European Journal of Operational Research 139 (2002) 230–244

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/dsw

Production, Manufacturing and Logistics

Using real time information for effective dynamic scheduling

Peter Cowling [a,*], Marcus Johansson [b]

[a] Department of Computing, University of Bradford, Bradford, West Yorkshire, BD7 1DB, UK
[b] Division of Manufacturing Engineering and Operations Management, University of Nottingham, Nottingham NG7 2RD, UK

Received 28 February 1999; accepted 02 May 2001

Abstract

In many production processes real time information may be obtained from process control computers and other monitoring systems, but most existing scheduling models are unable to use this information to effectively influence scheduling decisions in real time. In this paper we develop a general framework for using real time information to improve scheduling decisions, which allows us to trade off the quality of the revised schedule against the production disturbance which results from changing the planned schedule. We illustrate how our framework can be used to select a strategy for using real time information for a single machine scheduling model and discuss how it may be used to incorporate real time information into scheduling the complex production processes of steel continuous caster planning. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Scheduling; Production; Rescheduling; Real time information

1. Introduction

Although scheduling is a well researched area, and numerous articles and books have been published, classical scheduling theory has been little used in real production environments (Stoop and Wiers, 1996). We believe that scheduling research has much to offer industry and commerce, but that more work is needed to address the "gap" between scheduling theory and practice (MacCarthy and Liu, 1993). One frequent assumption of scheduling theory, which rarely holds in practice, is that the scheduling environment is static. In many production and service systems, schedules must be revised frequently in response to both instantaneous events, which occur without warning, and anticipated events where information is given in advance by, for example, process control computers or customers. In this paper we develop a framework for handling real time information concerning anticipated future events. Note that in this case the time of arrival of the information is important in deciding the best schedule revision strategy to adopt.

Many manufacturing environments use an material requirements planning (MRP), manufacturing resources planning (MRPII) or enterprise resources planning (ERP) system for medium term planning. Such a system divides the planning

* Corresponding author.
E-mail addresses: peter.cowling@scm.brad.ac.uk (P. Cowling), marcus.b.johansson@uk.andersen.com (M. Johansson).

horizon into discrete time buckets and requires a medium term production plan for several future time buckets, which is used to provide due dates and release dates for detailed production scheduling. We gain the advantage of being able to divide a complex medium term capacity planning problem into smaller and more manageable pieces at the cost of high rigidity in the production plan. The question as to how a scheduler should respond to changes in a dynamic system in this environment is a fruitful area for research. Between schedule creation and execution one or several assumptions may have changed concerning, for example, machine availability or material supply. The obvious question is how to react to this type of information and how different scheduling strategies affect the original plan or sequence of jobs? To answer this question one needs to consider effects on both upstream and downstream operations as well as the effects on longer term plans. This has previously been difficult, since there has been a lack of real time information concerning system status. However, in many current production and service systems a great deal of real time information is captured for control and monitoring purposes. In deciding how to react we must consider not only the quality of the revised schedule but also the disruption caused by schedule revision. Our framework will consider the trade off between these two factors.

Scheduling research has failed to keep pace with technological developments in process control and monitoring systems. In this paper we present a framework for effectively incorporating real time information produced by process control and monitoring systems into scheduling models and in so doing to address an important aspect of the "gap" between scheduling theory and practice.

In Section 2 we consider the types of real time information encountered in practice and survey the literature on dynamic scheduling. Here we present our notions of schedule repair and rescheduling. In Section 3 we present two measures for determining the value of real time information, *utility* which measures the improvement in our original scheduling objectives due to schedule revision and *stability* which measures the disruption caused by schedule revision. In Sections 4 and 5 we apply our notions of utility and stability to the simple $n/1//\overline{C}$ single machine scheduling model. Section 4 investigates in detail the response of the model to a single piece of real time information. In Section 5 we carry out a simulation study to investigate the behaviour of several strategies for dealing with multiple pieces of real time information. In Section 6 we discuss how our ideas may be applied to the complex scheduling environment of the steel continuous caster, where a great deal of information is produced by process control computers. Conclusions are presented in Section 7.

2. Using real time data

Historically, one of the major reasons for uncertainty in real scheduling environments has been the lack of accurate information (Ovacik and Uzsoy, 1994, 1997). However, the advent of computerised information systems capable of tracking job and machine status in real time has changed this situation. In many of the process industries, including the steel making example, which we will discuss later, information is generated in real time by process control computers. In discrete parts manufacture, computer systems for the entry and dissemination of data, such as VDU terminals and bar code scanners, are placed at various locations on the shop floor, to record information concerning the location and status of jobs and resources and to display this information for control purposes. Feedback can be generated from several or all work centres to track jobs and update their progress. This technology is now comparatively cheap and very effective (Singh, 1996).

Real time information is commonly used to improve estimated values of some parameters, such as processing time or worker performance, based on larger sample sizes. Real time information is only rarely used to improve schedules and then only in an ad hoc manner to locally correct short-term problems which might arise due to machine failure, etc. In this paper we develop a systematic approach to the use of real time information in scheduling.

Lindau et al. (1994) and Fredendall et al. (1996) have made empirical studies on the impact of real time information for specific industrial scheduling models. Both studies show that the performance of a system without shop floor information is inferior to a system where real time information is used to make real time scheduling decisions. Ovacik and Uzsoy (1994) exploit shop floor information to consider not only the jobs available at the machine at the time of the scheduling decision but also jobs that are going to become available within a certain time window. They studied a dynamic single-machine problem with sequence-dependent set-ups by comparing heuristic rules that use global information in making local scheduling decisions at the machine level to several myopic dispatching rules which use only local information. Chang (1997) considered a dynamic job shop where the arrival and nature of jobs is governed by known probability distributions. He showed that if estimates of queue length are updated in response to real time information, then the performance of several dispatching rules could be improved.

A dynamic scheduling system is one that uses real time information as it arrives. The planning and scheduling process then consists of one or more nested feedback loops, where each feedback loop corresponds to a scheduling period (month, week, day) and some group of processes which are undergone by the jobs. This group of processes might involve something as simple as being processed on a single machine, right up to a full manufacturing process. First, we formulate a static schedule for each period. Then we obtain real time information concerning, for example, the progress of each job and the shop floor situation, and react to that information to revise the schedule for the current period and processes, when circumstances make schedule changes desirable or necessary. Each feedback loop defines a *dynamic scheduling* problem. Each of our dynamic schedules will interact with other dynamic schedules at different time horizons and upstream and downstream processes, so that the effects of modifying a local schedule in response to a given piece of real time information must be considered throughout the system.

