Hybrid Evolutionary Algorithm with Adaptive Crossover, Mutation and Simulated Annealing Processes to Project Scheduling

Virginia Yannibelli[1,2(✉)] and Analía Amandi[1,2]

[1] ISISTAN Research Institute, UNCPBA University, Campus Universitario, Paraje Arroyo Seco, 7000 Tandil, Argentina
{vyannibe, amandi}@exa.unicen.edu.ar
[2] CONICET, National Council of Scientific and Technological Research, Buenos Aires, Argentina

Abstract. In this paper, we address a project scheduling problem that considers a priority optimization objective for project managers. This objective involves assigning the most effective set of human resources to each project activity. To solve the problem, we propose a hybrid evolutionary algorithm. This algorithm uses adaptive crossover, mutation and simulated annealing processes in order to improve the performance of the evolutionary search. These processes adapt their behavior based on the diversity of the evolutionary algorithm population. We compare the performance of the hybrid evolutionary algorithm with those of the algorithms previously proposed in the literature for solving the addressed problem. The obtained results indicate that the hybrid evolutionary algorithm significantly outperforms the previous algorithms.

Keywords: Project scheduling · Human resource assignment · Multi-skilled resources · Hybrid evolutionary algorithms · Evolutionary algorithms · Simulated annealing algorithms

1 Introduction

Project scheduling involves defining feasible start times and feasible human resource assignments for project activities in such a way that a predefined optimization objective is reached. To define human resource assignments, it is essential to have knowledge of the effectiveness of the available human resources in respect of the project activities. This is because the development and also the results of an activity mainly depend on the effectiveness of the human resources assigned to it [1, 2].

In the literature, many different kinds of project scheduling problems have been formally described and addressed. Nevertheless, to the best of our knowledge, only few project scheduling problems have considered human resources with different levels of effectiveness [3–6, 10], a fundamental aspect in real project scheduling problems. These project scheduling problems state different assumptions about the effectiveness of the human resources.

The project scheduling problem presented in [6] considers that the effectiveness of a human resource depends on various factors inherent to its work context (i.e., the

© Springer International Publishing Switzerland 2015

DOI: 10.1007/978-3-319-24834-9_40

activity to which the resource is assigned, the skill to which the resource is assigned within the activity, the set of human resources that has been assigned to the activity, and the attributes of the resource). This is a really significant aspect of the project scheduling problem presented in [6]. This is because, in real project scheduling problems, the human resources usually have different effectiveness levels in respect of different work contexts [1, 2] and, thus, the effectiveness of a human resource needs to be considered in respect of its work context. To the best of our knowledge, the influence of the work context on the effectiveness of the human resources has not been considered in other project scheduling problems. Based on the above-mentioned, we consider that the project scheduling problem presented in [6] states valuable and novel assumptions about the effectiveness of the human resources in the context of project scheduling problems. Besides, this problem considers a priority optimization objective for managers at the early stage of project scheduling. This objective implies assigning the most effective set of human resources to each project activity.

The project scheduling problem presented in [6] is considered as a special case of the RCPSP (Resource Constrained Project Scheduling Problem) [9] and, thus, is an NP-Hard optimization problem. Because of this, exhaustive search and optimization algorithms only can solve very small instances of the problem in a reasonable period of time. Therefore, heuristic search and optimization algorithms have been proposed in the literature to solve the problem: an evolutionary algorithm was proposed in [6], a memetic algorithm was proposed in [7] that incorporates a hill-climbing algorithm into the framework of an evolutionary algorithm, and a hybrid evolutionary algorithm was proposed in [8] that integrates an adaptive simulated annealing algorithm into the framework of an evolutionary algorithm. These three algorithms use non-adaptive crossover and mutation processes to develop the evolutionary search.

In this paper, we address the project scheduling problem presented in [6] with the aim of proposing a better heuristic search and optimization algorithm to solve it. In this regards, we propose a hybrid evolutionary algorithm that uses adaptive crossover, mutation and simulated annealing processes. The behavior of these processes is adaptive according to the diversity of the evolutionary algorithm population. The utilization of adaptive crossover, mutation and simulated annealing processes is meant to improve the performance of the evolutionary search [18–20].

