Ponz Tienda, JL.; Yepes Piqueras, V.; Pellicer Armiñana, E.; Moreno Flores, J. (2013). The Resource Leveling Problem with multiple resources using an adaptive genetic algorithm. Automation in Construction. 29(1):161-172. doi:10.1016/j.autcon.2012.10.003.

THE RESOURCE LEVELING PROBLEM WITH MULTIPLE RESOURCES USING AN ADAPTIVE GENETIC ALGORITHM

Ponz-Tienda, J.L.

Yepes, V.

Pellicer, E.

Moreno-Flores, J.

THE RESOURCE LEVELING PROBLEM WITH MULTIPLE RESOURCES USING AN ADAPTIVE GENETIC ALGORITHM

Abstract

Resources management ensures that a project is completed on time and at cost, and that its quality is as previously defined; nevertheless, resources are scarce and their use in the activities of the project leads to conflicts in the schedule. Resource leveling problems consider how to make the resource consumption as efficient as possible. This paper presents a new Adaptive Genetic Algorithm for the Resource Leveling Problem with multiple resources, and its novelty lies in using the Weibull distribution to establish an estimation of the global optimum as a termination condition. The extension of the project deadline with a penalty is allowed, avoiding the increase in the project criticality punishing the shift of activities. The algorithm is tested with the standard Project Scheduling Problem Library PSPLIB, and a complete analysis and benchmarking test instances are presented. The proposed algorithm is implemented using VBA for Excel 2010 in order to provide a flexible and powerful decision support system that enables practitioners to choose between different feasible solutions to a problem, and in addition it is easily adjustable to the constraints and particular needs of each project in realistic environments.

Keywords: Project Scheduling, Resource leveling, Genetic algorithms, Benchmarking

1. Introduction

Project management is the process of the coordination and integration of activities in an efficient and effective manner using limited resources. It consists of linking resources to their respective deliverables and assembling them into the whole project [1]. Resource management is an intrinsic element of project management [2,3,4]; resource management ensures that the project is completed on time and at cost and that the quality is as previously defined [5,6,7]. This is even more necessary for project-based companies such as contractors [3,8,9]. In fact, project scheduling problems are one of most important problems that practitioners deal with in scheduling, especially when they need to achieve the most efficient resource consumption without increasing the prescribed makespan of the project.

However, because resources are scarce, the use of resources in the activities of the project leads to conflicts in the schedule [10]. Project scheduling problems comprise not only resource-constrained problems but also resource leveling problems, among others [11]. These two kinds of problem consider resource consumption in two different ways: in the former it is seen as a constraint, and in the latter the problem is to make it as efficient as possible. Even though these two approaches may seem similar, they are conceptually different. Both have been widely studied by researchers and applied by practitioners, although these two groups are unaware of the differences between the approaches and the serious limitations imposed by the heuristics used in the commercial software.

These two problems are defined as non-deterministic polynomial-time hard (NP-hard) problems [12]. The first approach is a regular problem known as the Resource Constrained Project Scheduling Problem; its objective is to reduce the makespan without exceeding the constraints of resource availability [13,12]. The second, known as the Resource Leveling Problem (from now on, RLP) is a non-regular problem; its objective is to achieve the most efficient resource consumption without increasing the prescribed makespan of the project [14,12]. The two problems can be combined together as a multi-objective optimization problem, but there is always one main objective (usually the makespan); the other objective (usually the efficient resource consumption) is secondary.

Nevertheless, conventional analytical and heuristic methods are neither flexible nor productive when solving the RLP [15]. Some reasons for this inefficiency are, on the one hand, that exact procedures simplify the real problems so are not useful at offering optimal solutions

with acceptable computational effort [16] and, on the other hand, that heuristics offer solutions which are far from optimal, so that it is necessary to apply metaheuristic algorithms to complex and realistic projects [17]. Recently, important approaches have been made by researchers to improve the efficiency of resource consumption, proposing different heuristics which are applicable to small projects; simple examples try to show the merits of a particular algorithm, without establishing clear criteria for a performance comparison between the different algorithms [18].

Following this line of work, Liao et al. [11] proposed some ideas to advance the RLP in realistic environments; these authors made several proposals for the development or the improvement of the RLP. Regarding resources allocation, these authors proposed the use of a decision support system to assist project managers, as well as the development of benchmarking tests for performance assessment and comparison [11]. Concerning resources leveling, they suggested the use of multiple resources allowing the extension of the project deadline with a penalty [11]. We take these proposals as challenges to be overcome in this paper, contributing a little to the corpus of knowledge in this field.

Therefore, in this paper we present an Adaptive Genetic Algorithm (AGA) for the RLP with multiple resources allowing the extension of the project deadline with a penalty; for this purpose, we use the Weibull distribution as a termination condition, establishing an estimation of the global optimum. The proposed algorithm is tested with the standard "project scheduling problem library" (PSPLIB) [18], presenting a complete set of benchmarking tests. A decision support system is also used in order to implement this algorithm. Without loss of generality, we consider the classical resource leveling objective function: the total squared utilization cost for a given schedule.

The remainder of this paper is organized as follows. Section 2 provides the classification and formulation of the RLP. Section 3 details the different solving procedures: exact, heuristic, and metaheuristic algorithms with the new use of the Weibull distribution as a termination condition. Section 4 describes the algorithm proposed for the RLP with multiple resources. Computational results and the benchmarking test are explained in Section 5. Finally, conclusions are drawn.

2. Classification and Formulation of the Resource Leveling Problem

The general formulation of the RLP requires us to consider the following elements:

1. The set of activities, N :
   $$N = \{j_1, j_2, \cdots, j_n\} \tag{1}$$
   n being the total number of activities.
2. The set of durations, D :
   $$D = \{d_1, d_2, \cdots, d_n\} \tag{2}$$
   where $d_i, 1 \le i \le n$ is the assigned duration for each activity.
3. The set of periods of time in which these activities have to be distributed:
   $$T = \{t_1, t_2, \cdots, t_p\} \tag{3}$$
   t_p being the deadline of the project, from now on denoted $\overline{T}$.
4. The set of resources, R:
   $$R = \{r_1, r_2, \cdots, r_k\} \tag{4}$$
   k being the total number of resources.
5. The set of availabilities of the resources, A :
   $$A = \{a_{it}, 1 \le i \le k, 1 \le t \le p\} \tag{5}$$
   where a_{it} is the availability of the resource r_i in the period t .
6. The set of costs, C :
   $$C = \{c_1, c_2, \cdots, c_k\} \tag{6}$$

7. The set SS : to distribute the performance of the activities along the elements of the set T one needs to allocate a starting time for each activity, given by the ordered set, SS :

$$SS = \{SS_1, SS_2, \cdots, SS_n\} \tag{7}$$

$SS_i, 1 \le i \le n$, is the starting time of the activity j_i . $\overline{T}$ can be considered as the starting time of a finish dummy activity SS_{finish} , and then SS becomes:

$$SS = \{SS_1, SS_2, \cdots, SS_n, SS_{finish}\} \tag{8}$$

Obviously, the schedule SS is not unique; on the contrary, there are a large number of different possibilities, according to the logic and restrictions of the project to be performed. Each of these schedules has significant differences in the efficiency of resources consumption, and this is the reason for finding the values of SS which optimize this efficiency.

8. The functions $r_i(S,t)$, $1 \le i \le k$: given a schedule SS , the function $r_i(S,t)$ is defined as the consumption of the resource r_i in the period of time t , belonging to the set T , in such a way that the consumption of the resource r_i throughout the project is given by:

$$u_{i1} = r_i(S,t_1), u_{i2} = r_i(S,t_2), \cdots, u_{ip} = r_i(S,t_p) \tag{9}$$

9. The function f : Given a schedule SS , the efficiency of resources consumption depends on the layout of its use. Therefore, it becomes fundamental to establish an optimal criterion for the distribution of the resources. This is the role we want f to play in the development of the problem. Hence, the function f will be different for each optimization criterion to be considered.