Practical scheduling systems need to be able to react to significant real time events within an acceptable response time and revise schedules appropriately. *Rescheduling* occurs when we restart the scheduling process from scratch. *Schedule repair* refers to some local adjustment of the current schedule and may be preferable for many reasons, not least because feasible schedules may be difficult to generate within acceptable time limits in many environments. The practical importance of the decision whether to reschedule or repair has been noted in recent papers by Lee et al. (1996) and Dorn and Kerr (1994). In order to decide what action we should take in response to an event, we should have some idea of the value of modifying our current schedules in response to the event and some measure of the overall impact of making schedule changes. In the following section we will use the quantitative measures of *utility* and *stability* to assess the value and impact of schedule changes.

Schedule repair plays an important role in some knowledge-based systems which have been developed in the Artificial Intelligence community. ISIS (Fox and Smith, 1984), OPIS (Smith et al., 1990) and IOSS (Park et al., 1996) are systems which have used knowledge-based scheduling methods to generate a feasible schedule and interactive scheduling methods to revise the existing schedule. CABINS (Miyashita, 1995) is an intelligent scheduling system which integrates case-based reasoning mechanisms for incremental accumulation and re-use of past schedule repair experiences to achieve efficiency of the revision process while preserving the quality of the resulting schedule. Suresh and Chaudhari (1993) survey several other knowledge based systems in this area.

Some work on schedule repair based on heuristic rules shows that schedule repair has the potential to improve the efficiency and flexibility of scheduling systems. Zweben et al. (1994) use simulated annealing and constraint propagation to repair schedules for space shuttle ground operations. Efstathiou (1996) introduced a software package, developed at Rover, to help schedulers carry out manual or semi-automatic schedule repair in response to real time events.

Work on dynamic scheduling is surveyed in the paper of Suresh and Chaudhari (1993) and, more recently, in the thesis of Guo (1999). When machine failure requires rescheduling within a job shop or flow shop environment we may use the match-up scheduling approach, which has been considered in several papers including Akturk and Gorgolu (1999) and Bean et al. (1991). When machine failure requires revision of the current schedule, this revision is carried out subject to ensuring that the revised schedule "matches up" to the original schedule as soon as possible after the machine breakdown, allowing some consideration of the stability of the shop. Jain and Elmaraghy (1997) used genetic algorithms to obtain an initial schedule and then heuristic rules to handle shop floor disruption. Daniels and Kouvelis (1995) and Leon et al. (1994) discussed the concept of schedule "robustness". Here we consider how adverse the effects of our chosen schedule repair strategy may be in response to machine failure in order to design a robust initial schedule to minimise these effects. None of the above work deals with the trade off between schedule quality and schedule stability in choosing an appropriate schedule repair strategy. Wu et al. (1993) consider this trade off for events taking place in real time, in order to compare the performance of three schedule repair strategies. All these approaches consider the best way of dealing with events as they occur, rather than the arrival of real time information concerning anticipated future events, where the time of arrival of the information is critical to the way in which the information may be effectively handled.

Ehlers and Van Rensburg (1994) consider eight different types of real time information. Such a taxonomy may be useful but we consider that there are essentially two kinds of real time information, illustrated in Fig. 1: that relating to the status of resources and that relating to the status of jobs.

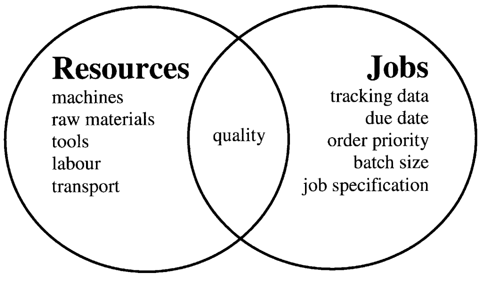


Fig. 1. Classification of real time information.

Real time information relating to the status of resources includes information concerning machines, raw materials, tools, labour, etc. Real time information relating to the status of jobs includes tracking data for each operation, data concerning successfully completed processing stages and information about schedule adherence. Information on actual or potential disruptions may relate to resources or jobs. Machine breakdowns, material or tool shortages and longer-than-expected processing times give resource problems. Job related disturbances arising from planning systems and customers include changes in priority, reassignments of jobs to orders and the emergence of new jobs. Quality problems may relate to both resources and jobs.

By having well-defined procedures for handling real time data we may both reduce the nervousness of the system and opportunistically improve schedule performance, compared with using ad hoc approaches. Most particularly, we can make a priori decisions as to what levels of system disruption are tolerable for a given level of performance improvement. When real time data arrives it is put through a four stage process: *detection*, *classification*, *identification* and *diagnosis*. Since real time events may occur every few minutes in a system (Stoop and Wiers, 1996) it is important that the procedure is standardised, with automatic computer intervention where possible. *Detection*: real time data are detected by, for example, sensors, barcode scanners or operators. Understanding the detection process will lead to effective use of real time data capture devices, and removal of unnecessary and useless devices. *Classification*: we must classify the event and decide whether it may be handled automatically, or requires a human decision-maker. *Identification*: after the real time information has been classified and possibly dealt with automatically, there will often remain a need for a more detailed analysis of the disturbance type. For example, we may wish more information as to why a machine breakdown has occurred, e.g. lack of maintenance or sabotage. Frequently occurring types of disturbance need deeper investigation, both for prevention and improved prediction. *Diagnosis*: here we decide what action to take in response to the piece of real time information. It is possible that we should take no action, conduct a limited repair or reschedule from scratch. We will use the quantitative measures of utility and stability to help us make this decision as to what action should be taken.

3. Measures of utility and stability

If we have a range of good techniques for repairing schedules and rescheduling in the presence of real time information, we must still address the important issue of whether to repair or reschedule and which schedule repair or rescheduling strategy should be used in response to any given collection of real time information. When we receive information concerning a highly significant anticipated future event, such as unplanned machine maintenance, the most appropriate course of action may well be to discard the current schedule and reschedule from scratch. In doing so, however, we must take into account the impact that this decision will have on current schedules for upstream and downstream processes and on future plans and schedules. When information concerning less significant anticipated future events is received, it may be most appropriate to adopt a schedule repair strategy, which attempts to find a revised schedule while minimising the disturbance to current and future plans. This "wait and see" approach means that we may do nothing or carry out only a small local repair in response to events. Of course we find, at some stage, that the accumulation of small anticipated events mean that the disruption caused by rescheduling is justified. It is important to note that in any operation where there are schedules which may be subject to unforeseen change, there must necessarily be a strategy for dealing with these events. Our experience of the steel, furniture and paper industries suggests that the strategies which are used in industry are, however, often ad hoc and not subject to the same kind of analytical rigour or sophisticated techniques which are applied to the scheduling decisions themselves.

