Decision Support Tool for Dynamic Scheduling

L. Ferreirinha[1], A. S. Santos[12], A. M. Madureira[1], M. L. R. Varela[2] and J. A. Bastos[1]

[1] School of Engineering, Polytechnic of Porto (ISEP/IPP)
[2] Department of production and Systems, University of Minho (UM)
l.ferreirinha96@gmail.com, abg@isep.ipp.pt, amd@isep.ipp.pt, leonilde@dps.uminho.pt, jab@isep.ipp.pt

Abstract. Production scheduling in the presence of real-time events is of great importance for the successful implementation of real-world scheduling systems. Most manufacturing systems operate in dynamic environments vulnerable to various stochastic real-time events which continuously forces reconsideration and revision of pre-established schedules. In an uncertain environment, efficient ways to adapt current solutions to unexpected events, are preferable to solutions that soon become obsolete. This reality motivated us to develop a tool that attempts to start filling the gap between scheduling theory and practice. The developed prototype is connected to the MRP software and uses meta heuristics to generate a predictive schedule. Then, whenever disruptions happen, like arrival of new tasks or cancelation of others, the tool starts rescheduling through a dynamic-event module that combines dispatching rules that best fit the performance measures pre-classified by Kano's model. The proposed tool was tested in an in-depth computational study with dynamic task releases and stochastic execution time. The results demonstrate the effectiveness of the model.

Keywords: Dynamic scheduling, Meta heuristics, Hyper heuristics, Dispatching rules, Decision support tool, Kano's Model.

1 Introduction

Scheduling in real-world manufacturing facilities can be extremely difficult. Most manufacturing systems operate in dynamic environments vulnerable to various stochastic real-time events like, machine breakdowns, rush orders, unavailable material, and many others which easily turn preschedules obsolete [1, 2, 3]. Those manufacturing systems generate and update production schedules, which are plans that serve as basis for many external activities (e.g., material procurement, preventive maintenance). In an uncertain environment, industries can benefit from better understanding how (re)scheduling strategies affect system's performance, because in this environment the focus of the foreman tends to be on finding efficient and effective ways to adapt current solutions to unexpected events, rather than finding a high-quality schedule that rapidly becomes outdated. In practice, rescheduling is done in a hybrid way, either periodically, i.e., plans for the next period based on current status of the system, or occasionally in response to nonplanned events [1,4,5].

When it comes to scheduling systems in dynamic environments, two key elements stand out, the *schedule generation*, which acts as a predictive mechanism and serves as an overall plan for other shop activities, and *schedule monitorization/updating*, which is viewed as the reactive part of the system that attempts to minimize the effect of the disturbances in the performance of the system [4, 5, 6].

In view of the above, this paper proposes a decision support tool to dynamic environments that attempts to fill the gap between scheduling theory and practice stated by the author in [7]. Continuing the model proposed in [8] this paper presents a prototype that does not require any interaction with the user besides the definition of the criteria of performance. The definition of the criteria of performance are classified through the degree of satisfaction of the Kano's Model, which classifies the performance measures and balance the interests of the stakeholders. The tool is also connected to the MRP of the company, and when the MRP launches the jobs into the system a meta heuristic is used to generate the schedule. When disruptions happen, the tool begins to reschedule the tasks according to priority rules that best fit the intended performance. To validate the tool, an instance composed of a vast set of tasks with normally distributed stochastic characteristics were executed.

The remaining sections of this paper are organized as follows: section 2 revises production scheduling problems in dynamic environments. Section 3 briefly introduces concepts related to optimization in dynamic environments. The prototype is presented in section 4. In section 5 is presented the computational results. And finally, section 6 presents some conclusions and provides some ideas for future work.

2 Production Scheduling Problems

Production Scheduling (PS) implies the definition of an initial and final moment of the processing of each task and its allocation to the resources, fulfilling certain restrictions that may involve the tasks and/or resources. The purpose of scheduling is to optimize a certain measurement of economic and operational performance. So, a typical production scheduling problem can be seen as, *n* parts must be processed by using *m* machines, and each part must be processed in a given order on the respective machine to find the schedule that optimizes certain performance metric [3,8,9,10].

