Int J Adv Manuf Technol (2002) 19:756–763

The International Journal of
Advanced
Manufacturing
Technology

Reactive Recovery of Job Shop Schedules – A Review

A. S. Raheja and V. Subramaniam

Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore, Republic of Singapore

In the recent past, a great deal of time and effort has been devoted to the development of job shop schedules. These schedules are able to cope with the dynamic and stochastic nature of job shops that consist of the uncertainties both at the planning stages and on on-line execution. Deviations from predictive schedules occur when the job shop experiences both external disturbances (e.g. urgent job arrivals) and internal disruptions (e.g. machine breakdowns). To avoid complete rescheduling of the job shop a repair and recovery strategy for the schedule becomes essential. This paper provides a comprehensive review of the literature on the reactive recovery of job shop schedules and proposes further research work in this area.

Keywords: Job shop schedules; Reactive scheduling; Schedule recovery; Schedule repair

1. Introduction

Job shops face uncertainties owing to disruptions occurring on the shop floor. In the real world, machines break down, wrong job quantities are produced, a machine other than the one that was planned performs the work, supplies do not arrive when expected, power fails during critical operations, set-up takes longer than anticipated, parts are misplaced, personnel do not perform as expected, etc. [1]. These events cause the predictive schedules to be disrupted and this in turn renders the original schedule useless. In such cases, a simple approach would be to gather the data from the shop floor when the deviation occurs and totally reschedule the system [2]. This process is time-consuming, as there could be a large amount of information to be collected from the shop floor and it could be difficult to collect. In addition, the time involved in collecting the information could be excessive leading to the failure of the scheduling mechanism.

Therefore, it is becoming increasingly established that the predominant scheduling activity in the real world is *reactive scheduling*, which can be defined broadly as the continuous adaptation and improvement of precomputed predictive schedules [3]. Reactive scheduling has a number of similarities and differences to predictive scheduling. In one sense, many predictive scheduling approaches, which may be considered to operate "off-line", are concerned with the iterative improvement of some initial schedule (for example, by interchanging two jobs and evaluating the performance measures). Reactive scheduling can be construed as a similar activity, albeit conducted "on-line" in which the previously accepted schedule, which has now been flawed owing to an unexpected event, is repaired by techniques that can be essentially similar to those used to improve a predictive schedule iteratively. It is, however, the on-line nature of reactive scheduling and associated real-time execution requirements that constitute one of the major differences between predictive and reactive scheduling. The reactive schedule must be computable in the time window within which the schedule remains valid. In a complex real-time environment, this timespan could be very short.

Reactive scheduling can be seen as an upgrade of the predictive schedule with the addition of an on-line schedule recovery repair strategy built on the predictive schedule. Schedule recovery, therefore, is a concept that largely amalgamates the off-line and on-line environment for reactive scheduling. Better schedule recovery requires human knowledge of the scheduling environment. As the scheduling systems have gradually moved from theoretical formulations to computerised shop floor implementations, emphasis has been laid on schedule recovery principles that can deal effectively with the reactive environment in the shortest time frame. Much work has been devoted to this area in recent years, but no attempt has been made to focus specifically on the study of repair methodologies. Thus, we are reviewing these methods and bringing them to a common platform for discussion. The purpose of this paper is to explore the scope and status of both the existing and the proposed schedule recovery approaches. In addition, the different approaches will be compared and a robust plan for future work will be formulated so that effort can be channelled into a common and beneficial direction.

In the next section, we describe the various scheduling approaches in static and dynamic job shop environments. In Section 3 an overview of the various schedule recovery methods implemented and proposed in the literature is

Correspondence and offprint requests to: Dr V. Subramaniam, Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, 119260 Singapore. E-mail: mpesubra@nus.edu.sg

presented. Section 4 evaluates various methods of schedule recovery and outlines a course for future work. Finally, we conclude with a summary of the work presented in the literature and the scope for future work.

2. Job Shop Scheduling Approaches

The scheduling environment in the job shop can be classified into two distinct categories: *static* and *dynamic and stochastic*. In a static job shop it is assumed that a number of jobs, each consisting of a sequence of operations with given processing times are to be manufactured on a number of machines and the aim is to sequence the operations optimally. In a dynamic and stochastic job shop, real-life is represented by jobs arriving randomly over time, and the uncertainties concerning job attributes, such as processing times and shop floor availabilities are considered.

2.1 Static Scheduling

The static scheduling problem has been widely investigated. Many techniques have been proposed and implemented, as shown in Fig. 1. The branch and bound technique [4], mathematical integer programming [5,6] and simple heuristic scheduling rules [7–10], have been used successfully. However, these methods are usually applied to relatively simple problems. The use of dispatching rules is another approach that has been widely studied, and these rules have been applied individually [11–14] as well as in combinations [15–18] with other dispatching rules. There is no single dispatching rule that can with certainty give good results for all performance measures. Recently, artificial intelligence related techniques such as genetic algorithms [19,20], neural networks [21], fuzzy logic [22], tabu search [23–25] and simulated annealing [26,27] have also been used to solve static job shop problems.