The practical importance of the decision whether to do as little as possible, locally repair the schedule or to reschedule from scratch in response to a real time event is identified in Lee et al. (1996) and Dorn and Kerr (1994). Both these papers discuss decision support for scheduling of primary steel making processes, where a great deal of real time process control information is available.

In this section we will define two measures to guide the decision as to what strategy should be used to repair a schedule in response to real time information, *utility* and *stability*.

Utility will measure the benefit which may be gained by using a particular rescheduling strategy. Suppose that we have a mathematical model M of a scheduling process, where we have n numerically defined objective criteria $(O_1, O_2, \dots, O_n)$. Without loss of generality we suppose that each objective criterion is to be maximised. For example a piece of real time information arising from process control computer might be that upstream process controllers report that "the true size of job A123 is 50% greater than the scheduled size". Clearly the value of this information for rescheduling purposes depends heavily on the time at which it arrives. We suppose that for a (potentially compound) piece of real time information E there is a strategy S_0, corresponding to the notion of "do nothing", which yields objective function values $(a_1, a_2, \dots, a_n)$. For example S_0 might correspond to the strategy "make the originally scheduled orders to stock in response to the customer order cancellation in the hope of a later sale" or "create a buffer of work to be done in front of the malfunctioning machine and leave other machine schedules unchanged", with this latter strategy corresponding to the *pushback* strategy of Bean et al. (1991). A further example is given in the following section. Suppose further that we have a strategy S_1 which will produce a unique solution for the scheduling problem modified by this real time information, yielding objective function values $(b_1, b_2, \dots, b_n)$. Then the utility U is the multiple valued function given by

$$U(M, S_0, S_1, E) = (b_1 - a_1, b_2 - a_2, \dots, b_n - a_n).$$

When the model M has only a single objective criterion we will have a strategy S_{opt} which will give an optimal solution in response to the real time information E arriving at time t. If there is a clearly defined "do nothing" strategy S_0 then for any given set of real time events E we may simply say that the utility of the real time information concerning real time events E, arriving at time t, $U(M,E,t) = v_{\text{opt}} - v_0$, where v_{opt} is the objective value given by strategy S_{opt} and v_0 is the objective value given by strategy S_0. By deriving such a function for utility, we may make minor changes to many scheduling models in order to evaluate the benefit which may be gained from the use of real time information within that model.

To further increase the usefulness of our utility measure, we must also consider the disbenefits which occur due to disruption of the planned production processes for dynamic schedules at different planning horizons or for upstream and downstream processes. In the matchup scheduling approach discussed earlier (see for example Akturk and Gorgolu, 1999) we measure the disruption of schedule repair due to machine failure as the time until the repaired schedule matches up again with the planned schedule. Here we propose a more general approach. The problems associated with changing a scheduling decision may be very complex. Any model built to support scheduling decisions is unlikely to be able to take all of the possible factors into account and it may be appropriate to model the effect on stability in some simplified way. To accurately model the stability effects of a given local schedule change we would need to solve a global resource allocation, planning and scheduling problem across all upstream and downstream processes and within a time horizon of those schedules which are affected by the local change. Since this is unlikely to be practically possible we suggest here some simple stability measures based upon the temporal movement of tasks, using the idea that the most significant effects of the movement of jobs is in changing release dates for upstream and downstream processes. Our measure is related to that of Wu et al. (1993) which incorporates starting time movements and a measure of the change in job sequence. Suppose that in our model M we have jobs $J_1, J_2, \dots, J_n$ to schedule through a subset of the production process, and that some strategy S_{static} gives us a schedule where the start times are $t_1, t_2, \dots, t_n$ and the completion times are $C_1, C_2, \dots, C_n$, respectively. Suppose that in response to the set of real time events E we apply scheduling strategy S_{react} and the revised schedule has new start times $t'_1, t'_2, \dots, t'_n$ and new completion times $C'_1, C'_2, \dots, C'_n$, respectively. For real time information E arriving at time t we define the stability $S(M, S_{\text{static}}, S_{\text{react}}, E, t)$ to be some function of the start and completion times before and after rescheduling. Our stability measure cannot be obtained directly from our scheduling model in the same way that utility was obtained from the objective criteria. However, it seems sensible to require that the stability function is non-decreasing with each of $|t_i - t'_i|$ and $|C_i - C'_i|$. A simple example of a stability function might then be

$$S(M, S_{\text{static}}, S_{\text{react}}, E, t) = \sum_{i=1}^{n} \min\{\alpha(|t_i - t'_i| + |C_i - C'_i|), D_i\},$$

where D_i represents the maximum possible disturbance which might occur due to movement of the job J_i (e.g. the cost of outsourcing the job) and α is a parameter which represents the "cost" of the destabilising effect per time unit which the job is moved. We will explore the properties of a slightly different stability measure in the following section.

Given measures for the utility and stability of a strategy for the use of real time information in scheduling decision making we may trade off the desirable effects of high utility with the undesirable effects of a large impact on the stability. In Section 4 we will consider how this might be done in one particular case. The use of utility and stability measures as described in this section gives us a basis upon which to analyse a given planning and scheduling environment and arrive at concrete recommendations to choose between strategies for dealing with real time information. We should be able to choose among a range of strategies, which are not dominated in terms of their stability and utility performance, for dealing with real time information effectively.

4. Stability and utility of a single event for $n/1//\overline{C}$

In order to illustrate our notions of utility and stability, we will investigate these measures for the $n/1//\overline{C}$ scheduling model, where n jobs $J_1, J_2, \ldots, J_n$ having processing times $p_1 \leqslant p_2 \leqslant \cdots \leqslant p_n$, are to be scheduled on a single machine in order to minimise the average completion time and preemption is not allowed. The optimal sequence S_0, given by the shortest processing time (SPT) rule, is $(J_1, J_2, \ldots, J_n)$. In this optimal sequence job J_i will have completion time C_i given by $C_i = \sum_{j=1}^{i} p_i$ $(i = 1, 2, \ldots, n)$ and the average completion time $\overline{C}$ is given by

$$\overline{C} = \frac{1}{n}\sum_{i=1}^{n} C_i = \frac{1}{n}\sum_{i=1}^{n}(n - i + 1)p_i.$$

Suppose that we receive real time information concerning an event E at time t and in response to this information we create a new schedule S' in which job J_i has processing time p'_i and completion time C'_i, where the job sequence and completion times may or may not be changed by the event and the strategy for dealing with it.

