International Journal of Engineering Innovation & Research
Volume 2, Issue 2, ISSN: 2277 – 5668

Solving the Job-Shop Scheduling Problem by Arena Simulation Software

Prof. Dr. Gamal M. Nawara
Zagazig University, Faculty of Engineering,
Industrial Engineering Department

Eng. Wael S. Hassanein
Zagazig University, Faculty of Engineering,
Industrial Engineering Department
Email: wsalsaid@gmail.com

Abstract **– The job-Shop Scheduling problem (JSSP) attracted a lot of researchers from various research disciplines, mainly Operations Research, Management Science, Computer Science, and Manufacture Science for the last 50 years. JSSP is a typical NP-hard problem in the strong sense. Although the literature is full of researches concerning the JSSP, practitioners are not able to get benefit of the majority of these researches because of the assumptions which take the problem very far away from the real life JSSP. The aim of our research is to build a simulation model for the JSSP to be able to relax some of these assumptions to simulate the real life JSSP. We used discrete event simulation as it's suitable for the JSSP. We used Arena simulation software version 14 to build the model on a Dell® Vostro PC (Intel® Core(TM) i5–2400 CPU @ 3.10GHZ with 4 GB RAM).In this paper we will just show the basic model which is able to solve the famous benchmarks for the JSSP to prove that our model is ready for the real life JSSP. In the following papers we will show how to relax some of these assumptions one by one. The computational results for 9 benchmarks of different sizes showed that the proposed model is both effective and efficient. It gave good solutions in reasonable amounts of time.**

Keywords **— Job-Shop Scheduling Problem, Makespan, Simulation, Arena.**

I. Introduction

The job shop scheduling problem is a practical problem and has numerous applications in manufacturing and supply chain scheduling problems [1]. Job-Shop scheduling refers to assigning jobs to machines satisfying the precedence and resource constraints over time such that certain objective(s) is optimized, These objectives can be minimization of makespan, minimization of maximum lateness, or minimization of the number of tardy jobs etc. The JSSP for the number of machines ≥ 2 is a typical NP-hard problem in the strong sense[2], which means that the number of possible schedules grows exponentially with the number of orders. In history, the famous test problem named FT10 which consists of 10 jobs and 10 machines, developed by Muth and Thompson [3] took more than 20 years until solved by Carlier and Pinson [4]. Because of the JSSP complexity some researchers tried to solve the problem just to justify their models or heuristics and prove that they are capable of handling such complex problems. After these decades and this huge number of researches, still there is a big gap between the analytical work and the real life problem. Pinedo [1] stated that advances in scheduling theory have had only a limited impact on scheduling in practice, this doesn’t mean that the theoretical research has been a complete waste of time because it has given insights into the scheduling problem. In this research we want to reduce this gap, we also want to give the practitioners a friendly interface program which they can deal with comfortably. Discrete event simulation is very promising in solving the real life JSSP. We chose Arena simulation software as it's a very powerful tool plus it's very famous as simulation software and a lot of the practitioners are familiar with it. Also we can get very beneficial statistical reports from Arena after each run containing some valuable information like; machine utilization, idle time for each machine, queue length, waiting time, and other information.

II. Literature Review

A very big number of different methods were used to solve the JSSP. These methods are classified into different approaches, two of them are common in all the researches like for example these recent papers [5, 6, 7, and 8]. First approach is exact methods, such as branch and bound, linear programming and Lagrangian relaxation. These exact methods guarantee global convergence and have been successful in solving small instances. However, they require a very high computing time as the size of problem increases plus they are not capable of dealing with stochastic problems. Second approach is Approximation Methods, such as the shifting bottleneck approach, particle swarm optimization, ant colony optimization, simulated annealing, Tabu search, genetic algorithm, neural network, immune algorithm, differential evolution and others. Some of the researchers added one or two additional approaches for example in [9] the authors added dispatching rules and simulation-based approach as a third one, in which we can build a simulation model based on dispatching rules and this is what we will do in this research. Additional two approaches were added by [10] which are, rule-based approach, and simulation approach. Recently researchers are adopting hybrid techniques in which two or more methods are combined in order to get the advantages of all used methods as in [5, 11, 12, and 13]. The author in [5] collected 14 techniques which are called state-of-the-art algorithms, these algorithms are a combination of recent and older solution techniques, some of them constituting the best approximation algorithms for the JSSP. These state-of-the-art algorithms are; SB [14], SBGA [15], TSAB [16], I-TSAB [17], GRASP [18], HGA [19], ACOFT [20], TSSA [21], BV [22], TSSB [23], DS [24], FL [25], SVS [26], and HA [27].

