Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

SIMULATION ASSISTED OPTIMIZATION AND REAL-TIME CONTROL ASPECTS OF FLEXIBLE PRODUCTION SYSTEMS SUBJECT TO DISTURBANCES

Wilhelm Dangelmaier
Kiran R Mahajan
Thomas Seeger
Benjamin Klöpper
Mark Aufenanger

Heinz Nixdorf Institute
Business Computing, esp. CIM
University of Paderborn
Fürstenallee 11, 33102 Paderborn, GERMANY

ABSTRACT

Several types of production systems have been studied and researched in the past using either simulation and\or optimization methods. In this paper we describe the design and development of a simulation assisted predictive-reactive system for scheduling and rescheduling a typical flexible production system configuration. Aspects like the combined use of simulation and optimization to solve complex scheduling and rescheduling tasks are described in view of system stability and some of the broader production system elements like buffer sizing and material handling equipment. Results show that combining simulation and optimization for predictive scheduling resulted in better and valid performance measures for a typical example. Results also show that some newly addressed aspects of stability and real-time control can be handled efficiently using a combination of simulation and optimization. Our discussions only bolster the claim that simulation is an indispensable tool in managing complex production systems.

1 INTRODUCTION

Production systems are getting more and more complex these days. The complexity is due to several factors like complex production systems, product variety and uncertain business processes. Efficient techniques for planning and re-planning the entire supply chain to cope with the complexity and business dynamics are more and more apparent. Within the manufacturing and distribution supply chain, production scheduling and re-scheduling is one area, which will be vital to the success of the manufacturing organization. Especially, day to day operations and the disturbances arising out of uncertain characteristics of the system have to be handled efficiently.

As one could imagine there are several configurations of flexible production systems, each with its own characteristics. In this report, we seek to address scheduling and control of one such flexible production system configuration – the flexible flow shop or more specifically the flow shop with parallel machines (FSPM) at one or more stages. Mathematically, this problem is a combination of the classical flow shop scheduling problem and the flow shop with multiple processors at some stages. The application of this type of problem occurs more often than one would imagine. Many high volume production facilities have several separate flow shops. The process in such facilities is such that machines are flexible or interchangeable at each stage and therefore practically similar. Some production facilities also have special expertise in machining a family of parts, where each part follows the same sequence, but each machine is flexible to accommodate the slight variety in parts. Assembly lines, in which more than one type of product may be manufactured and each work station has multiple machines, is also an obvious application of this problem. Similarly, the situation where a parallel machine is added to relax strain on a bottleneck facility, and\or to increase production capacities can be viewed as an application of the suggested problem.

For such kind of system configuration, several researchers have developed heuristic methods. However, these methods do not consider the broader production system elements and there do not yet exist re-scheduling methods for real-time control. Besides this, the ability to schedule such systems considering flexible and fixed part flows (mixed flows) is not available yet.

Comparatively inadequate amount of work has been published in the area of the parallel machine flow-shop scheduling problem. Brah and Hunsucker (1991) have developed a branch and bound algorithm for the FSPM problem. This algorithm was noted to have worked consistently

with a fair amount of computational speed for medium sized problems. For large sized problems, further improvements were suggested. More recently, Cheng et al. (2001) developed a heuristic using a combination of the property of reversibility of the FSPM problem and the shifting bottleneck procedure. Phadnis and Irani (2003) developed such an algorithm for the FSPM problem using the shifting bottleneck procedure, and have compared their results with other algorithms, namely that of Cheng et al. and others. Improvements were shown again for improving optimality. As discussed earlier, the algorithms are exact in the sense that they use only information on machines, jobs and processing times to compute schedules. They do not encompass the other elements of a production system, like buffer sizing, transportation systems and the like. In our opinion, the result of these algorithms could be considered theoretical at best because of the fact that we do not know if they can be executed exactly as computed in the real world. Besides this, the problem of how to schedule and re-schedule in the presence of deterministic and random disturbances and jobs with flexible or fixed routes is still not answered by all the previous research.

Over the last decades, a significant volume of research on the issues of scheduling with executional uncertainties has begun to emerge for several system configurations. Haldun et al. (2005) present an excellent overview of this research, some of which combined with our opinion are discussed next. They mention approaches like completely reactive approaches, robust scheduling approaches and predictive-reactive scheduling. The latter is by far the most studied, and we therefore examine a number of specific issues related to this approach in more detail, some of which are addressed in this paper. Completely reactive approaches are characterized by least commitment strategies such as real-time dispatching that create partial schedules based on local information. This approach has many practical advantages. Its computational burden is in general extremely low, and rules are usually intuitive and easy to explain to users.

A natural extension of the dispatching approach is to allow the system to select dispatching rules dynamically as the state of the shop changes. Early work in this area is that of Wu and Wysk (1989), who examine the problem of dispatching rule selection in the flexible production system environment. They divide the time horizon into shorter intervals. At the beginning of each interval a variety of dispatching rules are simulated, and the rule that yields the best performance is selected and implemented for the next time period. However, for complex systems with relatively low uncertainty, global scheduling instead could have the potential to significantly improve system performance compared to localized or myopic dispatching. Secondly, we believe that when the frequency of disturbances is high, then management efforts could be better spent on reducing those uncertainties, than developing complicated scheduling logic.

A number of Authors have extended this approach in various ways. Harmonosky et al. (1997) present their work in the areas of real-time selective re-routing and scheduling algorithms based on simulation. They iteratively use simulation as a tool to find out the best policy from a set of alternative policies in real-time. Using simulation as a tool for real-time scheduling presents several benefits and drawbacks. The most important benefit is that simulation can provide accurate information about a certain policy. Secondly, simulation proves to be effective when the system behavior is probabilistic, and cannot be easily captured by analytic methods. The ability to provide accurate information about a certain policy becomes a drawback when it comes to modeling a complex production system. As the underlying production system gets bigger encompassing other elements, and as the number of "decision points" (location where a decision to handle a part has to be taken) increase with each point providing several alternatives (due also to part variety), the more difficult it is to employ simulation, especially in real-time to obtain information about the best alternative policy. Other aspects that cannot be considered by using a purely simulation based approach are discussed in the third last paragraph of this section.

In the areas of robust scheduling approaches, probable disturbances are considered into the planning phase, so that the effect of executional uncertainties are reduced. One way in which this is done is to minimize the expected degradation in performance measure (Leon et al. 1993; Wu et al. 1999), where the degradation is measured as the difference in objective function value between the predictive and realized schedule. Leon et al. (1993) also include a number of re-configuration related costs, such as cost of changes in the start times and the cost of sequence changes. These approaches do not explicitly consider execution issues, since the formulation accounts for the fact that there will be disruptions prior to the execution of the schedule. The issue of predictability and graceful transition from a current system state is thus not considered.

