EngOpt 2012 – 3rd International Conference on Engineering Optimization
Rio de Janeiro, Brazil, 01 - 05 July 2012.

Sequencing Activities in a Project Network considering Resource Complementarity

Helder Cruz da Silva[1], Anabela Pereira Tereso[2], José António Oliveira[2]

[1] IFAM – Instituto Federal de Educação Tecnológica do Amazonas – Brazil (helder@ifam.edu.br)
[2] Universidade do Minho – Portugal (anabelat,zan@dps.uminho.pt)

Abstract
Project management is a methodology widely used in organizations that believe in innovation and choose to organize their resources around projects. This paper presents new results and developments of a model that address the issue of optimal resource allocation, and more specifically, the analysis of complementarity of resources (primary resource and supportive resource) in a project. The concept of complementarity, which has been discussed based on an economic view, can be incorporated into the engineering domain as an enhancement of the efficacy of a "primary" resource (P-resource) by adding to it another "supportive" resource (S-resource). No replacement takes place. The gain achieved from such action is manifested in improved performance; e.g., shorter duration or improved quality, because of the enhanced performance of the P-resource. But such gain is usually achieved at an increased cost; namely the cost of the support resource(s).

We developed a conceptual system capable of determining the ideal timing, and the ideal mixture of resources allocated to the activities of a project, such that the project is completed on time, if not earlier, with minimal cost. We present new computational results of a Genetic Algorithm, based in a random keys alphabet, with an optimized process that allowed reaching better results. The sequence of activities and the resource combinations for each pair activity/resources were obtained, respecting network constraints, showing the flexibility of the solution considering resources distribution and early resources release.

Keywords: Project Management, Scheduling, Complementarity of Resources, Operational Research, Genetic Algorithms.

1. Introduction

Project Management is recognized as an important activity in many companies nowadays. It is crucial to have a clear perception of the different phases of a project life cycle, the processes, techniques and tools appropriate to its management, taking into account the specific environment in which the project takes place as well as its size and complexity.

This paper deals with the issue of optimal resource allocation in activity networks under conditions of resource complementarity. Another related topic is the study of the resource substitutability, in which one resource replaces another; for example, one may use semi-skilled labor instead of high skilled labor, or an old machine (m/c) instead of a new (and more efficient) one. A certain loss (or gain) is realized, perhaps in time or quality, which is offset by the gain in cost or availability. Alternatively, there are several studies dealing with the problem of multiple skills; see [1], [2], [3], [4] and [5]. The problem posed in this context is usually framed as seeking the most economical diversity that satisfies an uncertain demand with high enough probability. In such context there is cost incurred by the increased diversity of skills [2] (e.g. a travel guide who speaks several languages, or a hand tool that can serve as a pair of scissors and a screwdriver) and there is gain secured by having a smaller number of service mechanisms.

The concept of complementarity which has been discussed based on an economic view [6] can be incorporated into the engineering domain as an enhancement of the efficacy of a "primary" resource (P resource) by adding to it another "supportive" resource (S-resource). No replacement takes place. The gain achieved from such action is manifested in improved performance; e.g., shorter duration or improved quality, because of the enhanced performance of the P-resource. But such gain is usually achieved at an increased cost; namely the cost of the support resource(s).

The issue then becomes: how much additional support should be allocated to project activities to achieve improved results most economically? After answering to this question, we will evaluate the effect of early supportive resources release for the results.

2. Problem Description

Consider a project network in the activity-on-arc (AoA) representation: $G = (N, A)$ with the set of nodes $|N| = n$ (representing the "events") and the set of arcs $|A| = m$ (representing the "activities"). Each activity may require the simultaneous use of several resources [7], [8], [9] and [10].

There is a set of "primary" resources, denoted by P, with $|P| = \rho$. Typically, a primary resource has a capacity of several units (say workers, machines, processors; etc) [3]. Additionally, there is a pool of "support" resources, denoted by S, with $|S| = \sigma$ (such as less-skilled labor, or computers and electronic devices; etc.) that may be used in conjunction with the primary resources to enhance their performance.

The number of support resources varies with the resource, and the relevance of each to the P-resources may best be represented in matrix format as shown in Table 1 (ϕ indicates inapplicability).