Once we have the elements that compose the problem, a general formulation could be:

$$Minimize \sum_{i=1}^{k} c_i f\left[r_i(S,t)\right] \tag{10}$$

subject to:

$$SS_{finish} \le \overline{T} \tag{11}$$

$$SS_i + d_i + \gamma_{ij} \le SS_j \text{ , for all } i \text{ which are successors to } j \tag{12}$$

$$u_{ij} \le a_{ij} \tag{13}$$

where γ_{ij} is the lead/lag between i and j

Having done this, the choice of the function f , which defines the criterion for the optimization of the resources consumption, provides different ways of solving the problem. In the case of the RLP, the optimization criterion focuses on getting the resource consumption as level as possible. Consequently, a suitable choice of f could be:

$$f\left[r_i(S,t)\right] = \sum_{t=1}^{\overline{T}} \frac{(u_{it} - a_{it})^2}{\overline{T}} \tag{14}$$

And Eq. (10) turns into:

$$Minimize \sum_{i=1}^{k} c_i \cdot f\left[r_i(S,t)\right] = \sum_{i=1}^{k} \sum_{t=1}^{\overline{T}} c_i \frac{(u_{it} - a_{it})^2}{\overline{T}} \tag{15}$$

Next, we can simplify the problem by taking into account the following:

1. $c_i = 1, \text{for all } 1 \le i \le k$
2. $a_{it} = a, \text{for all } 1 \le i \le k \text{ and for all } 1 \le t \le p$

And then Eq. (15) becomes:

$$Minimize \sum_{i=1}^{k} \sum_{t=1}^{\overline{T}} \frac{(u_{it} - a)^2}{\overline{T}} \tag{16}$$

As the minimum depends neither on how far the variable is shifted nor on the constant values, we can take $a = 0$ and $\overline{T} = 1$:

$$Minimize \sum_{r=1}^{k} \sum_{t=1}^{\overline{T}} u_{rt}^{\ 2} \tag{17}$$

The objective function expressed in Eq. (17) is known as a *minimum squares optimization*, and was introduced by Burgess and Killebrew [19] in a heuristic algorithm in which the near-optimality is determined by the schedule with the minimum total sum of the squares of resource consumption for each period, as illustrated in Fig. 1. Other measures for the objective function are the Minimum Moment (MOM) proposed by Harris [20], and more recently the entropy-maximization proposed by Christodoulou et al. [21], using the maximality and sub-additivity properties of the entropy function.

Fig. 1 Initial $\sum_{t=1}^{\overline{T}} u_{kt}^{\ 2} = 10.669$; levelled $\sum_{t=1}^{\overline{T}} u_{kt}^{\ 2} = 6.477$

To compare the leveling effectiveness between different projects, we use the *Resource Improvement Coefficient (RIC)* developed by Robert Harris [20], a measure that is independent of the total resource demand and is given by Eq. (18). The RIC relates the variation of a selected resource histogram to an ideal resource histogram which is a rectangle-shaped resource histogram (the ideal leveled schedule corresponds to a RIC value of one):

$$RIC = \frac{\overline{T} \cdot \sum_{t=1}^{\overline{T}} u_t^{\ 2}}{\left(\sum_{t=1}^{\overline{T}} u_t \right)^2} \tag{18}$$

A different kind of non-regular objective function for Eq. (14) can be considered if the objective function to be optimized represents the *Net Present Value Problem*. In this case, the objective function represents the net present value of the project, which is to be maximized, and this is used in practice when expensive resources have to be purchased.

3. Solving Procedures

The previous conceptual linear programming cannot be solved directly, because there is no easy way to translate the set S, used in Eq. (10), into a linear programming formulation. Other linear programming formulations have to be used in order to be able to specify the resource constraints in a correct and solvable form.

The most efficient formulations are based on integer and binary programming, and can be of two kinds, depending on whether the decision variables establish the period of execution or the finishing time of the activities.

The first formulation for the *Resource Leveling Problem* in the multi-mode case is based on a set of binary decision variables x_{jst} [22] that establish the period in which the activities are finished:

$$x_{jst} = \begin{cases} 1, & \textit{if job } j \text{ is finished in mode } s \text{ at the end of period } t \\ 0, & \text{otherwise} \end{cases}, \tag{19}$$

$$\text{for } j \in J, s \in M_j, \text{and } t \in \left[ES_j + d_j, \cdots, LF_j \right]$$

The variables x_{jst} can only be executed in one mode, and are defined over the interval between the earliest and latest finishing times (the delimiting periods) of the activities of the project. These limits are established using the traditional forward and backward pass calculations for the unconstrained problem.

The vector *SS* of *Starting Scheduled* period for each activity is defined by:

$$SS_j = \sum_{s=1}^{P_j} \left(\sum_{t=EF_j}^{LF_j} t \cdot x_{jst} - d_{js} \right) \tag{20}$$

The objective function to minimize the minimum total sum of the squares of resource consumption for each period is:

$$Minimize \sum_{k=1}^{R}\sum_{t=1}^{\bar{T}} c_k \cdot \left(\sum_{j\in E(t)} \sum_{s=1}^{P_j} u_{rst} \cdot \sum_{q=EF_j}^{LF_j} x_{jst} \right)^2 \tag{21}$$

Subject to:

$$\sum_{s=1}^{P_j} \sum_{t=EF_j}^{LF_j} x_{jst} = 1 \quad \text{for } j \in J \tag{22}$$

$$\sum_{s=1}^{P_j} \sum_{t=EF_j}^{LF_j} \left(t - d_{js}\right) \cdot x_{jst} - \gamma_{ij} - \sum_{s=1}^{P_j} \sum_{t=EF_j}^{LF_j} t \cdot x_{ist} \geq 0 \quad \text{for } j \in J \text{ and } i \in P_j \tag{23}$$

$$x_{jst} \in \{0,1\} \quad \text{for } j \in J, s \in \left[1,\cdots,P_j\right] \text{ and } t \in \left[ES_j + d_j,\cdots,LF_j\right] \tag{24}$$

The objective function modeled in Eq. (21) minimizes the total sum of the squares of resource consumption for each period. Eq. (22) specifies that only one mode and one completion time are allowed for every activity. The precedence constraints are given in Eq. (23). Finally Eq. (24) specifies that the decision variables are binary.

Another possible formulation for the RLP is also based on binary programming, but the decision variables x_{jst} establish the period in which the activities are executed [23]:

$$x_{jst} = \begin{cases} 1, & \textit{if job } j \text{ is processed in mode } s \text{ in period } t \\ 0, & \text{otherwise} \end{cases}, \tag{25}$$

$$\text{for } j \in J, s \in M_j, \text{and } t \in \left[ES_j + 1,\cdots,LF_j\right]$$

The vector *SS* of *Starting Scheduled* period for each activity is defined by:

$$SS_j = \sum_{s=1}^{P_j} \left(\sum_{ES_j}^{LF_j} t \cdot x_{jst} \cdot \left(x_{jst-1} - x_{jst}\right) \right) \tag{26}$$

The objective function will be:

$$Minimize \sum_{k=1}^{R}\sum_{t=1}^{\bar{T}} c_k \cdot \left(\sum_{j\in E(t)} \sum_{s=1}^{P_j} u_{rst} \cdot x_{jst} \right)^2 \tag{27}$$

subject to:

$$\sum_{s=1}^{P_j} \sum_{t=ES_j+1}^{LF_j} x_{jst} = d_j \quad \text{for } j \in J \tag{28}$$

$$d_{js} \cdot \left(x_{jst} - x_{j,t+1}\right) - \sum_{q=ES_j+1}^{t} x_{jsq} \leq 0 \quad \text{for } j \in J, s \in M_j, \text{and } t \in \left[ES_j + 1,\cdots,LF_j - 1\right] \tag{29}$$

$$d_i \cdot x_{jst} - \gamma_{ij} - \sum_{q=ES_j+1}^{t-1} x_{ist} \leq 0 \quad \text{for } j \in J, i \in P_j, s \in M_j, \text{and } t \in \left[ES_j + 1,\cdots,LF_i\right] \tag{30}$$

$$x_{jst} \in \{0,1\} \quad \text{for } j \in J, s \in \left[1,\cdots,P_j\right] \text{ and } t \in \left[ES_j + 1,\cdots,LF_j\right] \tag{31}$$

The objective function in Eq. (27) minimizes the total sum of the squares of resource consumption for each period. Eq. (28) ensures that each activity is executed for d_j time units. The execution of the activities without pre-emption is modeled in Eq. (29). The precedence constraints are given in Eq. (30). Finally, Eq. (31) specifies that the decision variables are binary. Easa [24] proposed a different binary integer formulation, preserving the precedence constraints by limiting the shifting of the activities to the free float, which must be equal to or greater than zero.

The *Resource Leveling Problem* is NP-Hard even if only one resource is considered [25,12]. The time complexity function is of the order of $O\left(q^n\right)$ with q being a positive constant. The universe of schedules for an instance is:

$$\prod_{j=1}^{J}\prod_{s=1}^{M_j}(Ht_{js}+\overline{T}-\overline{mkp}+1) \tag{32}$$

with Ht_{js} being the total float of the activity *j* processed in mode *s* , $\overline{T}$ the prescribed makespan, $\overline{mkp}$ the *makespan* for the resource unconstrained problem (RUPSP or resource relaxation of the RCPSP) and M_j the execution modes of activity *j* .