Predictive-reactive scheduling is presented as a two-step process. First, a predictive schedule representing the desired behavior of the shop floor over the time horizon considered is generated. This schedule is then modified during the execution in response to unexpected disruptions. The schedule that is actually executed on the shop floor after these modifications is called realized schedule. The two main questions are when to initiate a rescheduling action and what the rescheduling action should be.

Church and Uzsoy (1992) provide a rough taxonomy of existing approaches of when to reschedule, beginning with two extremes. Continuous rescheduling approaches take rescheduling action each time an event that is recognized by the system, such as the arrival of a new job, occurs. Periodic rescheduling, on the other hand, defines a basic interval T between rescheduling actions during which rescheduling actions are not permitted. Finally, they define

event-driven rescheduling, in which a rescheduling action can be initiated upon the recognition of an event with potential to cause significant disruption to the system. Both continuous and periodic rescheduling can be viewed as special cases of event-driven rescheduling.

Clearly, continuous rescheduling runs the risk of initiating rescheduling activity in the face of events that do not cause significant disruption, expending computational resources and potentially causing unnecessary changes in the schedule with associated poor effects on the shop floor. The obvious drawback of periodic rescheduling is that it ignores events occurring between rescheduling points, which in an extreme case may render the current schedule impossible to execute, and in less serious situations runs the risk of yielding poor schedules. Hence, a combination of the periodic and event driven approaches appears attractive, in which a periodic rescheduling approach is implemented, but rescheduling activity can be invoked between rescheduling points if a disruption that is deemed sufficiently serious is observed. This latter approach is more commonly observed where schedules are often developed for some base horizon, such as a day or a shift, but are modified as needed during that period.

In our opinion, there is something more to the aspect of when to re-schedule. The problem is which point in time to re-schedule\start making changes after the occurrence of a disturbance event or a deviation? Another problem is how much to re-schedule? Further questions are, how can we deviate from a predictive plan as less as possible? In case of deviations from a predictive plan, how can we bring back the system to its original\planned trajectory? How can we reschedule as less as possible, to result in system stability? We define system stability in terms of job starting time deviations and job sequence deviations. System stability is important because changing from a predictive plan asks for energy and resources from the management, especially in systems where materials and supplies are made available Just-In-Time. But most importantly, we also want to ask, is rescheduling really required? If yes, how can we make sure that future execution is not affected as a result of this rescheduling action?

In this paper, we describe a simulation assisted predictive-reactive system for scheduling and rescheduling of a typical complex environment to answer the above questions. In both phases of the system, simulation is combined with optimization. We think, it is best to employ a combination of simulation and optimization algorithms (for scheduling and re-scheduling) as this will reduce the computational burden on simulation, consider the effects of the broader system elements, besides using simulation to provide a problem free execution of the predictive plans. Optimization can reduce drastically the options to simulate, while simulation can be used to check the validity of the schedule, and most importantly, to improve or fine tune the schedule.

We show here how simulation is an vital tool in the planning phase as well as the rescheduling phase. We also show that simulation provides answers to some difficult problems, said to be at least NP-hard by others (Phadnis, Irani (2003)). We mainly describe our concepts, some prototype software and experimental results on simulation assisted predictive scheduling and rescheduling to address the problems we discussed earlier. Our results are promising. In the next section, we describe the overall framework of our system. In Section 3 we describe the implementation and results. Our conclusions are presented in Section 4.

2 THE OVERALL FRAMEWORK

Based on the problem descriptions of previous section, we now describe our concepts in details. Figure 1 shows the overall concept of our simulation assisted predictive-reactive approach for production scheduling and rescheduling. The predictive system develops a schedule\plan to implement in the next planning horizon or shift. As the execution of the plan proceeds in the next shift, the reactive system is used.

2.1 The predictive system

To begin with, the user first models the production system with all the deterministic parameters using standard and custom built object libraries. Here, we let the user define the alternative control policies at so called “decision points” in the simulation model, along with modeling other elements. The user also models several executional conditions at each decision point that could be expected to occur when implemented in the real world. The user models these conditions based on his expert knowledge of the system. For each decision point a rule generator is implemented for handling various conditions arising at the decision point during simulation run-time. All this information is modeled during simulation modeling using customized object libraries and standard simulation functions, both of which are implemented and used for optimization.

The predictive system works in two steps. In the first step, data required for optimization (processing times, jobs, machines, stages) is passed on to the optimization algorithm object, which acts like a black box. The optimization algorithm may compute a semi-feasible plan (one which cannot be completely executed as generated without problems) or a predictive schedule to be used in the system with sub optimal makespan Key Performance Indicator (KPI), which includes part routing\sequencing decisions. Current system status is considered in terms of earliest available machine times. During the optimization process, known events like machine maintenance schedules, material unavailability, fixed job routings in the system and other problems are considered (see Figure1) that are perceived to occur in the next period (for which the schedule

is being developed). The optimization algorithm works by determining bottleneck stages, and sequencing each job with longest tails at the earliest available machines on the bottleneck stage through all the stages. The system also has a section to accommodate due date conformance for jobs with fixed routings using iterative simulation in addition to makespan KPI. The reason for using simulation to assist in due date conformance is the fact that the problem in hand is atleast N-P hard to be solved by an algorithm optimally. Note that this semi-feasible plan may result in a performance measure value (makespan) which could be lower than the one obtained in the real world. This is because the optimization algorithm does not consider other elements like buffer sizes, other material flow objects, etc in its computations. This degree of increase in value of the performance measure in the real-world will depend on the travelling times, waiting times for other material flow objects, and the number of resources available to serve the machining stations. This sub optimal schedule is given back to the simulation function (see Figure 2) to complete the second step of the predictive phase.

In the second step, the partial schedule obtained from the optimization algorithm, is simulated and further analyzed for feasibility and optimality using a Flow Analyzer Module (FAM), shown in Figure 2. Analysing feasibility will cover problems that may occur during the real execution of the schedule obtained from the optimization algorithm, specifically problems like bottlenecks, and the effect of including buffer sizing and other elements of the production system. Analysing optimality will cover assessing the result of the optimization algorithm. This is done as follows. The plan is simulated once completely to the end, and the results are stored in a database. This will result in KPI values including the effect of problems, buffer sizing, etc that might occur during the simulation run. Then the simulation is run for a second time with the Flow Analyzer Module activated. This analyzer will generate its own rules (with the same objectives as that of the optimization algorithm) for routing jobs based on conditions that occur during the simulation run. As explained earlier, these conditions are modeled by the user during the initial modeling process. As an example, if step 1 of the predictive system computed that job 1 on decision point 3 has to take decision (route) 4, then the FA component will analyze the situation at the moment (for feasibility or optimality) and if required over-ride the decision of the optimization algorithm, to result in say decision (route) 5.