We propose the above-mentioned hybrid evolutionary algorithm because of the following reason. Evolutionary algorithms with adaptive crossover and mutation processes have been proven to be more effective than evolutionary algorithms with non-adaptive crossover and mutation processes in the resolution of a wide variety of NP-Hard optimization problems [18–20]. Therefore, we consider that the proposed hybrid evolutionary algorithm could outperform the heuristic algorithms previously proposed to solve the problem.

The remainder of the paper is organized as follows. In Sect. 2, we give a brief review of published works that describe project scheduling problems in which the effectiveness of human resources is considered. In Sect. 3, we describe the problem addressed in this paper. In Sect. 4, we present the proposed hybrid evolutionary algorithm. In Sect. 5, we present the computational experiments carried out to evaluate the performance of the hybrid evolutionary algorithm and an analysis of the results obtained. Finally, in Sect. 6 we present the conclusions of the present work.

2 Related Works

In the literature, different project scheduling problems that consider the effectiveness of human resources have been described. Nevertheless, these project scheduling problems state different assumptions about the effectiveness of human resources. In this regards, only few project scheduling problems consider human resources with different levels of effectiveness [3–6, 10], a fundamental aspect in real project scheduling problems. In this section, we focus the attention on analyzing the way in which the effectiveness of human resources is considered in project scheduling problems described in the literature.

In [12–17], multi-skill project scheduling problems are described. In these problems, each project activity requires specific skills and a given number of human resources (employees) for each required skill. Each available employee masters one or several skills, and all the employees that master a given skill have the same effectiveness level in relation to the skill (homogeneous levels of effectiveness in relation to each skill).

In [3], a multi-skill project scheduling problem with hierarchical levels of skills is described. In this problem, given a skill, for each employee that masters the skill, an effectiveness level is defined in relation to the skill. Therefore, the employees that master a given skill have different levels of effectiveness in relation to the skill (heterogeneous levels of effectiveness in relation to each skill). Then, each project activity requires one or several skills, a minimum effectiveness level for each skill, and a number of employees for each pair skill-level. This work considers that all sets of employees that can be assigned to a given activity have the same effectiveness on the development of the activity. Specifically, with respect to effectiveness, such sets are merely treated as unary resources with homogeneous levels of effectiveness.

In [4, 5], multi-skill project scheduling problems are described. In these problems, most activities require only one employee with a particular skill, and each available employee masters different skills. Besides, the employees that master a given skill have different levels of effectiveness in relation to the skill. Then, the effectiveness of an employee in a given activity is defined by considering only the effectiveness level of the employee in relation to the skill required for the activity.

Unlike the above-mentioned problems, the project scheduling problem presented in [6] considers that the effectiveness of a human resource depends on various factors inherent to its work context. Thus, for each human resource, it is possible to define different effectiveness levels in relation to different work contexts. This is a really significant aspect of the project scheduling problem presented in [6]. This is because, in real project scheduling problems, the human resources have different effectiveness levels in respect of different work contexts [1, 2] and, thus, the effectiveness of a human resource needs to be considered in respect of its work context. Taking into account the above-mentioned, we consider that the project scheduling problem presented in [6] states valuable assumptions regarding the effectiveness of human resources in the context of project scheduling problems.

3 Problem Description

In this paper, we address the project scheduling problem described in [6]. We present below a description of this problem.

A project contains a set A of N activities, $A = \{1, \ldots, N\}$, to be scheduled (i.e., the starting time and the human resources of each activity have to be defined). The duration, precedence relations and resource requirements of each activity are known.

The duration of each activity j is notated as d_j. Moreover, it is considered that pre-emption of activities is not allowed (i.e., the d_j periods of time must be consecutive).

Among some project activities, there are precedence relations. The precedence relations establish that each activity j cannot start until all its immediate predecessors, given by the set P_j, have completely finished.

Project activities require human resources – employees – skilled in different knowledge areas. Specifically, each activity requires one or several skills as well as a given number of employees for each skill.

It is considered that organizations and companies have a qualified workforce to develop their projects. This workforce is made up of a number of employees, and each employee masters one or several skills.

Considering a given project, set SK represents the K skills required to develop the project, $SK = \{1, \ldots, K\}$, and set AR_k represents the available employees with skill k. Then, the term $r_{j,k}$ represents the number of employees with skill k required for activity j of the project. The values of the terms $r_{j,k}$ are known for each project activity.