Define the utility, $U(S_0, S', E, t)$, of information, arriving at time t, concerning real time event E as the difference in the objective $\overline{C}$ between a strategy S_0 which ignores the real time event E, thus staying with the scheduled sequence, and strategy S' which uses it. When S' makes optimal use of the information given the nature of the anticipated future event and the time t at which it arrives, we may simply write $U(E, t)$ in the notation of the previous section.

The stability measure we shall consider for the $n/1//\overline{C}$ model is the sum over all jobs of the absolute change of start and finish times divided by the number of jobs. Hence our measure of the impact of moving from schedule S_0 to schedule $S', S(S_0, S', E, t)$ is given by

$$S(S_0, S', E, t) = \frac{1}{n}\sum_{i=1}^{n}\left(|C_i - C'_i| + |(C_i - p_i) - (C'_i - p'_i)|\right).$$

In writing our expressions for U and S above we have omitted the model identifier M since we are referring throughout to the $n/1//\overline{C}$ model.

Consider the real time event E "the processing time for job J_k changes from p_k to p'_k". The time t at which this information arrives may be before, during or after the processing of job J_k in the original schedule. We will derive expressions for the utility of this piece of real time information and the effect on stability of adopting a strategy which maximises utility without considering stability.

In Fig. 2 we illustrate the three separate cases where job J_k will be moved in response to real time data E arriving at time t. In (i) we see that the processing time for J_k has become shorter and the revised processing time information has arrived in time so that job J_k may be placed in the sequence at the position suggested by the SPT rule to produce a new sequence to optimise utility. In (ii) we see that the processing time for J_k has become shorter, but this time the revised processing time information has arrived too late for job J_k may be placed in the sequence at the position suggested by the SPT rule, so job J_k must simply be placed as soon as possible after time t to make optimal use of the real time information with respect to utility. In (iii) we see that the processing time for job J_k has increased and information about the revised processing time has arrived in time to revise the schedule in accordance with the SPT rule.

Let the set of jobs which are moved in the final sequence, apart from J_k, be denoted D. Let a be the number of jobs coming after all the jobs whose position in the sequence has changed, then

$$a = \begin{cases} 0, & k = n, \\ n - k, & k < n,\ p'_k \leqslant p_{k+1}, \\ |\{i : p_i > p'_k\}|, & k < n,\ p'_k > p_{k+1}. \end{cases}$$

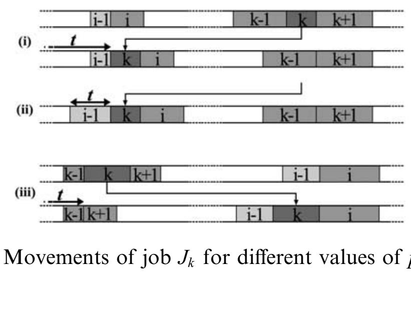


Fig. 2. Movements of job J_k for different values of p'_k and t.

Define $p_0 = 0$, $p_{n+1} = \infty$ for notational convenience. Then we have

$$D = \begin{cases} \{J_i, J_{i+1}, \ldots, J_{k-1}\} \\ \quad t \leqslant C_{i-1},\ p_{i-1} \leqslant p'_k < p_i < p_k, \\ \{J_i, J_{i+1}, \ldots, J_{k-1}\} \\ \quad i > 1,\ C_{i-2} < t \leqslant C_{i-1},\ p'_k < p_i < p_k, \\ \{J_{k+1}, J_{k+2}, \ldots, J_i\} \\ \quad t \leqslant C_{k-1},\ p_{k+1} < p'_k \leqslant p_{i+1}, \\ \emptyset \quad \text{otherwise,} \end{cases}$$

and we have

$$U(S_0, S', E, t) = U((p_1, p_2, \ldots, p_n), k, p'_k, t) = \frac{1}{n} \sum_{\{i:J_i \in D\}}^{n} |p_i - p'_k|$$

and

$$S(S_0, S', E, t) = \frac{1}{n}\left((2a+1)|p_k - p'_k| + 2\sum_{\{i:J_i \in D\}}^{n} (p_i + \min(p_k, p'_k))\right).$$

The expression for utility arises by considering the difference in completion times between the original sequence, using a "do nothing" strategy in response to the new processing time p'_k for job J_k, and the sequence following the movement of job J_k by strategy S'. If job J_k moves earlier in the sequence, then the jobs in D will have their completion times increased by p'_k, while job J_k will have its completion decreased by $\sum_{\{i:J_i \in D\}} p_i$. If job J_k moves later in the sequence, then the jobs in D will have their completion times decreased by p'_k, while job J_k will have its completion time increased by $\sum_{\{i:J_i \in D\}} p_i$.

The expression for stability arises by considering the change in start and finish times between the original and revised schedule. If job J_k moves earlier in the sequence, then the jobs in D will have their start and finish times increased by p'_k, while job J_k will have its start time decreased by $\sum_{\{i:J_i \in D\}} p_i$ and its finish time decreased by $|p_k - p'_k| + \sum_{\{i:J_i \in D\}} p_i$. Jobs $J_{k+1}, J_{k+2}, \ldots, J_n$ will have their start and finish times decreased by $|p_k - p'_k|$. If job J_k moves later in the sequence, then the jobs in D will have their start and finish times decreased by p_k, while job J_k will have its start time increased by $\sum_{\{i:J_i \in D\}} p_i$ and its finish time increased by $|p_k - p'_k| + \sum_{\{i:J_i \in D\}} p_i$. Jobs $J_{i+1}, J_{i+2}, \ldots, J_n$ will have their start and finish times increased by $|p_k - p'_k|$.

To investigate the behaviour of utility and stability, we simulate a 20 job problem with processing times uniformly distributed on the interval $[0, 10]$. At time t, we receive the real time information that the processing time for J_{10}, originally 5.53 changes to p'_{10}. We will explore the behaviour of utility and stability for different values of p'_{10} and t. In Fig. 3 we graph utility against p'_{10} and t assuming that we reschedule to optimise utility. In Fig. 4 we graph stability against p'_{10} and t, both for the case where we reschedule so as to optimise utility and in the case where we do not change the job sequence.