III. Problem Description

The definition and problem assumptions for the JSSP is available in the literature in many references, here is the definition and problem assumption from two recent papers [6, 7]. In the JSSP there are n jobs and m machines. Each job has a fixed processing route which traverses some or all the machines in a predetermined order. The manufacturing process to be performed on one machine is called an operation of the job, and each operation should be processed on a particular machine. The objective is to schedule operations on machines so that the makespan (maximum completion time) is minimized. Problem assumptions can be stated as follows:

1. Processing times are deterministic.
2. All jobs are ready to be processed at time zero.
3. Only one job can be processed on each machine at a given period of time.
4. Each job visits each machine once at most.
5. The machines are continuously available.
6. There's only one machine of each type of machines.
7. No preemption of operations is allowed.
8. The transportation times between different machines are neglected.
9. The setup times for different jobs are neglected or added to the processing time.

The previous definition and assumptions are for the classical JSSP which we will handle in this paper to show our basic model by Arena software. Even this classical JSSP with these assumptions is an NP-hard problem which researchers for the last 5 decades plus handled its benchmarks to find optimum solutions for them or to prove their different methods with this complex problem.

IV. Proposed Model

Now we will build our model using Arena software. The model will export some information to Microsoft Excel file as we will show. In the beginning we have to prepare an Excel file, we have to define names for certain fields which we will use to export the information, we will name it results. After preparing the excel file we are ready to build the model. We will use an example to show our solution methodology, the example consists of 4 machines, 4 jobs as shown in table 1 it's required to give the best schedule with the minimum *makespan*.

Table 1: A 4x4 JSS example

J	Processing Sequence (Processing time)			
1	2(2)	1(1)	4(2)	3(1)
2	4(3)	2(1)	3(3)	1(2)
3	1(1)	3(3)	4(2)	2(1)
4	3(2)	2(2)	1(2)	4(3)

The model will be divided into three parts:

- *Part one:* will be responsible for creating jobs, defining all variables and attributes, releasing jobs, and starting sequencing.
- *Part two:* will be responsible for the machines, the manufacturing process, and the exportation of the information to the Excel file. This part will be done for each machine. The same process will be repeated for all machines.
- *Part three:* In this part jobs will exit the manufacturing system, and the system will dispose the parts.

Now we will discuss the three parts in details:
Part one will include the four modules shown in Fig. 1 as follows:

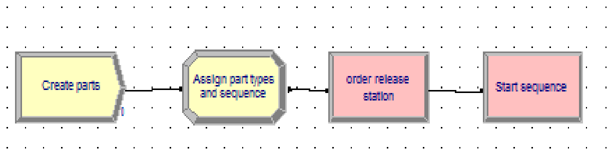

Fig.1. Part 1

- *Create* module from the Basic Process project bar, *time between arrivals* will be constant, value will be (0), *entities per arrival* will be (1), and *maximum arrivals* will be (4).
- *Assign* module also from the Basic Process project bar in which we will define all variables and attributes concerning: part index, entity type, entity sequence, and an attribute to generate random numbers which we will call random.
- *Station* module from Advanced Transfer project bar which we will use to release jobs.
- *Route* module from Advanced Transfer project bar which we will use to start sequencing.

Part two will include the four modules for each machine shown in Fig. 2 as follows:

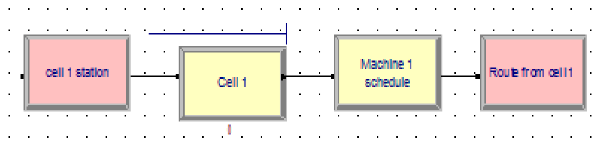

Fig.2. Part 2

- *Station* module from Advanced Transfer project bar which we will use to receive jobs.
- *Process* module from the Basic Process project bar, it will represent the machine and simulating the manufacturing process, in the field *Action* we will select *Seize Delay Release*, for *Resources* we will add a resource, name it (for example cell1), and put the quantity for it (here it's one), in the *Delay Type* we will choose *Expression* and it will be *process time*.
- *ReadWrite* module from Advanced Process project bar which we will use to export some information to the Excel file. In *type* we will select *Write to file*, in *Arena file name* we will select the predefined Excel file, in the *Recordset ID* we will select the predefined name for the three fields in the Excel sheet, and in *assignments* we will add the *Entity.Type* as an

attribute, the *Process Time* as an attribute, and *Entity.StartTime* as *Other*.

