Genetic algorithm constraint project scheduling

M.A. Shouman [a], M.S. Ibrahim [b], M. Khater [b] and A.A Forgani [b]
[a] *Research and Decision Support System Dept., Faculty of Computers and Informatics, Zagazig university, Zagazig, Egypt*
[a] *Industrial Eng. Dept., Faculty of Eng., Zagazig university, Zagazig, Egypt*

The current paper describes a genetic algorithm approach to single and multiple resource-constrained project scheduling problems using the permutation encoding-based representation. The gene of each chromosome represents an activity and its position represents the sequences of the activity to be scheduled. The approach is tested on a set of fifty test problems as benchmark problems. The computational results validate the effectiveness of the proposed algorithm.

تعتبر مشكلة جدولة المشروعات ذات الموارد المحدودة الأحادية والمتعددة من المشاكل الصعبة فى حلها حيث أن هناك العديد من كل من المتغيرات والحلول الممكنة والتى يصعب معها إيجاد حل فى وقت معقول فى هذا البحث تم إقتراح خوارزم جينى لجدولة المشروعات ذات الموارد المحدودة الأحادية والمتعددة وقد تم إختيار هذا الخوارزم على عدد خمسون مشروعا مرجعيا قد تم استخدامهم فى أبحاث سابقة بالإضافة إلى مقارنة نتائج جدولة هذه المشروعات باستخدام الخوارزم المقترح مع جدولتها بإستخدام عدد (٥٥) قاعدة استكشافية للجدولة وقد حقق الخوارزم المقترح نتائج أفضل من نتائج القواعد الإستكشافية بما يزيد عن ٧٠% من المشاريع المستخدمة فى تقييم الخوارزم بالإضافة إلى نتائج المشاريع الأخرى قد حقق تجاوز مقدارة ٠,٠٨٩% من نتائج القواعد الإستكشافية مما يدل على قوة ومستوى أداء الخارزم المقترح فى جدولة المشروعات ذات الموارد المحدودة الأحادية والمتعددة.

Keywords: Project management, Genetic algorithm, Scheduling techniques, Metaheuristics

1. Introduction and literature survey

Within the classical Resource-Constrained Project Scheduling Problem (RCPSP), the activities of a project have to be scheduled such that the make span of the project is minimized. Thereby, technological precedence constraints have to be observed as well as limitation of renewable resources required to accomplish the activities. The traditional used tools have serious limitation in practices although they assume unlimited availability of resources [1]. Further more, they are applied to only one project at a time. In the current paper a genetic algorithm approach for single and multiple resource-constrained project-scheduling problem is proposed as a solver scheme for project scheduling process. The proposed genetic algorithm uses the permutation encoding-based representation. Intensive computational experimentation tests are used to evaluate the performance of the proposed algorithm using data set of fifty test problems are benchmark instances.