We can see that when p_{10} undergoes a small change then the utility of the information and the effects of rescheduling are both small. When p_{10} undergoes a larger change then the utility of the information is much higher as is the change in schedule stability caused by acting upon the information. When the processing time for job J_{10} is reduced, then the utility of the information exhibits a nearly linear dependence on the new processing time, increasing with the time of arrival of the information. When the processing time for job J_{10} is increased, the utility's dependence on the time is only insofar as the utility will be zero when the information arrives too late. The important point to note from the two graphs is that the utility

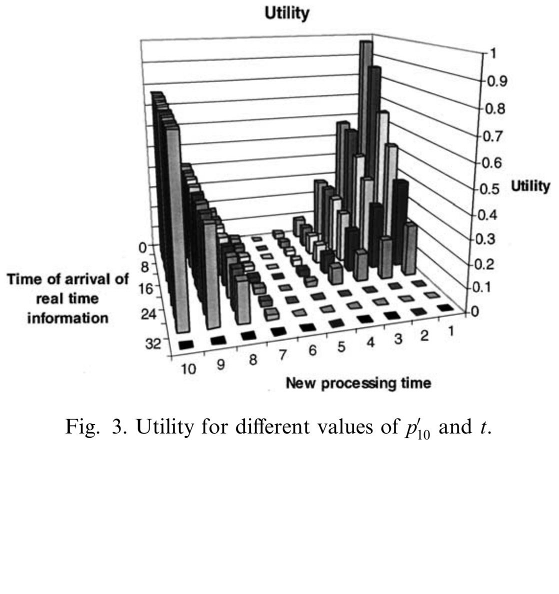
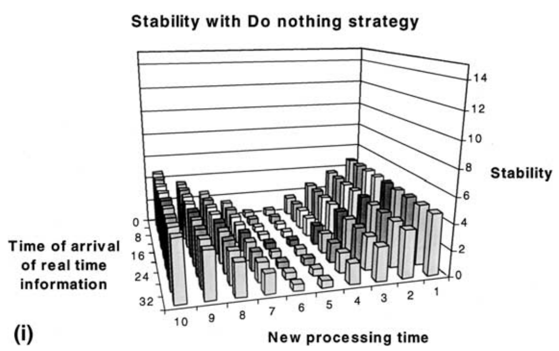


Fig. 3. Utility for different values of p'_{10} and t.



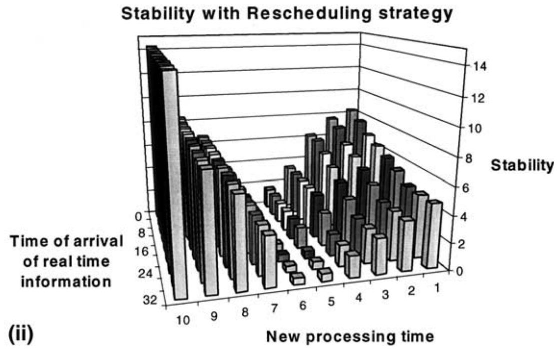


Fig. 4. (i) Change of stability for different values of p'_{10} and t, assuming we do not change the job sequence. (ii) Change of stability for different values of p'_{10} and t, assuming we reschedule to optimise utility.

and stability give rise to surfaces of different shapes and depend not only on the nature of the anticipated future event, but also on the time of arrival of the information. For example, we may see that when p'_{10} is much smaller than p_{10} and we receive limited prior warning, then rescheduling has substantial utility, but we see from Fig. 4(i) and (ii) that the difference in stability between rescheduling and not rescheduling is limited. We may consider a combination of utility and stability in deciding what action to perform in response to real time information to measure both the local improvement of schedule repair or rescheduling and any potential global adverse effects of rescheduling. Using this combination of utility and stability, together with a variety of schedule repair and rescheduling strategies, will allow us to trade off the effects of utility and stability to provide a consistent response to real time information. We will consider the behaviour of a number of these strategies in the following section.

5. Strategies for handling real time data for $n/1//\overline{C}$

In the previous section we have seen the behaviour of measures of utility and stability within the $n/1//\overline{C}$ scheduling model for a single piece of real time information. In this section we will consider how we may use these measures to decide on a schedule repair or rescheduling strategy when there are multiple real time events. We use simulation in order to compare different strategies.

Consider an $n/1//\overline{C}$ scheduling model, where processing times are chosen in advance from a uniform distribution on the interval $[0, 10]$. The pieces of real time information all have the same form, as considered previously: "at time t we find out that the processing time for job J_k changes from p_k to p'_k". Then our events are all chosen in advance with job J_k chosen at random from the set of jobs, p'_k chosen uniformly at random from $[0, 10]$, and t chosen uniformly at random in the interval $[0, (5n/2)]$. It is then perfectly possible that a given job may be affected several times by different pieces of real time information and that real time information may arrive too late to be acted upon. In order that a significant number of pieces of real time information do not arrive too late, our distribution ensures that real time information will arrive during the first half of the schedule.

When real time information arrives we may put the action performed in one of four categories:

1. *No move*: leave the job sequence unchanged, i.e. ignore the information.
2. *Repair* the sequence by moving only the job affected by the real time information. Try all possible positions in the sequence.
3. *Reschedule* the remaining jobs from the finish time of the job being processed currently using the SPT rule.
4. The event is *infeasible* since the information arrived too late to affect the schedule, i.e., it refers to a job already completed.

The course of action that we should take is determined by the resulting effect on average completion time (i.e., the utility of the real time information) and the effect upon the schedule stability, together with our chosen strategy. The strategies which we consider are: "do nothing" in response to real time information or maximise A (utility) − B (stability) for several different values of A and B. When $A = 1, B = 0$ we have the strategy "maximise utility with no consideration of stability". When $A = 0, B = 1$ we have the strategy "minimise the effect on stability without considering utility". For $A, B > 0$ we have a range of strategies which bridge the gap between these two extremes.

In order to compare our strategies, we measure the overall utility by considering the average processing time of the final sequence which each strategy produced in response to the real time information and the sum of stability effects of all of the moves which were made, divided by the number of jobs. The effects on stability come from two sources, first, simply from changes in the processing time of a single job and second, from any change in the job sequence resulting from use of the real time information. Hence the total stability effect of even the "do nothing" strategy will usually be greater than zero.