Each job at each decision point will be analyzed for the current situation and condition, and where required the results of the optimization algorithm over-ridden. The rules within the FAM component are largely based on waiting times for jobs within the system. When the entire plan is analyzed and simulated in such a way, the second simulation run stops and the user is presented with the results of both simulation runs. The first simulation will show the result of the optimization algorithm and the second simulation will show the result of the combination of the optimization algorithm and the FAM.

The optimization algorithm primarily seeks to reduce complexity due to the alternative policies by using improvement procedures in the first step, whilst the simulation function and the FAM component (in the second step) serves to improve the feasibility and optimality of the schedule and to obtain better and actual performance measures. The simulation function also serves the purpose of real-time (re)scheduling (explained later). Of course, the end result of combining optimization and simulation would also depend on how effective the optimization algorithm is.

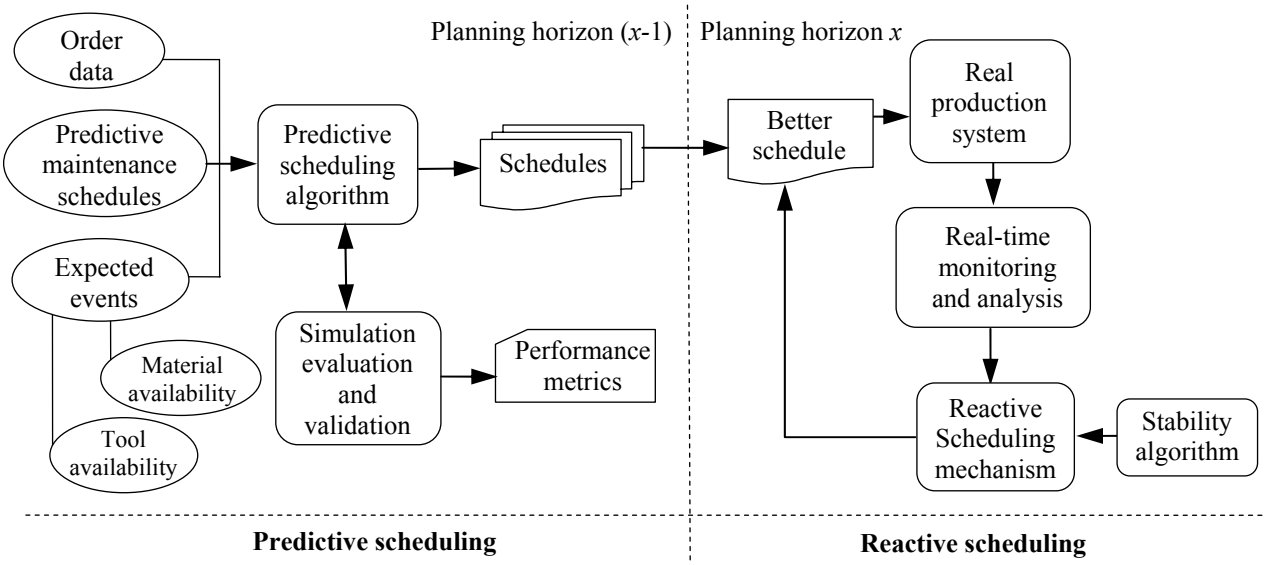

Figure 1: Overview of the Simulation Assisted Production Scheduling and Rescheduling System

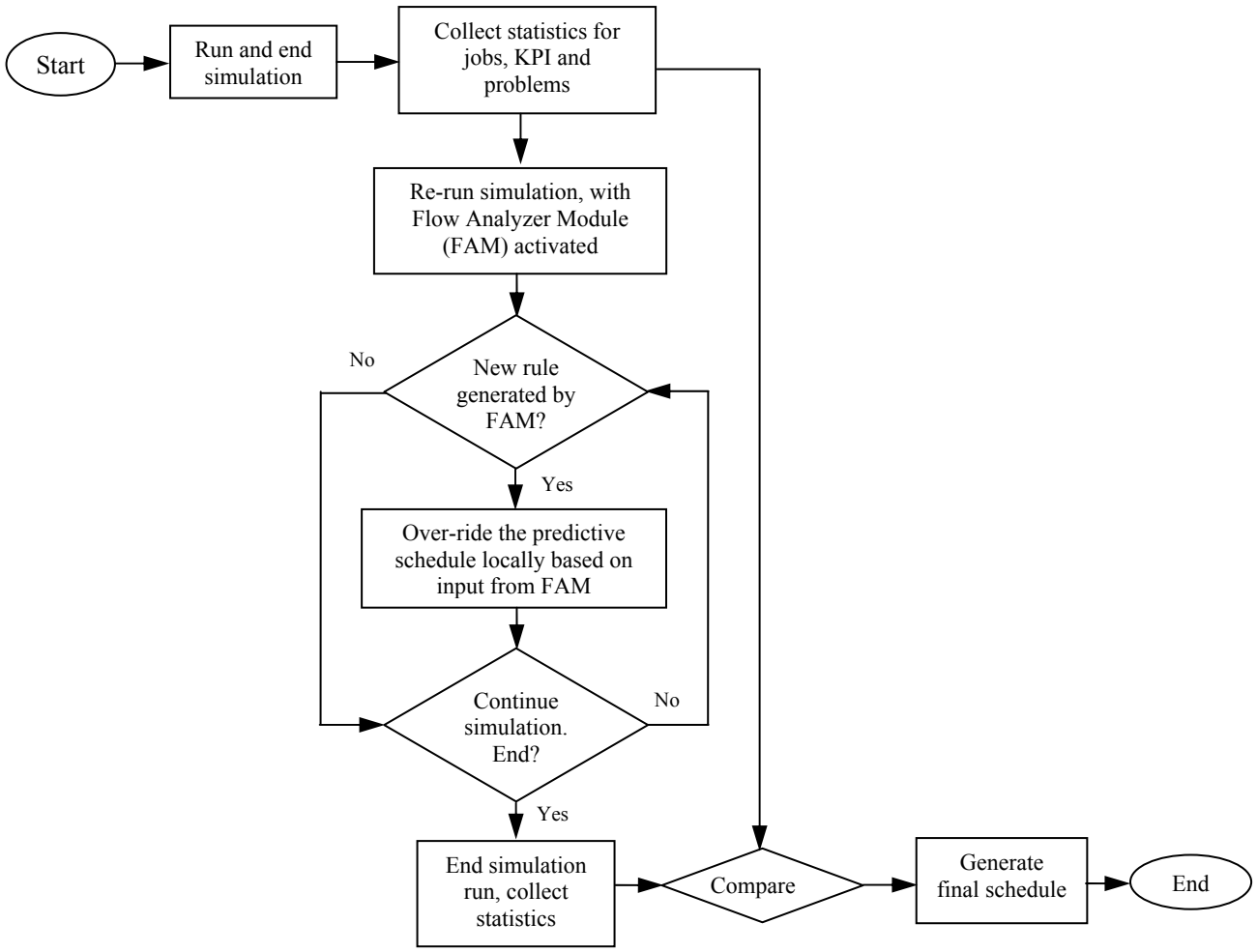

Figure 2: Combining Simulation and Optimization to Achieve Validity and Optimality of the Predictive Schedule

In any case, if the optimization algorithm provides a good solution, the combination with simulation will improve the solution further. This method of combining simulation and optimization for predictive scheduling makes sense because it solves a problem before they occur in the real world, whilst also improving the schedule and providing actual system performance measures. This has been quantified later in Section 3 of the paper.