Recently, some researchers have suited various algorithms for solving large-scale RCPSP problems. Their works have dealt with a variety of situation in which one or both of constraints types are relaxed, or at least simplified. Goncalves et al. [1] presents a genetic algorithm of Resource Constrained Multi-Project Scheduling Problem (RCMPSP). The schedules are constructed using a heuristic that builds parameterized active schedules based on priorities, delay times, and release dates defined by the genetic algorithm. The approach is tested on a set of randomly generated problems. The computational results validate the effectiveness of the proposed algorithm. Kim et al. [2] develop a hybrid genetic algorithm with fuzzy logic controller to solve RCPSP. The approach is based on the design of genetic operator with fuzzy logic controller and of serial method initialization. The hybrid system is tested with the different genetic operators in order to have better optimal make span schedule. Hartmann [3] presents a genetic algorithm for scheduling projects of multiple models of activity execution. The genetic encoding is based on a precedence feasible list of activities and a mode assignment. The results obtained show that the proposed algorithm out performs the other heuristic procedures with regards to a lower average deviation from the optimal make span. Valls et al. [4] introduce a new met a heuristic algorithm for RCPSP. The algorithm is non-standard implementation of fundamentals concepts of Tabu search without explicitly using memory structures embedded in a population-based framework. The procedures use a fan search strategy to intensify the search and implementation employs the technological order representation of schedules. Extensive computational tests show the merit of the proposed solution methodology. Fleszar and Himdi [5] present a solution scheme based on variable neighborhood search for solving RCPSP. The solution is ordered by using activity sequences that valid in terms of precedence constraints. The sequences are turned into valid active schedules through a serial scheduler. The search of solution space is carried out vial generating valid sequences using two types of move strategy. The effectiveness of the solution scheme is demonstrated through extensive experimentation with standard set of benchmark problem instances. Zhang et al. [6] develop partial swarm optimization based schemes for RCPSP. The potential solution to RCPSP in view of minimizing project duration is presented by the multidimensional particle, where two-solution representation, i.e, priority-based representation and permutation-based representation, are presented to investing the performance of the proposed algorithm. Kolisch and Hartmann [7] present an experimental investigation of heuristic for RCPSP. The investigation considers the heuristics for the well-known RCPSP. The study summarizes and categorizes a large number of heuristics. These heuristics are evaluated in a computational study and compared on the basis of a standard experimental design. The study discussed the features of good heuristics and presented the recent developments in heuristics for RCPSP. Drezet and Tecquerd [8] present a multi-constrained project scheduling scheme in which the financial aspects of project scheduling are considered as an objective with the make span minimization. This scheme classifies RCPSP into different categories and presents how the financial aspects can be treated for each category. Hartmann [9] develops a self-adapting genetic algorithm for project scheduling under resources constraints conditions. The scheme employs the well-known activity list representation and considers two different decoding procedures. An additional gene in the representation determines which of the two decoding procedures is actually used to compute a schedule for an individual. Computational experiments shows that the proposed mechanism is capable of exploit the benefits of both decoding procedures and are considered as one of the best ones for RCPSP. The metaheuristic methods or the new generation of heuristic algorithms normally include Simulated Annealing (SA), Tabu Search (TS), and Genetic Algorithm (GA). SA searches for better solutions through repetitive improvement on current solutions. Boctor [10], Hee and Kim [11], and Bouleimen and lococqu [12] have applied SA on RCPSP. TS start with a feasible solution and keep improving it in successive iterations so that a local optimum may be escaped in pursuit of global optimum. Its application to RCPSP includes the works of Pinson et al. [13], Lee and Kim [11], and Baar et al [14]. GA is based on the mechanisms of evaluation and natural genetics and has been applied to solve RCPSP [2,3,9,11,15, and 16]. The three metaheuristic schemes have some common features such as starting with initial solutions and updating them from iteration to iteration. Comparisons of the solution solving schemes for RCPSP show that GA and SA have better performance than TS in addition that the metaheuristic schemes generally outperform the exact or heuristic methods [6].

2. Resource-constrained project scheduling problem

RCPSP is normally characterized by objective functions, features of resources and permutation condition [11]. Minimization of project duration is often used as an objective of the RCPSP, while other objectives such as minimization of total project cost and leveling of resource usage are also considered. Resources involved in a project can be renewable or non renewable. Preemption means the activities can be interrupted while non-preemption means the activities are not allowed to stop once in progress. The traditional classical version of RCPSP can be characterized by:

- Single project consist of number of activities with definite specified duration,
- The start time of each activity is dependant upon the completion of some other activities (precedence and dependency constraints),
- The resources considered may be single or multiple resource types and are available in limited quantities and of renewable mode from period to another,
- Only one execution mode for each activity is available where no interruption is allowed, and
- The targeted objective is to minimize the project makespan.

The classical RCPSP that consider renewable resources, non preemption and minimizing project makespan can be mathematical formulated as cited in [6] as:

$$Min\,\{max\ F_i \mid i = 1,2,\ldots,N\}, \tag{1}$$

subject to

$$F_j \leq F_i\text{-}D_i,\ \text{for } j \in P_i,\ i = 1,2,\ldots,N \tag{2}$$

$$\sum_{At} r_{ik} \leq R_k,\ K=1,2,\ldots,K;\ \ t=s1,\ s2,\ldots,SN. \tag{3}$$

Where N is the number of the activities involved in a project.

F_i is the finish time of activity Ai,
D_i is the duration of activity a_i,
P_i is a set of preceding activities or predecessors of activity Ai,
R_k is available amount of resource k,
K is the number of resource types, and
R_{ik} is the amount of resource K required by activity Ai.
At is a set of ongoing activities Ai.