In [11] it is mentioned the PS problems can be classified in three levels: 1-*Requirements generation, 2-Processing complexity* and 3-*Scheduling criteria.* The first level refers to *open shop* versus a *closed shop*, i.e., in an *open shop* the production orders are requested by costumers and there is no inventory, while in a *closed shop* the inventory is used to serve the costumers requests, which introduces an inventory replenishment decision. So, in its simplest form, the *open shop* production scheduling is a *sequencing problem.* As for the *closed shop,* PS involve sequencing and also *lot-sizing* decisions associated to the inventory replenishment process. The second level refers to the number of processing steps that are needed for each production task, *One*-stage (one processor or Parallel processors) and Multistage (flow shop or job shop). The third level refers to the performance measures that the schedule should optimize. Sometimes these three levels are not sufficient to classify the PS problem, whereby the authors in [11] refer two additional levels: 4-*Nature of the requirement specification* and 5-*Scheduling environment*. The fourth level refers to the nature of the parameters, when all the parameters are known and fixed, the problem is classified as *deterministic*, otherwise it is *stochastic*, i.e., when all the parameters are uncertain with specified probability distribution. In the fifth level, when all the tasks are known at the beginning of the schedule the problem is said to be *static*, while when new tasks can arrive unexpectedly, the environment of the problem is *dynamic* [3,11,12].

2.1 Dynamic Scheduling

There has been a recent increasing interest in modelling and solving scheduling problems in dynamic environments. In industrial environments, scheduling is an ongoing reactive process where real time events forces reconsideration and revision of pre-established schedules. As mentioned before, in this scheduling environment the tasks are not known at the beginning of the problem which makes the system vulnerable to random, inevitable and unpredictable real-time events that cause a change in the scheduled plans, which makes a previously feasible schedule turn infeasible when it is released to the shop floor. Such unexpected events can occur for a variety of reasons, like related to the resources (e.g., breakdowns, rework, operator illness), or the jobs (e.g., order cancelation, changes in delivery times, late arrivals) [3,6, 13,14].

The main goal in solving this problem is no longer to find optimum schedules, because they quickly will become obsolete, but instead be able to efficiently adapt current solutions to the dynamic environment, since near optimal solutions that are easily adaptable, will be preferable to optimum ones. So, when non-planned perturbations occur, it is necessary to find a new schedule with the quality close to the schedule that could have been executed if all of the uncertainty had been revealed *a priori* [8,15].

Dynamic scheduling has been defined under three categories [4,6,14,16], which are summarized in table 1.

Table 1. Categories of dynamic scheduling

Completely reactive scheduling	**Predictive-reactive scheduling**
In completely reactive scheduling, no firm schedule is generated in advance and decisions are made locally in real-time usually by priority dispatching rules.	Predictive-reactive scheduling is an iterative process and has two primary steps. The first step generates a production scheduling. The second step updates the schedule in response to a real-time event.
Robust pro-active scheduling	
Robust pro-active scheduling is based on predictive schedules that satisfy performance requirements predictably in a dynamic environment.	

When it comes to reschedule due to real-time events, two major issues emerge: how and when to react to those events. The first issue lies on the strategy to use to reschedule, which usually leads to schedule repair or complete schedule [4, 5]. Schedule repair refers to adjusting the current schedule, saving CPU times and without compromising the stability of the system. Regarding complete schedule, it literally refers to schedule from scratch. Although this last may be better to maintain the optimal solution, in practice there is usually no time to reschedule [4,6,14,16]. Regarding the second issue, when to reschedule, it basically aims to answer when an event has sufficient impact that a new schedule is necessary. Three policies can be find in the literature [4,5,14,16]: *Continuous rescheduling* or *event driven,* which reschedules each time an event occur, such as an arrival of new tasks, *Periodic rescheduling,* where schedules are generated at regular intervals *T* and any disruptions between periods of rescheduling are ignored until the next period where it gathers all available information from the shop floor. And last the *hybrid,* which reschedules the system periodically and also when some particular events happen, like machine breakdowns, arrival of urgent jobs, cancelation of jobs, among others [2,5,16,17].

3 Optimization Techniques

3.1 Dispatching Rules

Heuristics are problem specific schedule repair methods, which have the ability to find near-optimum solutions quickly and with little computational effort. Dispatching rules are also heuristics with major importance in completely reactive scheduling and they are used to sort the jobs in the machines queue by some criteria. So, similar to pull mechanisms, like *Kanban cards*, dispatching rules are used to control production.