If the prescribed *makespan* is established to be equal to the makespan for the RUPSP and only one execution mode is considered, Eq. (32) can be simplified to:

$$\prod_{j=1}^{J}(Ht_{js}+1) \tag{33}$$

3.1. Exact Algorithms

Exact algorithms based upon implicit enumeration, integer programming, and dynamic programming techniques, have been proposed to solve the RLP. Easa [24] used a mixed binary-integer programming technique that guarantees the optimum leveling. Exhaustive enumeration procedures were presented by Ahuja [26]. Bandelloni et al. [27] developed an optimal technique based on dynamic programming. Recently, Rieck, Zimmermann & Gather [28] proposed a new mixed-integer linear model formulation and domain-reducing pre-processing techniques for the RLP based on smart discrete-time formulations.

The branch-and-bound technique [29] is probably the most widely used exact solution technique for solving project scheduling problems, as it is the only technique which allows for the generation of optimal solutions with acceptable computational effort. Neumann and Zimmermann [30] describe a branch-and-bound procedure that reduces the set of all feasible solutions by successively scheduling activities for approximately solving the problem. Recently, a lower bound improved method (Maximum Allowable Daily Resources Method) to branch the nodes has been developed by Mutlu [31], which may form a basis for the performance evaluation of heuristic and metaheuristic procedures for the RLP. Recently, Gather et al. [32] presented a new enumeration scheme embedded into a branch-and-bound framework using a constructive lower bound as well as pre-processing techniques and Hariga and El-Sayegh [33] a new mixed integer binary linear optimization model that allow activity splitting and minimizes its associated costs.

The RLP as an NP-Hard problem has a phenomenon of "*combinatorial explosion*" [15], especially for large-scale projects. For this reason, exact algorithms are only efficient for small projects.

3.2. Heuristics Algorithms

To avoid the explosion problem, heuristic rules are mostly used to solve the RLP. These are simple rules or sets of rules which aim to obtain a "good" solution (locally optimal) for a difficult problem but do not guarantee the best solution (globally optimal). Heuristic algorithms can be of two kinds, construction and improvement procedures. Construction procedures are used to establish a feasible solution to the problem, and the other procedures are used to improve it.

The first heuristic procedure for the RLP was proposed by Burgess and Killebrew [19], establishing the *minimum squares* as the performance measure. The Burgess-Killebrew algorithm is an improvement procedure that readjusts the starting time of each activity and reduces the variability to a near optimum (local optimum). The first step of the algorithm consists in establishing the *starting time scheduled* of each activity at their earliest starting time. Later, the activities are selected one by one according to the priority rule (earliest finishing time), selecting the best starting time, which is the one that minimizes the total sum of the squares of resource consumption for each period.

The Burgess-Killebrew algorithm is a single-pass algorithm that requires one to be aware of the precedence relationships between activities because this is not considered in the formulation. This problem was considered and solved by Burman [33] using the free float as the limit for the activity shifting.

Harris proposed a multi-pass algorithm [20] called the MOM, which established the *minimum moment* as the performance measure for minimizing the daily fluctuations in resource use while keeping the total project duration unchanged. Later, Harris improved the MOM with the *packing method* (PACK) [34], which recognizes network interactions with a more in-depth analysis. Martinez and Ioannou [35] improved this method by introducing the *Modified Minimum Moment Method* to level resources in construction projects. Hiyassat [36] devised a heuristic to reduce the calculations needed for the minimum moment approach. Jeetendra et al. [37] proposed the use of *Petri Nets* (*PNs*) and a *P-matrix* to help in token movements, describing an algorithm for the RLP.

3.3. Metaheuristic algorithms

Given the variety of network structures and resources, no single heuristic rule can always produce the optimal solution for all the RLP, and exact algorithms require extensive computational effort. Therefore they are unable to deal with real large problems.

Metaheuristic methods are general purpose high level search frameworks grounded in physical, biological and animal behavior that can be applied to any optimization problem. Several metaheuristic methods have been proposed, such as Grasp (Greedy Randomized Adaptive Search Procedure). Other improved methods are evolutionary algorithms (EA), genetic algorithms (GA), scatter search (SS), simulated annealing (SA), ant colony optimization (ACO), particle swarm optimization (PSO), and shuffled frog-leaping (SFL).

Evolutionary and genetic algorithms (EAs/GAs) [39,40] and simulated annealing (SA) [41] are the most popular metaheuristic methods for solving the RLP. Hybrid methods are presented by Son and Skibniewski [41], and by Alsayegh & Hariga [42], with a new approach for the RLP which considers the cost of splitting the non-critical activities and combines the particle swarm optimization (PSO) and simulated annealing (SA) methods. A hyperheuristic approach implemented within commercial project management software has been proposed by Koulinas and Anagnostopoulus [43]. Hyperheuristics chooses a sequence of "knowledge poor" heuristics that are used to choose the most appropriate low-level heuristic from a set of heuristics during the search.

Metaheuristic algorithms search for a better solution until a termination condition (or one of a set of termination conditions) has been satisfied. Possible reasonable termination conditions include: a certain amount of time, a maximum CPU time, a fixed maximum number of iterations, a vector SS of *Starting time Scheduled* with a measured value less than a predefined threshold value (*Objective Bound*), and a fixed maximum number of iterations without improvements in the best solution found so far.

An *Objective Bound* as a termination condition should be established by computing a *Lower Bound* (LB), by relaxing the restrictions of the RLP. Some LB methods are *LP-relaxation*, cutting planes relaxing the integrality restrictions, *Lagrangean Relaxation*, removing a set of constraints from the original problem and incorporating them into the objective function, or *Set Covering Based Approach*, discarding the precedence constraints and/or allowing the pre-emption of activities [44].

Recently, Paya-Zaforteza et al. [45] and Carbonell, Yepes and González-Vidosa [46] used the three-parameter Weibull distribution [47] to establish an estimation of the global optimum. They based their research on the fact, proved by Fisher and Tippet [48] that, as the number of independent samples of size m grows, their distribution approaches a three-parameter Weibull distribution with location parameter γ as an estimation of the global optimum.

4. An Adaptive Genetic Algorithm (AGA) for the RLP with multiple resources

Genetic Algorithms (GAs) [49] are stochastic search techniques based upon the principles of Darwinian evolution, and simulate biological evolution. Basically, a collection of candidate

solutions (*initial population*) is manipulated (*evolved*) through a number of iterations (*generations*). A candidate solution, called a *chromosome* or *individual*, is represented by a set of integer values (*genes*).

In the RLP, the most usual representation for encoding a candidate solution is the vector SS of *Starting time Scheduled* for each activity [15]:

$$individual_i = \left\{SS_{i1}, SS_{i2}, \ldots, SS_{ij}, \ldots, SS_{iJ}\right\} \tag{34}$$

The *initial population* is generated randomly by *heuristic constructive algorithms*, and the quality of individuals is evaluated and ranked using Eq. (10) as a *fitness function.* In the RLP the fitness function can be any of the performance measures: the *minimum moment* [20], the *minimum squares* (Eq. 17), the *RIC* (Eq. 18) or the entropy function [21].

The initial population (P) evolves, under a set of cyclic genetic operators. First, two *parents* are randomly selected, based on their fitness values (a mechanism is used to ensure that a solution with a better fitness has a higher chance of being selected as a mate). One or more point *crossovers* divides the *parents* into different sub-parts, generating, by recombination, two *children* or *offspring* (P') as new candidate solutions. Some genes of the offspring can mutate producing different offspring (P''), adding them to the population and reducing the population by selection. The previous algorithm can be structured as follows:

begin :
$P \leftarrow Generate\ InitialPopulation()$
$Evaluate(P)$
Do
$P' \leftarrow Recombine(P)$
$P'' \leftarrow Mutate(P')$
$Evaluate(P'')$
$P \leftarrow Select(P'' \cup P')$
Loop until termination condition is met
end

4.1. The formulation for the AGA

The formulation used to solve the RLP with the proposed AGA is based on an integer programming formulation with decision variables δ_j that represents the shifting of the initial early start (ES_j) of each activity using the traditional forward and backward pass calculations for the resource unconstrained problem.

The vector *SS* of *Starting time Scheduled* (not period) for each activity is defined by:

$$SS_j = \delta_j + ES_j \tag{35}$$

The objective function to minimize the minimum total sum of the squares of resource consumption for each period is:

$$Minimize \sum_{k=1}^{R}\sum_{t=1}^{\overline{T}} c_k \cdot \left(\sum_{j \in E(t)} u_{kt}\right)^2 + \varepsilon \cdot \sum_{j=1}^{J} \delta_j + \rho \cdot \Delta\overline{T} \tag{36}$$

Subject to:

$$\delta_j + ES_j - \left(\delta_i + ES_i + d_i + \gamma_{ij}\right) \geq 0 \quad \text{for } j \in J \text{ and } i \in P_j \tag{37}$$

$$ES_{finish} \leq \overline{T} + \Delta\overline{T} \tag{38}$$

$$\delta_j \geq 0 \text{ and integral} \quad \text{for } j \in J \tag{39}$$

Consequently, the individuals are encoded as follows:

$$individual_i = \left\{\delta_{i1}, \delta_{i2}, \ldots, \delta_{ij}, \ldots, \delta_{iJ}\right\} \tag{40}$$

The objective function Eq. (36) minimizes the total sum of the squares of resource consumption for each period. The precedence constraints are given in Eq. (37). The prescribed makespan, as the maximum project duration, is preserved by Eq. (38) and, finally, Eq. (39) specifies that the decision variables are non-negative integer values.