2.2 Dynamic and Stochastic Scheduling

The dynamic and stochastic job shops are subject to uncertainties, and reflect shop floor conditions that are difficult to model and handle. Shop floors are prone to disruptions such as machine breakdown, urgent jobs, and absence of workers. Since the operational environment in a job shop is dynamic and stochastic, it is clear that any effective scheduling system has to be reactive, i.e. perform schedule revisions in response to the unforeseen events during the schedule execution [28]. These schedule revisions can be dealt with during both off-line and on-line phases of schedule generation and execution, as shown in Fig. 2.

The initial focus in dealing with this stochasticity was in the off-line mode, i.e. while building the static schedule. The methods that have evolved to produce schedules at the planning stage itself are able to anticipate and absorb the disruptions that might occur during the execution phase and are collectively also known as *robust* [29,30] and *predictable* [1,31]. A scheduling method is said to be robust if it provides a schedule, the performance of which remains high in the presence of uncertainties [29]. This process focuses on minimising the effects of disruptions on the performance measures. In predictable scheduling, the focus of the scheduling is to complete jobs as planned while maintaining an acceptable realised schedule performance. This is achieved by inserting idle-times into the predictive schedule to allow it to recover from the disruptions that may occur [31].

Online scheduling practices comprise *reactive scheduling* and *totally reactive scheduling*. In totally reactive scheduling, a rolling-time approach is followed in which no firm schedule is generated in advance, and decisions are made locally in real-time [32,33]. A frequently used reactive approach is to use local priority dispatch rules. In this approach, whenever a machine becomes free, a job currently in its queue is selected for processing based on a priority index or performance measure that is calculated from the job and machine attributes [34]. Reactive scheduling is a process of revising a given schedule in real-time owing to unexpected events occurring during the execution of the schedule [35]. An intuitive approach would be to resolve the problem from scratch, i.e. rerun the predictive schedule and generate a new optimised schedule. This approach is not encouraged in industry as the new schedule can differ considerably from the old one and this is not desirable since many other decisions such as assignment of personnel, delivery of raw material and the subsequent processing of the jobs in other facilities, may be severely disrupted. This phenomenon is commonly referred to as shop floor nervousness [35]. Therefore, a better approach would be to adapt the old schedule to the new situation, by implementing a specialised repair mechanism to repair the schedule in an iterative mode as and when disruptions occur.

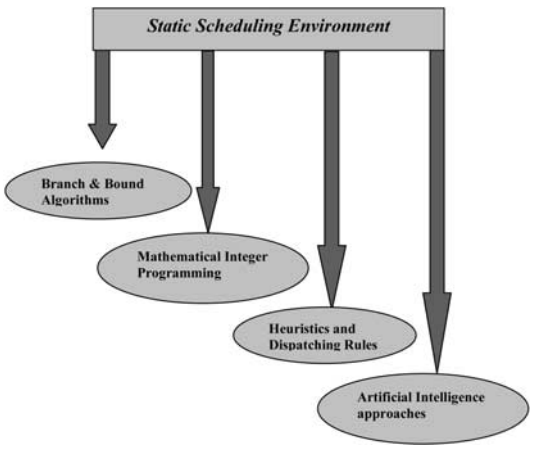

Fig. 1. Static scheduling approaches.

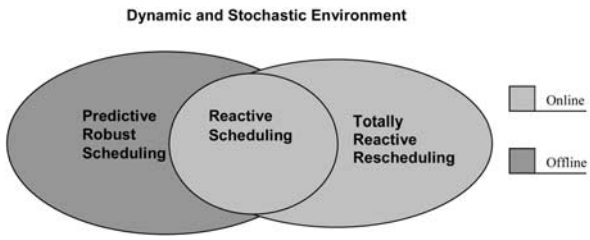

Fig. 2. Dynamic scheduling approaches.

3. Schedule Recovery

3.1 Definition

Schedule recovery and repair [36,37] is a procedure for modifying the original predictive schedule to accommodate sudden temporal changes in the job shop. Examples of these temporal changes include internal disruptions such as machine breakdown and external disruption such as arrival of an urgent job. The goal of schedule recovery is to avoid wastage of time and resources in capturing the status of the job shop for rescheduling whenever a minor disruption occurs. If a disruption continues for a longer period of time, or when very frequent disruptions are encountered, rescheduling is preferred, i.e. another predictive schedule considering the present status of the shop floor is generated. Therefore, schedule recovery can be described as an intermediate stage introduced in automated scheduling for monitoring in real-time and avoiding the total rescheduling of the system as far as possible [38].

3.2 Factors Affecting Schedule Recovery

The predictive schedules are usually based on heuristics and optimisation principles. Therefore, any correction or repair strategy applied will cause a deviation from the originally optimised schedule. In addition, the performance measures will no longer be optimal after the schedule is repaired. A better schedule recovery strategy is one that leads to minimum deviation of the performance measures while incorporating the necessary modifications and repair objectives. The focus is to implement a recovery method that can handle the repair of the schedules without compromising on the quality of the schedules [39].