It is considered that an employee cannot take over more than one skill within a given activity. In addition, an employee cannot be assigned more than one activity at the same time.

Based on the previous assumptions, an employee can be assigned different activities but not at the same time, can take over different skills required for an activity but not simultaneously, and can belong to different possible sets of employees for each activity.

As a result, it is possible to define different work contexts for each available employee. It is considered that the work context of an employee r, denoted as $C_{r,j,k,g}$, is made up of four main components. The first component refers to the activity j which r is assigned (i.e., the complexity of j, its domain, etc.). The second component refers to the skill k which r is assigned within activity j (i.e., the tasks associated to k within j). The third component is the set of employees g that has been assigned j and that includes r (i.e., r must work in collaboration with the other employees assigned to j). The fourth component refers to the attributes of r (i.e., his or her experience level in relation to different tasks and domains, the kind of labor relation between r and the other employees of g, his or her educational level in relation to different knowledge areas, his or her level with respect to different skills, etc.). It is considered that the attributes of r could be quantified from available information about r (e.g., curriculum vitae of r, results of evaluations made to r, information about the participation of r in already executed projects, etc.).

The four components described above are considered the main factors that determine the effectiveness level of an employee. For this reason, it is assumed that the effectiveness of an employee depends on all the components of his or her work context. Then, for each employee, it is possible to consider different effectiveness levels in relation to different work contexts.

The effectiveness level of an employee r, in relation to a possible context $C_{r,j,k,g}$ for r, is notated as $e_{rCr,j,k,g}$. The term $e_{rCr,j,k,g}$ represents how well r can handle, within activity j, the tasks associated to skill k, considering that r must work in collaboration with the other employees of set g. The mentioned term $e_{rCr,j,k,g}$ takes a real value over the range [0, 1]. The values of the terms $e_{rCr,j,k,g}$ inherent to each employee available for the project are known. It is considered that these values could be obtained from available information about the participation of the employees in already executed projects.

The problem of scheduling a project entails defining feasible start times (i.e., the precedence relations between the activities must not be violated) and feasible human resource assignments (i.e., the human resource requirements must be met) for project activities in such a way that the optimization objective is reached. In this sense, a priority objective is considered for project managers at the early stage of the project schedule design. The objective is that the most effective set of employees be assigned each project activity. This objective is modeled by Formulas (1) and (2).

Formula (1) maximizes the effectiveness of the sets of employees assigned to the N activities of a given project. In this formula, set S contains all the feasible schedules for the project in question. The term $e(s)$ represents the effectiveness level of the sets of employees assigned to project activities by schedule s. Then, $R(j,s)$ is the set of employees assigned to activity j by schedule s, and the term $e_{R(j,s)}$ represents the effectiveness level corresponding to $R(j,s)$.

Formula (2) estimates the effectiveness level of the set of employees $R(j,s)$. This effectiveness level is estimated calculating the mean effectiveness level of the employees belonging to $R(j,s)$.

For a more detailed discussion of Formulas (1) and (2), we refer to [6].

$$\max_{\forall s \in S} \left(e(s) = \sum_{j=1}^{N} e_{R(j,s)} \right) \tag{1}$$

$$e_{R(j,s)} = \frac{\sum_{r=1}^{|R(j,s)|} e_{rC_{r,j,k(r,j,s),R(j,s)}}}{|R(j,s)|} \tag{2}$$

4 Hybrid Evolutionary Algorithm

To solve the problem, we propose a hybrid evolutionary algorithm. This algorithm uses adaptive crossover, mutation and simulated annealing processes. The behavior of these processes is adaptive according to the diversity of the evolutionary algorithm population. The utilization of adaptive crossover, mutation and simulated annealing processes is meant to improve the performance of the evolutionary search in both exploration and exploitation [18–20].

The general behavior of the hybrid evolutionary algorithm is described as follows. Considering a given project to be scheduled, the algorithm starts the evolution from a random initial population of solutions in which each solution codifies a feasible project schedule. Then, each solution of the population is decoded (i.e., the related schedule is built), and evaluated according to the optimization objective of the problem by a fitness function. As explained in Sect. 3, the objective is to maximize the effectiveness of the sets of employees assigned to project activities. In respect of this objective, the fitness function evaluates the assignments of each solution based on knowledge about the effectiveness of the employees involved in the solution.