The objective function in Eq. (36) has been improved by "punishing" the minimum total sum of the squares, ensuring a better solution with the lower shifting and consequently increasing the float of the activities. Besides this, Eq. (36) has been completed by allowing an extension ($\Delta\overline{T}$) of the project deadline ($\overline{T}$) or the prescribed makespan, with penalty (ρ). To avoid undermining the main objective ε must be very small; we recommend a value such that the sum of the total float for all the activities multiplied by ε is less than one, as expressed in Eq. (41):

$$\varepsilon \cdot \sum_{j=1}^{J} Totalfloat_j < 1 \tag{41}$$

One of the major difficulties of *GA* is the premature convergence toward suboptimal solutions, and the loss of genetic variability that is caused by inbreeding (crossing individuals with common parentage). To avoid this, a *GA* must find an equilibrium point between intensification and diversification strategies. Intensification produces convergence to a local optimum with high levels of inbreeding, and diversification produces unfeasible individuals. Static and dynamic strategies have been proposed by researchers; the static methodologies modify the operation parameters according to a linear or logarithmic function [50]; the dynamic strategies adjust the mutation probability [51], or crossover and mutation [52], dynamically, with the aim of overcoming the premature problems of convergence and closing to a local optimum.

4.2. The initial population

The initial population is randomly generated by a heuristic backward constructive procedure to preserve the maximum diversification of the population with feasible individuals. The procedure in pseudocode is as follows:

begin :
Topological ordering
$Popsize = \max(50, 2 \times Totaljobs)$
For$(k = 1; Popsize; 1)$
  For$(j = J; 1; -1)$ in a decreasing topological order
    $\delta_{ij} = random(0, Free\ Float + \Delta\overline{T})$
  ;
;
end

Note that in the previous procedure, the shifting values δ_{ij} are randomly established between 0 and the free float of each activity, to guarantee that all the individuals are feasible solutions to the problem.

4.3. The evolution

Once the initial population is generated, individuals evolve by a principle inspired by natural evolution, each transmitting its genetic material to its offspring. Only the most adapted individuals (the *elite*) are able to survive and breed new individuals. Additionally, the genetic information of the offspring is subject to small mutations.

The population is evaluated and ranked, and Eq. (42) is applied to establish the individuals of the elite as a fraction of the total population:

$0 < Elitesize < 1$
$Elite \leftarrow (Elitesize \times Popsize)$ Best individuals

Based on the knowledge of the problem, the global optimum is in the neighborhood of the best individuals. Thus, parents are selected from the elite population through the classic roulette-wheel mechanism to probabilistically select individuals based on the fitness measure (f_i):

$$P(S_i) = \frac{f_i}{\sum_{i=1}^{Elitesize \times Popsize} f_i}; 1 \le i \le Elitesize \times Popsize \tag{42}$$

$Parents \leftarrow \text{random } P(S_i)$

4.4. Recombination and mutation

Like its counterpart in nature, recombination or crossover produces offspring (s-on and d-aughter) that have some parts of both (f-ather father and m-other) parent's genetic material. For a one-point crossover C , the pseudo-code is:

$$individual_f = \{\delta_{f1}, \delta_{f2}, \ldots, \delta_{fc}, \delta_{fc+1}, \ldots, \delta_{fJ}\}$$

$$individual_m = \{\delta_{m1}, \delta_{m2}, \ldots, \delta_{mc}, \delta_{mc+1}, \ldots, \delta_{mJ}\}$$

$$Crossover(C) = random(1, J)$$

$$offspring_s = \{\delta_{f1}, \delta_{f2}, \ldots, \delta_{fc}, \delta_{mc+1}, \ldots, \delta_{mJ}\}$$

$$offspring_d = \{\delta_{m1}, \delta_{m2}, \ldots, \delta_{mc}, \delta_{fc+1}, \ldots, \delta_{fJ}\}$$

For a multi-point crossover, the procedure is very similar. Crossover points are chosen randomly and sorted without duplicates. Then the genes between successive crossover points are exchanged between the parents to produce new offspring.

A mutation operator acts as natural mutation does in natural evolution, perturbing the genome. In the mutation process, randomly selected genes of a chromosome are probabilistically replaced by other genes from the valid domain of the problem, to produce a new genetic structure. The mutation process is controlled by a mutation probability range ($mutprob$) that must lie between 0 and 1 ($0 < mutprob < 1$) and a maximum number of mutation genes ($maxmut$).

4.5. Evaluation and Unfeasibility

Along the evolution process, offspring are potentially infeasible. Infeasible individuals are penalized in the function that measures the quality of an individual, but are not rejected; this guarantees the diversification of the population.

The feasibility range of individuals in the evolved population is controlled using the feasibility lower and upper ranges (Flr, Fur). The feasibility range guarantees the equilibrium between intensification and diversification strategies, changing the mutation probability range $(mutprob)$ by a value of f and the maximum number of mutation genes ($maxmut$) by η, dynamically maintaining the genetic evolution, the diversity in the population and the convergence capacity of the AGA. The proposed adaptive procedure in pseudocode is as follows:

begin :

$P \leftarrow Generate\ InitialPopulation()$

$Evaluate(P)$

$f = 1; \eta = 0$

Do

$P' \leftarrow Recombine(P)$

$P'' \leftarrow Mutate(P')$

$Evaluate(P'')$

if feasibility in last g generations is out of range adjust $mutprob$ and $maxmut$:

if feasibility $< Flr$; $maxmut$ - = f; $mutprob -= \eta$

if feasibility $> Fur$; $maxmut$ += f; $mutprob += \eta$

$P \leftarrow Select(P'' \cup P')$

Loop until termination condition is met

end

Due to the conditions of the problem, it is very difficult to establish a general rule for the feasibility range. The best performance values have been obtained with values for the feasibility lower range equal to 10% ($Flr = 0.1$) and for the feasibility upper range equal to 35% ($Fur = 0.35$).The effect on the evolution process with different values for the feasibility lower and upper ranges can be seen in Fig. 2:

Fig. 2 Effect of different values for the feasibility lower and upper ranges

4.6. Termination conditions; the Weibull distribution

4.6.1. Formulation of the Weibull distribution

The formulation of the Weibull distribution to compute the location parameter (γ) or, as we call it for this special case the Weibull Bound (WB), to be used as termination condition, is explained in the following lines. Let F be the distribution of a random variable X, in other words: $F(x) = \Pr ob[X \le x]$, such that, given a sample of size n, the equality:

$$\Pr ob\left[X_1 \le x_1, \cdots, X_n \le x_n\right] = \Pr ob\left[X_1 \le x_1\right] \cdots \Pr ob\left[X_n \le x_n\right] \tag{43}$$

holds. Let Ln be a new random variable, given by:

$$L_n = \min_{i=1,2,\cdots n} \{X_i\} \tag{44}$$

The problem we want to solve is to determine the distribution that L_n follows, in particular at the limit of large samples, and the asymptotic behavior of the random variable L_n. Fisher and Tippet [48] solved this problem based on the so-called *Stability Postulate*: The distribution of the largest (smallest) value in samples of size $N \times n$ will tend to the same asymptotic expression as the distribution of the largest (smallest) value in samples of size n, since the largest (smallest) of the minimums of N samples of size n is the same as the minimum of the sample of size $N \times n$. Consequently the asymptotic distribution must be the same for both cases.

Since a linear transformation does not change the form of the distribution, the Stability Postulate can be formulated as:

$$F^n(x) = F(a_n x + b_n) \tag{45}$$

Solving this functional equation for F, the Weibull distribution for the smallest value is obtained as:

$$W(x) = \begin{cases} 1 - e^{-\left(\frac{x-\gamma}{\eta}\right)^{\beta}}, & \text{if } x > \gamma \\ 0, & \text{if } x \le \gamma \end{cases} \tag{46}$$

with $\eta, \beta > 0$ and where γ is called the location parameter, η the scale parameter, and β the shaper parameter.

4.6.2. How to compute the parameters γ, η and β when $F(x)$ is unknown

First: We take a sample of size n in a random way.