Since rules and constraints are involved when schedules are adjusted, schedule recovery and repair can therefore also be viewed as a constrained scheduling problem. The objective of the problem could be to minimise the deviation in performance during the repair while satisfying the constraints [40]. Constraints in a job shop are decided based on a number of factors, e.g. routeing of jobs, availability of machines, and priority assignment for jobs and machines. The constraints are built into the schedule recovery algorithm, and can be in the form of a set of "if–then" rules or in the form of case-based reasoning data from past experience, or in terms of an expert's knowledge of the domain. A reactive recovery approach can then be evaluated on the basis of the ability of the algorithm to satisfy the constraints to an acceptable degree, while preparing the feasible revised schedule.

Since the schedule recovery and repair integrates unexpected events or disruptions into the original predictive schedule, the time required to carry out the repair should be minimised so that the schedule is still active and not outdated by the time it is ready to replace the original schedule. When this requirement is not met, the basic objective of the repair is lost and the process is flawed. Therefore, the minimum time frame required to react to the disruptions in the original predictive schedule may be viewed as another critical factor by which the schedule recovery performance is evaluated.

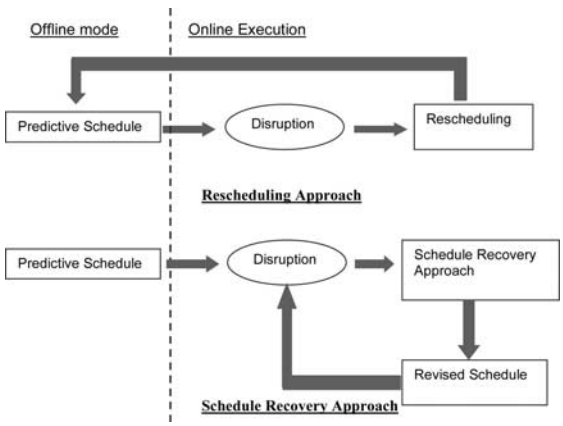

Fig. 3. Rescheduling vs. schedule recovery

3.3 Methods of Schedule Recovery

A summary of repair methods that have been successfully applied in a reactive environment for schedule recovery is presented in Table 1. These methods appear similar to the predictive schedule generation approaches. However, the difference lies in the reactive mode in which the methods are applied on a real-time basis.

The *right–shift rescheduling* approach [40] is the simplest schedule repair approach reported. The process consists of shifting the operations globally and expanding the schedule towards the right on the time axis in order to accommodate the disruptions. This "right-shifting" to accommodate the disruptions arising from machine breakdown, results in gaps in the schedule, during which the machines are idle and waiting for jobs [41]. The schedule obtained using this approach is generally of poor quality, and the method can be used only if the job shop is generally stable and only minor deviations occur from the original schedule.

Heuristic-based approaches [41–43] consist largely of explicit algorithms for schedule repair and optimisation. One such heuristic-based approach that has been reported to be effective in the literature is the affected operation rescheduling (AOR). AOR has been implemented, and its performance with respect to measures of efficiency and stability are compared with that of total rescheduling and right-shift rescheduling [41]. In this algorithm, only the operations that are affected by the disruption are rescheduled. The basic concept of AOR is to accommodate any disruption by pushing the starting times of some operations forward by the minimum amount possible so as to keep the technological constraints satisfied and to preserve the initial sequence of the operations on each machine. Another heuristic approach based on games theory [43] performs better when compared to total rescheduling and right-shift scheduling. Heuristic-based algorithms are easy to use and implement, and can handle disruptions such as machine breakdown and urgent job arrivals effectively. However, it does not perform a search for optimum repair and constraint satisfaction. The schedule generated is better in performance than right-shift rescheduling [41] but lacks the optimisation and quality improvement fea-

Table 1. Summary of reactive repair approaches.

Schedule recovery method		Advantages	Disadvantages	Performance measures	Prerequisites for effective application	References
Right shift rescheduling (RSR)	Heuristic based approaches	Simplest method. No algorithm is needed.	Poor quality of schedule. Constraints and rules are not resolved.	Makespan and deviation from the original schedule.	Job shop should usually be stable with minor disruptions at prolonged intervals.	[37, 40, 42]
Affected operation rescheduling (AOR)		Simple algorithm to implement. Better quality schedule than RSR.	Response time is greater. Limited disruptions can be handled. Schedule quality is not recovered after repair.	Makespan and deviation from the original schedule.	Job shop should usually be stable with minor disruptions at prolonged intervals. The technological constraints and processing times are predetermined and fixed.	[40, 44]
Multi-agents in DAI (active scheduling approach)		Complete automated approach. Module for repair, refinement and rescheduling. Responsiveness of system is good. Multithreaded operations are possible.	Coordination between the agents is difficult to achieve. Better integration between human and automated agents is difficult to achieve.	Computation time reactiveness and quality measures.	Dynamic job shop with uncertainties and random disruptions can be modelled.	[45, 46, 47]
Knowledge based scheduling and artificial intelligence approaches	Case-based reasoning (CBR)	Well-suited to domain specific problems. Continuous learning from past cases. Multiple disruptions can be modelled and addressed.	Extensive search through the database is time-consuming. An extensive experience database is essential.	Schedule quality. Reactive efficiency in terms of deviation from the original schedule.	Job shops with specific rule sets and multiple disruptions where scheduling experience is available as expert's advice or case database.	[28, 32, 52, 53]
	Constraint-based scheduling	Human interaction and supervision is better. Timely response is possible in stipulated time frame. Performs better than CBR as it includes both knowledge base and constraint satisfaction modules.	Real-time approach requires further refinement. Multiple agents architecture is needed for multithreaded operations for random disruptions.	CPU time (execution responsiveness). Schedule quality and repaired weightage tardiness.	Dynamic job shops with multiple disruptions and events occurring in a knowledge extensive system.	[28, 54, 55]
	Fuzzy logic	Complete scan of the schedule for constraint violation after every repair. Random processing times can be used for disruptions. Response is fast as the same module is used for schedule generation and repair.	The knowledge of the domain has to be built into the algorithm. Learning and growth of the algorithm is not possible.	CPU time (execution responsiveness). Schedule quality and repaired weightage tardiness.	Job shops with variability in processing time and large number of constraints to be adhered to either fully or partially.	[48, 49, 50, 51]
	Neural network	Response time is very fast for trained neural net. Predictions are extrapolated from past experience and are reliable.	Carefully prepared training sets are required for accurate prediction. Extensive knowledge base and expert advice has to be formulated in the form of a knowledge base.	CPU time (execution responsiveness). Schedule quality and repaired weightage tardiness.	More applicable in job shops with a continuous flow and repetitiveness in the type of disturbances.	[56, 57]

