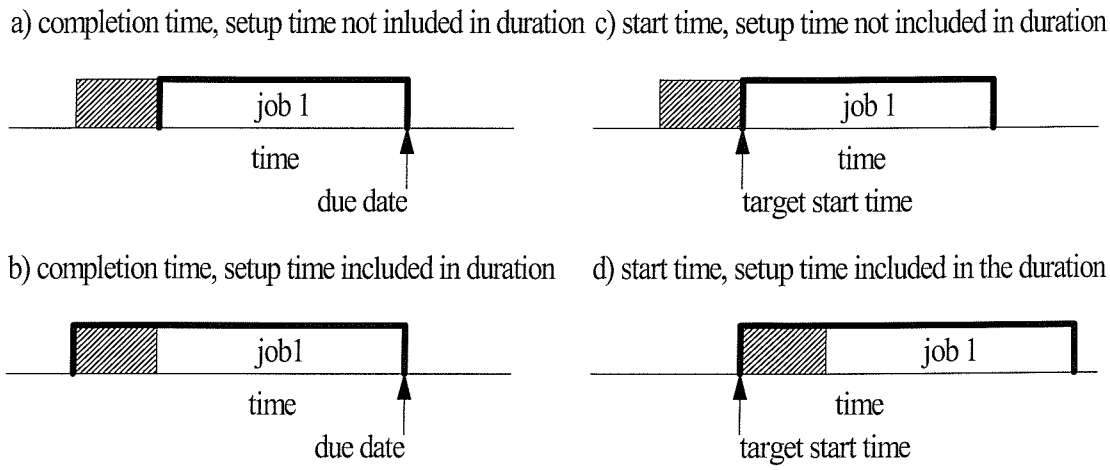


Figure D.1 Comparison completion time and start time

Figure D.1a shows a job that needs to be completed by a certain time, the due date. The setup time for this job is not included in the duration. The job is scheduled to finish exactly on time. Figure D.1b shows the same job, but now the setup time is included in the duration. The inclusion of the setup time has caused the job to start earlier but the job is still scheduled to finish on time. This shows that if completion times are considered setup times can indeed safely be included in the duration. Figure D.1c shows that a job needs to start on a certain time, the target start time. The setup time is not included in the duration. The job is scheduled to start exactly on time. Figure D.1d shows the same job, but now the setup time is included in the duration. It can be seen that the job is scheduled to start on time, but this is the time the setup time starts, not the time the actual job starts. This means that when start times are considered setup times cannot be included in the duration and need to be considered separately.

In the example used in Chapter 3, setup times are included in the irrigation duration. The example above shows that this is not entirely correct. However, as explained the example only serves to demonstrate the principles of Models 1, 2a, 2b and 2c. It is believed that the fact that in the example the start of the travel time is optimised rather than the start of the irrigation duration, does not make a difference to those principles. However to be able to compare Model 1 and Model 3, Model 1 needs to be adjusted slightly so that the start of the irrigation duration is optimised. This adjustment follows below.

The non-contiguous single machine model allows scheduling of a number of jobs (user requests) according to their target start times. Let $\boldsymbol{Q} = \{1,2,...,N\}$ be a set of jobs to be scheduled. The following parameters are specified for each job $i \in \boldsymbol{Q}$: duration, target start time, earliness cost and tardiness cost. In this model there is one decision to be

made, what is the scheduled start time of each job. If the answer to this question is known, then the schedule is known too. The objective of the model is to minimize the difference between target start time and scheduled start time. Therefore

$$Z = \min \sum_{i=1}^{N} (\alpha_i E_i + \beta_i T_i) \tag{D.1}$$

As any job can precede any other job in the schedule a variable is used to define which job precedes which

$$\begin{aligned} \delta_{ij} &= 1 \quad \text{if job } i \text{ precedes job } j \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{D.2}$$