In [18] an extensive list of dispatching rules is presented and the difficulty of the choice of a dispatching rule arises from the fact that there are ***n!*** ways of sequencing ***n*** jobs. In the literature, the performance of dispatching rules is usually evaluated experimentally, although in some cases a dispatching rule can be shown to be optimal like the Shortest Processing Time *(SPT)* that minimizes the Mean Flow Time and the Earliest Due Date *(EDD)* that minimizes the Maximum Tardiness, among others [2, 8, 6,14,15,19]. A dynamic scheduling can be viewed as a collection of linked static problems, which implies that methods developed for static scheduling problems become applicable to dynamic ones. Such methods can effectively deal with complex problems and can optimize the quality of the schedules for each static sub-problem. [10]. The authors of [18, 19, 20] present a state-of-the-art survey of dispatching rules.

3.2 Meta Heuristics

Scheduling problems are becoming more complex and the exhaustive search for optimal solutions has become impractical, given the computational effort required to find them. In the last decades a new family of approximate algorithms has arisen that dominated the research. These methods are called Meta Heuristics (MHs), a term that was first introduced by Glover [21] in the 1980s. According to [22] meta-heuristics are more advantageous than most heuristics in terms of solution robustness. However, they are more difficult to implement and tune, since they need information about the problem in order to achieve satisfactory results. The author of [23] further states that unlike exact methods, meta heuristics allow dealing with instances of large-scale problems and find satisfactory solutions within a reasonable amount of time.

4 Prototype

The tool was designed with the purpose of not requiring any interaction with the user besides the definition of the performance criteria. The definition of the performance criteria is classified through the degree of satisfaction of the Kano's Model. As an example of how the performance measures can be classified in the tool, see Figure. 1. For the defined objectives in Figure. 1, it is assumed that the user does not want the Maximum Tardiness to surpass a value and, at the same time, wants the Mean Flow Time to be the smallest possible. As Figure. 1 shows, the Maximum Tardiness represents a Must-be attribute in the Kano's model, so if it is not achieved, it results in an extreme dissatisfaction. As for the Mean Flow Time, it represents a One-dimensional, since it corresponds to a degree of satisfaction proportional to the degree of performance of the attribute, i.e., the smaller the better [24,25].

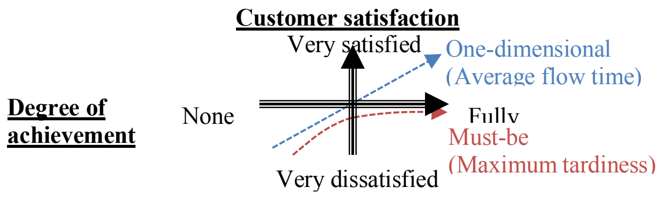


Figure. 1. Example of Performance measures classified through Kano's Model

After the definition of the performance criteria, the software itself schedules and reschedules the tasks in the system, over time, even when a real-time event, like the cancelation of a task, changes on due dates and so on, occurs. To generate acceptable solutions in such circumstances the tool is connected to the MRP of the company to generate a predictive schedule using the available information and then, whenever disruptions occur in the system during the execution of the pre-established schedule, rescheduling is performed. Since the prototype is connected to the MRP and most of the time the MRP runs periodically, we are aware that disturbances between two immediate periods of time may happen, which lead us to a hybrid strategy, i.e., rescheduling the system periodically and also when some particular events happen.

When MRP runs, a large number of tasks usually enter the system, so the search for the satisfactory solution is time consuming, and it will hardly remain viable in industrial environment. Thus, the proposed tool generates the predictive schedule by meta heuristics, which can deal with large-scale problems and find satisfactory solutions within a reasonable amount of time. It is worth emphasizing that the ideal would be that all the tasks launched in period $\boldsymbol{T}$ of the MRP would end before the period $\boldsymbol{T+1}$. However, such a scenario is not likely to occur in real-world environments, so, when the system reaches $\boldsymbol{T+1}$, the MH will schedule the new tasks that the MRP will launch, as well as the tasks that have not yet been processed. The parameterization of the MHs is done by the design of experiments (DOE) of Taguchi, where the values that are going to compete either are calculated by metrics or are generated randomly within intervals that have been shown to be effective in solving problems of this type in the scheduling community. See for example [23, 26, 27, 28] where some metrics and ranges are presented for Simulated annealing (SA). Whenever disruptions happen, the prototype reschedules the tasks with dispatching rules. A brief outline of what has been described here is shown in Figure. 3