Formula (1) represents the objective, while, formula (2), and (3), respectively, represents precedence constraints and resource constraints.

3. Proposed genetic algorithm

Genetic search is implemented through genetic operators and directed by selection pressure. Usually, crossover operator is used as the main genetic operator and the performance of a genetic system is heavily depended on it. Mutation operator is used as a background operator, which produces spontaneous random change in various chromosomes. In order to find the best schedule with minimum makespan and alternate schedules, several genetic operators for solving RCPSPs are used. In the following subsections, the proposed genetic variables will be discussed in details. Such chromosome representation, initial population, selection method, crossover and mutation operators, and reproduction system…etc.

4. Chromosome representation

The permutation encoding is used to represent the problem. Where each gene in the chromosome represents an activity and its position represents the sequence of that activity to be scheduled (i.e the order of an activity in the permutation of the activities means the priority the activity is scheduled to start. So the permutation-based representation actually indicates the sequence to start the activities). An activity in the permutation must appear in a location after all its predecessors. Fig. 1 exhibits a project through which parent 1 and 2 in fig. 2 are considered as feasible solutions for that project. This project has eleven activities and the arrows of the figure present the dependency relationships. The starting activities are activity one and two while the terminating activities are activity eleven and eight.

5. Initial population

The initial population of chromosome is generated randomly. For not generating illegal chromosome, each process of generating a random gene (activity) checks the previous genes. This procedure to ensure that each gene is only once chosen in a chromosome and no violence in the precedence relationships.

6. Evaluation (fitness)

The evaluation criterion in the current work is the makespan of the RCPSP problems.

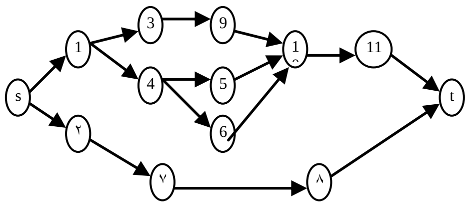

Fig. 1. Example project.

7. Selection method

The selection criterion is used to select two parents to apply the crossover operation. In the literature, a typical selection method gives a higher priority to fitter individuals since this leads to a faster convergence of the GA. The tournament selection method [17] is applied in the proposed GA in order to control convergence speed by the tournament size. In this study, the tournament size is 2.

8. Crossover operation

The crossover operation corresponds of concept of mating. It is hoped that the crossover of good parents may produce good offspring. The partially mapped crossover [18,19] is used with probability 0.5 to generate two offspring. And because of existence the precedence constraints, the manner of PMX to maintain the precedence relationships among the activities is changed to generate feasible solutions. Essentially, it takes some genes that are located between two cutting points generated at random from one parent and fills vacuum position with genes from the other parent by a left-to-right scan fig. 2.

9. Mutation operation

If the entire population has only one type of string then the crossover of two chromosomes does not produce any new ones. The mutation operation is used to escape from this scenario. The swap mutation [20] operator is used here, which simply selects two positions (activities) at random and swaps their contents with no violence on the precedence constraints. That is, in fig. 3. positions two and five will be legally swapped because their precedence constraint satisfied. The mutation probability that found to be very efficient in the current study is 0.03.

10. Reproduction system

The generation-based system is used. That is, γ offsprings from μ parents (population size) are produced and the best μ chromosome of γ are retained. Elitism method [21] is also used to prevent losing the best solution in old population from the new population. Elitism means that at least one best solution is copied without changing to the new population, so the best solution found can survive to end run. In this study, the number of elite solution is the best one only.

				Two cut points						
Parent 1	1	3	4	5	2	7	8	9	6	10	11
Parent 2	2	7	1	4	3	6	5	9	10	11	8
Offspring	1	3	4	2	7	5	9	8	6	10	11

Fig. 2. PMX crossover operation used in the proposed GA.

Parent	1	3	4	5	2	7	8	9	6	10	11
Child	1	2	4	5	3	7	8	9	6	10	11

Fig. 3. Swap mutation method.

11. Termination criteria

The GA is stopped when the number of iteration equals to the maximum iteration number. In the current study, maximum iteration number is 200.