2.2 The reactive system

The plan obtained in the predictive phase is executed in the shift under consideration. During execution, as soon as there is a disturbance, the rescheduling mechanism is activated. The criteria for activating the reactive system are set during the predictive planning phase itself. The disturbances we consider so far are process disturbances. Anything to do with new order arrival would be dealt with by the predictive system. Here too, a combination of simulation and optimization is suggested for the same reasons mentioned in the previous section. Figure 3 shows the method. The method is somewhat similar to the one proposed by Chong et. al (2003), but differs significantly in operating principles.

In this system, the real-time monitoring and control module receives events from the real production system. Upon occurrence of a disturbance, the simulation evaluation function is activated which simulates the result of the disturbance to compute the effect of the disturbance. This will be the upper bound of the system. The control is then passed on to the rescheduling optimization algorithm. The rescheduling optimization algorithm is divided in two sub algorithms both achieving some or all aspects of stability with different aims. These algorithms are the selective rescheduling algorithm (which is described in more details in this paper) and the match-up rescheduling algorithm. The selective rescheduling algorithm provides real-time control by selectively rescheduling the fewest jobs possible, while the match-up rescheduling algorithm tries to match-up the entire schedule to the original planned trajectory. Note that these optimization algorithms are different to that used in the predictive part.

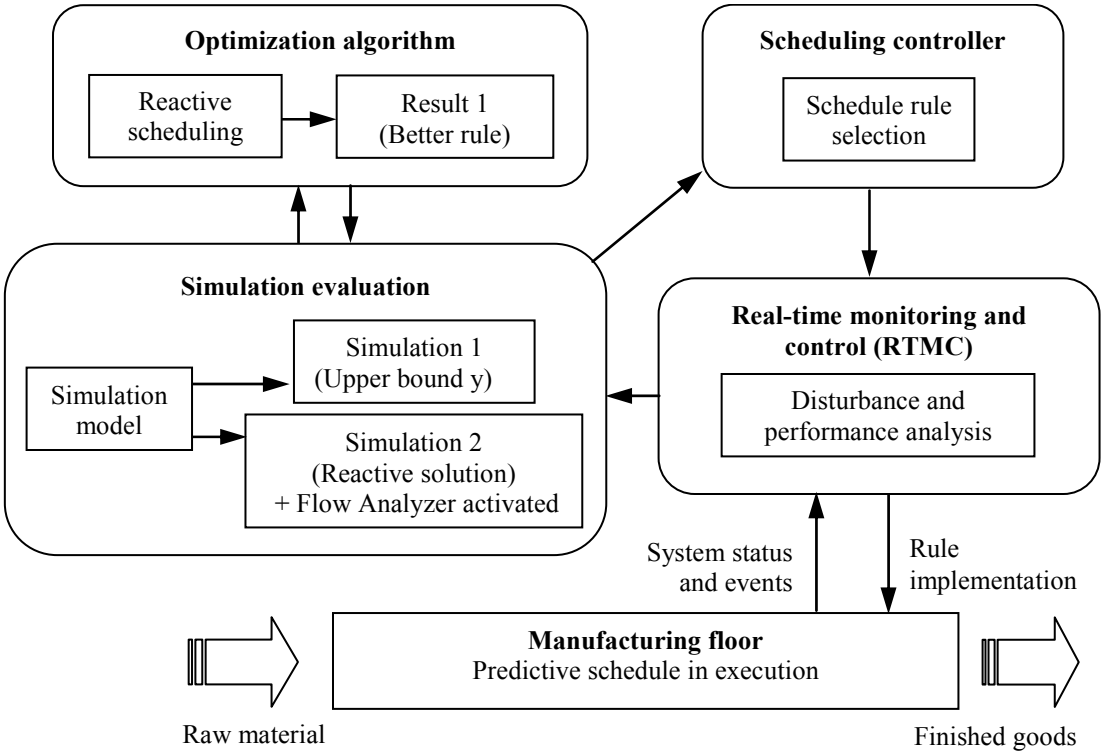

Figure 3: Combination of Simulation and Optimization for Rescheduling and Real-time Control

The optimization algorithm(s) then passes on the result to the simulation function, which simulates this result, but this time, it also activates the same Flow Analyzer (FA) component which was used in the predictive part. Note that this time, the FA component will stick to the predictive plan as much as possible, and only over-ride the predictive plan where problems such as bottleneck and related problems happen. It will not check each job, at each decision point for further overriding as was done in the predictive part. This second simulation assisted FAM will not only simulate the result to check optimality considering the broader system elements but will also check validity by solving problems which might occur due to this rescheduling solution. At the end of the second simulation run, the user can compare the result of the simulated upper bound and the rescheduling solution. He could decide which solution he wishes to implement in the scheduling controller and pass on the result to the real-time control module. In the next section the selective rescheduling optimization algorithm is described.

2.2.1 Selective rescheduling stability algorithm

Figure 4 shows the system stability algorithm which is part of the simulation assisted rescheduling mechanism. The algorithm takes as an input the upper bound computed by the simulation as shown in Figure 3. Then the stage\machine finishing last is analyzed for additional capacity. If there exists capacity, the user is prompted for selecting all machines where some capacity exists. If capacity exists, this means theoretically, system stability and rescheduling could take place. On the other hand, when capacity does not exist, the system informs the user that the rescheduling does not help the situation for the current disturbance. The user selects the other machines on the stage that the system asked him to. The system then asks the user to enter a probable number of jobs that he thinks could be a good candidate for a rescheduling try. Note that the system selects jobs for rescheduling based on criteria such as jobs finishing last on the stages under consideration. The heuristic then runs and computes a rescheduling solution for each job cumulatively. In other words, if the user selects three jobs, the system first tries to reschedule only one job, followed by rescheduling the first and the second job, followed by rescheduling the first, second and the third job and computing the job completion times. Upon completion of this procedure, the user has the results on rescheduling for all the three cases. The system will automatically select the solution which provides system stability, for further evaluation with the simulation based Flow Analyzer Module (FAM). The results are given in next section.