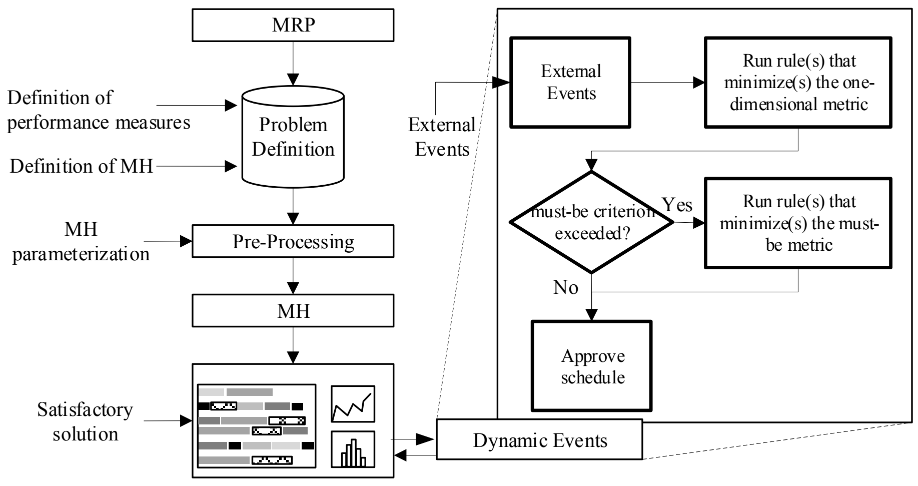

MRP
Definition of performance measures
Definition of MH
Problem Definition
MH parameterization
Pre-Processing
MH
Satisfactory solution
External Events
Dynamic Events
External Events
Run rule(s) that minimize(s) the one-dimensional metric
must-be criterion exceeded?
Yes
No
Run rule(s) that minimize(s) the must-be metric
Approve schedule

Figure. 2. Scheme of the prototype logic

In Figure 2 is shown the logic of the prototype. When the MRP runs, the user has to define the performance measures and the MH that is going to generate the predictive schedule, otherwise the tool will maintain the previous choices. After the parameterization a predictive schedule is generated and then is released to the shop floor by a panel where several performance indicators are shown. When disruptions happen, the "Dynamic Events Module" reschedules the tasks. Despite of the prototype presented being a totally autonomous tool, there will be situations where the foreman may want to validate alternative schedules. In this initial phase of the project, such functionality is not operational, but it is in the authors' interest to implement it in the near future.

The main difference of the prototype presented here in relation to other methodologies in the literature, is that it is independent of the machine environment. Much of the literature that analyzes dynamic scheduling presents problem specific methodologies and not a generic model adaptable to several scenarios. For example, in [29] the authors present an efficient hybrid Genetic Algorithm (GA) methodologies where a new KK heuristic + swap and well-known dispatching rules + swap, for minimizing makespan in dynamic job shop problems. In [30] the authors proposed a multi-objective methodology for the FJSP scheduling problem in machine breakdown situations. Another example can be in [31] where the authors proposed a Variable Neighborhood Search (VNS) algorithm to solve a dynamic Flexible Flow Shop (FFS) problem considering unexpected arrival of new jobs. Although the methodologies presented by the aforementioned authors are efficient for the problems in question, they are hardly immediately applicable to other machine environments.

5 Computational study

To demonstrate how the prototype operates, 250 jobs with stochastic attributes, shown in table 2, were created. For this study it was defined that the period between executions of the MRP is 250 time units, the "one-dimensional" criteria is the Mean Flow Time and the "must be" criteria is the Maximum Tardiness, where the maximum value is zero. SA was chosen as the meta heuristic and the obtained results will be compare to the ones achieved if only the rule that best fits the “must be criteria” was used.