It is not possible for a job to incur a negative earliness or tardiness. Therefore for T_i and E_i the following constraints need to be satisfied

$$T_i \geq 0 \qquad \forall i = 1,2,...,N \tag{D.3}$$

$$E_i \geq 0 \qquad \forall i = 1,2,...,N \tag{D.4}$$

The scheduled start time of a job is determined from the target start time, the earliness and the tardiness

$$S_i = r_i - E_i + T_i \qquad \forall i = 1,2,...,N \tag{D.5}$$

Jobs may not be processed simultaneously, therefore

$$S_j - S_i + M(1 - \delta_{ij}) \geq d_i + t_j \qquad \forall i = 1,2,...,N;\ \forall j = 1,2,...,N;\ i \neq j \tag{D.6}$$

and

$$S_i - S_j + M\delta_{ij} \geq d_j + t_i \qquad \forall i = 1,2,...,N;\ \forall j = 1,2,...,N;\ i \neq j \tag{D.7}$$

All jobs must be processed within the irrigation interval.

$$S_i + d_i \leq g \qquad \forall i = 1,2,...,N \tag{D.8}$$

Therefore the adjusted version of Model 1 is defined by the objective function (D.1) and constraints (D.2), (D.3), (D.4), (D.5), (D.6), (D.7) and (D.8).

3.2 Analysis and development of contiguous scheduling model

The contiguous scheduling model, herein referred to as Model 2, is similar to Model 1 in that it allows scheduling of a number of jobs according to their target start times. The main difference is that to reduce water spillage and/or gate operations idle time between jobs is not allowed. The objective function and most of the constraints of Model 1 remain valid.

3.2.1 Contiguous single machine scheduling: Model 2a

To ensure idle time is only inserted after the last job has been processed, no job may finish later than the sum of all durations. Therefore constraint (3.11) of Model 1 must be replaced by

$$S_i + d_i \leq \sum_{i=1}^{N} d_i \qquad \forall i = 1,2,...,N \tag{D.9}$$

Model 2a is therefore defined by the objective function (3.1) and constraints (3.2), (3.3), (3.4), (3.6), (3.9), (3.10) and (3.12).

3.2.2 Contiguous single machine scheduling: Model 2b

In Model 2b all job are scheduled contiguously and idle time is inserted prior to the start of the first job. Constraint (3.11) becomes

$$S_i \geq g - \sum_{i=1}^{N} d_i \qquad \forall i = 1,2,...,N \tag{D.10}$$

Model 2b is defined by the objective function (3.1) and constraints (3.2), (3.3), (3.4), (3.6), (3.9), (3.10) and (3.13).

3.2.3 Contiguous single machine scheduling: Model 2c

In Model 2c idle time may precede the start of the first job and/or follow the end of the last job, therefore (3.11) is altered to

$$g = \sum_{i=1}^{N} d_i + X_a + X_b \tag{D.11}$$

where X_a = idle time preceding the start of the first job in the schedule; and X_b = idle time following the completion of the last job in the schedule. The following additional constraints also need to be included for Model 2c

$$S_i + d_i \leq g - X_b \qquad \forall i = 1,2,...,N \tag{D.12}$$

$$S_i \geq X_a \qquad \forall i = 1,2,...,N \tag{D.13}$$

Model 2c is therefore defined by the objective function (3.1) and constraints (3.2), (3.3), (3.4), (3.6), (3.9), (3.10), (3.14), (3.15) and (3.16).

References

Akhand, N. A., Larson, D. L., and Slack, D. C. (1993). "An irrigation allocation planning model." *Management of Irrigation and Drainage Systems, Integrated Perspectives*, July 21-23, 1993, Park City, Utah, ASCE, 842-849.

Alidaee, B., Kochenberger, G. A., and Ahmadian, A. (1994). "0-1 Quadratic-programming approach for optimum solutions of 2 scheduling problems." *International Journal of Systems Science*, Taylor and Francis, 25(2), 401-408.

Allahverdi, A., Gupta, J. N. D., and Aldowaisan, T. (1999). "A review of scheduling research involving setup considerations." *Omega-International Journal of Management Science*, Elsevier, 27(2), 219-239.

Anwar, A. A., and Clarke, D. (2001). "Irrigation scheduling using mixed-integer linear programming." *Journal of Irrigation and Drainage Engineering*, ASCE, 127(2), 63-69.

Arkin, E. M., and Roundy, R. O. (1991). "Weighted-tardiness scheduling on parallel machines with proportional weights." *Operations Research*, Institute for Operations Research and the Management Sciences, 39(1), 64-81.

Baker, K. R. (1974). *Introduction to sequencing and scheduling*, Wiley and Sons, New York, USA.

Baker, K. R., and Scudder, G. D. (1990). "Sequencing with earliness and tardiness penalties - a review." *Operations Research*, 38(1), 22-36.