3 IMPLEMENTATION

At the time of writing of this paper, the predictive-reactive scheduling system was partly implemented in the simulation software *Technomatix eM-Plant*. The entire system is implemented as a custom object library in eM-Plant.

This means that the user can build his own simulation model using standard simulation objects, and place the objects for his tasks like scheduling and or\re-scheduling within the model frame, and start the analysis. For the moment, we use the simulation system itself (not a real production system) for rescheduling, as a means to test our approaches. Figure 5 shows the user interface developed for the predictive system. It shows facilities such as editing predictive information on the right hand side including setting for activating and using the simulation based FAM (Figure 6 and 7), and setting for activating the reactive system based on user defined criteria. Figure 6 shows the rescheduling dialog which is activated as soon as a disturbance occurs. It shows information such as the time, location and the nature of the disturbance, and method of rescheduling selection. As described earlier, the system informs the user if rescheduling is possible at all. This is shown by Figure 9. Figure 10 shows the options the user has to select alternative machines for rescheduling. Figure 11 shows the number of jobs the user can try for rescheduling. Examples were taken to demonstrate the system functionality and results for both the predictive and reactive part. Table 1 shows the data used for testing the predictive scheduling system. We have tested the predictive system with 1000 jobs and 30 machines with improvements in job ending times and makespan KPI.