12. Algorithm steps

Fig. 4 exhibits flow of logic in the proposed GA search procedure for RCPSPs problems and the algorithm steps as follow:
Step 1: Input the project data (no. of activities, precedence relationships, and resources…).
Step 2: Randomly generate the initial population.
Step 3: Evaluate the population's individuals and elite the best one.
Step 4: Using tournament selection method with tournament size 2, select the first and the second parent.
Step 5: Apply the crossover operation with probability 0.5 to generate two feasible children; otherwise the two parents become two children.
Step 6: Apply the mutation operation with probability 0.03 through each generated child.
Step 7: Apply steps (4 to 7) until the new population is completed. Apply elitism by copying the best individual in the old population to the new population.
Step 8: Apply steps (3 to 8) until the stop criterion is achieved.
Step 9: Save best project schedule.

13. GA parameters

The evolutionary environment for the fifty tested projects is set as recommended for the optimization problems in [4] as follow: population size of 30, crossover rate (probability) of 0.5, mutation rate of 0.03, tournament size of 2, elitism size is 1, and maximum generation of 200.

14. Test projects

The present genetic algorithm is applied on 50 data set projects. Most of these projects have been used as investigated projects in references [22-37]. The number of activities of these test problems ranges from 10 to 65, the length of the calculated critical path of these projects ranges from 10 units of time to 121 units of time, the number of nodes from 7 to 40 nodes, maximum number of critical paths exists in project is three critical paths where the number of critical paths is considered as a parameter of project complexity, and the degree of complexity measure suggested by Shouman et al. [37] for these data set ranges from 8.13 to 98.46 complexity index value. The experiments have been done considering only single orientor critical resource (R1) for scheduling process then the experiments repeated for only single orientor critical resource (R2) as main parameter for scheduling process and finally the scheduling process is directed for multiple critical resources (R1) and (R2) as dual required resources for each project of the data set. The max resource (s) required by any activity included in the project is (are) considered as the availability limit (s) through which the scheduling processes are obtained by the proposed algorithm.

15. Results

Table 1 lists the best and average project makespan after 10 runs for the test projects when single and multiple resources are considered. The average of ten runs each of 200 iterations is considered for the exhibited data. These 200 iterations are considered as the termination criterion and recommended to attain all the best schedules for the project under consideration. Table 2 presents the best project makespan using fifty-five heuristic rules advised by Shouman et al. [38] for the same fifty test projects. The makespan is considered as a measuring performance criterion for the proposed genetic algorithm. The proposed algorithm achieved the same best results that have been achieved by the advised heuristic rules [38] for twelve projects from the fifty test projects under consideration. These projects are P4, P5, P11, P12, P13, P15, P18, P20, P29, P38, P46, and P48. The proposed algorithm achieved better results than the heuristics for fourteen projects. These projects are P21, P23, P24, P25, P33, P35, P36, P37, P39, P41, P42, P45, P47, and P49. This means that the proposed genetic algorithm achieved 52% results better than