In Tables 1 and 2 we summarise our results. We have carried out two series of experiments, one with 25 jobs and 15 events, and one with 50 jobs and 15 events. We have randomly generated 5 sets of jobs together with 5 sets of events, giving 25 simulation runs for each series. For each simula-

Table 1
Summary of performance of the different strategies with 25 jobs and 15 events

Strategy	Rank Cbar	Rank stability	Final Cbar	Overall stability	Reschedule frequency	No move frequency	Repair frequency	Infeasible frequency
Do nothing	9.36	2.24	40.82	6.06	0.00	6.96	0.00	8.04
$A=0$, $B=1$	10.52	1.00	40.89	5.54	0.00	6.08	1.00	7.92
$A=1$, $B=1$	9.84	1.48	40.80	5.61	0.00	6.44	0.64	7.92
$A=2$, $B=1$	8.84	2.80	40.46	6.25	0.00	6.00	1.12	7.88
$A=3$, $B=1$	7.36	4.40	38.85	9.80	0.16	4.44	2.32	8.08
$A=5$, $B=1$	5.48	5.84	37.73	13.26	0.24	3.32	3.24	8.20
$A=6$, $B=1$	4.60	6.52	37.62	13.80	0.24	3.16	3.48	8.12
$A=8$, $B=1$	4.48	7.32	37.54	14.41	0.28	2.8	3.80	8.12
$A=10$, $B=1$	3.36	7.80	37.36	15.48	0.32	2.48	3.96	8.24
$A=15$, $B=1$	2.48	9.32	37.22	16.36	0.44	1.92	4.44	8.20
$A=25$, $B=1$	1.60	10.12	37.06	17.10	0.44	1.80	4.60	8.16
$A=1$, $B=0$	1.72	10.96	37.16	19.12	0.00	1.08	5.80	8.12

All values are averages over 25 simulation runs.

Table 2
Summary of performance of the different strategies with 50 jobs and 15 events

Strategy	Rank Cbar	Rank stability	Final Cbar	Overall stability	Reschedule frequency	No move frequency	Repair frequency	Infeasible frequency
Do nothing	7.60	2.56	82.25	7.85	0.00	7.12	0.00	7.88
$A=0$, $B=1$	8.32	1.36	82.05	7.10	0.04	5.52	1.72	7.72
$A=1$, $B=1$	7.80	1.72	81.96	7.31	0.04	5.72	1.52	7.72
$A=2$, $B=1$	6.64	2.84	81.43	7.77	0.08	5.64	1.56	7.72
$A=3$, $B=1$	5.60	4.12	80.08	10.98	0.16	4.40	2.76	7.68
$A=5$, $B=1$	4.68	4.96	79.31	13.90	0.20	3.52	3.56	7.72
$A=8$, $B=1$	3.68	6.68	78.87	15.37	0.16	2.92	4.16	7.76
$A=10$, $B=1$	2.92	7.68	78.70	16.79	0.20	2.24	4.80	7.76
$A=20$, $B=1$	2.04	8.84	78.48	19.48	0.16	1.44	5.68	7.72
$A=1$, $B=0$	1.20	9.96	78.36	24.68	0.00	0.28	6.92	7.80

All values are averages over 25 simulation runs.

tion run, we consider several strategies by giving different values of parameters A and B, as well as the "do nothing" strategy. For each simulation run, we rank the strategies in terms of their final value of $\overline{C}$ (Cbar) and the sum over all events of the effect on stability (stability). Recall that we would wish to minimise both of these quantities. The average rank for each strategy over all simulation runs is given in columns 2 and 3 of the tables. The average values of final Cbar and stability sum are given in columns 4 and 5. In columns 6, 7, 8 and 9 we show the average frequency over all simulation runs of the category of action which was taken in response to the real time information.

Figs. 5 and 6 illustrate the relationship between Cbar and Stability for each series and strategy. They clearly illustrate that there is a continuum of different strategic approaches to dealing with real time information between "do nothing" and "always reschedule to maximise utility without considering stability", where we may trade-off Cbar and stability. This enables a choice of behaviour in response to real time information based upon a consistent, rational approach. In an environment where we may tolerate significant changes in stability, it still seems that strategies such as $(A = 10, B = 1)$ give a marked improvement in stability performance over the strategy which ignores stability, for very minor changes in mean completion time. Indeed, we see in Fig. 5 that the $(A = 20, B = 1)$ strategy strictly outperforms the $(A = 1, B = 0)$ strategy, however this is principally due to a single simulation run where a minor change in schedule sequence made a piece of real time information infeasible for one strategy but not the other. We see that in both Figs. 5 and 6, the "Do Nothing" strategy is dominated since the $(A = 1, B = 1)$ strategy gives strictly better performance with respect to both Cbar and stability. We may conclude that even in an environment where schedule stability is critical, using real time information is better than ignoring it.

Figs. 7 and 8 show the average over all simulation runs of the frequency of each response cat-

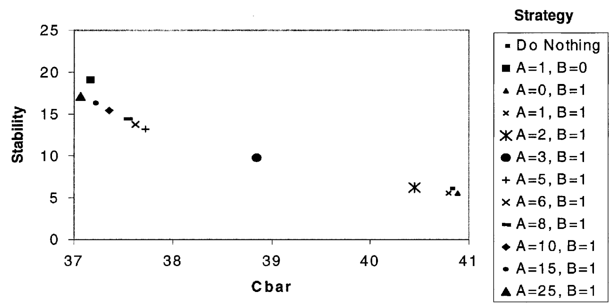


Fig. 5. Average performance for each real time information strategy over 25 simulation runs of 25 jobs and 15 pieces of real time information.

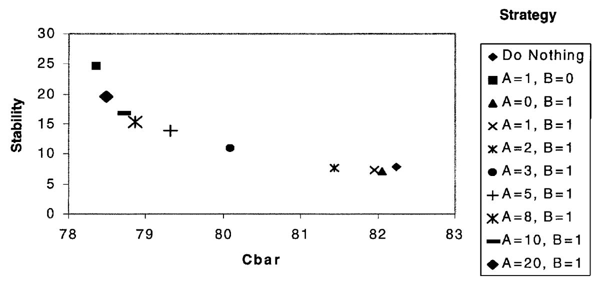


Fig. 6. Average performance for each real time information strategy over 25 simulation runs of 50 jobs and 15 pieces of real time information.