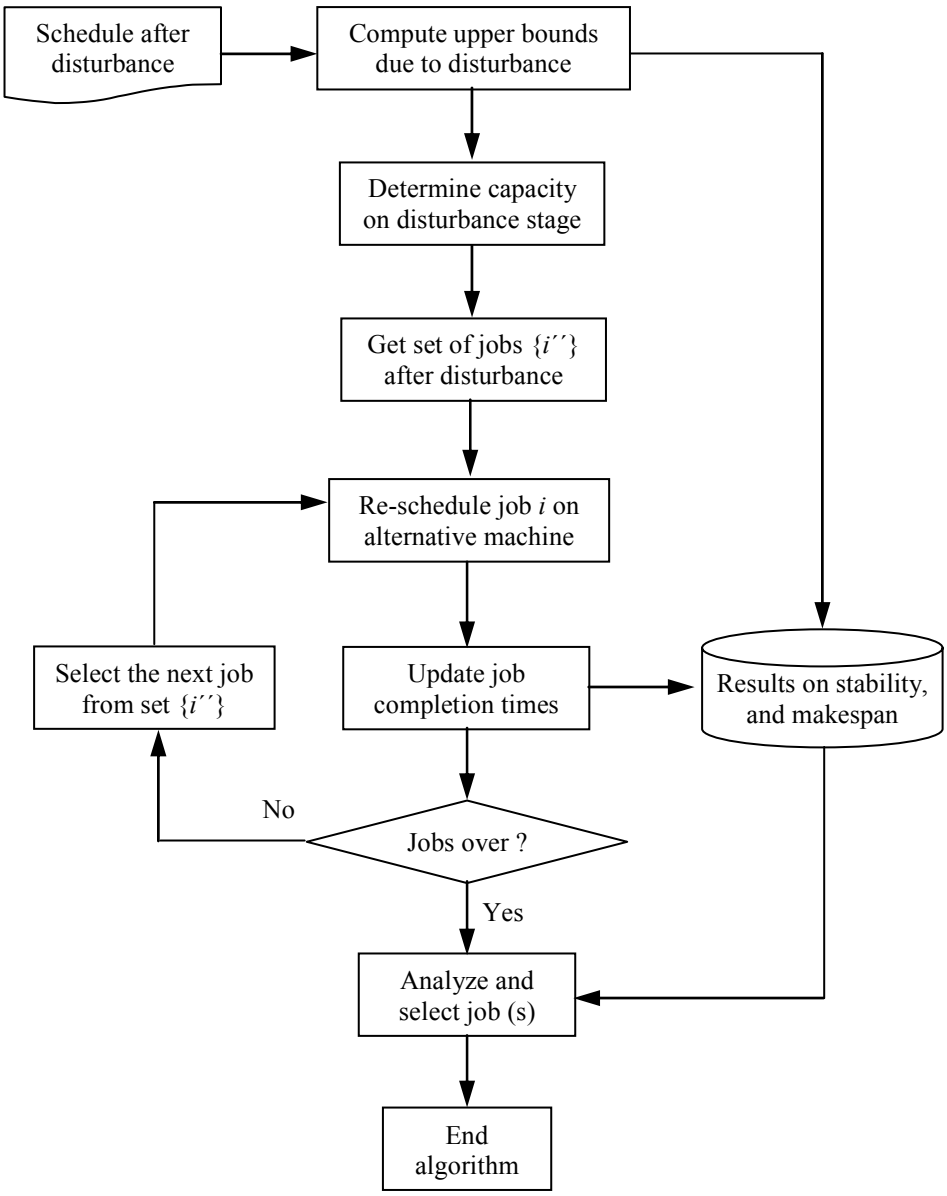

Figure 4: Selective Rescheduling Stability Algorithm

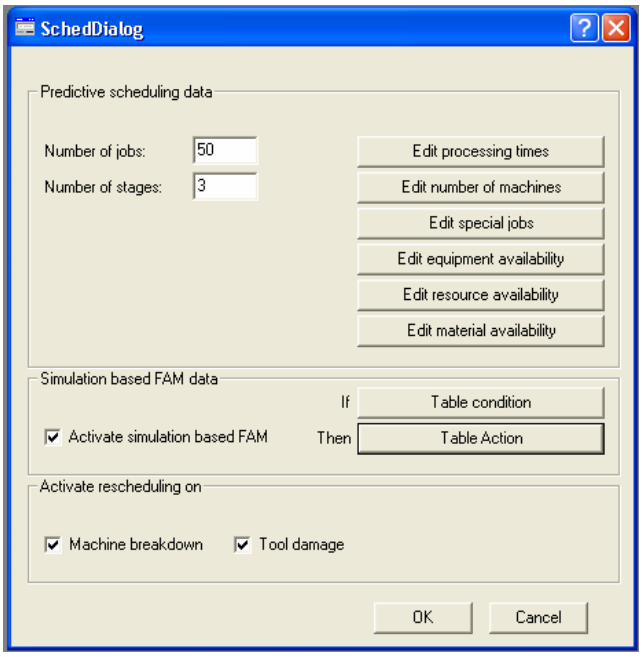

Figure 5: Predictive Scheduling Dialog

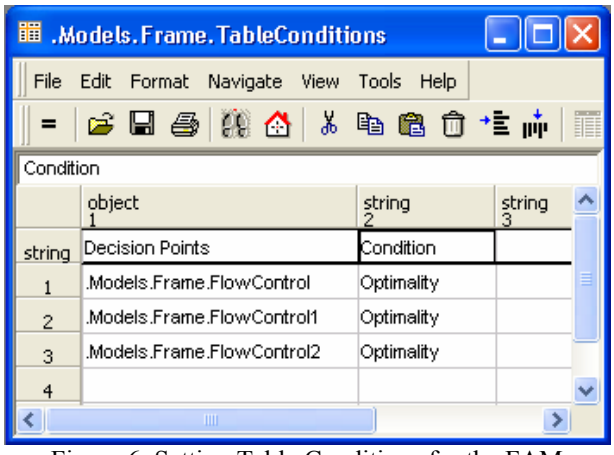

Figure 6: Setting Table Conditions for the FAM

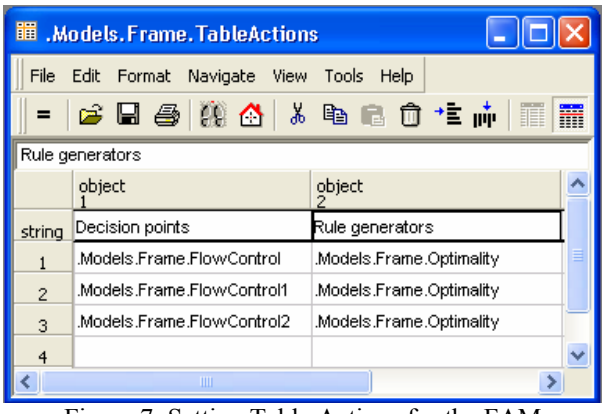

Figure 7: Setting Table Actions for the FAM

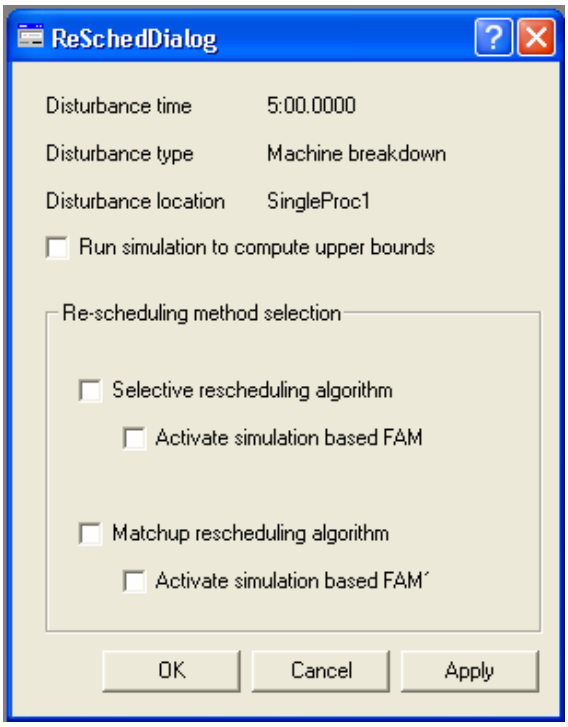

Figure 8: Rescheduling Dialog

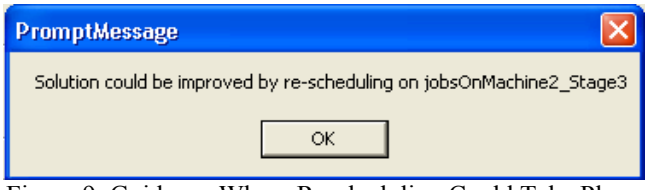

Figure 9: Guidance Where Rescheduling Could Take Place

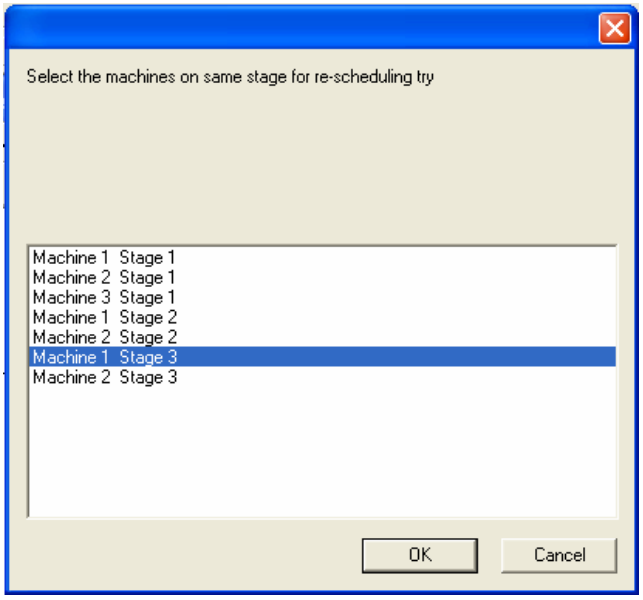

Figure 10: User Alternative Resource Selection

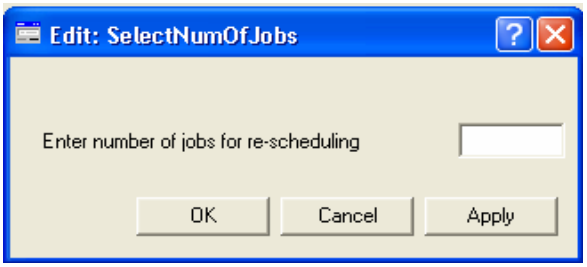

Figure 11: System Prompt for Input on Number of Jobs

Table 1 : Production System Size for Predictive Test Case

Number of stages			Nr. of jobs	Proc. time range (minutes)
Nr. of machines Stage 1	Nr. of machines Stage 2	Nr. of machines Stage 3		
5	5	3	50	(5-100)

Figure 12 shows the comparison of the combination of simulation and optimization with pure optimization and random scheduling methods for a smaller problem size. Table 2 shows the percentage reduction in job finishing times (JFT). Note that the percentage reduction for the optimization plus simulation is with respect to the optimization method. Results show that 41 jobs showed reduction in JFT out of 50 jobs, with minimum improvement of 0.9 percent and maximum improvement of 9.9 percent. It was observed that for different problem sizes, the simulation plus optimization method of scheduling was rarely worser than pure optimization, and always resulted in some solution improvements.

