Genetic Algorithm for Flexible Job Shop Scheduling Problem - a Case Study

Gabriela Guevara[*], Ana I. Pereira[†,**], Adriano Ferreira[*], José Barbosa[*] and Paulo Leitão[*,‡]

[*]*Polytechnic Institute of Bragança, Campus Sta Apolónia, Apartado 1134, 5301-857 Bragança, Portugal*
[†]*Polytechnic Institute of Bragança, Portugal*
[**]*ALGORITMI, University of Minho, Portugal*
[‡]*LIACC - Artificial Intelligence and Computer Science Laboratory, Rua Dr. Roberto Frias, 4200- 465 Porto, Portugal*

Abstract. This paper proposes the impact assessment of the workers in the optimal time of operations in a Flexible Job Shop Scheduling Problem. In this work, a real enterprise was studied. The problem consists in finding the workers operations schedule, taking into account the precedence constraints. The main objective is to minimize the finish time of the last task completed in the schedule. The genetic algorithm was used to solve the optimization problem and some numerical results are presented.

Keywords: Scheduling Problem. Genetic algorithm. Flexible Job Shop Problem.
PACS: 02.60.Pn

INTRODUCTION

The scheduling of operations is a complex problem consisting in determining the optimal allocation of tasks to resources under a set of constraints, which in enterprise business assumes a critical issue. Solving this issue requires the use of optimization techniques that guarantees the achievement of acceptable solutions as optimized as possible. Among several optimization software's available on the literature, it is possible to point out the IBM ILOG software, however as most of the software's it is not freely distributed [3].

In industrial environments, characterized by the frequent occurrence of unplanned disturbances and changes, the optimal plans becomes inapplicable and obsolete very fast. This introduces a new requirement to the use of optimization techniques, where besides the quality of the calculated solution, it is also crucial to consider the time to compute the solution.

This paper studies the application of an genetic algorithm approach to determine the scheduling in an industrial factory plant organized as flexible job shop.

Flexible Job Shop Scheduling Problem (FJSSP) is an extension of the traditional Job Shop Scheduling Problem, differing from this in the sense that some workers may be capable of performing more than one type of tasks. Additionally, for each task there is at least, one worker that is capable of performing the operation.

The main objective of this work is to find the schedule of the workers, taking into account the precedence constraints, which minimizes the makespan, i.e., the finish time of the last task completed in the schedule [1, 8, 9].

This paper is organized as follows: Section 2 presents the case study to be analysed in this paper and Section 3 formalizes the optimization problem. Section 4 describes the scheduling algorithm based on the genetic algorithm principles and Section 5 analyses the achieved numerical results. At last, the last section rounds up the paper with the conclusions and points out future work.

CASE STUDY

The case study is based on a real enterprise that has a set of works orders. Consider $O = \{o_k : k = 1,...,K\}$ the set of works orders. Each work order is decomposed into one or more tasks. Consider $T_k = \{t_{kj} : j = 1,...,J_k\}$ the set of tasks of the work order $o_k \in O$. Tasks can be dependent on status of execution or the existence of dependencies between tasks. For example, the o_3 has J_3 tasks: $T_3 = \{t_{31}, t_{32}, ..., t_{3J_3}\}$.

Each task t_{kj} is associated with a start time δ_{kj}^0 and a finish time δ_{kj}^f, representable by the pair $\Delta_{kj} = \left(\delta_{kj}^0, \delta_{kj}^f\right)$. The cycle time can be extended until all tasks are finished.

The minimum execution time of a group of tasks assigned to a work order, since tasks are sequential, is given by the sum of all the assigned tasks. This execution time can be bigger if a worker, or set of workers, are scheduled to different task being executed in parallel. In this case, in the tasks where the worker is not available, the execution time must consider the busy time of the worker.

Tasks that are not fulfilled within their work order may be shifted into the subsequent work order. When this happens, the transportation time needed to move the task into the new work order has be added into the normal execution time.

Consider $R_{kj} = r_{kji} : i = 1, ..., I_{kj}$ the set of resources needed to accomplish the task t_{kj} of the work order o_k.

Each task may require a variable number of human resources depending on the skills required, if the task is delayed or passed to another work order.

Consider $W_{kj} = W_{kj}^L \cup W_{kj}^G = \left\{w_{kjp} : p = 1, ..., P_{kj}\right\}$ the set of workers with the skills to address work order o_k. Where W_{kj}^L represents the number of workers from the workers' local pool and W_{kj}^G represents the number of workers of the workers' global pool.

Each worker W_{kj} can be assigned to different tasks in different work orders o_k.