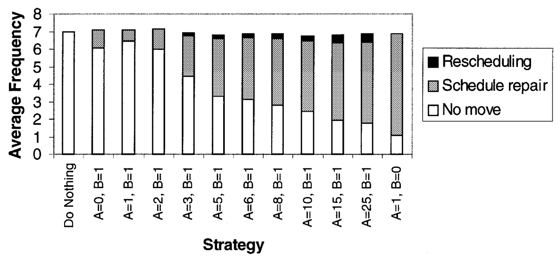


Fig. 7. Average frequency of each category of response to real time information over 25 simulation runs of 25 jobs and 15 pieces of real time information.

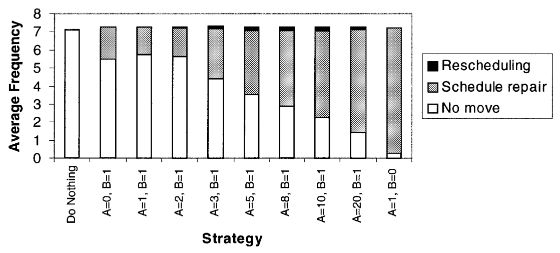


Fig. 8. Average frequency of each category of response to real time information over 25 simulation runs of 50 jobs and 15 pieces of real time information.

egory to each piece of real time information. Intuitively a large ratio A/B should give rise to more frequent rescheduling and schedule repair, in order to reduce Cbar. This behaviour is shown in both graphs. Note that in the $(A = 1, B = 0)$ strategy, where we wish to minimise Cbar with no consideration of stability, there is no need for rescheduling since all schedule repairs give rise to the same sequence that would be produced by rescheduling the remaining jobs using the SPT rule. For strategies with appropriately balanced A and B, we see precisely the behaviour which we might wish in practise, where minor or no changes are made in order to minimise the impact on schedule stability until such time as rescheduling is justified by an event having very large utility.

Figs. 9 and 10 show the marked difference in behaviour, in terms of final Cbar and stability values, of the strategies for different sets of events and the same initial schedule. We see a similar "reflected J" shape for each of the event sets. However some of the event sets give rise to very large discontinuities in the "reflected J". In general

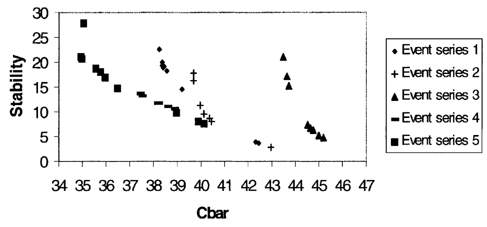


Fig. 9. The performance of each strategy for five different sets of events and the same initial set of 25 jobs.

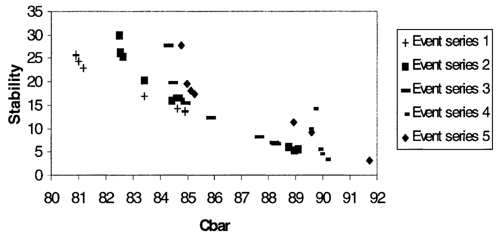


Fig. 10. The performance of each strategy for five different sets of events and the same initial set of 50 jobs.

these are due to one strategy perceiving insufficient utility to justify the large change in stability for a rescheduling move, whereas for another strategy this is justified.

To summarise then, even in an environment where utility and thus our original scheduling objective dominates, giving stability a small amount of weight will often produce schedules of comparable utility and greatly improved stability, and vice versa. When neither consideration dominates, we have continuum of strategies which allow us to make a balanced choice, reflecting our view as to the relative importance of utility and stability.

6. Using real time information for scheduling a steel continuous caster

A steel continuous caster uses one to eight parallel oscillating moulds to transform large ladles of liquid steel, each weighing about 200 tonne, of a known chemical composition, into hot steel slabs, weighing 10–20 tonne, whose chemistry and dimensions match customer orders. Several ladles are scheduled into a continuous sequence which should be as long as possible to maximise throughput and minimise setup costs. The scheduling problem is unusual in that in addition to the production pull due to the need to satisfy customer orders, there is a production push due to the need to cast all the liquid steel coming from the blast furnace. Lee et al. (1996), Cowling (2002) and Cowling and Rezig (2000) give more details of the scheduling problem. Our aim here is to discuss how the techniques of dynamic scheduling discussed in this paper might be applied to this scheduling problem.

The ladles of molten steel go through a complex production process which may take a few hours between leaving the blast furnace and arriving the continuous caster, with many parallel production routes. Most of these processes are computer controlled and give rise to real time information. It is not uncommon that we have available real time information from process controllers as well as sales and marketing staff, a few hours before the impact of the real time information will be felt at the caster, but usually after the caster schedule has been generated. Principal real time events for short-term scheduling of the continuous caster include: information concerning the true weight of ladle contents, ladle timing information, ladles having the wrong chemistry, upstream machine failures which restrict the steel chemistries which can be manufactured and changes in downstream steel demand to balance material flows, particularly in response to machine failure. When real time information arrives we can consider several strategies for dealing with this information:

1. Stay with the existing plan and make to stock.
2. Search order book for an unscheduled future order, to change only the order which will be made in the current ladle (schedule repair).
3. Allow localised changes to the current schedule in addition to 2, by changing the orders which are made in ladles of the current sequence, which will not have significant impact on upstream/downstream schedules (schedule repair).
4. Allow major changes which may affect upstream and downstream plans, and change plans at longer time horizons (rescheduling).

Our measure of utility of a piece of real time information will be the difference in some measure of profitability between strategy 1 and optimal application of strategy 4. Once we have a model which has quantified scheduling objectives, see for example (Cowling and Rezig, 2000), then we can immediately use this to provide a measure of utility. Our measure of stability of a scheduling strategy will be a measure of the cost of the disruption caused by changes in scheduled processing times, for the continuous caster as well as for the upstream steel making, downstream milling processes and medium term plans. Note that providing a suitable measure of disruption will be a difficult modelling task, but is an important part of putting our continuous caster scheduling model in a global context and avoiding global suboptimisation through local optimisation of the casting process. Our measures of utility and stability will give us a consistent method for handling real time information in a scheduling environment where dealing with this information is critical. Work is continuing in this area.

7. Conclusion

In this paper we have addressed an important gap between scheduling theory and scheduling practice: that scheduling models and algorithms are unable to make use of real time information, which is widely available from process control computers and other monitoring systems. We have surveyed and categorised the range of real time information that may be captured and given approaches for dealing with it. We have reviewed work in the literature which looks at techniques for repairing schedules in response to real time events, pointing out that the issue of constructively using real time information concerning an anticipated future event has been little considered.