Table 1: Applicability and impact of support resources

S-Resource →	s_1	$\cdots$	s_q	$\cdots$	s_σ
↓ *P-Resource*					
r_1	$v(1,1)$	$\cdots$	$\emptyset$	$\cdots$	$v(1,\sigma)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
r_p	$\emptyset$	$\cdots$	$v(p,q)$	$\cdots$	$v(p,\sigma)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
r_ρ	$v(\rho,1)$	$\cdots$	$v(\rho,q)$	$\cdots$	$v(\rho,\sigma)$

In Table 1 an entry $v(r_p, s_q) \neq \emptyset$, measures the enhancement offered by *S*-resource s_q to *P*-resource r_p.

Although various models of the impact of the support resource may be constructed, we will discuss only two. The choice of the applicable model is decided empirically from data on the actual performance of the process.

If $0 < v(r_p, s_q) \leq 1$, then it indicates the fraction by which the support resource s$_q$ improves the performance of primary resource r_p. Typically $v(r_p, s_q) \in [0.10, 0.50]$. In this case the performance of the allocation of P-resource r_p to activity *a*, which is denoted by $x_a(r_p)$, is augmented to,

$$x_a(r_p, s_p) = x_a(r_p) + v(r_p, s_p) \tag{1}$$

If $1 < v(r_p, s_q) < U < \infty$, then it indicates the multiplier of the *P*-resource allocation.

Typically $1.10 \leq v(r_p, s_q) \leq 2.0$. In this case the performance of the allocation of *P*-resource r_p is augmented to

$$x_a(r_p, s_p) = x_a(r_p) \cdot v(r_p, s_q) \tag{2}$$

In the treatment below, we shall adopt mode denoted by Eq.(1). For the sake of simplicity, we make the following assumptions.

2.1. First Assumption

The impact of the *S*-resources is additive: if a subset $\{s_q\}_{q=1}^{\sigma}$ of the *S*-resources is used in support of *P*-resource r_p in activity *a* then the performance of the former is enhanced to,

$$x_a\left(r_p, \{s_q\}_{q=1}^{\sigma}\right) = x_a(r_p) + \sum_{q=1}^{\sigma} v(r_p, s_q) \tag{3}$$

In the sequel we consider the possible addition of only a single *S*-resource; the discussion can be easily extended to multiple *S*-resources.

The primary resource $r_p \in P$ would accomplish activity a in time $y_a(r_p)$. If it is enhanced by the addition of *S*-resource s_q then its processing time decreases to $y_a(r_p, s_q)$, with $y_a(r_p, s_q) < y_a(r_p)$. The issue now is to express the functional relationship between the resource allocation (both primary and support) and the activity duration.

Let $w_a(r_p)$ denote the work content of activity a of *P*-resource r_p. Let $x_a(r_p)$ denote, as suggested above, the amount of primary resource *r$_p$* allocated to activity *a*.

2.2. Second Assumption

The duration of activity *a* when using resource *r$_p$* is given by [11]:

$$y_a(r_p) = \frac{w_a(r_p)}{x_a(r_p)} \tag{4}$$

If support resource $s_q$is added to the primary resource *r$_p$* then the duration becomes (considering model (1)),

$$y_a(r_p, s_q) = \frac{w_a(r_p)}{x_a(r_p, s_q)} \tag{5}$$

To illustrate, suppose an activity has work content $w_a(r_p) = 36$ man-days. Further, assume the *S*-resource s_q yields a rate $v(r_p, s_q) = 0.50$. If $x_a(r_p) = 0.85$ then in the absence of the support resource the duration of the activity would be
$y_a(r_p) = 36/0.85 = 42.35$ days.

But in the presence of the *S*-resource the duration would be reduced to $y_a(r_p, s_q) = 36$ / ((0.85 + 0.5) = 26.67 days, a saving of approximately 37%.

If $x_a(r_p) = 1.5$ then in the absence of the *S*-resource the duration of the activity would be $y_a(r_p) = 36/1.5 = 24$ days.

But in the presence of the *S*-resource the duration would be only $y_a(r_p, s_q) = {}^{36}/_{(1.5\,+\,0.5)}$ = 18 days, a saving of 25%.

An activity normally requires the simultaneous utilization of more than one *P*-resource for its execution. The problem then becomes: "At what level should each resource be utilized and which supportive resource(s) should be added to it (if any) in order to optimize a given objective?"