FORMALIZATION OF THE OPTIMIZATION PROBLEM

The scheduling problem is constituted by a group of work orders, being each one composed with a set of tasks. Each task has an execution time and is preformed according to its precedence. However, tasks of different work orders can be simultaneously executed. The end time of execution of a task that has precedences will be the ending time of the previous task plus the current one. The main objective of this problem formalization is to minimize the execution time of each work order and globally of all work orders.

Consider x_{ij} the variable that represents the worker associated to the task i and work order j, for $i = 1, ..., \max(J_k)$, $\forall k \in \{1, ..., K\}$ and $j = 1, ..., K$. And t_{ij} represents the execution time of each task i and work order j calculated using x.

Each worker needs to be allocated in a non-overlapped slot of time building the best distribution task/worker in order to minimize the execution time of all work orders $(f(x))$. So,

$$f(x) = max(t).$$

The optimization problem is given by

$$\min f(x).$$

To associate a worker to the task t_{kj} the workers must match with the set of necessary skills fo fulfil the designated task. If the worker has more than one task to execute then its necessary to calculate the time before and after the fixed task, i.e. check whether the worker has spare execution time. If the duration time of the new task is equal or less than the worker's free time then the task can be included in the free slot.

If the task does not have precedences then the start time of the task is delimited by the available time of the worker that will perform the task. If the task has precedences, then the start time of the task is delimited by the ending time of the previous task with the spare time of the worker that will execute the task.

The human resources will be allocated in an scheduling matrix in order to have a global view of their availability, ensuring their proper allocation. The dimension of this matrix is given by the number of workers (W_{kj}) with the total number of tasks T_k. In short, the scheduling matrix allocates the worker (W_{kj}) from the matrix $x_{i,j}$ that will execute the task T_k.

This will create a schedule matrix with the start and end times for each allocated human resource avoiding overlapped schedules.

SCHEDULING USING GENETIC ALGORITHM

To solve the flexible job shop problem the genetic algorithm (GA) is used. The genetic algorithm is a stochastic method, whose mechanism is based on the simplifications of an evolutionary process observed in nature, namely selection, mutation and crossover [5]. As opposed to many other optimization methods, genetic algorithm works with

a population of solutions instead of one single solution. In the GA, the solutions are combined to obtain new ones until a satisfactory solution is obtained, i.e. the stop criteria ie meet.

The GA uses the crossover process, where the genes of the best individuals are crossed with genes from other individuals which also have good performance. Additionally, the algorithm also applies the concept of mutation, improving the optimization process by introducing values that were not present in the previous generations. Finally, if the stop criteria is not meet, the genetic algorithm selects the best individuals to participate in the next population [6].

The optimization procedure, described before, is based by the following algorithm:

```
P = InitialPopulation();
while not StoppingCondition(P) do
    P'=Crossover(P);
    P''=Mutation(P);
    P=CalculateFitness(P ∪ P' ∪ P'');
    P=RemoveWorse(P);
end
```

Where the initial population, P, consists of N, feasible, schedules. In the context of the problem, feasibility means that all constraints are satisfied, in particular those related to time, precedence of work orders and tasks, allocation of physical resources and workers (considering the workers' skills).

The initial number of station schedules, N, is an integer - usually between 10 and 100, depending on the construction time of one schedule, and is defined by the user.

As for the stop criteria, it is fixed an upper limit for iterations and an upper limit for function evaluations.

ANALYSIS OF THE NUMERICAL RESULTS

The GA method was implemented in the Matlab software and results were compared with the ones obtained using the ILOG CPLEX [7]. The results are presented in Table 1.

Table 1 contains the basic description of work orders (o_k), tasks (t_{kj}), including their duration (Δ_{kj}) and the workers (w_{kj}) with skills to do the task t_{kj}.

TABLE 1. Initial parameters.

o_k	t_{kj}	Δ_{kj}	w_{kj}
o_1	t_{11}	900	3,5,6,12,14,25
	t_{11}	61	3,5,6,12,14,25
	t_{12}	55	3,5,6,12,14,25
	t_{13}	1	3,5,6,12,14,25
	t_{14}	1	3,5,6,12,14,25
	t_{15}	1	1,2,3,5,6,7,11,12,13,14,15,16,22,23,25,26,28,29,30,31,32,33,35,36
o_2	t_{21}	1	3,5,6,9,12,15,25
	t_{22}	50	3,5,6,9,12,15,25
	t_{23}	900	3,5,6,9,12,15,25
	t_{24}	1	3,5,6,12,14,25
	t_{25}	1	9,32
	t_{26}	1	3,5,6,12,14,25
	t_{27}	1	3,5,11,32,33,35
	t_{28}	1	2,3,5,6,7,11,12,13,14,15,16,22,23,25,26,28,29,30,31,32,33,35,36
o_3	t_{31}	900	3,5,6,12,14
	t_{32}	5	5 3,5,6,12,14,254
	t_{33}	1	54
	t_{34}	90	3,5,6,12,14,254
	t_{35}	1	3,5,11,32,33,354
	t_{36}	1	1,2,3,5,6,7,11,12,13,14,15,16,22,23,25,26,28,29,30,31,32,33,35,364
o_4	t_{41}	900	5
	t_{42}	61	5
	t_{43}	1	5
	t_{44}	1	5
	t_{45}	1	3,5,11,32,33,35
	t_{46}	1	1,2,3,5,6,7,11,12,13,14,15,16,22,23,25,26,28,29,30,31,32,33,35,36