Once the solutions of the population are evaluated, a parent selection process is used to decide which solutions of the population will compose the mating pool. The solutions with the highest fitness values will have more probability of being selected. After the mating pool is composed, the solutions in the mating pool are paired. Then, a crossover process is applied to each pair of solutions with an adaptive probability P_c to generate new feasible ones. Then, a mutation process is applied to each solution generated by the crossover process, with an adaptive probability P_m. Then, a survival selection process is applied in order to define which solutions from the solutions in the population and the solutions generated from the mating pool will compose the new population. Finally, an adaptive simulated annealing algorithm is applied to the solutions of the new population.

This process is repeated until a predetermined number of iterations is reached.

4.1 Encoding of Solutions and Fitness Function

To encode the solutions, we used the representation proposed in [6]. Each solution is represented by two lists having as many positions as activities in the project. The first list is a standard activity list. This list is a feasible precedence list of the activities involved in the project (i.e., each activity j can appear on the list in any position higher than the positions of all its predecessors). The activity list describes the order in which activities shall be added to the schedule.

The second list is an assigned resources list. This list contains information about the employees assigned to each activity of the project. Specifically, position j on this list details the employees of every skill k assigned to activity j.

To build the schedule related to the representation, we used the serial schedule generation method proposed in [6]. In this method, each activity j is scheduled at the earliest possible time.

In order to evaluate a given encoded solution, we used a fitness function. This function decodes the schedule s related to the solution by using the serial method above-mentioned. Then, the function calculates the value of the term $e(s)$ corresponding to s (Formulas (1) and (2)). This value determines the fitness level of the solution. The term $e(s)$ takes a real value over [0, …, N].

To calculate the term *e(s)*, the function uses the values of the terms $e_{rCr,j,k,g}$ inherent to *s* (Formula 2). As mentioned in Sect. 3, the values of the terms $e_{rCr,j,k,g}$ inherent to each available employee *r* are known.

4.2 Parent Selection, Adaptive Crossover, Adaptive Mutation, and Survival Selection

In order to develop the parent selection, we applied the traditional roulette wheel selection process [18].

To develop the crossover and the mutation, we applied feasible processes for the representation of the solutions. The crossover process contains a feasible crossover operation for activity lists and a feasible crossover operation for assigned resources lists. For activity lists, we used the two-point crossover proposed by Hartmann [21]. For assigned resources lists, we used the traditional uniform crossover [18].

The mutation process contains a feasible mutation operation for activity lists and a feasible mutation operation for assigned resources lists. For activity lists, we used the adjacent pairwise interchange operator described in [21]. For assigned resources lists, we used the traditional random resetting [18].

The crossover and mutation processes are applied with adaptive probabilities P_c and P_m, respectively. In this regards, we used the well-known adaptive probabilities proposed by Srinivas [11, 18]. These probabilities are calculated as detailed in Formulas (3) and (4), where f_{max} is the maximal fitness into the population, f_{avg} is the average fitness of the population, and $(f_{max} - f_{avg})$ is used as a measure of the diversity of the population. In Formula (3), f' is the higher fitness of the two solutions to be crossed, and P_{c1} and P_{c2} are predetermined values for the crossover probability, considering $0 \le P_{c1}, P_{c2} \le 1$. In Formula (4), f'' is the fitness of the solution to be mutated, and P_{m1} and P_{m2} are predetermined values for the mutation probability, considering $0 \le P_{m1}, P_{m2} \le 1$.

By Formulas (3) and (4), when the diversity of the population decreases, P_c and P_m are increased to promote the exploration of unvisited regions of the search space and thus to prevent the premature convergence of the evolutionary search. When the population is diverse, P_c and P_m are decreased to promote the exploitation of visited regions of the search space. Thus, probabilities P_c and P_m are adaptive to promote either the exploration or exploitation according to the diversity of the population.

$$P_c = \begin{cases} \frac{P_{c2}(f_{max} - f')}{(f_{max} - f_{avg})} & f' \ge f_{avg} \\ P_{c1} & f' < f_{avg} \end{cases} \tag{3}$$

$$P_m = \begin{cases} \frac{P_{m2}(f_{max} - f'')}{(f_{max} - f_{avg})} & f'' \ge f_{avg} \\ P_{m1} & f'' < f_{avg} \end{cases} \tag{4}$$

In order to develop the survival selection, we applied the traditional fitness-based steady-state selection scheme [18]. In this scheme, the worst λ solutions of the current population are replaced by the best λ solutions generated from the mating pool. This scheme preserves the best solutions found by the hybrid evolutionary algorithm [18].