tures present in other repair approaches. These approaches are not suitable for highly stochastic job shops as they handle the schedule disruptions independently of each other and only one disruption is considered at any one time.

Multi-agents in a distributed artificial intelligence (DAI) environment has been widely reported [44–47]. In DAI-based approaches, reactive schedule recovery is achieved using multi-agents. It allows the independent agents to coordinate their knowledge and solve subproblems while working toward a common goal. In this approach, intelligent agents possess the knowledge pertaining to the schedule repair. In addition, the agents are also actively engaged in improving the schedule. The human scheduler can also act as one of the intelligent agents and become actively involved in the decision process. DAI has also been used as a blackboard-based opportunistic approach [46]. This method supports a cooperative problem-solving effort of independent agents communicating by posting messages on the blackboard to modify and improve the schedule. The advantage of the DAI approach is that it provides effective integration of the human agent and the automated knowledge agents. The central problem with the use of DAI is how to achieve coordination among such agents, so that they accomplish more as a group than individually.

Active scheduling [45] is a concept proposed on the principles of multi-agents. The concept acknowledges that rescheduling has to be very fast and the stream of data to be collected from the shop floor has to be automated. In addition, the schedule has to be optimised with regard to some predefined quality functions. The following hypothesis is stated and presented in [45]:

$$\text{Rescheduling} = \text{Checking} + \text{Repairing} + \text{Improving} \quad (1)$$

An algorithm has been reported [45] in which an initial schedule is generated and continuously rescheduled in an infinite loop to accommodate the disruptions. Thus, active scheduling can also be defined as

$$\text{Active scheduling} = \text{Predictive scheduling} + \text{Continuous rescheduling} \quad (2)$$

Active scheduling is a completely automated rescheduling concept that makes it more reactive than other approaches. It handles multiple events simultaneously and also performs an overall quality check of the repaired schedule while performing local repair. The process, however, lacks a module for a human interface that is essential for expert advice. It also lacks a case base for if–then analysis, which could be exploited to react to various disruptions.

Fuzzy logic has also been reported in the literature for reactive schedule repair [48–51]. The fuzzy processing times are substituted for the crisp processing times used by heuristic-based scheduler. When jobs are scheduled in order of their criticality, some constraints may be violated. Crisp processing times are replaced by fuzzy possibility distributions, and temporal constraint violations are expressed as a matter of degree. The extent to which the due date is met, is measured by the degree of overlap between the due date and the fuzzy estimate of the finishing time of the job. Threshold values can be defined for acceptable or unacceptable degrees of constraints. If the threshold value is not met, the schedule requires rescheduling or repair by progressive interchange of jobs. Uncertain events or disruptions will lead to constraint violations, and these are then evaluated by progressive shuffling of jobs. In this manner, the fuzzy repair strategy is applied until the violated constraints are satisfied [48]. This approach has the advantage of a complete scan of the schedule for constraint violations every time a new event is integrated, and the schedule is optimised globally during the repair. In addition, high variabilities in the processing times can be accommodated easily using this approach. Another advantage is that the same process can be used for both the generation of robust predictive schedules and the subsequent schedule recovery [49]. On the other hand, since fuzzy logic is a hill-climbing search methodology, the repair could frequently be localised, leading to poor quality in terms of overlaps in processing times or unsatisfied constraints.

Case-based reasoning, [28,35,52,53] has also been used for reactive scheduling and the goal is to find a case that best suits the disrupted schedule. Schedules are often complex, and to model all details of the schedule or the production environment in a case is not feasible. Instead, some characteristic surface knowledge is captured in a case. This abstraction of the problem also supports the application of the case for a new problem. The case database is created by domain experts and is capable of specifying the correct reaction to schedule disruptions. Gradually, the cases are refined and a better knowledge base is created. The implementation of case-based reasoning in an experimental set-up for schedule correction has been reported [35] in which case-based reasoning is combined with an iterative improvement method using fuzzy constraints and a tabu search for repair. Case-based reasoning combines the advantages of both heuristics and experience modelling, as the case contains explicit heuristics for a given problem and it uses the solutions to old cases to solve a new problem. Another advantage of the approach is the possibility of finding solutions to problems that have never been encountered before using analogous cases from the case base. However this approach is not very responsive, as it has to search through an extensive case base [35].