Second: From the sample, we calculate the density and distribution functions:

$f(x) = \frac{N_x}{n}$; N_x being the number of times that the value x appears in the sample

$$F(x) = \sum_{X \le x} f(x) \tag{47}$$

Third: We approach the Weibull distribution using the experimental values of $F(x)$.

$$F(x) = 1 - e^{-\left(\frac{x-\gamma}{\eta}\right)^{\beta}} \rightarrow \frac{1}{1 - F(x)} = e^{-\left(\frac{x-\gamma}{\eta}\right)^{\beta}} \tag{48}$$

Taking logarithms:

$$\log\left(\frac{1}{1-F(x)}\right)=\left(\frac{x-\gamma}{\eta}\right)^{\beta} \tag{49}$$

Taking logarithms again:

$$\log\left(\log\left(\frac{1}{1-F(x)}\right)\right)=\beta\log(x-\gamma)-\beta\log(\eta) \tag{50}$$

Changing the variables:

$$y=\log\left(\log\left(\frac{1}{1-F(x)}\right)\right);z=\log(x-\gamma) \tag{51}$$

Eq. (49) becomes:

$$y=\beta z-\beta\log(x-\gamma) \tag{52}$$

which can be seen to be a linear regression problem in the variables y and z;

$$\delta(\gamma)^2=\sigma_{yz}^2(\gamma)/\left(\sigma_z^2(\gamma)\sigma_y^2\right) \tag{53}$$

The value of γ that minimizes Eq. (53), γ_0, is found using derivation techniques. In this way we get the regression for the best correlation between y and z:

$$y=az+b \tag{54}$$

Comparing Eq. (51) with Eq. (53) we get:

$$\beta=a;-\beta\log(\eta)=b \tag{55}$$

which allows us to compute the other parameters β and η.

Finally, the termination conditions for the proposed AGA are:

$$or\begin{cases}\#iteration=\text{established number of } iterations\\ BB\le Objective\ Bound,\ \text{where } BB \text{ is the better bound obtained}\\ W(BB)=P\left(x\le BB\right)\le 0.001,\ \text{where } W \text{ is the Weibull distribution}\end{cases}$$

And then the adaptive procedure in pseudocode is as follows:

begin :

$P\leftarrow Generate\ InitialPopulation()$

$Evaluate(P)$

$Compute\ Weibull\ Bound\ (WB=\lfloor\gamma\rfloor)$ with $InitialPopulation()$

$f=1;\varepsilon=0$

Do

$P'\leftarrow Recombine(P)$

$P''\leftarrow Mutate(P')$

$Evaluate(P'')$

if feasibility in last g generations is out of range adjust *mutprob* and *maxmut :*

if feasibility $<Flr$; $maxmut\ -=f;mutprob-=\varepsilon$

if feasibility $>Fur$; $maxmut\ +=f;mutprob+=\varepsilon$

$P\leftarrow Select(P''\cup P')$

Loop until termination condition is met

end

4.6.3. Numerical example

For a better understanding and to illustrate the application of the Weibull distribution for the estimation of the global optimum as a termination condition, we will show a step by step calculation, based on the first instance of the set of problems j30. The exact location parameter (γ) of the Weibull distribution is obtained when n, the size of the sample, goes to

infinity. For each sample of size n, a location parameter (γ_n) is obtained, in such a way that $\lim_{n\to\infty}\gamma_n=\gamma$. So there is a monotonous subsequence $\gamma_{n_k};(k=1,2,3,\cdots)$, either increasing or decreasing, that tends to γ when k goes to infinity. Then, by performing a heuristic search for such a monotonous subsequence of samples of size $n_1=32$, $n_2=64$, $n_3=96$, we obtain the following results step by step:

First: We choose three samples of feasible solutions of sizes $1\times n, 2\times n, 3\times n$, ($S_1, S_2, S_3$ respectively), n being the number of jobs of the instance j30_1, in a random way, for example, the S_2 sample:

S_2={7863, 8115, 7905, 7931, 7983, 8151, 7969, 7957, 8113, 8077, 7803, 7903, 8055, 8163, 8193, 8127, 8145, 7963, 8051, 7959, 7733, 8151, 8005, 7767, 7883, 7937, 7939, 7877, 7945, 8001, 7745, 7811, 7949, 7865, 8043, 8161, 8033, 7763, 8039, 8259, 7881, 8129, 7993, 8187, 7957, 8019, 7921, 7965, 8057, 7985, 8137, 7989, 7981, 7999, 8003, 7775, 8171, 8081, 7969, 8193, 7855, 7953, 8075, 7975}

Second: From S_1, S_2, S_3, we calculate the density and distribution functions:

$f(x)=\frac{N_x}{n}; F(x)=\sum_{X\le x} f(x)$; (from Eq. (46) and Eq. (47)).

Third: We approach the Weibull distribution using the experimental values of $F(x)$ in accordance with Eqs. (48) to (55), obtaining the location parameters $(\gamma_{32},\gamma_{64},\gamma_{96})$ respectively, and the other parameters $(\beta_1,\eta_1\beta_2,\eta_2\beta_3,\eta_3)$ to compute the three Weibull distributions (see Fig. 3):

$\gamma_{32}=7517.19$ with $R^2=0.972123$

$\gamma_{64}=7494.65$ with $R^2=0.982858$

$\gamma_{96}=7488.38$ with $R^2=0.968511$

Fig. 3 Approach to the Weibull distribution for W_{32}, W_{64} and W_{96}

Fourth: if the algorithm finds a value for BB (Better Bound) which satisfies the inequalities $W_n(BB)=P(x\le BB)\le 0,001; n=32,64,96$, then it stops.

$W_{32}(7549)=0.000017\le 0.001$; where BB=7549

$W_{64}(7549)=0.000036\le 0.001$; where BB=7549

$W_{96}(7549)=0.000313\le 0.001$; where BB=7549

5. Computational results and benchmarking test

As test instances, we have used the standard sets j30, j60 and j120 for the RCPSP (Resource Constrained Project Scheduling Problem). The instances, represented as activity-on-node networks, were generated with a full-size instance generator, named ProGen [53], for a general class of project scheduling problems with four renewable resources. The PSPLIB library is fully accessible in the Project Scheduling Project Library (PSPLIB) at http://129.187.106.231/psplib/library.html [18].

The values used to compute the instances are shown in Table 1 and Table 2:

Table 1 GA parameters (1)

The instances are solved with $\varepsilon=0$ and $C_k=1$, the prescribed makespan as that obtained for the RUPSP (the resource relaxation of the RCPSP), and the penalty as zero ($\rho=0$). The termination conditions are established at a maximum number of iterations and a probability for WB of 0.001%, as can be seen in Table 2:

Table 2 GA parameters (2)

The AGA is programmed with VBA (visual Basic for applications) for Excel 2010, applying a matrix algorithm for the RUPSP (the resource relaxation of the RCPSP), as can be seen in Fig. 4 [54,55]. We have used a computer with an Intel Core i7 processor, 3.6Gh of velocity and 8 GB of RAM with an average time consumed for the j30 set of 15 seconds with 1000 iterations.

Fig. 4 Decision support system application (VBA for Excel 2010)

The main statistics of the best improvement values found over the initial value of the measure function (total sum of the squares of resource consumption for each period) are shown in Table 3:

Table 3 Test statistics

The complete information about the benchmarking test can be downloaded from the "Benchmarking for Resource Leveling Problem (PSPLIB j30, j60, j120)", (https://docs.google.com/spreadsheet/ccc?key=0AgisRN826029dHdEZmNDbmxBZDI1QmZNSGQ4a1VhY3c), with the initial and leveled value for all the analyzed instances, the improvements and the vector SS of *Starting time Scheduled* for each activity.

The correlation values obtained as a function of the logarithm of the time complexity and the initial value of the measure function for the solved instances are shown in Table 4, Fig.5 and Fig.6:

Table 4 Test correlation

Fig. 5 Correlation (r_{xy}); Improvement over logarithm of the time complexity $O(q^n)$

Fig. 6 Correlation (r_{xy}); Improvement over initial sum of the squares

Finally, as there are no known results from other researchers for the PSPLIB, and to determine how good the proposed AGA algorithm is, we have computed the same instances with three different parallel scheduling schemes [56]: a parallel backward scheme with earliest finishing time as the priority rule and maximum free float as the secondary one, a parallel forward scheme with latest finishing time as the priority rule and maximum total float as the secondary one, and a forward-backward scheme with the priority rule depending on the direction of the scheduling. The results obtained compared with the proposed AGA are shown in Table 5 and Fig. 7.

Table 5 Heuristic results vs. proposed AGA

Fig. 7 Heuristic results vs. proposed AGA

The results obtained in the test instances show that there is a significant positive correlation between the improvement and the logarithm of the time complexity $O(q^n)$, but that this is not statistically significant when compared with the initial sum of the squares.

This is due to the fact that algorithmic complexity is determined by the slack in the tasks, and, in the majority of cases, if the slacks are regularly distributed over the geometry of the graph they will provide a greater number of feasible solutions to the problem, and consequently a greater improvement capacity.