4.3 Adaptive Simulated Annealing Algorithm

Once obtained a new population by the survival selection process, we applied an adaptive simulated annealing algorithm to each solution of this population, except to the best solution of this population which is maintained. This adaptive simulated annealing algorithm is a variant of the one proposed in [8], and is described below.

The adaptive simulated annealing algorithm is an iterative process which starts from a given encoded solution s for the problem, considering a given initial value T_0 for a parameter called temperature. In each iteration, a new solution s' is generated from the current solution s by a move operator. When the new solution s' is better than the current solution s (i.e., the fitness value of s' is higher than the fitness value of s), the current solution s is replaced by s'. Otherwise, when the new solution s' is worse than the current solution s, the current solution s is replaced by s' with a probability equal to $exp(-delta/T)$, where T is the current temperature value and *delta* is the difference between the fitness value of s and the fitness value of s'. Thus, the probability of accepting a new solution s' that is worse than the current solution s mainly depends on the temperature value. If the temperature is high, the acceptance probability is also high, and vice versa. The temperature value is decreased by a cooling factor at the end of each iteration. The described process is repeated until a predefined number of iterations is reached.

The initial temperature value T_0 is defined before applying the simulated annealing algorithm to the solutions of the population. In this case, T_0 is inversely proportional to the diversity of the population, and is calculated as follows: $T_0 = 1/\left(f_{max} - f_{avg}\right)$, where $(f_{max} - f_{avg})$ is used as a measure of the diversity of the population. Therefore, when the population is diverse, the value of T_0 is low, and thus the simulated annealing algorithm behaves like an exploitation process, fine-tuning the solutions of the population. When the diversity of the population decreases, the value of T_0 rises, and thus, the simulated annealing algorithm changes its behavior from exploitation to exploration in order to introduce diversity in the population and therefore to prevent the premature convergence of the evolutionary search. Thus, the behavior of the simulated annealing algorithm is adaptive to either an exploitation or exploration behavior according to the diversity of the population.

In respect of the move operator applied by the simulated annealing algorithm to generate a new solution from the current solution, we used a feasible move operator for the representation of the solutions. The move operator contains a feasible move operation for activity lists and a feasible move operation for assigned resources lists. For activity lists, we used a move operator called simple shift [21]. For assigned resources lists, we used a move operator which is a variant of the traditional random resetting [18]. This variant modifies only one randomly selected position of the list.

5 Computational Experiments

In order to develop the experiments, we utilized the six instance sets presented in [7]. The main characteristics of these instance sets are shown in Table 1. Each instance of these instance sets contains information about a number of activities to be scheduled, and information about a number of available employees to develop the activities. For a detailed description of these six instance sets, we refer to [7].

Each instance of these instance sets has a known optimal solution with a fitness level *e(s)* equal to *N* (*N* refers to the number of activities in the instance). The known optimal solutions of the instances are considered here as references.

Table 1. Characteristics of instance sets

Instance set	Activities per instance	Possible sets of employees per activity	Instances
j30_5	30	1 to 5	40
j30_10	30	1 to 10	40
j60_5	60	1 to 5	40
j60_10	60	1 to 10	40
j120_5	120	1 to 5	40
j120_10	120	1 to 10	40

The hybrid evolutionary algorithm was run 30 times on each instance of the six instance sets. To carry out these runs, the algorithm parameters were set with the values detailed in Table 2. The algorithm parameters were set based on preliminary experiments that showed that these values led to the best and most stable results.

Table 3 presents the results obtained by the experiments. Column 2 presents the average percentage deviation from the optimal solution (Dev. (%)) for each instance set. Column 3 presents the percentage of instances for which the value of the optimal solution is achieved at least once among the 30 generated solutions (Opt. (%)).

The results obtained by the algorithm for j30_5, j30_10, j60_5 and j60_10 indicate that the algorithm has found an optimal solution in each of the 30 runs carried out on each instance of these sets.