- *Route* module from Advanced Transfer project bar which we will use to release the jobs by sequence. In *Destination Type* we will choose *By Sequence*.

Part three will include the two modules shown in fig 3 as follows:

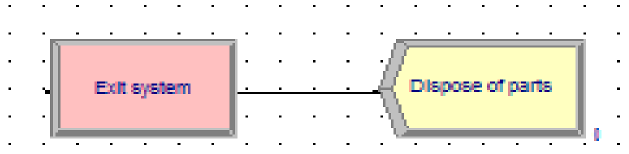

Fig.3. Part 3

- ***Station*** module from Advanced Transfer project bar which we will use for the jobs to exit the manufacturing system.
- ***Dispose*** module from the Basic Process project bar which means that the system will dispose the jobs.

Still two things we have to do in the Arena model to be ready to run it:

1. From the Advanced Process project bar we will select *Statistic*, in the *Name* we will write makespan, in *type* we will select *output*, in Expression we will build an expression to get the makespan which is the completion time of all jobs, and finally in *output file* we will name a file makespan.dat and put it in the same folder with the model. This file (makespan.dat) will record the makespan for each replication, we will open this file later with the Output Analyzer.
2. From the Advanced Process project bar we will select *File*, in the *Name* we will write results in *Access type* we will select Microsoft Excel, in *Operating System File Name* we will select the predefined Excel File, and finally in *Recordsets* we will add the named ranges from the Excel file. This is the file which will receive the exported data from the *ReadWrite* module.

V. Verification and Validation

Model verification and validation are critical in the development of a simulation model [28]. Verification ensures that the model is conducted correctly, while validation ensures that the model represents the real system and that the model is truly representative of that system. Sargent [28] mentioned different approaches to verify and validate any simulation model, then in his research summary he said ". Unfortunately, there is no set of specific tests that can easily be applied to determine the correctness of a model. Furthermore, no algorithm exists to determine what techniques or procedures to use. Every simulation project presents a new and unique challenge to the model development team". To verify our model here we should just review the data entered to the model. We will run the model with FIFO as a scheduling rule and go to the Excel file to check the resulting schedule for each machine. Table 2 shows the schedule for machine one as an example, column 1 represents the job sequence, column 2 represents the operation time, column 3 represents the operation start time, and finally column 4 represents the operation finish time. We can check the operations times and the feasibility of the solution, if it’s ok so our model is verified. We can also generate the Ganttchart as in Fig. 4. Because our example here is small we can make the FIFO schedule manually and compare it to the generated one from the model, and this will be the validation for the model. Now we will use the model to solve 9 famous benchmarks with different sizes from the literature to make sure that our model is representing the JSSP and it's capable of handling such problems.

Table 2: Schedule for machine 1 for the FIFO scheduling rule

Machine1			
Job No.	Processing time	Start time	Finish time
3	1	0	1
1	1	2	3
4	2	4	6
2	2	8	10

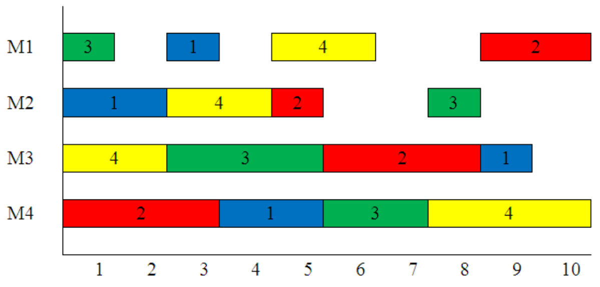

Fig.4. Ganttchart for the FIFO scheduling rule

VI. Results and Discussion

Now we will show how to run the model for any number of replications then get the minimum makespan and the related schedule. From *Run* menu we will select *setup* then *Replication Parameters* and put for example *100* in the *Number of Replications*. We have to change the dispatching rule for all queues to Highest Attribute Value, from the Basic Process project bar we will select *Queue*, go to *Type*, and select *Highest Attribute Value*, then in the *Attribute Name* we will select *Random* the attribute we prepared before in the *Assign* module to generate random numbers. Now we are ready again to run the model, this time for 100 replications and random queue discipline.