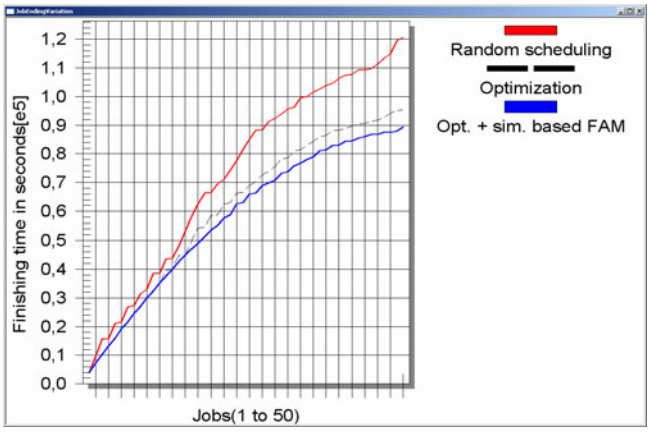

Figure 12: Assessing the Opt. + Sim. Based FAM Results

Table 3 and 4 shows the data used for testing the rescheduling solution using the selective rescheduling method. At the time of writing of this paper the match up rescheduling system was under initial stages of implementation (we have the algorithm). Figure 13 shows the rescheduling solution obtained for the first case.

Table 2: Sample Predictive Test Results

Job	Methods of scheduling				
	Random	Optimization		Opt. + Simulation based FAM	
	JFT* (minutes)	JFT (minutes)	% Reduction in JFT	JFT (minutes)	% Reduction in JFT
1	70	70	0.0 %	70	0.0 %
2	160	120	25.0 %	120	0.0 %
3	260	170	34.6 %	170	0.0 %
4	260	220	15.3 %	220	0.0 %
5	350	270	22.8 %	265	1.85 %
6	360	315	12.5 %	320	-1.6 %
7	445	365	17.9 %	360	1.3 %
8	455	405	10.9 %	410	-1.2 %
9	525	455	13.3 %	450	1.1 %
10	550	495	10.0 %	500	-1.0 %
11	640	545	14.8 %	540	0.9 %
12	640	580	9.3 %	585	-0.8 %
13	725	660	8.9 %	625	5.3 %
14	730	665	6.8 %	665	0.0 %
15	800	745	15.8 %	710	4.70 %
16	885	745	14.4 %	745	0.0 %
17	970	830	12.9 %	785	5.4 %
18	1040	905	17.6 %	815	9.9 %
19	1105	910	12.1 %	855	6.0 %
20	1110	975	15.5 %	895	8.2 %
21	1160	980	12.2 %	920	6.1 %
22	1185	1040	15.3 %	960	7.6 %
23	1240	1050	14.6 %	980	6.6 %
24	1295	1105	18.3 %	1045	5.4 %
25	1360	1110	19.0 %	1050	5.4 %
26	1420	1150	20.0 %	1100	4.3 %
27	1470	1175	17.6 %	1105	5.6 %
28	1470	1210	19.0 %	1150	4.9 %
29	1520	1230	18.1 %	1205	2.0 %
30	1540	1260	16.6 %	1180	6.3 %
31	1565	1305	17.8 %	1220	6.5 %
32	1595	1310	15.8 %	1230	6.1 %
33	1605	1350	18.3 %	1265	6.3 %
34	1660	1355	16.8 %	1280	5.5 %
35	1665	1385	17.1 %	1300	6.1 %
36	1695	1405	16.0 %	1315	6.4 %
37	1710	1435	16.7 %	1350	5.9 %
38	1730	1440	15.7 %	1355	5.9 %
39	1745	1470	16.9 %	1380	6.1 %
40	1770	1470	16.7 %	1385	5.7 %
41	1790	1490	16.7 %	1405	5.7 %
42	1795	1495	17.3 %	1410	5.7 %
43	1820	1505	17.0 %	1425	5.3 %
44	1820	1510	16.9 %	1435	4.9 %
45	1830	1520	16.9 %	1445	4.9 %
46	1855	1530	17.5 %	1445	5.5 %
47	1890	1545	18.2 %	1460	5.5 %
48	1915	1570	18.0 %	1460	7.0 %
49	1990	1595	19.8 %	1465	8.1 %
50	2010	1600	20.4 %	1490	6.8 %

* JFT = Job Finishing Times

Table 3 : Production System Size for Reactive Test Case

Number of stages			Nr. of jobs	Proc. time range (minutes)
Nr. of machines Stage 1	Nr. of machines Stage 2	Nr. of machines Stage 3		
3	3	2	20	(5-60)

Table 4: Data Used for Reactive Test Cases

Case	Disturbance duration	Shift length	Number of jobs	Nr. of machines
1	15 minutes	4 hours	20	7
2	45 minutes	2 hours	10	7