The concept of *constraint-based scheduling* has been reported [28,54–55] for schedule repair of a prototype scheduling system (CABINS). The incremental accumulation and reuse of past experience is achieved through case-based reasoning, whereas constraint-based scheduling has been used for the propagation and resolution of the effect of repair. The dominant constraints satisfied during the schedule recovery are activity precedence and resource capacity. Experimental results have been reported to show CABINS being able to capture and effectively use the user's scheduling preferences to outperform other scheduling methods in both predictive schedule generation and the reactive response to unpredictable events. Constraint-based scheduling models use "spreadsheet" or "blackboard" style architecture that provide better interaction for "what if" analysis and interactive decision support between the system and the user. The advantage is that the user has direct control of the repair action formulation and performs an active role in the decision-making in choosing the repair strategy. CABINS

also shows far better responsiveness and better schedule quality when compared to the case-based reasoning approach [55].

Artificial intelligence techniques such as neural networks [56, 57] and genetic algorithms [58] have also been used in schedule recovery. Neural networks have been applied in the recovery of the production schedule of glass coating (AIRCO process) [56] in which rich control strategies for reactive scheduling have been identified. The purpose is to maintain control over the environment where the predetermined schedule has been interrupted, because of an urgent order, maintenance, or reassignment of job priorities. The training of a neural network is usually performed in a single pass and in some cases excessive re-training time can prove disadvantageous [57]. A trained neural network is used to predict the repair strategy by extrapolating the acquired knowledge, which is contained in an extensive knowledge database. For accurate predictions, appropriate training sets are required for each combination of circumstances. The response time of the trained neural network is very fast and predictions are usually reliable, making it a suitable tool for reactive repair. Neural networks are also advantageous as they can learn and grow with the growth in the knowledge database and accumulation of experience. Genetic algorithms mimic the natural selection process of genes to perform schedule repair. Crossover and mutation operators are used to generate successive population of better schedules [58] to accommodate the deviations in the original schedule. Genetic algorithms are efficient and produce nearly optimum schedules but require a high computational effort.

4. Discussion

Reactive scheduling techniques currently available do not consider all types of disruptions that occur in shop floors. In the approaches discussed in the previous section, most of the work in schedule recovery has concentrated on urgent job arrivals and machine breakdowns. Other factors of uncertainty in the shop floor have yet to be modelled and considered in the schedule recovery approaches. A prototype model REAKTION [59] has been proposed to consider various types of disruption that may be external in nature (e.g. short-term acceptance of a high-priority order or delay in material delivery) or internal in nature (e.g. machine intensity, change of start time, change of priority and order splitting). In order to have a fully automated rescheduler in an on-line reactive environment, it is necessary to model and repair all types of disruption and event that can occur in a stochastic job shop. The approaches capable of handling independent repairs and continuous quality improvement hold better prospects for reactive scheduling and repair.

Another area in which the current schedule recovery approaches are deficient is interaction with the human scheduler. It would be useful to have a tool that explores the benefit of real-time integration and interaction between the human scheduler and the scheduling system so that the human scheduler's experience and spontaneous intelligence are exploited. The need for such an interactive scheduler has been proposed [60] and the schedule recovery system is required to possess the ability to generate automated schedules, and the ability to react to unexpected events using a schedule editor and a schedule recovery module. An intelligent supervisory function assisted by a human scheduler for revising an active but flawed schedule has also been proposed [52].

In the future, schedule recovery must play a stronger role in the development of the predictive reactive schedule that has continuous off-line and on-line control of the shop floor. A few expert models like REAKTION and CABINS that have been proposed are currently in the development stages. The deficiency of most of the reactive recovery approaches lies in the handling of varied types of deviation and in the active involvement of the human agent. The authors are therefore proposing a new model that supports the integration of the

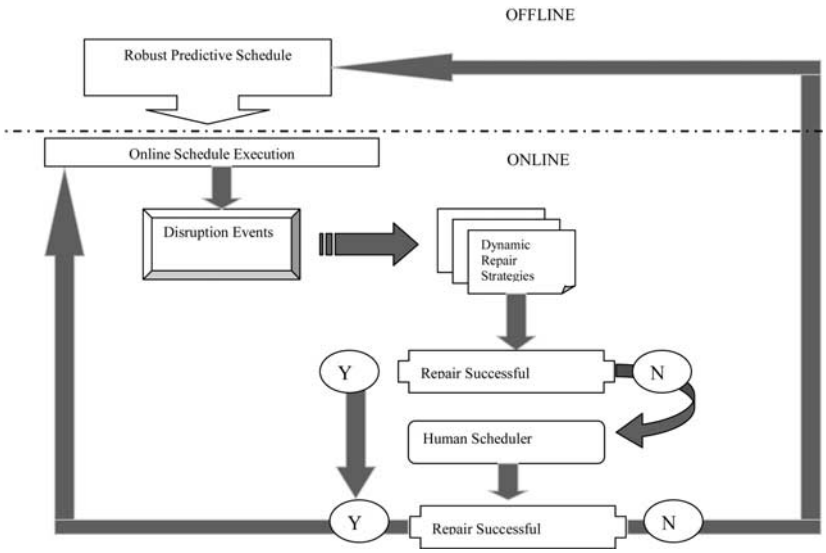

Fig. 4. Architecture of proposed system (APRS).