Recall that the processing time of an activity is given by the maximum of the durations, as in Eq.(6), that would result from a specific allocation to each resource (see a previous discussion on the evaluation of the duration considering multiple resources in [8], [9] and [10].

$$y(a) = \underbrace{max}_{all\, r_p}\{y_a(r_p)\} \tag{6}$$

To better understand this representation, consider the small project of Figure 1 and Figure 2 with three activities. Assume that the project requires the utilization of four *P*-resources; not all resources are required by all the activities. The resource requirements of each activity are indicated in Table 2.

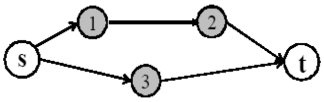


Figure 1: Project with 3 activities AoN.

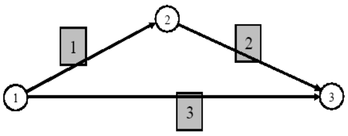


Figure 2: AoA representation.

Table 2: Work content (in man-days) of the activities of project 1.

P-Resource →	*1*	*2*	*3*	*4*
Availability	2	1	3	2
Activity				
A1	16	0	12	12
A2	0	7	0	8
A3	20	22	0	0

Table 2 is to be read as follows. There are two units available of resources #1 & #4; one unit of resource #2 and 3 units of resource #3. Activity 1 requires 16 man-days of resource #1 and 12 man-days of each of resources #3 and #4. It does not require resource #2.
The relevance and impact of the support resources are represented in Table 3, which may be read as follows: *S*-resources 1 and 2 have availability of one unit each. *S*-resource 1 can support *P*-resources 1 and 3 and *S*-resource 2 can support *P*-resources 1 and 2; no support is available for *P*-resource 4.

Table 3: The P-S matrix: Impact of S-resources on P-resources.

	P-RES →	*1*	*2*	*3*	*4*
↓*S-RES*	↓ *AVAILABILITY*				
1	*1*	0.25	ϕ	0.25	ϕ
2	*1*	0.15	0.35	ϕ	ϕ

With little additional data processing, the problem can be enriched with the inclusion of the cost of the resource utilization at each level. Then in each cell in both the primary and support resource tables there shall be added the marginal cost for the resource per unit time. If the project gains a bonus for early completion and incurs a penalty for late completion then one can easily include such costs in the objective function.
At time 0 we may initiate both activities A1 and A3 because their required *P*-resources are available (A1 requires *P*-resources 1, 3 and 4 and A3 requires *P*-resources 1 and 2.). Assume for the moment that no support resource is allocated to either activity. Further, suppose that each unit of the primary resource is devoted to its respective activity at level 1; i.e.,

$$x_1(r_1) = 1 = x_1(r_3) = x_1(r_4) \tag{7}$$

$$x_3(r_1) = 1 = x_3(r_2) \tag{8}$$

Observe that the *P*-resource availabilities have been respected: the two units of *P*-resource 1 have been equally divided between the two activities; *P*-resource 2 is not required by A1 and the unit available is allocated to A3, *P*-resources 3 and 4 are required only by A1. The *P*-resource allocation would look as shown in Table 4.

Table 4: The P-resources allocation at time 0.

	P-Resource			
Activity	1	2	3	4
A1	1	0	1	1
A3	1	1	0	0
Total Allocation	2	1	1	1

The durations of the two activities shall be:

$$A1: y(1) = max\left\{\frac{16}{1}, \frac{12}{1}, \frac{12}{1}\right\} = 16 \text{ days}$$
$$A3: y(3) = max\left\{\frac{20}{1}, \frac{22}{1}\right\} = 22 \text{ days}$$

At time t = 16 activity A1 completes processing and A2 becomes sequence feasible. Unfortunately it cannot be initiated because *P*-resource 2, of which there is only one unit, is committed to A3 which is still on-going. Therefore activity 2 must wait for the completion of A3, which occurs at t = 22. When initiated at resource levels $x_2(r_2) = 1 = x_2(r_4)$, it will consume $y(2) = max\left\{\frac{7}{1}, \frac{8}{1}\right\} = 8$ days to complete.

The project duration (time of completion of node 3 in the AoA network) would be $t_3 = 22 + 8 = 30$. If the due date of the project were specified at $T_s = 24$, the project would be 6 days late.

2.3. Impact of the Support Resources

Suppose that at the start of the project both support resources were allocated to activity 3 as follows: $s_1 \rightarrow r_1$ and $s_2 \rightarrow r_2$; then $x_3(r_1, s_1) = 1 + 0.25$ and $x_3(r_2, s_2) = 1 + 0.35$.

The duration of the A3 would change to $y(3) = max\left\{\frac{20}{1.25}, \frac{22}{1.35}\right\} \approx 16.30$ days.