After running the model we will get a file with the name of results.dat, we will open this file with the output analyzer and generate a DIF file with the same name. This DIF file is readable by Microsoft Excel, we will find 2 rows the first one will contain the replication number while the second one will contain the Makespan. From Excel we can get the minimum makespan with the corresponding replication number(s), it's expected to have more than one solution with the same makespan and any one of them is accepted.

After getting the minimum makespan and the number of corresponding replication we will go to the Excel file (results), go to the required replication and we should find all required scheduling information for each machine containing; the processing sequence, the processing time for each operation, start and finish times for each operation. We can also generate Ganttchart from these information.

There's no guarantee that this is the optimum solution, and of course we can raise the number of replications as we want, the only constraint is the time consumed to get the solution for the model and it depends on the computer power.

As we said we will test our model with several benchmark instances with different sizes, these benchmarks and many others could be found in the OR library [29] and other references like [30, 31, 32, 33, 34, and 35]. Table 3 shows different values for best makespan for different number of replications with their corresponding computation times. We will run the model with the FIFO scheduling rule then with the random scheduling rule for 100, 1000, and 2000 replications. We will use the same formula as in [5] to calculate the Makespan Relative Error (MRE) of the best makespan found by our model. Makespan Relative Error of the best makespan f_{best} found, with respect to a reference makespan which is typically a lower bound LB. The relative error is expressed as a ratio (MRE = 0 indicates that the corresponding instance is solved optimally) and is computed according to the following formula, MRE = 100 × [(f_{best} – LB)/LB]. This formula is originally from [36], For example, for the first problem FT06 the optimum solution is 55 and the best makespan we got is 58 so the MRE = 100 x [(58-55)/55] = 5.45.

We recorded two times to the nearest minute, first one is the time for simulating all replications and the second one is the time to generate the reports with the required statistical analysis. We will not use the reports in the basic model as we exported all needed information to the DAT file, and the Excel file.

Table 3: Different benchmarks with their solutions

Problem	Size (jobs x machines)	Optimum or LB	FIFO	Proposed model			MRE
				No. of replications	Makespan	Time (min.)	
Ft06	6x6	55	65	100	59	0 + 0	5.45
				1000	58	1 + 1	
				2000	58	3 + 2	
Ft10	10x10	930	1184	100	1120	0 + 0	13.76
				1000	1075	2 + 1	
				2000	1058	3 + 3	
La01	10x5	666	772	100	690	0 + 0	3.6
				1000	690	2 + 1	
				2000	690	3 + 2	
La06	15x5	926	926	100	946	0 + 0	0
				1000	926	2 + 1	
La11	20x5	1222	1272	100	1272	0 + 0	0
				1000	1222	2 + 2	
La21	15x10	1046	1265	100	1228	0 + 0	12.7
				1000	1179	3 + 2	
				2000	1179	4 + 6	
Ta01	15x15	1231	1486	100	1385	0 + 0	12.5
				1000	1385	4 + 5	
				2000	1385	7 + 20	
Abz7	20x15	655	803	100	835	0 + 0	17.6
				1000	786	5 + 5	
				2000	770	8 + 25	
Yn1	20x20	846	1085	100	1056	0 + 0	20.8
				1000	1022	6 + 6	
				2000	1022	11 + 30	

VII. Conclusion

In this paper, we introduced a simulation model for the JSSP using Arena Simulation program which is a well-known simulation program and the majority of industrial and production engineers are familiar with it. We also integrated Arena and Microsoft Excel by exporting some information from Arena to Microsoft Excel for further analysis. We examined our model by solving 9 benchmarks with different sizes and the results were reasonable we got near optimum solutions in reasonable times.