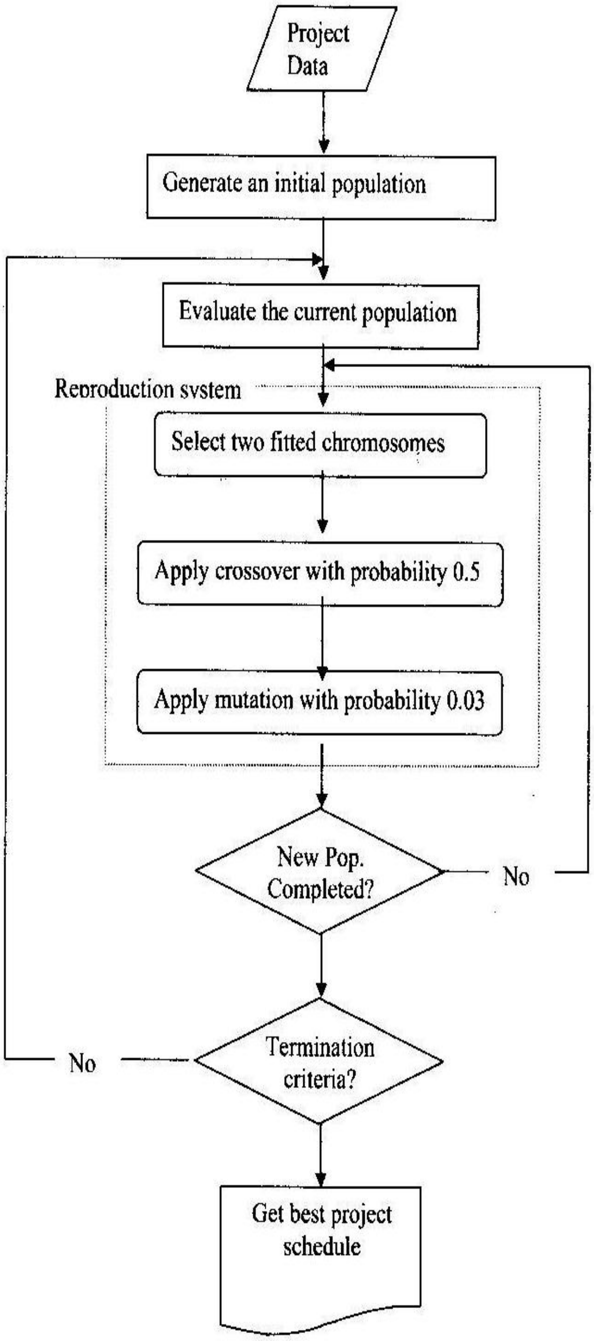

Fig. 4. Proposed GA flow chart.

the advised heuristics [38]. However, the average deviation per project from heuristics is 0.223% in case of this category of multiple resources (R1 and R2). In case of single constrained resource (R1) for scheduling process, the proposed algorithm achieved the same best results achieved by the advised heuristic rules [38] for nineteen projects. These projects are P1, P2, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P16, P25, P26, P27, P40, P45, and P49. The proposed algorithm achieved better results than the heuristics for twenty-three projects. These projects are P14, P15, P18, P19, P20, P21, P22, P23, P24, P30, P32, P33, P34, P37, P38, P39, P41, P42, P44, P46, P47, P48, and P50. This means that the proposed genetic algorithm achieved 84% results better than the heuristics. However, the average deviation per project from heuristics is 0.03% in case of this category. Incase of single constrained resource (R2) for scheduling process, the proposed algorithm achieved the same best results achieved by the advised heuristic rules [38] for twelve projects. These projects are P4, P5, P11, P15, P16, P22, P25, P27, P29, P34, P45, and P46. The proposed algorithm achieved better results than the heuristics for twenty-three projects. These projects are P1, P2, P7, P8, P9, P12, P18, P20, P21, P23, P24, P31, P33, P36, P37, P39, P41, P42, P43, P47, P48, P48, and P50. This means that the proposed genetic algorithm achieved 70% results better than the heuristics. However, the average deviation per project from heuristics is 0.04% in case of this category of single resource (R2).

16. Conclusion

RCPSP is normally characterized by makespane length as objective functions and features of resource limitations and permutation condition. In this article, a proposed genetic algorithm is presented for scheduling single and multiple resource-constrained projects. The proposed algorithm is tested using fifty test problems and compared with the recommended makespans derived by fifty-five heuristic rules. The proposed genetic algorithm achieved 70% results better than that rendered by the test benchmark problems while the average deviation per project from heuristics is 0.097% per project for the remainder 30%. Hence the proposed genetic algorithm is recommended as a powerful tool for scheduling single and multiple resource constrained projects.