At t = 16.30 activity 2 can be initiated because primary resource 2 would be freed. If we continue with $x_2(r_2) = 1 = x_2(r_4)$ it will consume the same 8 days to complete and the project duration would be $t_3 = 16.30 + 8 = 24.30$.

The project is almost on time!

Whether or not such allocation of the support resources is advisable shall depend on the relative costs of the *S*-resources and tardiness. In fact, again depending on the relative costs, it may be advisable to have allocated *S*-resource 1 to activity 1 when it is initiated at time 0 and, when completed, continue as above with activity 3, since the gain in the project completion time may secure some bonus payment that would more than offset the cost of the added support. It is also possible to allocate more than one *S*-resource to complement the *P*-resources in some activities. All these, and other, possibilities should be resolved by a formal mathematical model.

3. Mathematical Model

We assume that all costs are linear or piece-wise linear in their argument.

Let:

C^k*:* the *k*th uniformly directed cutset *(udc)* of the project network that is traversed by the project progression; $k = 1, \ldots, K$.

$x_a(r_p)$*:* level of allocation of (primary) resource r_p to activity a (assuming integer values from 1 to $Q_P(p)$ if the activity needs this resource).

$x_a^{r_p}(s_q)$: level of allocation of secondary resource s_q to primary resource r_p in activity a (assuming integer values from 0 to $Q_S(q)$).

$x_a\left(r_p, \{s_q\}_{q=1}^{\sigma}\right)$: total allocation of resource r_p (including complementary resources) to activity a.

$v(r_p, s_q)$: degree of enhancement of *P*-resource r_p by *S*-resource s_q.

$w_a(r_p)$: work content of activity a when *P*-resource r_p is used.

$y_a(r_p, \{s_q\}_{q=1}^{\sigma})$: duration of activity a imposed by primary resource r_p (including enhancement by complementary resources).

$y(a)$: duration of activity a (considering all resources).

ρ : number of primary resources, $\rho = |P|$.

σ : number of secondary resources, $\sigma = |S|$.

$Q_P(p)(Q_S(q))$: capacity of *P*-resource r_p (*S*-resource s_p) available.

γ_p : marginal cost of *P*-resource r_p.

γ_q : marginal cost of *S*-resource s_q.

γ_E : marginal gain from early completion of the project.

γ_L : marginal loss (penalty) from late completion of the project.

t_i : time of realization of node i (AoA representation), where node 1 is the “start node” of the project and node n its “end node”.

T_s : target completion time of the project (due date).

$c_R(a, r_p)$: cost of resources for activity a and resource r_P (including complementary resources).

$c_R(a)$: cost of resources for activity a (includes all resources).

e : earliness.

d : tardiness (delay).

c_E : cost of earliness.

c_T : cost of tardiness.

c_{ET}: cost of earliness and tardiness.

TC: total cost.

The constraints are enumerated next. To avoid confusion with node designation we refer to an activity as *“a”* and to a node as *i* or *j*. The notation $a \equiv (i,j)$ means that activity a is represented by arc (i,j).

Respect precedence among the activities:

$$t_j \geq t_i + y(a), \quad \forall a \equiv (i,j) \in A \qquad (9)$$

Define total allocation of resource r_p (including complementary resource) in activity a,

$$x_a\left(r_p, \{s_q\}_{q=1}^{\sigma}\right) = x_a(r_p) + \sum_{q=1}^{\sigma} v(r_p, s_q) * x_a^{r_p}(s_q) \qquad (10)$$

Define the duration of each activity when using each P-resource; then define the activity's duration as the maximum of individual resource durations:

$$y_a(r_p, \{s_q\}_{q=1}^{\sigma}) = \frac{w_a(r_p)}{x_a\left(r_p, \{s_q\}_{q=1}^{\sigma}\right)} \tag{11}$$

$$y(a) = \underbrace{max}_{all\, r_p} \left\{ y_a(r_p, \{s_q\}_{q=1}^{\sigma}) \right\} \tag{12}$$

Respect the P-resource availability at each *udc*[1] traversed by the project in its execution,

$$\textstyle\sum_{a \in C^k} x_a(r_p) \leq Q_P(p), \ \forall p \in P \tag{13}$$

in which *Q(p)* is the capacity (i.e., availability) of P-resource r_p (in the three activities example given above, the vector *Q (P)* = (2, 1, 3, 2)).