predictive schedule with multiple schedule recovery techniques in order to generate an iterative scheduling environment that can amalgamate off-line schedule generation and on-line schedule correction policies. In addition, a versatile interface with the human scheduler is planned to exploit his experience in schedule correction. A closer control of the quality of the repaired schedule and high responsiveness will be emphasised in the development of the model.

In the proposed future model, AutoSHARP (Automated Scheduling and Human Assisted Reactive Planner), a robust predictive schedule will be generated during off-line planning. This predictive scheduler is a scheduling algorithm that optimises multiple performance criteria. The schedule is robust, as artificial time gaps are inserted to accommodate minor deviations during the generation of the schedule. The schedule is then executed on the shop floor. As soon as a disruption on the shop floor is encountered, an algorithm is triggered that measures the severity of the disruption. If the disruption can be accommodated in the time allowances of the robust schedule, no action is required. Otherwise, the repair agents are used for the modification of the flawed schedule.

A significant step forward fig4 will be to make provisions in the repair agents for considering various types of disruption. Depending on the type of disruption, the specific agent is activated to invoke an event-specific strategy. Thus, the schedule recovery in AutoSHARP can be viewed as a continuous process in which agents are busy improving the schedule and integrating the new events as they arrive. The system checks every event to see if it violates the basic constraint and tries to reach a solution. The resultant schedule is checked for schedule quality. If a successful repair is achieved, the revised schedule replaces the original schedule and thereafter execution restarts. The time frame in which the repair is to be performed is critical, and is required to be as short as possible. If the repair is not satisfactory, the supervisory function of the human scheduler is activated. The human scheduler can provide a solution, based on experience when all other repair strategies fail. The human scheduler can also overrule some decisions and alter the schedule when necessary. Rescheduling is triggered only if all the repair strategies fail to give a feasible schedule and the intervention of the human scheduler is ineffective in salvaging the original schedule.

5. Conclusions

Reactive scheduling of a job shop using various approaches for schedule recovery and repair has been extensively reported. In this paper, the various methods have been summarised and compared on common measures. In addition, the factors that have been lacking in the current work have been identified and we have proposed a unified methodology, AutoSHARP, which is aimed at resolving the deficiencies in the existing approaches. The focus is to develop a new engine to integrate predictive and reactive scheduling using repair strategies based on multiple approaches to handle different events and to exploit the spontaneous intelligence and experience of a human scheduler. Such a system must be developed to consider the scheduling needs of the dynamic and stochastic job shop on a continuous basis, and provide qualitative schedules at all times. Reactive schedule recovery has matured as a concept, but it is strongly felt that in order to evolve as a successful shop floor tool, further effort is required in the development of a generic system that has inbuilt alternatives for handling the dynamics of the job shop under different problem scenarios and shop floor practices.

References

1. R. O'Donovah, R. Uzsoy and K. N. Mckay, "Predictable scheduling of single machine with break down and sensitive jobs", International Journal of Production Research, 37(18), pp. 4217–4233, 1999.
2. R. Shafaei and P. Brunn, "Workshop scheduling using practical (inaccurate) data. Part 2: An investigation of the robustness of the scheduling rules in a dynamic and stochastic environment", International Journal of Production Research, 37(18), pp. 4105–4117, 1999.
3. E. Szelke, and R. M. Kerr, "Foreword", IFIP Transactions B (Applications in Technology), B-15, p. 1, 1994.
4. D. Biskup and W. Piewitt, "A note on an efficient algorithm for the single machine tardiness problem", International Journal of Production Economics, 66, pp. 287–292, 2000.
5. K. Kim and P. J. Egbelu, "A mathematical mode for job shop scheduling with multiple process plan consideration per job", Production Planning and Control, 9(3), pp. 250–259, 1998.
6. K. Haase and A. Kimms, "Lot sizing and scheduling with sequence dependent set-up costs and times and efficient rescheduling opportunities", International Journal of Production Economics, 66, pp. 159–169, 2000.
7. M. Singer, "Forecasting policies for scheduling a stochastic due date job shop", International Journal of Production Research, 38(15), pp. 3623–3637, 2000.
8. S. O. Duffuaa, A. Raouf, M. Ben-Daya and M. Makki, "One machine scheduling to minimize mean tardiness with minimum number tardy", Production Planning and Control, 8(3), pp. 226–230, 1997.
9. H. Iima, R. Kudo, N. Sannomiya and Y. Kobayashi, "An autonomous decentralized scheduling algorithm for scheduling problem in a metal mould assembly process", Journal for Intelligent Manufacturing, 10, pp. 161–167, 1999.
10. S. Mohri, T. Masuda and H. Ishii, "Bi-criteria scheduling problem of three identical parallel machines", International Journal of Production Economics, 60–61, pp. 529–536, 2000.
11. M. K. Reeja and C. Rajendran, "Dispatching rules for scheduling in assembly job shops – Part 1", International Journal of Production Research, 38(9), pp. 2051–2066, 2000.
12. P. R. Philopoom, "The choice of dispatching rules in the shop using internally set due dates with quoted lead-time and tardiness costs", International Journal of Production Research, 38(7), pp. 1641–1655, 2000.
13. R. Cigolini, A. Comi, A. Micheletti, M. Perona and A. Portioli, "Implementing new dispatching rules at SGS-Thomson Microelectronics", Production Planning and Control, 10(1), pp. 97–106, 1999.
14. O. Holthaus and C. Rajendran, "Efficient job shop dispatching rules: further developments", Production Planning and Control, 11(2), pp. 171–178, 2000.
15. S. Barman, "Simple priority rules combinations: an approach to improve both the flow time and tardiness", International Journal of Production Research, 35(10), pp. 2857–2870, 1997.
16. M. K. Reeja and C. Rajendran, "Dispatching rules for scheduling in assembly job shops – Part 2", International Journal of Production Research, 38(10), pp. 2349–2360, 2000.
17. E. Hart and P. Ross, "A heuristic combination method for solving job shop scheduling problems", PPSN V, Lecture Notes in Computer Science, 1498, pp. 845–854, 1998.
18. H. Pierreval and N. Mebarki, "Dynamic selection of dispatching rules for manufacturing system scheduling", International Journal of Production Research, 35(6), pp. 1575–1591, 1997.
19. V. A. Armentana and R. Mazzini, "A genetic algorithm for scheduling on a single machine with set-up times and due dates", Production Planning and Control, 11(7), pp. 713–720, 2000.
20. L. P. Khoo, S. G. Lee and X. F. Yin, "A prototype genetic algorithm enhanced multi-objective scheduler for manufacturing systems", Inter-