VIII. Future Work

In the following papers, we will show how we can relax the assumptions one by one in order to reduce the gap between the analytical work and the real life problem. For example we are planning to include the transfer time between machines which is assumed to be zero as in the assumptions, we will allow parallel processing which means that there will be more than one machine of each type, Jobs will be allowed to use any machine more than once, we will consider machines breakdowns, we will consider stochastic processing times, all jobs will not be available at time zero, and other factors. Regarding the objective function we will consider other objectives like minimizing tardy jobs, minimizing lateness or tardiness, we will also consider multi objective function combining two or more objectives, and we might include costs related to jobs or machines.

References

[1] Pinedo M. Scheduling: theory, algorithms, and systems. 3rd Ed. New York, NY: Prentice Hall; 2008.
[2] J.K. Lenstra, A.H.G.R. Kan, P. Brucker, Complexity of machine scheduling problems, Annals of Discrete Mathematics 7 (1977) 343–362.
[3] J. F. Muth and G. L. Thompson. Industrial Scheduling. Englewood Cliffs, New Jersey,1963.
[4] J. Carlier and E. Pinson. An algorithm for solving the job-shop problem. Management Science, vol. 35,1985, pp.164-176.
[5] Antonin Ponsich, Carlos A. CoelloCoello, A hybrid Differential Evolution—Tabu Search algorithm for the solution of Job-Shop Scheduling Problems, Applied Soft Computing, Volume 13, Issue 1, January 2013, Pages 462-474, ISSN 1568-4946, 10.1016/j.asoc.2012.07.034.
[6] Abbas Ebadi, GhasemMoslehi, An optimal method for the preemptive job shop scheduling problem, Computers & Operations Research, Volume 40, Issue 5, May 2013, Pages 1314-1327, ISSN 0305-0548, 10.1016/j.cor.2012.12.004.
[7] Rui Zhang, Cheng Wu, Bottleneck machine identification method based on constraint transformation for job shop scheduling with genetic algorithm, Information Sciences, Volume 188, 1 April 2012, Pages 236-252, ISSN 0020-0255, 10.1016/j.ins.2011.11.013.
[8] Beizhi Li, Shanshan Wu, Jianguo Yang, Yaqin Zhou, Min Du, A three-fold approach for job shop problems: A divide-and-integrate strategy with immune algorithm, Journal of Manufacturing Systems, Volume 31, Issue 2, April 2012, Pages 195-203, ISSN 0278-6125, 10.1016/j.jmsy.2011.05.005.
[9] Andersson, M.; Grimm, H.; Persson, A.; Ng, A.; , "A web-based simulation optimization system for industrial scheduling," Simulation Conference, 2007 Winter, vol., no., pp.1844-1852, 9-12 Dec. 2007. doi: 10.1109/WSC.2007.4419811
[10] Kaban, a. K., Othman, Z., &Rohmah, D. S. (2012). Comparison of dispatching rules in job-shop scheduling problem using simulation: a case study. International Journal of Simulation Modelling, 11(3), 129–140. doi:10.2507/IJSIMM11(3)2.201
[11] Ren Qing-dao-er-ji, Yuping Wang, A new hybrid genetic algorithm for job shop scheduling problem, Computers & Operations Research, Volume 39, Issue 10, October 2012, Pages 2291-2299, ISSN 0305-0548, 10.1016/j.cor.2011.12.005.
[12] Mohammad Mahdi Nasiri, FarhadKianfar, A GES/TS algorithm for the job shop scheduling, Computers & Industrial Engineering, Volume 62, Issue 4, May 2012, Pages 946-952, ISSN 0360-8352, 10.1016/j.cie.2011.12.018.
[13] Meeran, S. and Morshed, M. S., 2012. Forthcoming. A hybrid genetic tabu search algorithm for solving job shop scheduling problems: a case study. Journal of Intelligent Manufacturing, 23 (4), pp. 1063-1078
[14] J. Adams, E. Balas, D. Zawack, The shifting bottleneck procedure for job shop scheduling, Management Science 34 (1988) 391–401.
[15] U. Dorndorf, E. Pesch, Evolution based learning in a job shop scheduling environment, Computers and Operations Research 22 (1) (1995) 25–40.
[16] E. Nowicki, C. Smutnicki, A fast Taboo Search algorithm for the job shop problem, Management Science 42 (6) (1996) 797–813.
[17] E. Nowicki, C. Smutnicki, An advanced Tabu Search algorithm for the job shop problem, Journal of Scheduling 8 (2) (2005) 145–159.
[18] R.M. Aiex, S. Binato, M.G.C. Resende, Parallel GRASP with path-relinking for job shop scheduling, Parallel Computing 29 (4) (2003) 393–430.
[19] J.F. Goncalves, J.J. de Magalhaes Mendes, M.G.C. Resende, A hybrid genetic algorithm for the job shop scheduling problem, European Journal of Operational Research 167 (1) (2005) 77–95.
[20] K.L. Huang, C.J. Liao, Ant colony optimization combined with Taboo Search for the job shop scheduling problem, Computers and Operations Research 35 (4) (2008) 1030–1046.
[21] C.Y. Zhang, P. Li, Y. Rao, Z. Guan, A very fast TS/SA algorithm for the job shop scheduling problem, Computers and Operations Research 35 (1) (2008) 282–294.
[22] E. Balas, A. Vazacopoulos, Guided local search with shifting bottleneck for job shop scheduling, Management Science 44 (2) (1998) 262–275.
[23] F. Pezzella, E. Merelli, A Tabu Search method guided by shifting bottleneck for job shop scheduling problem, European Journal of Operational Research 120 (2000) 297–310.
[24] U. Der, K. Steinhöfel, A parallel implementation of a job shop scheduling heuristic, in: PARA'00: Proceedings of the 5th International Workshop on Applied Parallel Computing, New Paradigms for HPC in Industry and Academia, Springer-Verlag, London, UK, 2001, pp. 215–222.
[25] S. Fernandes, H.R. Lourenc¸ o, A simple optimised search heuristic for the jobshop scheduling problem, Economics Working Papers 1050, Department of Economics and Business, UniversitatPompeuFabra, 2007, January.
[26] P.S. Velmurugan, V. Selladurai, A Tabu Search algorithm for job shop scheduling problem with industrial scheduling case study, International Journal of Soft Computing 2 (4) (2007) 531–537.
[27] S.M. KamrulHasan, R.A. Sarker, D. Essam, D. Cornforth, Memetic algorithms for solving job-shop scheduling problems, Memetic Computing 1 (1) (2009) 69–83.
[28] Robert G. Sargent, Verification and Validation of Simulation Models, Proceedings of the 2010 IEEE Winter Simulation Conference, B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, eds., pp. 166-183.
[29] http://people.brunel.ac.uk/~mastjjb/jeb/orlib/jobshopinfo.html
[30] J. Adams, E. Balas, D. Zawack, The shifting bottleneck procedure for job shop scheduling, Management Science 34 (1988) 391–401.
[31] D. Applegate, W. Cook, A computational study for the job-shop scheduling problem, ORSA Journal on Computing 3 (2) (1991) 149–156.