Respect also the S-resources availability, considering again the current *udc*,

$$\textstyle\sum_{a \in C^k} x_a^{r_p}(s_q) \leq Q_S(q) \ \ \forall q \in S \tag{14}$$

in which $Q_S(q)$ is the capacity of S-resource s_q (in the three-activities example given above, the vector $Q_S(q) = (1,1)$). Note that the requirement that an S-resource is applied only to its relevant P-resources is taken care of in the *P-S* matrix (see Table 3); what this constraint accomplishes is to limit its use to each resource's total availability.

The difficulty in implementing this constraint stems from the fact that we do not know a priori the identity of the *udc*'s that shall be traversed during the execution of the project, since that depends on the resource allocations (both the P- and S-resources). A circularity of logic is present here: the allocation of the resources is bounded by their availabilities at each *udc*, but these latter cannot be known except after the allocations have been determined. Unfortunately, this vicious cycle cannot be broken by a blanket enumeration of all the *udc*'s of the project because that would over-constrain the problem. There are several ways to resolve this circularity, formal as well as heuristic. The formal ones are of the integer programming genre which, when combined with the nonlinear mathematical programming model presented above, impose a formidable computing burden. The heuristic approaches are more amenable to computing; we propose such a heuristic approach below.

Define earliness and tardiness by,

$$e \geq T_s - t_n \tag{15}$$

$$d \geq t_n - T_s \tag{16}$$

$$e, d \geq 0 \tag{17}$$

The objective function is composed of two parts: the cost of use of the P- and S-resources, and the gain or loss due to earliness or tardiness, respectively, of the project completion time t_n relative to its due date.

For simplicity, we make the following two assumptions:

(i) The cost of resource utilization is quadratic in the resource allocation for the duration of the activity [7], [11], which renders the cost linear in work content (recall that the work content is assumed a known constant),

$$c_R(a, r_p) = \left(\gamma_p * x_a(r_p) + \gamma_q * \textstyle\sum_{q=1}^{\sigma} x_a^{r_p}(s_q)\right) * w(a, r_p) \tag{18}$$

$$c_R(a) = \textstyle\sum_{all\, r_p} c_R(a, r_p) \tag{19}$$

(ii) the earliness-tardiness costs are linear in their respective marginal values;

$$c_{ET} = c_E + c_T = \gamma_E \cdot e + \gamma_L \cdot d \tag{20}$$

The desired objective function may be written simply as

$$min\, TC = \textstyle\sum_{a \in A} c_R(a) + c_{ET} \tag{21}$$

4. Description of the Procedure Adopted

The mathematical model presented in the previous section is complex and represents a hard problem to solve. Actually this problem is a generalization of the well-known Resource Constraint Project Scheduling Problem (RCPSP). For such problems, the use of methods of exact solution is limited to small instances. Thus, we chose to use a heuristic approach. The procedure we have used to solve this problem will be presented below.

4.1. Genetic Algorithm

The heuristic method selected was a genetic algorithm (GA). The GA's simplicity to model more complex problems and its easy integration with other optimization methods were factors that were considered before it was chosen.

The same type of GA has previously been successfully applied to classic scheduling problems [12, 13] and its variants [14].

For the Activity Networks under Resource Complementarity, we define a chromosome with $n(\rho + \sigma + 1)$ genes. For each activity, the chromosome gives the quantity of each P-resource and the quantity of the complementary S-resource, as well as the priority.

The genetic algorithm has a very simple structure and can be represented by Algorithm 1. It begins with population generation and its evaluation. The first population is randomly generated. Attending to the fitness of the chromosomes the individuals are selected to be parents. We choose the parents selecting them for crossover using roulette-wheel selection method [15].

[1] The acronym *udc* stands for 'uniformly directed cutset', which is a cutset of the graph in which all arrows are directed from the subset of nodes H which contains the origin node, to the complementary subset $\overline{H} = N - H$ which contains the terminal node.

```
Step 1    begin
Step 2    P ← GenerateInitialPopulation()
Step 3    Evaluate(P)
Step 4    while termination conditions not meet do
              (1)  P' ← Recombine(P)  //UX
              (2)  P'' ← Mutate(P')
              (3)  Evaluate(P'')
              (4)  P ← Select(P ∪ P'')
Step 5    end while
```

Algorithm 1: Genetic algorithm

The crossover is applied and it generates a new temporary population that is also evaluated. Comparing the fitness of the new elements and of their progenitors the former population is updated.