Table 2 shows the results from both approaches, depicting the work orders (o_k), theirs tasks (t_{kj}) with the time intervals (Δ_{kj}) and the allocated workers (W_{kj}).

Regarding the optimization function, both the GA and ILOG obtained $C_{max} = 1053$. As for the computation time, the GA implementation obtained 243ms while the ILOG needed 600ms. In both cases the results are very similar changing only the workers that where selected for the tasks.

TABLE 2. Comparison betweem ILOG and MatLab results.

Task		ILOG			MatLab		
Task	Δ_{kj}	workers	t_0	t_f	workers	t_0	t_f
1	900	12	0	900	25	0	900
2	61	3	901	962	14	901	962
3	55	14	963	1018	6	963	1018
4	1	12	1019	1020	6	1019	1020
5	1	35	1021	1022	3	1021	1022
6	1	3	1023	1024	29	1023	1024
7	1	5	0	1	9	0	1
8	50	5	2	3	5	2	52
9	900	25	4	904	6	53	953
10	1	6	905	906	5	954	955
11	1	34	907	908	32	956	957
12	1	6	909	910	6	958	959
13	1	33	911	912	35	960	961
14	1	7	913	914	2	962	963
15	900	14	0	900	14	0	900
16	55	14	901	956	12	901	956
17	1	5	957	958	5	957	958
18	90	6	959	1049	12	959	1049
19	1	32	1050	1051	33	1050	1051
20	1	30	1052	1053	3	1052	1053
21	900	5	3	903	5	53	953
22	61	5	958	1019	5	959	1020
23	1	5	1020	1021	5	1021	1022
24	1	5	1022	1023	5	1023	1024
25	1	32	1024	1025	35	1025	1026
26	1	33	1026	1027	2	1027	1028

CONCLUSIONS AND FUTURE WORK

In this work it was analysed a real problem of flexible job shop scheduling. To solve the problem, two scheduling approaches were followed, namely the ILOG software and Genetic algorithm were used. The numerical results show that the GA implementation is capable to find the optimum solution in a short time.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007- 2013 under grant agreement n° 314056.

REFERENCES

1. N. Al-Hinai, T. ElMekkawy, Solving the Flexible Job Shop Scheduling Problem with Uniform Processing Time Uncertainty, *World Academy of Science, Engineering and Technology*, Vol. 64, pp. 996-1001 (2012).
2. B. Cai and S. Wang, Hybrid Genetic Algorithm for Minimizing Total Weighted Tardiness in Job Shop Scheduling Problem, *International Journal of Advancements in Computing Technology (IJACT)*, Vol 4, no. 12, July 2012.
3. L. Qin and Q. Li, A New Constrution of Job-Shop Scheduling System Integrating ILOG and MAS, *Journal of Software*, Vol 7, No. 2, pp. 269-276, 2012.
4. A. Dusseau, R. Arpaci and D. Culler, Effective Distributed Scheduling of Parallel Workloads, *SIMETRICS*, Philadelphia, PA, pp. 25-36 (1996).
5. A. Curralo, A. I. Pereira, J. Barbosa and P. Leitão, Sensibility Study in a Flexible Job Shop Scheduling Problem, *Numerical Analysis and Applied Mathematics*, Volumes I-III (1558) Book Series: AIP Conference Proceedings, September 2013.
6. M. Kumar, M. Husian, N. Upreti, D. Gupta, Genetic Algorithm: Review and Application, *International Journal of Information Technology and Knowledge Management*, 2(2), pp. 451-454 (2010).

7. IBM ILOG CPLEX Optimization Studio, 23-Jul-2014. [Online]. Available: http://www-03.ibm.com/software/products/en/ibmilogcpleoptistud. [Accessed: 23-Jul-2014].
8. C. Pach, A. Bekrar, T. Bonte, Y. Sallez, T. Berger, D. Trentesaux, P. Leitão and J. Barbosa, Benchmarking flexible job-shop scheduling and control systems, *Control Engineering Practice*, Vol. 21, pp. 1204-1225 (2013).
9. R. Zhang, S. Song and C. Wu, A hybrid artificial bee colony algorithm for the job shop scheduling problem, *Elsevier - Int. J. Production Economics*, Vol. 141, pp. 167-178 (2013).