[32] S. Lawrence, Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques (supplement), Technical report, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh (Pennsylvania), USA, 1984.
[33] T. Yamada, R. Nakano, A Genetic algorithm applicable to large-scale job-shop instances, in: Manner, Manderick (Eds.), Parallel instance solving from nature 2, North-Holland, Amsterdam, 1992, pp. 281–290.
[34] E.D. Taillard, Benchmarks for basic scheduling problems, European Journal of Operational Research 64 (2) (1993) 278–285.
[35] H. Fisher, G.L. Thompson, Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules, Industrial Scheduling Edition, Prentice Hall, Englewood Cliffs, NJ, USA, 1963, pp. 225–251.
[36] P.J.M. Van Laarhoven, E.H.L. Aarts, J.K. Lenstra, Job shop scheduling by simulated annealing, Operations Research 5 (1) (1992) 113–125.

AUTHOR'S PROFILE

Prof. Dr. Gamal M. Nawara

He has a Ph.D. in Industrial Engineering, Leipzig University, Germany, 1969. Senior member in IEE society. Ex-Vice, President for Graduate Studies and Research, Zagazig University Egypt. Ex-Dean, Faculty of Engineering, Zagazig University Egypt. Currently Secretary General Council of Private Universities, Egypt.He is interested inProduction and Maintenance Planning and Control.

Eng. Wael S. Hassanein

He got his B.Sc. and M.Sc. in industrial engineering, Zagazig University, Egypt, 1998 and 2005. Currently he is a PhD student, industrial engineering, Zagazig University, Egypt.He is interested in Operations research, production planning and simulation.