Table 2. Distributions used

T	**0**	**250**	**500**	**[0-750]**
Nº of Jobs	50	50	50	100
rj	0	250	500	N(400,150)
dj	N(300,25)	N(550,25)	N(800,25)	N(300,25)
pj	N(5,1)			
wj	N(10,3)			

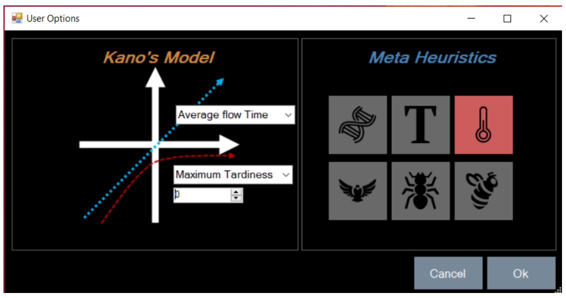


Figure. 3. User options

Depending on the performance criteria selected, the fitness calculation will vary. For the defined test, the fitness calculation is done according to expression 1.

$$F(x)=\alpha.\ \mathrm{Max}\{L\}+\frac{\Sigma cj}{n},\ \text{where } \alpha= \begin{cases} M, if\ Max\ \{L\} > Set\ value \\ 1, othewise \end{cases} \qquad (1)$$

In the expression 1, ***M*** stands for a major number that forces the meta heuristic to minimize the mean flow time whenever triggered. That ***M*** must be at least the maximum tardiness of a schedule through the LPT rule. So, whenever the maximum tardiness exceeds the set value, the fitness forces the meta heuristic to find a solution where the Mean Flow Time is minimum from the set of solutions that cannot meet the defined value. Otherwise, it tries to find one that minimizes both. Regarding the “Dynamic Events Module”, it will use dispatching rules like SPT and EDD, since these minimize the Mean Flow Time and the Maximum Tardiness, [8].

5.1 Results

The results, Figure 5, were compared with the results from the system based only on EDD in Figure 4. Figure 4 compares the Mean Flow Time and the Work in Progress (WIP) between the model and the system based only on EDD. As we can see, the Mean Flow Time in the system based on EDD tends to get a bit higher than the tool. As for the work in progress, the system based only on EDD tends to create a bigger WIP. Figure 5 shows the effectiveness of the tool, where the Maximum Tardiness was always less than zero and the Mean Flow Time was minimized whenever possible, reaching 958,89 in the final phase. The SPT rule was used 57 times, gray on the performance chart, the EDD rule was used 36 times, represented in red and the blue stands for the MH. There are 68 tasks in progress and 182 have been processed.

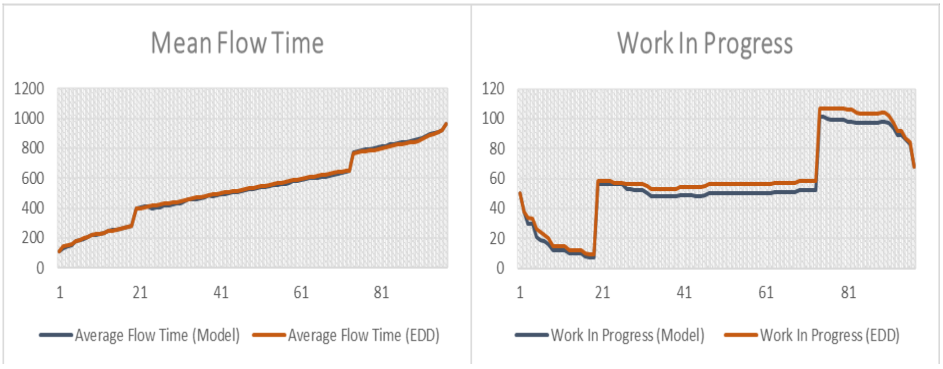
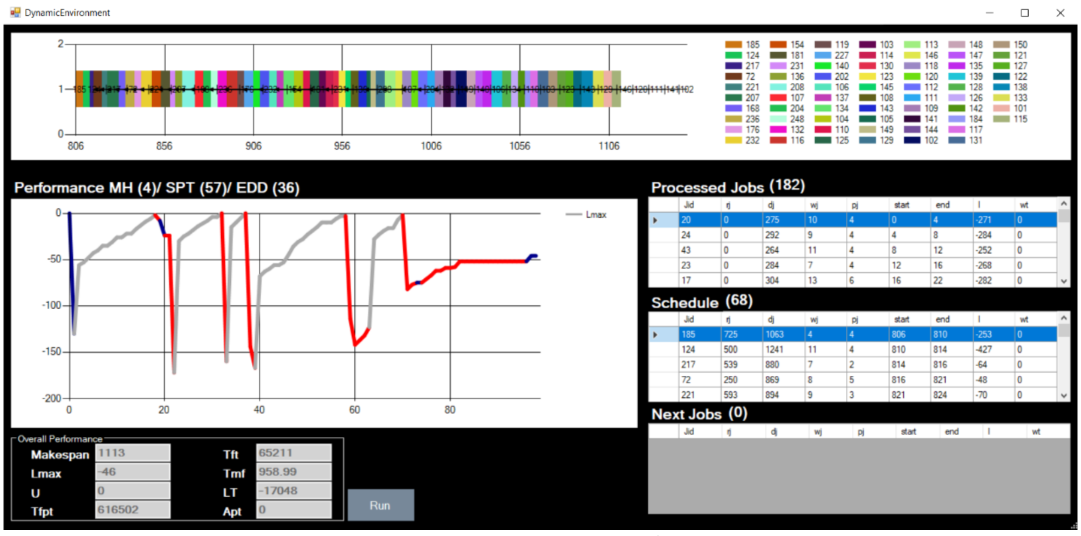