Table 1
Best and average project make span after 10 runs

Project No	No of activities	Max. available resources		Makespan for R1 and R2		Makespan for R1		Makespan for R2	
		Res1	Res2	Best	Average	Best	Average	Best	Average
1	9	4	3	18	18.0	12	127	18	18.0
2	13	5	3	40	40.0	35	35.0	40	40.0
3	12	9	3	78	78.0	51	51.0	78	78.0
4	12	7	3	36	36.0	36	36.0	34	34.0
5	11	4	3	25	25.0	22	22.0	25	25.0
6	11	9	3	35	35.0	19	19.0	35	35.0
7	15	9	3	77	77.0	49	49.0	77	77.0
8	11	8	5	20	20.0	13	13.0	17	17.0
9	12	6	5	70	70.0	68	68.0	70	70.0
10	14	4	5	48	48.0	36	6.0	48	48.0
11	13	7	6	40	40.0	37	37.0	38	38.0
12	15	9	6	267	267.0	211	211.0	224	224.0
13	15	8	5	26	26.0	18	18.0	26	26.0
14	18	8	6	135	135.0	84	85.1	135	135.0
15	24	6	7	86	86.0	53	53.0	81	81.0
16	21	8	6	100	100.0	66	66.0	100	100.0
17	22	4	7	65	65.0	50	53,3	62	62.8
18	24	8	8	50	50.0	39	39.3	45	45.0
19	28	8	6	66	67.5	41	42.3	65	65.4
20	31	5	5	86	86.1	51	51.7	86	86.0
21	40	14	8	192	196.4	143	146.2	177	178.8
22	30	7	6	98	98.7	77	77.4	91	91.0
23	38	5	5	117	118.7	76	78.7	113	114.7
24	43	5	6	193	194.7	153	153.9	176	178.6
25	54	9	6	143	144.1	78	79.2	142	144.2
26	18	4	8	62	62.0	56	56.0	49	49.0
27	10	4	8	35	35.0	28	28.2	32	32.0
28	13	5	8	60	60.0	47	47.0	52	52.0
29	12	6	8	30	30.0	25	25.0	30	30.0
30	12	4	6	39	39.0	27	27.2	39	39.0
31	18	4	7	71	71.0	61	61.0	58	58.8
32	18	10	6	82	82.0	48	48.3	82	83.0
33	37	12	8	129	133.4	113	116.1	120	121.5
34	28	10	6	180	180.6	143	146.6	153	154.2
35	39	15	8	247	249.5	209	213.3	224	226.4
36	28	12	10	68	68.4	66	66.9	57	58.8
37	23	10	10	46	46.2	31	31.6	46	46.0
38	23	10	10	46	46.5	31	31.7	46	46.0
39	24	8	8	64	64.7	36	36.9	64	64.4
40	24	8	8	64	65.6	35	36.4	64	64.4
41	33	10	9	227	231.5	164	169.3	217	222.4
42	42	8	8	133	137.0	123	126.7	1.1	102.1
43	30	8	8	115	118.0	107	109.5	79	81.9
44	13	5	6	42	42.0	28	28.0	42	42.0
45	11	5	9	34	35.6	33	33.0	32	32.2
46	12	4	6	60	60.0	40	40.0	60	60.0
47	18	6	9	66	66.3	46	46.7	62	62.8
48	22	5	10	65	65.3	47	47.8	65	65.4
49	16	6	8	53	53.0	40	40.0	53	53.0
50	27	5	10	98	100.4	71	72.5	96	97.6

Table 2
Best project make span using heuristics

Project no	No of activities	Max. Available Resources		Makespan for R1 and R2	Makespan for R1	Makespan for R2
		Res1	Res2	Best	Best	Best
1	9	4	3	12	12	19
2	13	5	3	35	35	40
3	12	9	3	69	42	69
4	12	7	3	36	36	34
5	11	4	3	25	22	25
6	11	9	3	19	19	27
7	15	9	3	53	53	78
8	11	8	5	13	13	18
9	12	6	5	68	68	71
10	14	4	5	31	31	47
11	13	7	6	40	37	38
12	15	9	6	267	211	234
13	15	8	5	26	18	25
14	18	8	6	126	89	126
15	24	6	7	86	63	81
16	21	8	6	66	66	100
17	22	4	7	64	45	61
18	24	8	8	50	40	47
19	28	8	6	59	42	50
20	31	5	5	86	53	87
21	40	14	8	203	147	186
22	30	7	6	94	80	91
23	38	5	5	119	80	119
24	43	5	6	213	165	222
25	54	9	6	146	78	142
26	18	4	8	61	56	47
27	10	4	8	34	28	32
28	13	5	8	38	43	46
29	12	6	8	30	20	30
30	12	4	6	35	32	31
31	18	4	7	68	32	62
32	18	10	6	75	49	74
33	37	12	8	133	117	126
34	28	10	6	161	150	153
35	39	15	8	251	208	208
36	28	12	10	76	65	60
37	23	10	10	146	100	130
38	23	10	10	46	33	40
39	24	8	8	211	114	206
40	24	8	8	60	35	60
41	33	10	9	238	169	228
42	42	8	8	144	129	107
43	30	8	8	103	102	93
44	13	5	6	37	31	35
45	11	5	9	36	33	32
46	12	4	6	60	41	60
47	18	6	9	69	48	64
48	22	5	10	65	49	66
49	16	6	8	55	40	55
50	27	5	10	89	74	98

References

[1] J.F. Gonocalves, J.J. Mendes, and M.G.C. Resende, A Genetic Algorithm for the Resource Constrained Multi-Project Scheduling Problem, AT and T labs, Technical Report TD-668 LM4, and (2004).