We have identified two principal types of reaction to a real time information, *rescheduling*, which involves significant system wide schedule changes and *schedule repair*, which makes only localised changes. In order to decide what action to take in response to a given collection of real time information, we have defined general measures of *utility* and *stability*. We may use these two measures to evaluate strategies for dealing with the real time information. We have then gone from this generally applicable framework to consider the specific $n/1//\overline{C}$ single machine scheduling model, first by considering the utility and stability for a single event and a strategy which optimises utility, in detail and then considering several strategies for dealing with multiple pieces of real time information. For multiple pieces of real time information arriving and the $n/1//\overline{C}$ model, we have designed a simulation experiment which demonstrates that there is a continuum of strategies for dealing with real time information, allowing the decision maker to trade off utility and stability performance in choosing an appropriate strategy. We anticipate that we may find such a continuum of strategies for a wide range of scheduling models and production processes.

Finally, we have discussed how we may incorporate our framework for dealing with real time information into the scheduling problem for the steel continuous caster. In this production environment there is both a great deal of process control information and a frequent need for schedule repair and rescheduling, which must take into account upstream and downstream production processes.

By incorporating an ability to constructively use real time information, using the framework provided by the repair/reschedule decision and the trade off between measures of utility and stability, we believe that techniques of scheduling theory may be made more effective and efficient at solving a range of practical scheduling problems. In this way we believe that we may attack one of the barriers to the industrial and commercial application of scheduling theory. Further work in this area would include the application of scheduling systems able to use real time information in appropriate industrial and commercial applications on the one hand, and the incorporation of the ability to handle real time information into models and algorithms from the scheduling theory.

References

Akturk, M.S., Gorgolu, E., 1999. Match-up scheduling under a machine breakdown. European Journal of Operational Research 112, 81–97.

Bean, J.C., Birge, J.R., Mittenthal, J., Noon, C.E., 1991. Matchup scheduling with multiple resources, release dates and disruptions. Operational Research 39 (3), 470–483.

Chang, F.P.R., 1997. Heuristics for dynamic job shop scheduling with real time updated queuing time estimates. International Journal of Production Research 35 (3), 651–665.

Cowling, P., Rezig, W., 2000. Integration of continuous caster and hot strip mill planning for steel production. Journal of Scheduling 3 (4), 185–208.

Cowling, P., 2002. A flexible decision support system for steel hot rolling mill scheduling. Computers and Industrial Engineering, forthcoming.

Daniels, R.L., Kouvelis, P., 1995. Robust scheduling to hedge against processing time uncertainty in single-stage production. Management Science 41 (2), 363–376.

Dorn, J., Kerr, R.M., 1994. Co-operating scheduling systems communicating through fuzzy sets. In: Proceedings of the 2nd IFAC/IFIP/IFORS Workshop on Intelligent Manufacturing Systems, pp. 687–704.

Efstathiou, J., 1996. Anytime heuristic schedule repair in manufacturing industry. IEE Proceedings – Control Theory and Applications 142 (2), 114–124.

Ehlers, E.M., Van Rensburg, E., 1994. An intelligent object-oriented scheduling systems for a dynamic manufacturing environment. In: Proceedings of ICIM Singapore, pp. 702–709.

Fox, M.S., Smith, S.F., 1984. ISIS: a knowledge-based system for factory scheduling. Expert Systems 1, 25–49.

Fredendall, L.D., Melnyk, S.A., Ragatz, G., 1996. Information and scheduling in a dual resource constrained job shop. International Journal of Production Research 34 (10), 2783–2802.

Guo, B., 1999. Scheduling problems with consideration of machine reliability, PhD thesis, Department of Industrial Management and Engineering, Science U. Tokyo.

Jain, A.K., Elmaraghy, H.A., 1997. Production scheduling/rescheduling in flexible manufacturing. International Journal of Production Research 35 (1), 281–309.

Lee, H.S., Murthy, S.S., Haider, S.W., Morse, D.V., 1996. Primary production scheduling at steelmaking industries. IBM Journal of Research Development 40 (2), 231–252.

Leon, V.J., Wu, D., Storer, R.H., 1994. Robustness measures and robust scheduling for job shops. IIE Transactions 26 (5), 32–41.

Lindau, R.A., Kanflo, T., Lumsden, K.R., 1994. Impact of real-time information for scheduling a car-body shop – a simulation study. International Journal of Operational and Production Management 14 (3), 114–125.

MacCarthy, B.L., Liu, J., 1993. Addressing the gap in scheduling research: A review of optimization and heuristic methods in production scheduling. International Journal of Production Research 31 (1), 59–79.

Miyashita, K., 1995. CABINS: A framework of knowledge acquisition and iterative revision for schedule improvement and reactive repair. AI 76, 377–426.

Ovacik, I.M., Uzsoy, R., 1994. Exploiting shop floor status information to schedule complex job shops. Journal of Manufacturing Systems 13 (2), 73–84.

Ovacik, I.M., Uzsoy, R., 1997. Decomposition Methods for Complex Factory Scheduling Problems. Kluwer, Dordrecht.

Park, J., Kang, M., Lee, K., 1996. Intelligent operations scheduling system in a job shop. International Journal of Advanced Manufacturing Technology 11, 111–119.

Singh, N., 1996. Systems Approach to Computer-integrated Design and Manufacturing. Wiley, New York.

Smith, S.F., Ow, P.S., Potvin, J.-Y., Muscotella, N., Matthys, D., 1990. An integrated framework for generating and revising factory schedules. Journal of Operational Research Society 41 (6), 539–552.

Stoop, P.P.M., Wiers, V.C.S., 1996. The complexity of scheduling in practice. International Journal of Operational and Production Management 16 (10), 37–53.

Suresh, V., Chaudhari, D., 1993. Dynamic scheduling – a survey of research. International Journal of Production Economy 32, 53–63.

Wu, S.D., Storer, R.H., Chang, P.-C., 1993. One-machine rescheduling heuristics with efficiency and stability as criteria. Computer Operational Research 20 (1), 1–14.

Zweben, M., Duan, B., Deale, M., 1994. Scheduling and rescheduling with iterative repair. In: Fox, M.S. (Ed.), Intelligent Scheduling. Wiley, New York, pp. 241–255.