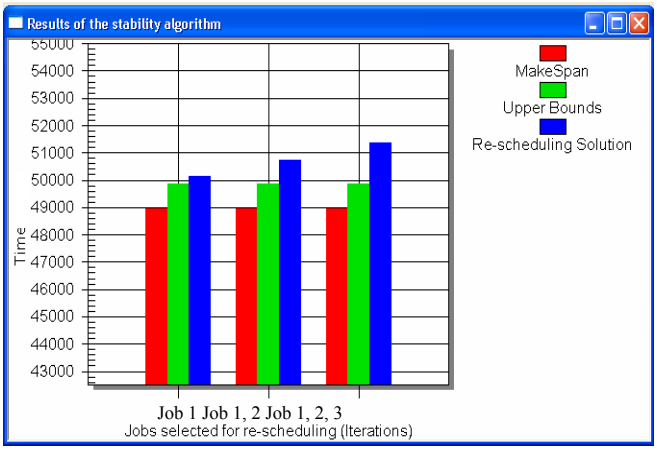

Figure 13: Rescheduling Solution Case 1

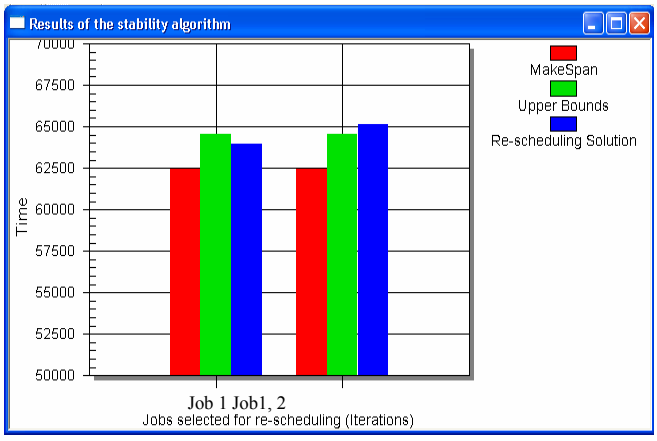

Figure 14: Rescheduling Solution Case 2

The solution was computed in about 10 seconds, which included time for the interactive data exchange with the rescheduling system using the developed user interface. Our discussion on real-time control should not be considered in a rigid sense, as we consider a semi-automated system, where a few minutes or seconds of decision making time, would not have a negative impact on the performance of the entire system. As seen in Figure 13, we see that the solution is worse when compared to the makespan and the upper bounds for all iterations. The Y axis in Figure 13 and 14 shows the time in seconds (in eM-Plant the simulation starts from the current time, not from zero, but the computation times shown start from zero with respect to the current time. It may appear that a large amount of time has passed giving the impression that the results are irrelevant– that is not the case!). This means, that in this case, rescheduling is not at all recommended – it will not improve the situation. Note that though the system has an earlier check system (by prompting the user) to filter out unnecessary computations when rescheduling cannot be done, it does still consider theoretical possibilities of rescheduling, when capacity exists. In this case, capacity did exist, so the algorithm went ahead with rescheduling, but found out that rescheduling is not feasible because the solution was worse. For the second test case, Figure 14 shows that the first iteration clearly helps the situation by resulting in makespan lower than the upper bounds. This figure also shows that only one job is sufficient to be rescheduled, and not 2 or more jobs. The computation times for the second test case were about the same as for the first case. The results so far have answered the question how much to reschedule, and if we should reschedule at all. The question when to reschedule will also be answered by the system as it computes the time when the job or jobs should be rescheduled depending on the precedence constraints.

4 CONCLUSIONS

In this paper, we have presented issues that exist when scheduling and rescheduling a special configuration of a flexible production system. We addressed how simulation can be used effectively for scheduling and rescheduling a complex system. The combination of the simulation based FAM and optimization for predictive scheduling can result in quantitative gains in performance measures considering some global elements of the system as seen from the results. We consider this combination of simulation and optimization as a small practical step in the state of the art simulation and optimization technologies. We also showed how simulation can be combined with optimization for rescheduling, for achieving newer performance indicators such as system stability.

The elements of stability like how much to reschedule, which point in time to reschedule, and if we should reschedule at all have been answered with certainty. From the results we can conclude that the rescheduling algorithms can start their computations as soon as the disturbance occurs, but do not essentially have to reschedule immediately upon occurrence of a disturbance. Our results show that the methods are promising at the outset, though the system needs to be completely developed, and tested thoroughly for it to be acclaimed as useful.

REFERENCES

Brah, S. A. and J. L. Hunsucker. 1991. Branch and Bound algorithm for the flow shop with multiple processors, *European Journal of Operational Research*, Vol. 51, pp- 88-99.

Cheng, J., Y. Karuno, and H. Kise. 2001. A shifting bottleneck procedure for a parallel machine flow shop scheduling problem. *Journal of Operations Research, Society of Japan*, 29, No. 2, pp. 140-156.

Chong, C. S., A. I. Sivakumar, and R. Gay. 2003. Simulation-based scheduling for dynamic discrete production. *In Proceedings of the 2003 Winter Simulation Conference*, ed. S. Chick, P. J. Sanchez, D. Ferrin, and D. J. Morrice, 1465 - 1473. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Church, L. K., and R. Uzsoy. 1992. Analysis of periodic and event driven rescheduling policies in dynamic shops. International *Journal of Computer Integrated Manufacturing 5*, 153-163.

Gupta, J. N. D., and A. J. Ruiz-Torres. 2000. Minimizing makespan subject to minimum total flow time on identical parallel machines, *European Journal of Operational Research*, Vol. 125, No.5, pp.616-625.

Haldun, A., M. Lawley, K. McKay, S. Mohan and R. Uzsoy. 2005. Executing production schedules in the face of disturbances: A review. *European Journal of Operational Research 161*, 86-110.

Harmonosky, C. M., R. H. Farr, and M. C. Ni. 1997. Selective rerouting using simulated steady state system data. *In Proceedings of the 1997 Winter Simulation Conference*, ed. S. Andradottir, K. J. Healy, D. H. Withers, and B. L. Nelson, 1293 - 1298. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Leon, V. J., S. D. Wu, and R. H. Storer. 1994. A game-theoretic control approach for job shops in the presence of disruptions. *International journal of production research 32*: 1451-1476.

Leon, V. J., S. D. Wu, and R. H. Storer. 1993. Robustness measures and robust scheduling for job shops. *IIE Transactions 26*, 32-43.

Phadniis, S., and S. Irani. 2003. Development of a new heuristic for scheduling flow-shops with parallel machines by prioritizing bottleneck stages. *Transactions of the SDPS*, Vol. 7, No. 1, pp 87-97.

Wu, S. D., and R. A. Wysk. 1989. An application of discrete-event simulation to on-line control and scheduling of flexible manufacturing. *International Journal of Production Research*, Vol. 27 (9).

Wu, S. D., E. Byeon, and R. H. Storer. 1999. A graph-theoretic decomposition of job shop scheduling problems to achieve scheduling robustness. *Operations Research*, Vol. 47, pp. 113-124.

AUTHOR BIOGRAPHIES

WILHELM DANGELMAIER studied Mechanical Engineering at the University of Stuttgart, Germany. Since 1981, he was director and head of the Department for Corporate Planning and Control at the Fraunhofer Institute for Manufacturing (Fraunhofer IPA). There he was responsible for the development of SIMPLE++ (today *eM-Plant* owned by UGS – Technomatix line of products). In 1991, Dr. Dangelmaier became Professor for Business Computing at the Heinz Nixdorf Institute; University of Paderborn, Germany. In 1996, he founded the Fraunhofer Center for Applied Logistics. Dr. Dangelmaier has published 508 technical articles in various magazines, journals and conferences. He has written chapters in 73 books, supervised 60 Ph.D thesis' and written 3 books on the subject of Supply Chain Management and Manufacturing. His principal interests today are models and tools for distributed production systems. His e-mail address is - <dangelmaier@hni.upb.de>.

KIRAN R MAHAJAN studied Mechanical Engineering at the Delft University of Technology (TUDelft) in The Netherlands with a specialization in Production Engineering. Since 2004, he is a research assistant at the Heinz Nixdorf Institute, group Business Computing, esp. CIM, Germany. He has a bachelor's degree in Production Engineering from the University of Pune (Govt. College of Engineering, Pune), India. His research interest is development of simulation based planning and scheduling systems. His e-mail address is <kiran@hni.upb.de>.

THOMAS SEEGER is a masters student in the group Business Computing, esp. CIM, Heinz Nixdorf Institute, Germany. His research interests are the development of simulation based planning and scheduling systems. His e-mail address is <tseeger@upb.de>.

BENJAMIN KLÖPPER studied business computing at the University of Paderborn. Since 2005, he is a research assistant at the group of Prof. Dangelmaier, Business Computing, esp. CIM. His research interests are planning and scheduling of complex systems. His e-mail address is <Klöpper@hni.upb.de>.

MARK AUFENANGER studied business computing at the University of Paderborn. Since 2005, he is a research assistant at the group of Prof. Dangelmaier, Business Computing, esp. CIM. He is mainly interested in simulation of logistic systems. His e-mail address is <marka@hni.upb.de>.