[2] K.W. Kim, M. Gen, and G. Yamazaki, Hybrid Genetic Algorithm with Fuzzy Logic for Resource-Constrained Project Scheduling, Applied Soft Computing, Elsevier, 2/3 F, pp. 174-188 (2003).

[3] S. Hartmann, "Project Scheduling with Multiple Models: A Genetic Algorithm", Annals of Operation Research, Vol. 102, pp. 111-135 (2001).

[4] V. Valls, S. Quintima, and F. Ballestin, "Resource-Constrained Project Scheduling: A critical Activity Reordering Heuristic", EJOR, Elsevier, Vol. 149, pp. 282-301 (2003).

[5] K. Fleszar, K.S. Hindi, "Solving the Resource Constrained Projects Scheduling Problem by a Variable Neighborhood Search", EJOR, Elsevier, Vol. 155, pp. 402-413 (2004).

[6] H. Zhang, X. Li, H. Li, and F. Uang, 'Particle Swarm Optimization-Based Schemes for Resource-Constrained Projects Scheduling, Automation in construction, Elsevier, Vol. 14, pp. 393-404 (2005).

[7] R. Kolisch, and S. Hartmann, "Experimental Investigation of Heuristics for Resource-Constrained Projects Scheduling: An update", EJOR, Elsevier (2005).

[8] L.E. Drezet, and C. Tacquard, Multi-Constrained Projects Scheduling, Laboratoire D'infortique, Ecole Polytechnique de I'Universite de Tours, 64 av Jean Partalis, 37200 Tours (2003).

[9] S. Hartmann, "A self-Adapting Genetic Algorithm for Project Scheduling under Resource Constrained", Naval Research Logistics, Vol. 49, pp. 433-448 (2002).

[10] F.F. Boctor, "Some Efficient Multi-heuristic Procedures for Resource-Constrained Projects Scheduling", EJOR, 49, pp. 3-13 (1990).

[11] J.K. Lee, and Y.D. Kinn, "Search Heuristics for Resource Constrained Project Scheduling", EJOR, Vol. 47 (51), pp. 678-689 (1996).

[12] K. Bouleimen, and H. Lecocq, "A new Efficient Simulated Annealing Algorithm for the Resource-Constrained Projects Scheduling Problem", Proceedings of the Sixth International Workshop on Project Management and Scheduling, Bogazici University pp. 19-22 (1998).

[13] E. Pinson, C. Prins, F. and Rullier, "Using Tabu Search for Solving the Resource-Constrained Projects Scheduling Problem", Proceedings of Fourth International Workshop on Project Management and Scheduling, Katholieke Universiteit Leuven, pp. 102-106 (1994).

[14] T. Baar, P. Brucker, and S. Knust, Tabu-Search Algorithms and Lower Bounds for the Resource-Constrained Projects Scheduling Problem, Meta Heuristics, Advances and Trends in Local Search Paradigms for Optimization, Kluwer, Academic, pp. 1-18 (1998).

[15] V.J. Leon, and R. Balakrishnan, "Strength and Adaptability of Problem-Space Based Neighbouhood for Resource-Constrained Scheduling", OR Spektrum Vol. 17, pp. 173-182 (1995).

[16] S. Hantmann, "A competitive Genetic Algorithm for Resource-Constrained Project Scheduling", Naval Research Logistic, Vol. 45, pp. 733-70 (1998).

[17] G. Syswerrda Scheduling Optimization Using Genetic Algorithms, in: Davis L. (ED.), Handbok of Genetic Algorithms, Van Nostrand, New York, pp. 332-249 (1991).

[18] K.W. Kim M. Gen and G. Yamazaki Hybrid Genetic Algorithm with Fuzzy Logic for Recource-Constrained Project Scheduling, Applied Soft Computing Vol. 2/3F, pp. 174-188 (2003).