The Dev. (%) obtained by the algorithm for j120_5 and j120_10 is greater than 0 %. Considering that the instances of j120_5 and j120_10 have known optimal solutions with a fitness level *e(s)* equal to 120, we analyzed the meaning of the average deviation obtained for each one of these sets. In the case of j120_5 and j120_10, average deviations equal to 0.1 % and 0.36 % indicate that the average value of the solutions obtained by the algorithm is 119.88 and 119.57 respectively. Thus, we may state that the algorithm has obtained very high quality solutions for the instances of j120_5 and j120_10. Besides, the Opt. (%) obtained by the algorithm for j120_5 and j120_10 is 100 %. These results indicate that the algorithm has found an optimal solution in at least one of the 30 runs carried out on each instance of the sets.

Table 2. Parameter values of the hybrid evolutionary algorithm

Parameter	Value
Population size	90
Number of generations	300
Crossover process	
P_{c1}	0.9
P_{c2}	0.6
Mutation process	
P_{m1}	0.1
P_{m2}	0.05
Survival selection process	
λ	45
Simulated annealing algorithm	
Number of iterations	25
Cooling factor	0.9

Table 3. Results obtained by the computational experiments

Instance set	Dev. (%)	Opt. (%)
j30_5	0	100
j30_10	0	100
j60_5	0	100
j60_10	0	100
j120_5	0.1	100
j120_10	0.36	100

5.1 Comparison with a Competing Algorithm

To the best of our knowledge, three algorithms have been previously proposed to solve the addressed problem: a classical evolutionary algorithm [6], a classical memetic algorithm [7] that incorporates a hill-climbing algorithm into the framework of an evolutionary algorithm, and a hybrid evolutionary algorithm [8] that integrates an adaptive simulated annealing algorithm into the framework of an evolutionary algorithm. These three algorithms use non-adaptive crossover and mutation processes to develop the evolutionary search.

According to the experiments reported in [7, 8], the three algorithms have been evaluated on the six instance sets presented in Table 1 and have obtained the results that are shown in Table 4. Based on these results, the algorithm proposed in [8] is the best of the three algorithms. Below, we compare the performance of this algorithm with that of the hybrid evolutionary algorithm proposed here. For simplicity, we will refer to the algorithm proposed in [8] as algorithm H.

The results in Tables 3 and 4 indicate that the hybrid evolutionary algorithm proposed here and the algorithm H have reached the same effectiveness level (i.e., an optimal effectiveness level) on the first four instance sets (i.e., the less complex sets).

However, the effectiveness level reached by the hybrid evolutionary algorithm on the last two instance sets (i.e., the more complex sets) is higher than the effectiveness level reached by the algorithm H on these sets. Thus, the performance of the hybrid evolutionary algorithm on the two more complex instance sets is better than that of the algorithm H. The main reason for this is that, in contrast with the algorithm H, the hybrid evolutionary algorithm uses adaptive crossover and mutation processes, and these processes adapt their behavior to promote either exploration or exploitation of the search space according to the diversity of the population. Thus, the hybrid evolutionary algorithm can reach better solutions than algorithm H on the more complex instance sets.

Table 4. Results obtained by the algorithms previously proposed for the addressed problem.

Instance set	Evolutionary algorithm [6]		Memetic algorithm [7]		Hybrid algorithm [8]	
	Dev. (%)	Opt. (%)	Dev. (%)	Opt. (%)	Dev. (%)	Opt. (%)
j30_5	0	100	0	100	0	100
j30_10	0	100	0	100	0	100
j60_5	0.42	100	0	100	0	100
j60_10	0.59	100	0.1	100	0	100
j120_5	1.1	100	0.75	100	0.64	100
j120_10	1.29	100	0.91	100	0.8	100

6 Conclusions

In this paper, we have proposed a hybrid evolutionary algorithm to solve the project scheduling problem presented in [6]. This algorithm uses adaptive crossover, mutation and simulated annealing processes in order to improve the performance of the evolutionary search. These processes adapt their behavior to promote either exploration or exploitation of the search space according to the diversity of the evolutionary algorithm population. The computational experiments developed indicate that the performance of the hybrid evolutionary algorithm on the used instance sets is better than those of the algorithms previously proposed for solving the problem.