Figure. 4. Obtained results



Processed Jobs (182)

Jid	rj	dj	wj	pj	start	end	l	wt
20	0	275	10	4	0	4	-271	0
24	0	292	9	4	4	8	-284	0
43	0	264	11	4	8	12	-252	0
23	0	284	7	4	12	16	-268	0
17	0	304	13	6	16	22	-282	0

Schedule (68)

Jid	rj	dj	wj	pj	start	end	l	wt
185	725	1063	4	4	806	810	-253	0
124	500	1241	11	4	810	814	-427	0
217	539	880	7	2	814	816	-64	0
72	250	869	8	5	816	821	-48	0
221	593	894	9	3	821	824	-70	0

Next Jobs (0)

Jid	rj	dj	wj	pj	start	end	l	wt

Overall Performance

Makespan	1113	Tft	65211
Lmax	-46	Tmf	958.99
U	0	LT	-17048
Tfpt	616502	Apt	0

Figure. 5. Prototype results

6 Conclusions

This paper intends to demonstrate an approach to the dynamic scheduling production without user intervention. The software itself is connected to the MRP of the company and according to the defined objectives through the Kano's model, whenever disruptions occur it schedules and reschedules through meta heuristics and "Dynamic Events Module", in order to comply with the objectives. The results show the effectiveness of the tool, where the set value of the "must be criteria" was never exceeded, and the "one dimensional" criteria was optimized whenever possible. Regarding future work, it is intended to prove through statistical evidence the operation of the tool, as well as to prepare the tool for more dynamic events such as machine breakdowns, repair times, among others, in order to lessen the gap between scheduling theory and practice. As mentioned previously, it is still in the authors' interest to enable manual intervention by the foreman in relation to the scheduling defined by the tool.

References

1. O'Donovan, R., Uzsoy, R., McKay, K. N.: Predictable scheduling of a single machine with breakdowns and sensitive jobs. International Journal of Production Research, 37(18), 4217-4233 (1999).
2. Sabuncuoglu, I., Karabuk, S.: Rescheduling frequency in an FMS with uncertain processing times and unreliable machines. Journal of Manufacturing Systems, 18(4), 268-283 (1999).
3. Madureira, A., Ramos, C., Silva, S. D. C.: Using genetic algorithms for dynamic scheduling. In I14th Annual Production and Operations Management Society Conference (POMS 2003). (2003).
4. Vieira, G. E., Herrmann, J. W., Lin, E.: Rescheduling manufacturing systems: a framework of strategies, policies, and methods. Journal of scheduling, 6(1), 39-62 (2003).
5. Sabuncuoglu, I., Bayız, M.: Analysis of reactive scheduling problems in a job shop environment. European Journal of operational research, 126(3), 567-586 (2000).
6. Varela, M. L. R., Ribeiro, R. A.: Distributed manufacturing scheduling based on a dynamic multi-criteria decision model. In Recent Developments and New Directions in Soft Computing (pp. 81-93). Springer, Cham. (2014).
7. Cowling, P., Johansson, M.: Using real time information for effective dynamic scheduling. European journal of operational research, 139(2), 230-244 (2002).
8. Ferreirinha, L., Baptista, S., Pereira, A., Santos, A. S., Bastos, J., Madureira, A. M., Varela, M. L. R.: A Dynamic Selection of Dispatching Rules Based on the Kano Model Satisfaction Scheduling Tool. In International Conference on Innovation, Engineering and Entrepreneurship. 339-346. Springer, Cham (2018).
9. Artiba, A., Elmaghraby, S.E. (ed.).: The planning and scheduling of production systems: methodologies and applications. Springer Science & Business Media (1996).
10. Akers Jr, S. B., Friedman, J.: A non-numerical approach to production scheduling problems. Journal of the Operations Research Society of America 3.4, 429-442 (1955).
11. Graves, S. C.: A review of production scheduling. Operations research 29.4, 646-675 (1981).
12. French, S.: Sequencing and Scheduling: An introduction to the Mathematics of the Job Shop. Ellis Horwood,Chichester. (1982).
13. Goren, S., Sabuncuoglu, I.: Robustness and stability measures for scheduling: single-machine environment. IIE Transactions 40.1, 66-83 (2008).