[19] R. Cheng M. Gen Genetic Algorithm and Engineering Optimization, Wiley, New York (2000).

[20] D.E. Goldberg Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley (1989).

[21] A. Brindle Genetic Algorithms for Function Optimization, Technical Report TR*!-02, Dept. of Computer Science, University of Alberta, Edmonton, Alberta, Canada, T6G 2H1 (1981).
[22] A. Pritsker, L. Watters, P. and Wolfe, P. "Multi-project Scheduling with Limited Resources: A Zero-one Programming Approach", Management Science, Vol. 16 pp. 93-107 (1969).
[23] E. Demeulemeester, and W. Herroelen, Project Scheduling - A Research Handbook., Kluwer Academic Publishers, Boston (2002).
[24] R. Alvarez-Valdes, and J.M. Tamarit, "Heuristic Algorithms for Resource-constrained Project Scheduling: A Review and an Empirical Analysis", In R.Slowinski and J. Weglarz, Editors, Advances in Project Scheduling, Pages Elsevier, Amsterdam, Netherlands pp. 113-134 (1989).
[25] D.F. Cooper, "Heuristics for Scheduling Resource-constrained Projects: An Experimental Investigation" Management Science Vol. 22, pp. 1186-1194 (1976).
[26] S.R. Lawrene, Resource Constrained Project Scheduling- A Computational Comparison of Heuristic Scheduling Techniques, Technical Report, Carnegie Mellon University, Pittsburgh, Pennsylvania (1985).
[27] A. Schrimer, "Case-based Reasoning and Improved Adaptive Search for Project Scheduling ", Naval Research Logistics, Vol. 47, pp. 201-222 (2000).
[28] A. Schrimer, and S. Riesenberg, "Parameterized Heuristics for Project Scheduling-Biased Random Sampling Methods," MANUSKRIPTE Aus Den Instituten fur Betriebswirtschaftslehre, 456, Universitat Kiel, Germany (1997).
[29] A. Schrimer, and S. Riesenberg, "Class-based Control Schemes for Parameterized Project Scheduling Heuristics", Manuskripte aus den Instituten fur Betriebswirtschaftslehre 471, Universitat kiel, Germany (1998).
[30] A. Alcaraz, C. Maroto, and R. Ruiz, "Improving the Performance of Genetic Algorithm for the RCPS Problem", Proceedings of the Ninth International Workshop on Project Management and Scheduling, Nnacy p. 3 (2004)
[31] K.S. Hindi, H. Yang, and K. Fleszar, "An Evolutionary Algorithm for Resource-constrained Project Scheduling", IEEE Transactions on Evolutionary Computation, Vol. 6, pp. 512-518 (2002).
[32] K. Bouleimen, and L. Lecocq, "A New Efficient Simulated Annealing Algorithm for the Resource–constrained Project Scheduling Problem and its Multiple Modes Version", European Journal of Operational Research, Vol. 149, pp. 268-281 (2003).
[33] E. Pinson, C. Prins, and F. Rullier, "Using Tabu Search for Solving the Resource-Constrained Project Scheduling Problem", In Proceedings of the Fourth International Workshop on Project Management and Scheduling, Leuven, Belgium Vol. 102-106 (1994).
[34] B. Pollack-Johnson, "Hybrid Structures and Improving Forecasting and Scheduling in Project Management", Journal of Operations Management, Vol. 12, pp. 101-117 (1995).
[35] S.E. Sampson, and E.N. Weiss, "Local Search Techniques for the Generalized Resource-Constrained Project Scheduling Problem", Naval Research Logistics, Vol. 40, pp. 665-675 (1993).
[36] L.R. Shaffer, B.J. Ritter, and W.L. Meyer, the Critical-path Method, McGraw-Hill, New York (1965).
[37] M.A. Shouman, A.Z. Ghafagy, M.A. Zaghloul, and A. Bou-Shaala, "New Heuristics for Scheduling Single Constrained Resource Projects", AEJ, Journal, Vol. 38 (3) (1999).
[38] M.A. Shouman, M.S. Ibrahim, M. Khater, and A.A. Forgani, "Heuristic Rules for Scheduling Single and Multiple Resources Constrained Projects", Accepted for publication in AEJ (2005).

Received July 30, 2005
Accepted December 31, 2005