The Uniform Crossover (UX) is used in this work. This genetic operator uses a new sequence of random numbers and swaps both progenitors' alleles if the random key is greater than a prefixed value. Figure 3 illustrates the UX's application on two parents (prnt1, prnt2), and swaps alleles if the random key is greater or equal than 0.75. The genes 3, 4 and 12 are changed and it originates two descendants (dscndt1, dscndt2). Descendant 1 is similar to parent 1, because it has about 75% of genes of this parent. In this preliminary version we do not implement a mutation operator. We intend to implement a mutation operator to select randomly a gene and replace the allele value by other value randomly generated. With this strategy we change the priority of an activity or we change the number of resources (primary or support) assigned to an activity.

gene	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	***11***	***12***
prnt1	0,89	0,49	0,24	0,03	0,41	0,11	0,24	0,12	0,33	0,30	0,27	0,18
prnt2	0,83	0,41	0,40	0,04	0,29	0,35	0,38	0,01	0,42	0,32	0,28	0,13
randkey	0,64	0,72	0,75	0,83	0,26	0,56	0,28	0,31	0,09	0,11	0,37	0,76
dscndt1	0,89	0,49	0,40	0,04	0,41	0,11	0,24	0,12	0,33	0,30	0,27	0,13
dscndt2	0,83	0,41	0,24	0,03	0,29	0,35	0,38	0,01	0,42	0,32	0,28	0,18

Figure 3: The UX crossover

4.2. Constructive Algorithm

In order to build a solution from the chromosome, we use a constructive algorithm based on Giffler and Thompson's algorithm (GT) [16]. While the GT algorithm can generate all the active plans for the JSSP, the constructive algorithm only generates the plan in agreement with the chromosome. As advantages of this strategy, we have pointed out the minor dimension of the solution space, which includes the optimum solution and the fact that it does not produce impossible or disinteresting solutions from the optimization point of view. On the other hand, since the dimensions between the representation space and the solution space are very different, this option can represent a problem because many chromosomes can represent the same solution.

The constructive algorithm has *n* stages and in each stage an activity is scheduled. To assist the algorithm's presentation, consider the following notation existing in stage *t*:

P_t- the partial schedule of the $(t-1)$ scheduled activities;

S_t- the set of activities schedulable at stage t, i.e. all the activities that must precede those in S_t are in P_t;

σ_k- the earliest time that activity a_k in S_t could be started. This time respects the conclusion of all predecessors of a_k and the availability of all resources that a_k will use (primary and supportive resources);

ϕ_k - the earliest time that activity a_k in S_t could be finished;

ϕ^* - the selected minimal value of $\emptyset_k$ considering a_1 in S_1 where $\left(\emptyset^* = \min_{a_1 \in S_1}\{\emptyset_k\}\right)$;

S_t^* - the conflict set formed by a_k in S_t, $\sigma_k < \phi^*$;

a_k^* - the selected activity to be scheduled at stage t.

The constructive algorithm of solutions is presented in a format, similar to the one used by [17] to present the GT algorithm (see Algorithm 2). In Step 3, instead of using a priority dispatching rule, the information given by the chromosome is used. If the maximum allele value is equal for two or more activities, one is chosen randomly.

```
Step 1    Let t=1 with P_1 being null. S_1 will be the set of all activities with no predecessors, in other words those that are
          connected to start vertex.
Step 2    Find $\emptyset^* = \min_{a_k \in S_t}\{\emptyset_k\}$ and identify $A^*$ from $S_t^*$.
Step 3    Select activity $a_k^*$ in $S_t^*$ with the greatest allele value greatest priority.
Step 4    Move to next stage by
              (1)  adding $a_k^*$ to $P_t$, so creating $P_{t+1}$ ;
              (2)  deleting $a_k^*$ from $S_t$ and creating $S_{t+1}$ by adding (if exist) to $S_t$ the activities that directly follows $a_k^*$ and
                   have all predecessors in $P_{t+1}$ ;
              (3)  incrementing $t$ by 1.
Step 5    If there are any activities left unscheduled ($t < N$), go to Step 2.
          Otherwise, stop.
```

Algorithm 2: Constructive algorithm

Consider the example presented in Figure 2, Table 1 and Table 2 with three activities $(\mathrm{A}_1, \mathrm{A}_2, \mathrm{A}_3)$. In this instance, there are $\rho = 4$ primary resources and $\sigma = 2$ supportive resources. A chromosome to represent a solution for this instance has 21 genes, since there are six genes for each activity to establish the number of elements of each resource that will be used, plus a gene A_k that defines the activity's priority. Table 5 presents a chromosome of random keys values for this instance. The values are generated randomly between 0 and 99.