national Journal of Advanced Manufacturing Technology, 16, pp. 131–138, 2000.
21. A. El-Bouri, S. Balakrishnan and N. Popplewell, "Sequencing jobs on a single machine; a neural network approach", IFIP Transactions B (Applications in Technology) B-15, pp. 39–55, 1994.
22. M. Kuroda and Z. Wang, "Fuzzy job shop scheduling", International Journal of Production Economics, 44, pp. 45–51, 1996.
23. V. A. Armentano and D. S. Yamashita, "Tabu search for scheduling on identical parallel machines to minimize mean tardiness", Journal for Intelligent Manufacturing, 11, pp. 453–460, 2000.
24. V. A. Armantano and R. C. Scrich, "Tabu search for minimizing total tardiness in a job shop", International Journal of Production Economics, 63, pp. 131–140, 2000.
25. S. G. Ponnambalam, P. Aravindan and S. V. Rajesh, "Tabu search algorithm for job shop scheduling", International Journal of Advanced Manufacturing Technology, 16, pp. 765–771, 2000.
26. S. G. Poonambalam, N. Jawahar and P. Arvindan, "A simulated annealing algorithm for job shop scheduling", Production Planning and Control, 10(8), pp. 767–777, 1999.
27. T. Satake, K. Morikawa, K. Takaahashi and N. Nakamura, "Simulated annealing approach for the minimizing the make span of the general job shop", International Journal of Production Economics, 60–61, pp. 515–522, 1999.
28. K. P. Sycara and K. Miyashita, "Adaptive schedule repair", IFIP TC5/WG5.7 International Workshop on Knowledge-Based Reactive Scheduling, IFIP Transactions B (Applications in Technology), B-15, pp. 107–123, 1994.
29. R. Shafaei and P. Brunn, "Workshop scheduling using practical (inaccurate) data – Part 1: The performance of the heuristic scheduling rules in a dynamic job shop environment using a rolling time horizon approach", International Journal of Production Research, 37, pp. 3913–3925, 1999.
30. E. J. Yelligi and G. T. Mackulak, "Robust deterministic scheduling in the stochastic environments: the method of capacity hedge points", International Journal of Production Research, 35, pp. 369–379, 1997.
31. S. V. Mehta and R. Uzsoy, "Predictable scheduling of a single machine subject to the breakdowns", International Journal of Computer Integrated Manufacturing, 12, pp. 15–38, 1999.
32. I. M. Ovacik and R. Uzsoy, "Rolling horizon algorithms for single machine dynamic scheduling problem with the sequence dependent setup times", International Journal of Production Research, 32, pp. 1243–1263, 1994.
33. I. M. Ovacik and R. Uzsoy, "Rolling horizon procedure for dynamic parallel machine scheduling with sequence dependent setup times", International Journal of Production Research, 33, pp. 3173–3192, 1995.
34. K. Bhaskaran and M. Pinedo, "Dispatching", Chap. 83, in G. Salvendy (ed.), Handbook of Industrial Engineering, Wiley, New York, 1991.
35. J. Dorn, "Case based reactive scheduling", Chap. 4, in Artificial Intelligence in Reactive Scheduling, Chapman and Hall (UK), pp. 32–50, 1994.
36. J. Efstathiou, "Anytime heuristic schedule repair in manufacturing industry", IEE Proceedings: Control Theory and Applications, 143(2), pp. 114–124, 1996.
37. J. Efstathiou, "Formalizing the repair of schedule through knowledge acquisitions", Proceedings, Advances in Knowledge Acquisition, 9th European Knowledge Acquisition Workshop, EKAW '96, pp. 306–320, 1996.
38. J. Efstathiou, "Automated schedule repair on the shop floor", IIA'96/SOCO'96. International ICSC Symposia on Intelligent Industrial Automation and Soft Computing, pp. A44–A50, 1996.
39. J. T. Lee, Y. S. Oh, G. S. Jo, "Minimum adjustment for repairing an initial solution in reactive scheduling", Journal of KISS (B) (Software and Applications), 25 (6), pp. 923–930, 1998.
40. V. J. Leon, S. D. Wu and R. H. Storer, "Robustness measures and robust scheduling for job shop". IEE Transactions, 26(5) pp. 32–43, 1994.