The heuristic does not differ significantly between the forward and backward scheduling schemes, but a forward-backward scheme presents significant improvements compared with the single direction scheduling scheme. The improvement of the forward-backward scheme with respect to case of the single direction scheme is constant, and it seems not to be related to the time complexity or the number of tasks. However, comparing the values obtained from the application of the AGA and the analyzed heuristics, the improvement over the best heuristic significantly increases with the number of tasks of the problem.

6. Conclusions

The Resource Leveling problem (RLP) is a non-regular problem whose objective is to achieve a resource consumption which is the most efficient possible, without increasing the

prescribed makespan of the project. Conventional analytical and heuristic methods are neither flexible nor productive when solving the RLP. Exact procedures are not useful for offering optimal solutions with acceptable computational effort; and, on the other hand, heuristics offer solutions which are far from the optimal so that it is necessary to apply metaheuristic algorithms for complex and real projects in realistic environments.

In this paper:

1. We describe the complete state of the art of the RLP, proposing different binary and integer mathematical formulations and incorporating modifications and improvements on previous contributions.
2. We propose an Adaptive Genetic Algorithm (AGA) for the RLP with multiple resources allowing the extension of the project deadline with a penalty.
3. We propose the use of the three-parameter Weibull distribution as a termination condition for the metaheuristic, with location parameter γ or the Weibull Bound (WB) as an estimation of the global optimum.

The previous contributions have been tested with the standard "project scheduling problem library" (PSPLIB), presenting a complete set of benchmarking tests solved with only the most usual and common parameters to provide clear criteria for comparison between the different algorithms. To prove the merits of the proposed algorithm, the results we obtained have been compared with the most common heuristic procedures and with a more efficient forward-backward scheduling scheme. The proposed AGA is always better than the heuristics, especially for the most difficult problems with 120 jobs.

The proposed AGA for the RLP has been implemented with VBA for Excel 2010 to provide a flexible and powerful decision support system that enables practitioners to choose between different feasible solutions to a problem, and that is in addition easily adjustable to the constraints and particular needs of a project in a realistic environment. This contribution is a tool that can be applied in a direct and simple way by practitioners; besides, it can serve as a starting point for specialists in order to develop user-friendly and practical computer applications to provide realistic and good solutions for production and project management.

The use of the Weibull distribution was applied following a heuristic process. Its improvement is proposed as a future research field, searching for a bound $(\varepsilon > 0)$ in such a way that the inequality $|\gamma_n - \gamma| \leq \varepsilon, \forall n \geq N_0$ holds for all sample sizes n greater than a given size N_0; the exact parameter would therefore be in the interval $\gamma_n - \varepsilon \leq \gamma \leq \gamma_n + \varepsilon$.

Another important area for future research is the consideration of the graph density as a new variable in the computational test, to determine its influence on the optimization and the correlation between the other variables.

Acknowledgments

This study was partially funded by the Spanish Ministry of Science and Innovation (research project BIA2011-23602).

References

[1] J. Turner, "Do you manage work, deliverables or resources?," International Journal of Project Management, vol. 18, no. 2, p. 83–84, 2000.

[2] E. Goldratt, Critical Chain, North River Press, Inc., Great Barrignton, MA, 1997.

[3] I. Clark and T. Colling, "The management of human resources in project management-led organizations," Personnel Review, vol. 34, no. 2, p. 178–191, 2005.

[4] J. Turner, "Towards a theory of project management: the nature of the project," International Journal of Project Management, vol. 24, no. 1, p. 1–3, 2006b.

[5] F. Karaa and A. Nasr, "Resource management in construction," Journal of Construction Engineering and Management, vol. 112, no. 3, p. 346–357, 1986.
[6] M. Park, "Model-based dynamic resource management for construction projects," Automation in Construction, vol. 14, no. 5, p. 585–598., 2005.
[7] J. Turner, "Towards a theory of project management: the functions of project management," International Journal of Project Management, vol. 24, no. 3, p. 187–189, 2006a.
[8] J. Söderlund and K. Bredin, "HRM in project-intensive firms: changes and challenges," Human Resource Management, vol. 45, no. 2, p. 249–265, 2006.
[9] M. Huemann, A. Keegan and J. Turner, "Human resource management in the project-oriented company: a review," International Journal of Project Management, vol. 25, p. 315–323, 2007.
[10] S. Al-Jibouri, "Effects of resource management regimes on project schedule," International Journal of Project Management, vol. 20, p. 271–277, 2002.
[11] T.W. Liao, P.J. Egbelu, B.R. Sarker and S.S. Leu, "Metaheuristics for project and construction management - A state-of-the-art review," Automation in Construction, vol. 20, no. 5, p. 491–505, 2011.
[12] K. Neumann, C. Schwindt and J. Zimmermann, Project Scheduling with Time Windows and Scare Resources, Springer,Berlin, 2003.
[13] W. Herroelen, B. de Reyck and E. Demeulemeester, "Resource-constrained project scheduling: a survey of recent developments," Computers & Operations Research, vol. 25, no. 4, p. 279–302, 1998.
[14] K. Brinkmann and K. Neumann, "Heuristic procedures for resource-constrained project scheduling with minimal and maximal time lags: the resource-levelling and minimum project-duration problems," Journal of Decision Systems, vol. 5, p. 129–156, 1996.
[15] S.S Leu, C.H. Yang and J.C. Huang, "Resource leveling in construction by genetic algorithm-based optimization and its decision support system application," Automation in Construction, vol. 10, p. 27-41, 2000.
[16] S. Hartmann, "Project scheduling with multiple modes: a genetic algorithm," Annals of Operations Research, vol. 102, p. 111-135, 2001.
[17] P. Jaśkowski and A. Sobotka, "Scheduling construction projects using evolutionary algorithm," Journal of Construction Engineering and Management, vol. 132, no. 8, p. 861-870, 2006.
[18] R. Kolisch and A. Sprecher, "PSPLIB - A project scheduling problem library," European Journal of Operational Research, vol. 96, p. 205-216, 1996.
[19] A. Burguess and J. Killebrew, "Variation in activity level on a cyclic arrow diagram," The Journal of Industrial Engineering, vol. 13, p. 76-83, 1962.
[20] R. Harris, Precedence and Arrow Networking Techniques for Construction, Wiley, New York, 1978.
[21] S. E. Christodoulou, G. Ellinas and A. Michaelidou-Kamenou, "Minimum moment method for resource leveling using entropy maximization," Journal of Construction Engineering and Management, vol. 136, no. 5, p. 518-527, 2010.
[22] A.A.B. Prisker, L.J. Watters and P.M. Wolfe, "Multiproject scheduling with limited resources: a zero-one programming approach," Management Science, vol. 16, p. 95-108, 1969.
[23] L. Kaplan, "Resource-constrained project scheduling with preemption of jobs," Unpublished Ph. D. Thesis, University of Michigan, 1988.
[24] F. Ballestin, C. Schwindt and J. Zimmermann, "Resource leveling in make-to-order production: modeling and heuristic solution method," International Journal of Operations Research, vol. 4, no. 1, p. 50-62, 2007.
[25] S.M. Easa, "Resource leveling in construction by optimization," Journal of Construction Engineering and Management, vol. 115, no. 2, p. 302-316, 1989.
[26] H. Ahuja, Construction Perfomance Control by Networks, Wiley, New York, 1976.