Balakrishnan, N., Kanet, J. J., and Sridharan, V. (1999). "Early/tardy scheduling with sequence dependent setups on uniform parallel machines." *Computers and Operations Research*, Elsevier, 26(2), 127-141.

Bard, J. F., Venkatraman, K., and Feo, T. A. (1993). "Single-machine scheduling with flow time and earliness penalties." *Journal of Global Optimization*, Kluwer, 3(3), 289-309.

Bianco, L., Ricciardelli, S., Rinaldi, G., and Sassano, A. (1988). "Scheduling tasks with sequence-dependent processing times." *Naval Research Logistics*, Wiley, 35(2), 177-184.

Bishop, A. A. and Long, A. K. (1983). "Irrigation water delivery for equity between users." *Journal of Irrigation and Drainage Engineering*, ASCE, 109(4), 349-356.

Burt, C. M. (2000). "Irrigation district modernization in the U.S. and worldwide - the necessary link for efficient on-farm irrigation." *4th National Decennial Irrigation Symposium*, Phoenix, Arizona, ASAE, 428-434.

Burt, C. M., and Styles, S. W. (2000). "Irrigation district service in the western United States." *Journal of Irrigation and Drainage Engineering*, ASCE, 126(5), 279-282.

Clemmens, A. J. (1987). "Arranged delivery schedules." *Planning, Operation, Rehabilitation and Automation of Irrigation Water Delivery Systems*, Proceedings symposium, ASCE Irrigation and Drainage Division Specialty Conference, ASCE, 57-67.

Clyma, W., and Reddy, J. M. (2000). "Optimal design and management of surface irrigation systems." *4th National Decennial Irrigation Symposium*, November 14-16, 2000, Phoenix, Arizona, ASAE, 298-303.

Crama, Y., and Spieksma, F. C. R. (1996). "Scheduling jobs of equal length: Complexity, facets and computational results." *Mathematical Programming*, The Mathematical Programming Society, 72(3), 207-227.

Cross, P. R. (2000). "Benefits of flexible irrigation water supply." *Journal of Irrigation and Drainage Engineering*, ASCE, 126(5), 275-278.

Davis, L. (1991). *A handbook of genetic algorithms*, Van Nostrad Reinhold, New York.
De Bièvre, B., Alvarado, A., Timbe, L., Célleri, R. and Feyen, J. (2003). "Night irrigation reduction for water saving in medium-sized systems." *Journal of Irrigation and Drainage Engineering*, ASCE, 129(2), 108-116.
Della Croce, F., Szwarc, W., Tadei, R., Baracco, P., and Di Tullio, R. (1995). "Minimizing the weighted sum of quadratic completion times on a single-machine." *Naval Research Logistics*, Wiley, 42(8), 1263-1270.
Dudley, N. J., Howell, D. T., and Musgrave, W. F. (1971). "Optimal intraseasonal irrigation water allocation." *Water Resources Research*, American Geophysical Union, 7(4), 770-788.
Fry, T.D. and Leong, G.K. (1987). "A bi-criterion approach to minimizing inventory costs on a single machine when early shipments are forbidden." *Computers & Operations Research*, Pergamon, 14(5), 363-368.
Hannan, T. C., and Coals, V. A. (1995). "Real-time water allocation for irrigation." *Journal of the Institution of Water and Environmental Management*, Institution of Water and Environmental Management, 9(1), 19-26.
Holland, J. H. (1975). *Adaption in natural and artificial systems*, MIT Press, Ann Arbor.
Holzapfel, E. A., Marino, M. A., and Chavezmorales, J. (1986). "Surface irrigation optimization models." *Journal of Irrigation and Drainage Engineering*, ASCE, 112(1), 1-19.
James, R. J. W., and Buchanan, J. T. (1998). "Robustness of single machine scheduling problems to earliness and tardiness penalty errors." *Annals of Operations Research*, 76, 219-232.
Javan, M., Sanaee-Jahromi, S. and Fiuzat, A.A. (2002). "Quantifying management of irrigation and drainage systems", *Journal of Irrigation and Drainage Engineering*, ASCE, 128(1), 19-25.
Khepar, S. D., Gulati, H. S., Yadav, A. K., and Brar, T. P. S. (2000). "A model for equitable distribution of canal water." *Irrigation Science*, Springer Verlag, 19(4), 191-197.
Kipkorir, E. C., Raes, D., and Labadie, J. (2001). "Optimal allocation of short-term irrigation supply." *Irrigation and Drainage Systems*, Kluwer, 15, 247-267.
Kumar, C. N., Indrasenan, N., and Elango, K. (1998). "Nonlinear programming model for extensive irrigation." *Journal of Irrigation and Drainage Engineering*, ASCE, 124(2), 123-126.
Kuo, S. F., Merkley, G. P., and Liu, C. W. (2000). "Decision support for irrigation project planning using a genetic algorithm." *Agricultural Water Management*, Elsevier, 45(3), 243-266.
Latif, M. and Sarwar, S. (1994). "Proposal for equitable water allocation for rotational irrigation in Pakistan." *Irrigation and Drainage Systems*, Kluwer, 8(1), 35-48.
Lembke, W. D., and Jones, B. A. (1972). "Selecting a method for scheduling irrigation, using a simulation model." *Transactions of the Asae*, ASAE, 15(2), 284-286.
Li, G. (1997). "Single machine earliness and tardiness scheduling." *European Journal of Operational Research*, Elsevier, 96(3), 546-558.
Liaw, C. F. (1999). "A branch-and-bound algorithm for the single machine earliness and tardiness scheduling problem." *Computers & Operations Research*, 26(7), 679-693.
Malek-Mohammadi, E. (1998). "Irrigation planning: integrated approach." *Journal of Water Resources Planning and Management*, ASCE, 124(5), 272-279.
McCornick, P. G. (1993). "Water management in arranged-demand canal." *Journal of*