41. R. J. Abumaizar and J.A Svestka, "Rescheduling job shops under random disruptions", International Journal of Production Research, 35(7), pp. 2065–2082, 1997.
42. P. Brandimarte, M. Rigodanza and L. Roero, "Conceptual modeling of an object oriented scheduling architecture based on the shifting bottleneck procedure", International Journal of Production Research, 35(7), pp. 2065–2082, 1997.
43. V. J. Leon, S. D. Wu and R. H. Storer, "Games theoretic control approach for job shops in the presence of disruptions", International Journal of Production Research. 32 (6), pp. 1451–1476, 1994.
44. G. Hasle and S. F. Smith, "Directing an opportunist scheduler: an empirical investigation on reactive scenario", Chap. 1, Artificial Intelligence in Reactive Scheduling, Chapman and Hall (UK), pp. 1–11, 1994.
45. H. Henseler, "From reactive to active scheduling by using multi agents", Chap. 2, Artificial Intelligence in Reactive scheduling, Chapman and Hall (UK), pp. 12–18, 1994.
46. E. Szelke and G. Markus, "A black board based perspective of reactive scheduling", Chap. 6, Artificial Intelligence in Reactive scheduling, Chapman and Hall (UK), pp. 60–77, 1994.
47. R. J. Rabelo and L. M. Camarinha-Matos, "A holistic control architecture infrastructure for dynamic scheduling", Chap. 7, Artificial Intelligence in Reactive scheduling, Chapman and Hall (UK), pp. 78–94, 1994.
48. J. Dorn, R. Kerr and G. Thalhammer, "Reactive scheduling in a fuzzy-temporal framework", Knowledge Based Reactive Scheduling, IFIP Transactions B (Applications in Technology), B-15, pp. 39–54, 1994.
49. G. Schmidt, "How to apply fuzzy logic to reactive production scheduling", Knowledge Based Reactive Scheduling, IFIP Transactions B (Applications in Technology), B-15, pp. 57–66, 1994.
50. J. Dorn, R. Kerr and G. Thalhammer, "Reactive scheduling: improving the robustness of schedules and restricting the effects of shop floor disturbances by fuzzy reasoning", International Journal of Human–Computer Studies, 42(6), pp. 687–704, 1995.
51. W. Slany, "Scheduling as a fuzzy multiple criteria optimization problem", Fuzzy Sets and System, 78(2), pp. 197–222, 1996.
52. K. Miyashita and K. P. Sycara, "Exploiting failure information for the case based schedule repair", Journal of Japanese Society of Artificial Intelligence, 9(4), pp. 559–568, 1994
53. E. Szelke and G. Markus, "A learning reactive scheduler using CBR/L", Computer in Industry, 33(1), pp. 31–46, 1997.
54. K. Miyashita, "Case based knowledge acquisition for schedule optimization", Artificial Intelligence in Engineering, 9(4), pp. 277–287, 1995.
55. J. E. Spargg, G. Fozzard and D. J. Tyler, "Constraint based reactive rescheduling in a stochastic environment", Proceedings, Recent Advances in AI Planning, 4th European Conference on Planning, ECP'97, pp. 403–413, 1997.
56. B. J. Garner and G. J. Ridley, "Application of neural network process models in reactive scheduling", Knowledge Based Reactive Scheduling, IFIP Transactions B (Applications in Technology), B-15, pp. 19–28, 1994.
57. G. A. Rovithakis, S. E. Perrakis and M. A. Christodoulou, "Application of a neural network scheduler on a real manufacturing system", IEEE Transactions on Control Systems Technology, 9(2), pp. 261–270, 2001.
58. J. G. Qi, G. R. Burns and D. K. Harrison, "The application of the parallel multi population genetic algorithms to dynamic job shop scheduling", International Journal of Advanced Manufacturing Technology, 16, pp. 609–615, 2000.
59. H. Henseler, "REAKTION: a system for event independent reactive scheduling". Chap. 3, Artificial Intelligence in Reactive Scheduling, Chapman and Hall (UK), pp. 19–31, 1994.
60. P. Jordan, J. Browne and M. Browne, "Production activity control for small manufacturing enterprises", Knowledge Based Reactive Scheduling, IFIP Transactions B (Applications in Technology), B-15, pp. 19–28, 1994.