Table 5: A Chromosome

A1	P1	P2	P3	P4	S1	S2	A2	P1	P2	P3	P4	S1	S2	A3	P1	P2	P3	P4	S1	S2
94	51	88	76	52	23	68	**36**	73	60	61	53	75	35	**47**	7	15	42	86	16	16

Activity A_1 has a priority 94, which is the greatest, while A_2 has the lowest priority "36". To define the number of elements of each resource, we use Table 2. The availability of P-resource 3 is 3 units. We define three equal intervals between 0 and 99. For this resource, if the allele is a value between 0 and 32 one unit is assigned. For values between 33 and 66, we assign two units, and for values between 67 and 99, three units will be assigned. To establish the number of units for the supportive resources, the procedure is similar, but it also includes the possibility to assign 0 units, because this is an optional type of resource. Considering these rules, the chromosome presented in Table 5 defines the following units of resources to be used, which are presented in Table 6.

Table 6: Amount of units of resources to be used

A1	P1	P2	P3	P4	S1	S2	A2	P1	P2	P3	P4	S1	S2	A3	P1	P2	P3	P4	S1	S2
Units	2	0	3	2	0	2	Units	0	1	2	2	1	1	Units	1	0	2	0	0	0

The procedure assigns zero units of a *P*-resource to an activity, if the activity does not use that primary resource, which is the case of P_2 in the activity A_1, according to Table 2. The assignment of supportive resources to the primary resources is performed considering the "amount" of work content existent after the assignment of primary resources. The first unit is assigned to the *P*-resource with the largest work content. After the assignment, the amount of work is recalculated, and then the next assignment is made, and so on. Activity A_1 has the following work content (see Table 2):

P-resource →	1	2	3	4
A1	16	0	12	12

Assigning the units of primary resources defined by the chromosome, the duration is then:

P-resource →	1	2	3	4
A1	8	0	4	6

The first unit of supportive resource S_2 is assigned to P_1. Recalculating the durations by Eq. (11), we have:

P-resource →	1	2	3	4
A1	7.442	0	4	6

The second unit of supportive resource S_2 is assigned to P_1. Recalculating the durations by Eq. (11), we have:

P-resource →	1	2	3	4
A1	6.957	0	4	6

Applying the same procedure to the remaining activities, we will have the durations presented in Table 7.

Table 7: Activities duration

P-resource →	1	2	3	4
A1	6.957	0	4	6
A2	0	5.185	4.444	4
A3	20	0	11	0

The result of applying the constructive algorithm can be seen in Figure 4. It presents the evolution of set S^* and the corresponding starting and conclusion times for each activity during the execution of the constructive algorithm. The final Gantt Chart of the project is also presented for the AoN network, and it shows the occupation of all the resource units. Figure 4 shows that one can start earlier the processing of activities considering the time of availability of resources. Consider the case of activity 2 and activity 3. Although activity 3 continues its execution with primary resource P11, it releases resources P31 and P32 earlier. From this moment it is possible to start activity 2 because all the resources that are needed are available.
Consider the due date for the project equal to 24 units of time and the following parameters:

- *P*-resource cost: 4 per unit of work content;
- *S*-resource cost: 1 per unit of work content;
- Delay cost: 60 per unit of time;
- Earliness cost: 40 per unit of time.

The project is complete at time 26.96 with 2.96 units of lateness. The resources cost is equal to 845, and the delay cost is 177.39. The total cost of the solution is 1022.39. (See Eq. (18) - (20)).

S_1	1	3
σ_k	0	0
ϕ_k	6.957	20
a_k^*	1	
S_2	2	3
σ_k	6.957	6.957
ϕ_k	12.14	26.96
a_k^*		3
S_3	2	
σ_k	17.97	
ϕ_k	23.14	
a_k^*	2	

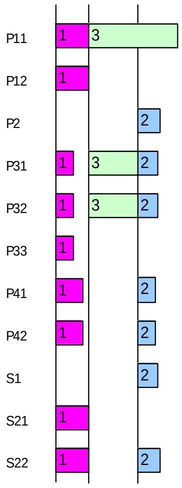


Figure 4: Constructive Algorithm execution

5. Computational Results

We performed several computational experiments with instances proposed by [18]. These experiments allow measuring the effectiveness and efficiency of the methodology. We have chosen five instances with 3, 5, 11, 17 and 24 activities. Despite being small instances, they are sufficient to characterize the performance of the genetic algorithm in terms of evolution of the best solution found throughout the iterations, the robustness of the search, among other features. The networks associated with the selected instances are presented in Figure 5.