Irrigation and Drainage Engineering, ASCE, 119(2), 251-264.
Merriam J.L. (1987). "Symposium introduction." *Planning, Operation, Rehabilitation and Automation of Irrigation Water Delivery systems*, Proceedings symposium, ASCE Irrigation and Drainage Division Specialty Conference Portland, Oregon, USA, ASCE, 1-17.
Mondal, S. A., and Sen, A. K. (2000). "An improved precedence rule for single machine sequencing problems with quadratic penalty." *European Journal of Operational Research*, Elsevier, 125(2), 425-428.
Muspratt, M. A. (1971). "Optimal distribution of water to irrigation canals." *Journal of Hydrolgy*, North-Holland Publishing Co., 14(1), 19-28.
Naadimuthu, G., Raju, K. S., and Lee, E. S. (1999). "A heuristic dynamic optimization algorithm for irrigation scheduling." *Mathematical and Computer Modelling*, Elsevier, 30(7-8), 169-183.
Naresh, R., and Sharma, J. (2000). "Hydro system scheduling using ANN approach." *IEEE Transactions on Power Systems*, IEEE, 15(1), 388-395.
Onta, P. R., Das Gupta, A., and Paudyal, G. N. (1991). "Integrated irrigation development planning by multi-objective optimization." *Water Resources Development*, Butterworth-Heinemann, 7(3), 185-193.
Ow, P. S., and Morton, T. E. (1989). "The single machine early/tardy problem." *Management Science*, Institute of Management Sciences, 35(2), 177-191.
Paudyal, G. N., and Das Gupta, A. (1990). "Irrigation planning by multilevel optimization." *Journal of Irrigation and Drainage Engineering*, ASCE, 116(2), 273-291.
Paul, S., Panda, S. N., and Kumar, D. N. (2000). "Optimal irrigation allocation: A multilevel approach." *Journal of Irrigation and Drainage Engineering*, ASCE, 126(3), 149-156.
Palmer, J.D., Clemmens, A.J. and Dedrick, A.R. (1991). "Field study on irrigation delivery performance", *Journal of Irrigation and Drainage Engineering*, ASCE, 117(4), 567-577.
Perry, C. (2001). "Water at any price? Issues and options in charging for irrigation water." *Irrigation and Drainage*, Wiley, 50(1), 1-7.
Petriczek, G., and Uhrynowski, Z. (1986). "On the use of multilevel dynamic optimization in water management systems." *Systems Science*, 12(3), 53-68.
Pinedo, M. (1995). *Scheduling: theory, algorithms and systems*, Prentice-Hall, Englewood Cliffs, New Jersey, USA.
Pleban, S., Labadie, J. W., and Heermann, D. F. (1983). "Optimal short-term irrigation schedules." *Transactions of the ASAE*, ASAE, 26(1), 141-147.
Plusquellec, H., Burt, C., and Wolter, H. W. (1994). *Modern water control in irrigation, concepts, issues and applications*. World Bank technical paper 245, Irrigation and drainage series, International Bank for Reconstruction/World Bank.
Ponnambalam, K., and Adams, B. J. (1987). "Experiences with integrated irrigation systems optimization analysis." *Irrigation and Water Allocation*, Proceedings symposium, XIX Assembly of the International Union of Geodesy and Geophysics August 1987, Vancouver, Canada, IAHS, 229-244.
Rardin, R. L., and Uzsoy, R. (2001). "Experimental evaluation of heuristic optimization algorithms: A tutorial." *Journal of Heuristics*, 7(3), 261-304.
Rathod, K. G., and Prajapati, N. M. (1998). "Rotational water supply system." *International Water Resources Engineering Conference*, proceedings vol. 2, 1078-1083.