14. Ouelhadj, D., Petrovic, S.: A survey of dynamic scheduling in manufacturing systems. Journal of scheduling, 12(4), 417 (2009).
15. Terekhov, D., Down, D. G.,Beck, J. C.: Queueing-theoretic approaches for dynamic scheduling: a survey. Surveys in Operations Research and Management Science, 19(2), 105-129 (2014).
16. Aytug, H., Lawley, M. A., McKay, K., Mohan, S., Uzsoy, R.: Executing production schedules in the face of uncertainties: A review and some future directions. European Journal of Operational Research, 161(1), 86-110 (2005).
17. Church, L. K., Uzsoy, R.: Analysis of periodic and event-driven rescheduling policies in dynamic shops. International Journal of Computer Integrated Manufacturing, 5(3), 153-163 (1992).
18. Panwalkar, S. S., Iskander, W.: A survey of scheduling rules. Operations research, 25(1), 45-61 (1977).
19. Rajendran, C., Holthaus, O.: A comparative study of dispatching rules in dynamic flowshops and jobshops. European journal of operational research, 116(1), 156-170 (1999).
20. Ramasesh, R.: Dynamic job shop scheduling: a survey of simulation research. Omega, 18(1), 43-57 (1990).
21. Glover, F.: Future Paths for Integer Programing and Links to Artificial Intelligence. Computer & Operations Research, 13(5), 533-549 (1986).
22. Xhafa] F., Abraham A.: Metaheuristics for Scheduling in Industrial and Manufacturing Applications Series. Studies in Computational Intelligence. 1st edn. Springer. (2008).
23. Talbi, E. G.: Metaheuristics: From design to implementation. Wiley. Chichester. (2009).
24. Högström, C., Rosner, M., Gustafsson, A.: How to create attractive and unique custom-er experiences: An application of Kano's theory of attractive quality to recreational tourism. Marketing Intelligence & Planning 28(4), 385-402 (2010).
25. Löfgren, M., Witell, L., Gustafsson, A.: Theory of attractive quality and life cycles of quality attributes. The TQM Journal 23(2), 235-246 (2011).
26. Eglese, R. W.: Simulated annealing: a tool for operational research. European journal of operational research, 46(3), 271-281 (1990).
27. Park, M. W. Kim, Y. D.: A Systematic Procedure for Setting Parameters in Simulated Annealing Algorithms. Computers & Operations Research, 25(3), 207- 217 (1998).
28. Kirkpatrick, S., Gelatt, C. D. Vecchi, P. M.: Optimization by SimulatedAnnealing. Science, 220, 671-680 (1983).
29. Kundakcı, N., & Kulak, O.: Hybrid genetic algorithms for minimizing makespan in dynamic job shop scheduling problem. Computers & Industrial Engineering, 96, 31-51. (2016).
30. Ahmadi, E., Zandieh, M., Farrokh, M., & Emami, S. M.: A multi objective optimization approach for flexible job shop scheduling problem under random machine breakdown by evolutionary algorithms. Computers & Operations Research, 73, 56-66.(2016).
31. Rahmani, D., & Ramezanian, R.: A stable reactive approach in dynamic flexible flow shop scheduling with unexpected disruptions: A case study. Computers & Industrial Engineering, 98, 360-372.(2016).