[27] M. Bandelloni, M. Tucci and R. Rinaldi, "Optimal resource leveling using non-serial dynamic programming," European Journal of Operational Research, vol. 78, no. 2, p. 162-177, 1994.
[28] J. Rieck, J. Zimmermann and T. Gather, "Mixed-integer linear programming for resource leveling problems," European Journal of Operational Research, vol. 221, no. 1, p. 27-37, 2012.
[29] N. Agin, "Optimum seeking with branch and bound," Management Science, vol. 13, no. 4, p. B176-B185, 1966.
[30] K. Neumann and J. Zimmermann, "Procedures for resource leveling and net present value problems in project scheduling with general temporal and resource constraints," European Journal of Operations Research, vol. 127, no. 2, p. 425-443, 2000.
[31] M. Mutlu, A branch and bound algorithm for resource leveling problem, Middle East Technical University, Ankara (Turkey), 2010.
[32] T. Gather, J. Zimmermann and J.H. Bartels, "Exact methods for the resource levelling problem," Journal of Scheduling, vol. 14, no. 6, p. 557–569, 2011.
[33] M. Hariga and S.M. El-Sayegh, "Cost optimization model for the multiresource leveling problem with allowed activity splitting". Journal of Construction Engineering and Management, vol. 137, no.1, p. 56-64, 2011.
[34] P. Burman, Precedence Networks for Project Planning and Control, McGraw Hill, London, 1972.
[35] R. B. Harris, "Packing method for resource leveling (PACK)," Journal of Construction Engineering and Management, vol. 116, no. 2, p. 331-350, 1990.
[36] J. Martinez and P. Ioannou, "Resource leveling based on the modified minimum moment heuristic," Civil and Building Engineering, p. 287-294, 1993.
[37] M. Hiyassat, "Modification of minimum moment approach in resource leveling," Journal of Construction Engineering and Management, vol. 126, no. 4, p. 278-284, 2000.
[38] V.A. Jeentra, O.V. Ksishnaiah Chetty and J. Prashanth Redy, "Petri nets for project management and resource leveling," International Journal of Advanced Manufacturing Technology, vol. 16, p. 516-520, 2000.
[39] T. Hegazy, "Optimization of resource allocation and leveling using genetic algorithms," Journal of Construction Engineering and Management, vol. 125, no. 3, p. 167-175, 1999.
[40] J. Roca, E. Pugnaghi and G. Libert, "Solving an extended resource leveling problem with multiobjetive evolutionary algorithms," World Academy of Science, Engineering and Technology, vol. 46, p. 712-723, 2008.
[41] J. Son, and M. Skibniewski, "Multiheuristic approach for resource leveling problem in construction engineering: hybrid approach," Journal of Construction Engineering and Management, vol. 125, no. 1, p. 23-31, 1999.
[42] H. Alsayegh and M. Hariga, "Hybrid meta-heuristic methods for the multi-resource leveling problem with activity splitting," Automation in Construction, vol. 27, p. 89-98, 2012.
[43] G. K. Koulinas and K. P. Anagnostopoulus, "Construction resource allocation and leveling using a threshold accepting based hyperheuristic algorithm," Journal of Construction Engineering and Management, vol. 138, no. 7, p. 854-863, 2012.
[44] A. Mingozzi, V. Maniezzo, S. Ricciardelli and L. Bianco, "An exact algorithm for the resource-constrained project scheduling based on a new mathematical formulation," Management Science, vol. 44, no. 5, p. 714-729, 1998.
[45] I. Paya-Zaforteza, V. Yepes, F. González-Vidosa and A. Hospitaler, "On the Weibull cost estimation of building frames designed by simulated annealing," Meccanica, vol. 45, p. 693-704, 2010.
[46] A. Carbonell, V. Yepes and F. González-Vidosa. "Automatic design of concrete vaults using iterated local search and extreme value estimation," Latin American Journal of Solids and Structures (accepted, in press).

[47] W. Weibull, "A statical distribution function of wide applicability," Journal of Applied Mechanics, vol. 18, no. 3, p. 293-297, 1951.
[48] R. Fisher and L. Tippett, "Limiting forms of the frequency distribution of the largest or smallest member of a sample," Mathematical Proceedings of the Cambridge Philosophical Society, vol. 24, p. 180-190, 1928.
[49] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, 1975.
[50] M. Mahdavia, M. Fesangharyb and E. Damangirb, "An improved harmony search algorithm for solving optimization problem," Applied Mathematics and Computation, vol. 188, no. 2, p. 1567-1579, 2007.
[51] C. Lin, "An adaptive genetic algorithm based on population diversity strategy," Third International Conference on Genetic and Evolutionary Computing, Guilin, China, 2009.
[52] M. Srinivas and L. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," IEEE Transactions on Systems, Man and Cybernetics, vol. 24, no. 4, p. 656-667, 1994.
[53] R. Kolish, A. Sprecher and A. Drexl, "Characterization and generation of a general class of resource-constrained project scheduling problems," Management Science, vol. 41, no. 10, p. 1693-1703, 1995.
[54] J. L. Ponz-Tienda, J. Belloch-Marco, C. Andrés-Romano and D. Gil-Senabre, "A matrix algorithm for the RUPSP non spliting allowed," Revista de la Construcción, vol. 10, no. 2, p. 90-103, 2011.
[55] J.L. Ponz-Tienda, Gestión de Proyectos con Excel 2010, Anaya Multimedia (in Spanish), Madrid, 2011.
[56] R. Álvarez-Valdés and J.M. Tamarit, "Heuristic algorithms for resource-constrained project scheduling: a review and empirical analysis," in: R. Skovinski and J. Weglarz (Eds.), Advances in Project Scheduling, p. 113-134, 1989.

Figure 1

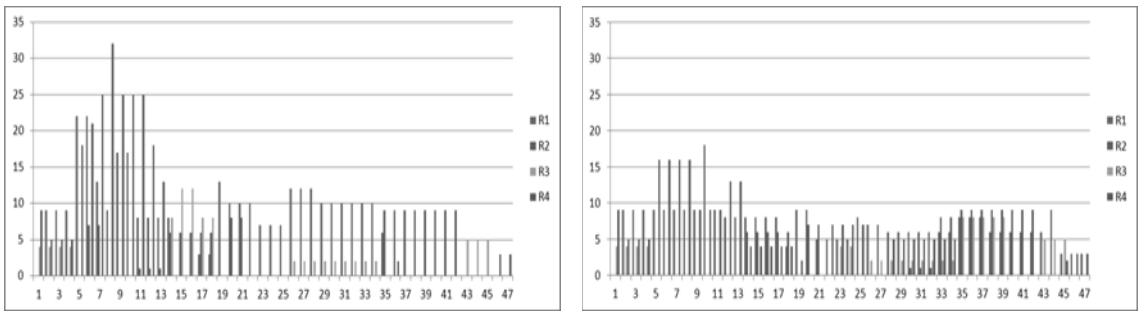

Fig. 1 Initial $\sum_{t=1}^{\bar{T}} u_{kt}^{\ 2} = 10.669$; leveled $\sum_{t=1}^{\bar{T}} u_{kt}^{\ 2} = 6.477$

Figure 2

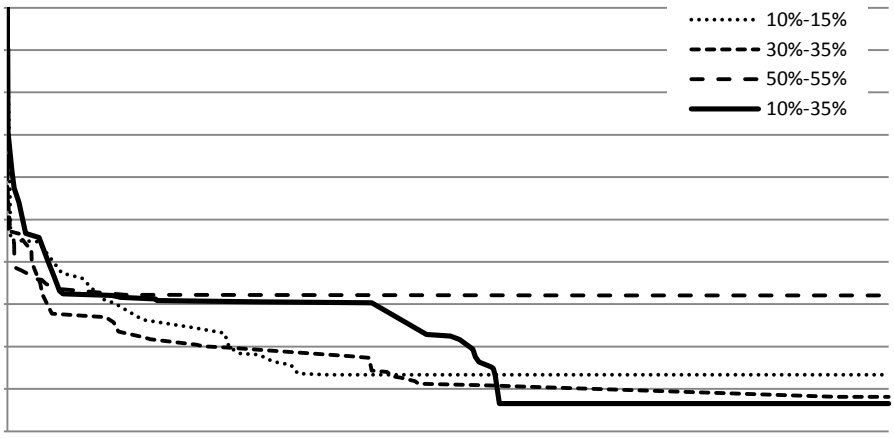

Fig. 2 Effect of different values for the feasibility lower and upper range

Figure 3

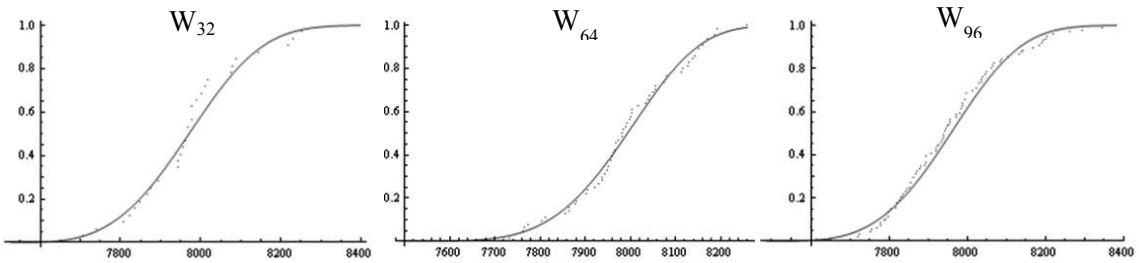

Fig. 3 Approach of Weibull distribution for W_{32}, W_{64} and W_{96}

Figure 4

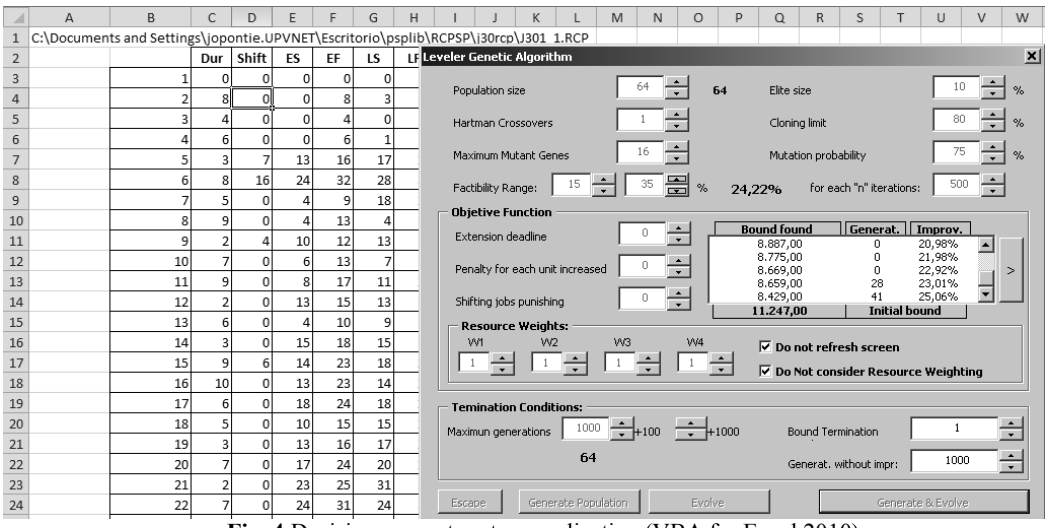

Fig. 4 Decision support system application (VBA for Excel 2010)