Replogle, J. A. (1987). "Irrigation water management with rotation scheduling policies." *Planning, operation, rehabilitation and automation of irrigation water delivery systems*, Proceedings symposium, ASCE Irrigation and Drainage Division Specialty Conference, ASCE, 35-44.

Rose, C. J. (1973). "Management science in the developing countries: a comparitive approach to irrigation feasibility." *Management Science*, The Institute of Management Sciences, 20(4), 423-438.

Santhi, C. and Pundarikanthan, N.V. (2000). "A new planning model for canal scheduling of rotational irrigation." *Agricultural Water Management*, Elsevier, 43(3), 327-343.

Sarwar, A., Bastiaansen, W. G. M., and Feddes, R. A. (2001). "Irrigation water distribution and long-term effects on crop and environment." *Agricultural Water Management*, Elsevier, 50(2), 125-140.

Schrage, L. (1999). *Optimization modelling with Lingo*, Lindo Sytems, Chicago, USA.

Sharma, D. N., and Oad, R. (1990). "Variable-time model for equitable irrigation water distribution." *Agricultural Water Management*, 17(4), 367-377.

Simpson, J., and Weiner, E. (1989). *Oxford English dictionary*, Oxford University Press.

Skogerboe, G. V. (2000). "Lessons from the American West." *4th National Decennial Irrigation Symposium*, November 14-16, 2000, Phoenix, Arizona, ASAE, 176-181.

Smith, W. E. (1956). "Various optimizers for single-stage production." *Naval Research Logistics Quarterly*, Wiley, 3, 59-66.

Soric, K. (2000). "A cutting plane algorithm for a single machine scheduling problem." *European Journal of Operational Research*, Elsevier, 127(2), 383-393.

Sun, J. U., and Hwang, H. (2001). "Scheduling problem in a two-machine flow line with the N-step prior-job-dependent set-up times." *International Journal of Systems Science*, Taylor and Francis, 32(3), 375-385.

Suryavanshi, A. R., and Reddy, J. M. (1986). "Optimal operation schedule of irrigation distribution-systems." *Agricultural Water Management*, Elsevier, 11(1), 23-30.

Townsend, W. (1978). "The single machine problem with quadratic penalty function of completion times: a branch-and-bound solution." *Management Science*, The Institute of Management Sciences, 24(5), 530-534.

Upcraft, M. J., Noble, D. H., and Carr, M. K. V. (1996). "An heuristic algorithm for short term irrigation scheduling using hosereel-raingun systems." *Irrigation Science*, Springer-Verlag, 16(4), 141-147.

Wang, Z., Reddy, J. M., and Feyen, J. (1995). "Improved 0-1 programming model for optimal flow scheduling in irrigation canals." *Irrigation and Drainage Systems*, Kluwer, 9, 105-116.

Wardlaw, R. (1999). "Computer optimisation for better water allocation." *Agricultural Water Management*, Elsevier, 40(1), 65-70.

Wardlaw, R., and Barnes, J. (1999). "Optimal allocation of irrigation water supplies in real time." *Journal of Irrigation and Drainage Engineering*, ASCE, 125(6), 345-354.

Winston, W. L. (1994). *Operations research, applications and algorithms*, Duxbury Press, Belmont, California,USA.