In future works, we will evaluate other adaptive crossover and mutation processes, and other selection processes. Moreover, we will evaluate the incorporation of other search and optimization techniques into the framework of the evolutionary algorithm.

References

1. Heerkens, G.R.: Project Management. McGraw-Hill, New York (2002)
2. Wysocki, R.K.: Effective Project Management, 3rd edn. Wiley Publishing, Hoboken (2003)

3. Bellenguez, O., Néron, E.: Lower bounds for the multi-skill project scheduling problem with hierarchical levels of skills. In: Burke, E.K., Trick, M.A. (eds.) PATAT 2004. LNCS, vol. 3616, pp. 229–243. Springer, Heidelberg (2005)
4. Hanne, T., Nickel, S.: A multiobjective evolutionary algorithm for scheduling and inspection planning in software development projects. Eur. J. Oper. Res. **167**, 663–678 (2005)
5. Gutjahr, W.J., Katzensteiner, S., Reiter, P., Stummer, Ch., Denk, M.: Competence-driven project portfolio selection, scheduling and staff assignment. CEJOR **16**(3), 281–306 (2008)
6. Yannibelli, V., Amandi, A.: A knowledge-based evolutionary assistant to software development project scheduling. Expert Syst. Appl. **38**(7), 8403–8413 (2011)
7. Yannibelli, V., Amandi, A.: A memetic approach to project scheduling that maximizes the effectiveness of the human resources assigned to project activities. In: Corchado, E., Snášel, V., Abraham, A., Woźniak, M., Graña, M., Cho, S.-B. (eds.) HAIS 2012, Part I. LNCS, vol. 7208, pp. 159–173. Springer, Heidelberg (2012)
8. Yannibelli, V., Amandi, A.: A diversity-adaptive hybrid evolutionary algorithm to solve a project scheduling problem. In: Corchado, E., Lozano, J.A., Quintián, H., Yin, H. (eds.) IDEAL 2014. LNCS, vol. 8669, pp. 412–423. Springer, Heidelberg (2014)
9. Blazewicz, J., Lenstra, J., Rinnooy Kan, A.: Scheduling subject to resource constraints: classification and complexity. Discrete Appl. Math. **5**, 11–24 (1983)
10. Yannibelli, V., Amandi, A.: Project scheduling: a multi-objective evolutionary algorithm that optimizes the effectiveness of human resources and the project makespan. Eng. Optim. **45**(1), 45–65 (2013)
11. Srinivas, M., Patnaik, L.M.: Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Trans. Syst. Man Cybern. **24**(4), 656–667 (1994)
12. Bellenguez, O., Néron, E.: A branch-and-bound method for solving multi-skill project scheduling problem. RAIRO – Oper. Res. **41**(2), 155–170 (2007)
13. Drezet, L.E., Billaut, J.C.: A project scheduling problem with labour constraints and time-dependent activities requirements. Int. J. Prod. Econ. **112**, 217–225 (2008)
14. Li, H., Womer, K.: Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm. J. Sched. **12**, 281–298 (2009)
15. Valls, V., Pérez, A., Quintanilla, S.: Skilled workforce scheduling in service centers. Eur. J. Oper. Res. **193**(3), 791–804 (2009)
16. Aickelin, U., Burke, E., Li, J.: An evolutionary squeaky wheel optimization approach to personnel scheduling. IEEE Trans. Evol. Comput. **13**(2), 433–443 (2009)
17. Heimerl, C., Kolisch, R.: Scheduling and staffing multiple projects with a multi-skilled workforce. OR Spectr. **32**(4), 343–368 (2010)
18. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing, 2nd edn. Springer, Berlin (2007)
19. Rodriguez, F.J., García-Martínez, C., Lozano, M.: Hybrid metaheuristics based on evolutionary algorithms and simulated annealing: taxonomy, comparison, and synergy test. IEEE Trans. Evol. Comput. **16**(6), 787–800 (2012)
20. Talbi, E. (ed.): Hybrid Metaheuristics. SCI 434. Springer, Berlin, Heidelberg (2013)
21. Kolisch, R., Hartmann, S.: Experimental investigation of heuristics for resource-constrained project scheduling: an update. Eur. J. Oper. Res. **174**, 23–37 (2006)