Figure 5

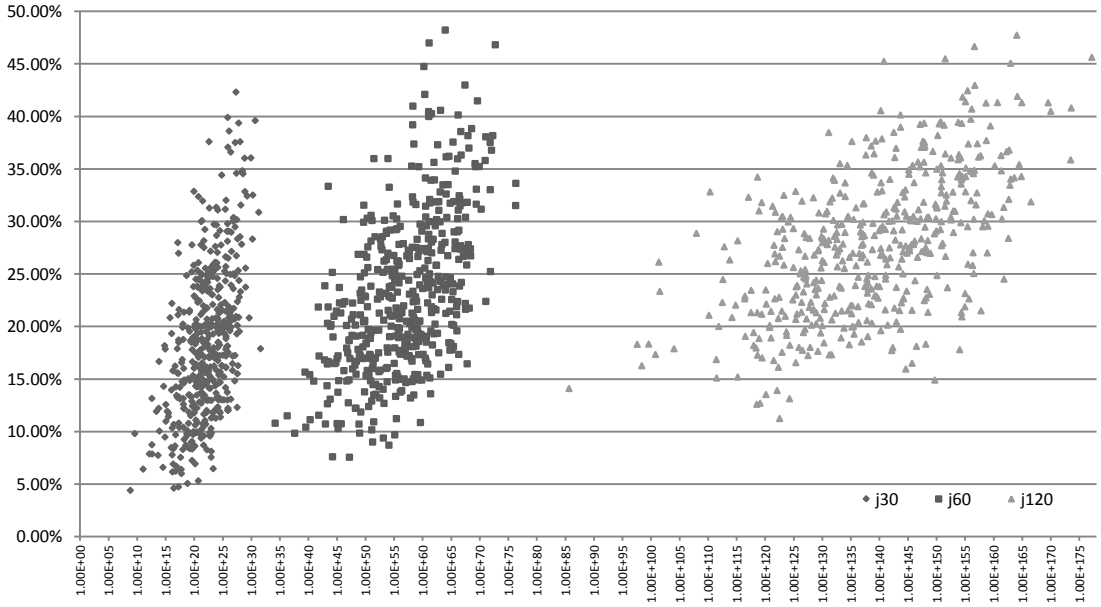

Fig. 5 Correlation (r_{xy}); Improvement over logarithm of the time complexity $O\left(q^n\right)$

Figure 6

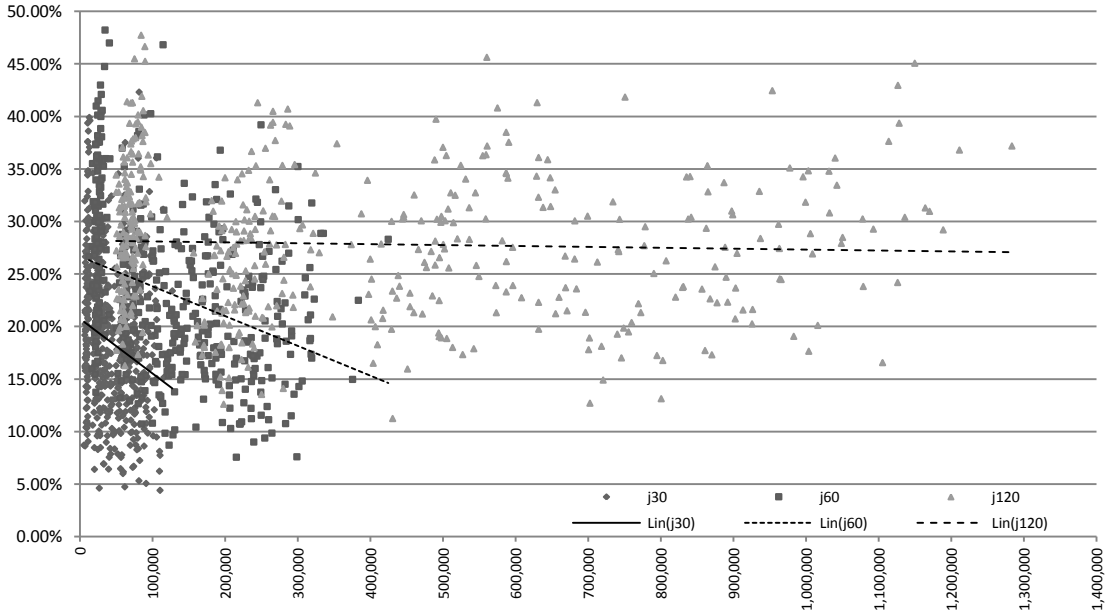

Fig. 6 Correlation (r_{xy}); Improvement over initial sum of the squares

Figure 7

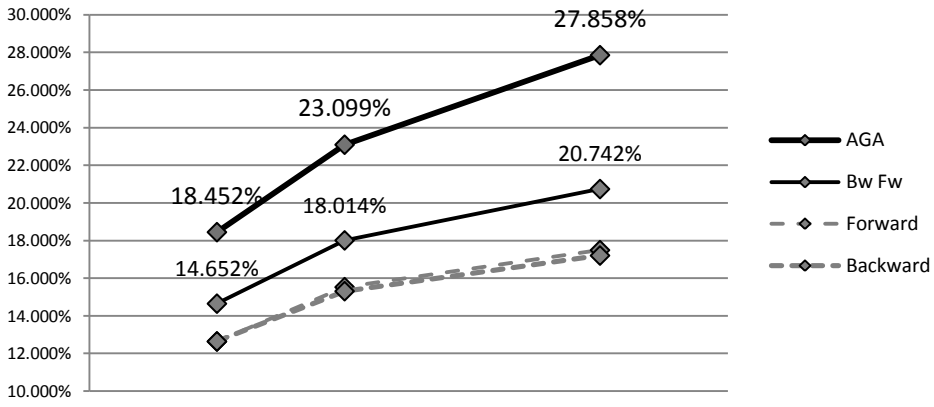

Fig. 7 Heuristic results vs. proposed AGA

Table 1
GA parameters (1)

	Totaljobs	*Popsize*	*Elitesize*	*Crosspoints*	*mutprob*	*Flr*	*Fur*
*j*30	32	32,64,96	0.1	1	0.1	15%	35%
*j*60	62	62,124,186	0.1	1	0.1	15%	35%
*j*120	122	122,244,366	0.1	1	0.1	15%	35%

Table 2
GA parameters (2)

	ε	$C_k; k = \{1,2,3,4\}$	$\overline{T}, \Delta\overline{T}$	ρ	instances	max *iterations*
*j*30	0	1	RUPSP, 0	0	480	1000
*j*60	0	1	RUPSP, 0	0	480	2000
*j*120	0	1	RUPSP, 0	0	480	3000

Table 3
Test statistics

	Average μ	Deviation σ	Min value	Max value	% values $\mu \pm 2\sigma$
*j*30	18.45%	0.07176	4.41%	42.33%	96.04%
*j*60	23.10%	0.07483	7.57%	48.24%	95.42%
*j*120	27.86%	0.06737	11.25%	47.74%	96.25%

Table 4
Test correlation

	Over complexity $O(q^n)$			Over initial Sum of the squares		
	j30	*j60*	*j120*	*j30*	*j60*	*j120*
Covariance	0.146	0.301	0.556	-449.025	-2295.646	-854.354
Pearson (r_{xy})	0.544	0.544	0.583	-0.213	-0.341	-0.040
r_{xy}^2	29.578%	29.624%	33.962%	4.544%	11.609%	0.160%

Table 5
Heuristic results vs. proposed AGA

	j30	j60	j120
Parallel forward scheme	12.645%	15.314%	17.208%
Parallel backward scheme	12.641%	15.530%	17.496%
Fw-Bw scheme	14.652%	18.014%	20.742%
AGA	18.452%	23.099%	27.858%