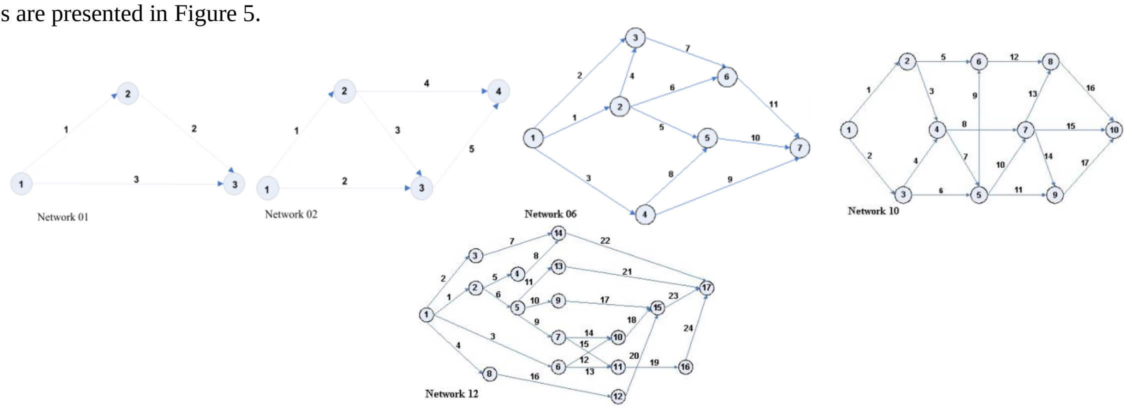


Figure 5: Representative Instances of the Networks

The summary of the results of the computational experiments is presented in Table 8 (instances *Network 01, Network 02, Network 06*, *Network 10* and *Network 12).* We have done 5 runs for each configuration. The average cost values and the best cost value obtained from five runs are presented. We tested different dimensions of population, 20, 100 and 300 individuals. It was intended to evaluate the performance of the fitness function established for the genetic algorithm.

Table 8: Computational Results for Network 01, 02, 06, 10 and 12

Instances	Population	Iterations	Run1	Run2	Run3	Run4	Run5	Average	Better
	20	1000	214	214	214	214	214	214	214
Network 01		5000	214	214	214	214	214	214	214
(3 act)	100	1000	214	214	214	214	214	214	214
		5000	214	214	214	214	214	214	214
	300	1000	214	214	214	214	214	214	214
	20	1000	631	631	631	631	631	631	631
Network 02		5000	631	631	631	631	631	631	631
(5 act)	100	1000	631	631	631	631	631	631	631
		5000	631	631	631	631	631	631	631
	300	1000	631	631	631	631	631	631	631

	20	1000	2584	2548	2529	2427	2444	2506.4	2427
Network 06		5000	2252	2252	2372	2361	2424	2332.2	2252
(11 act)	100	1000	2396	2458	2461	2475	2486	2455.2	2396
		5000	2367	2247	2273	2263	2268	2283.6	2247
	300	1000	2435	2511	2493	2441	2444	2464.8	2435
	20	1000	1569	1577	1574	1577	1579	1575.2	1569
Network 10		5000	1563	1572	1572	1571	1522	1560	1522
(17 act)	100	1000	1574	1572	1575	1569	1572	1572.4	1569
		5000	1567	1569	1575	1567	1568	1569.2	1567
	300	1000	1579	1589	1533	1582	1589	1574.4	1533
	20	1000	4615	4653	3986	4040	3634	4185.6	3634
		5000	3373	3137	3406	3750	3325	3398.2	3137
Network 12	100	1000	4416	4305	3380	4068	3783	3990.4	3380
(24 act)		5000	2943	2537	2731	2731	2770	2742.4	2537
	300	1000	4296	4170	4103	4095	4197	4172.2	4095
		5000	2654	2626	2637	2659	2770	2654	2626

In general, there is consistency in the values obtained, especially in the two smaller instances. With the increase in population, best values are achieved.

In order to have a perception of the evolution of the solution along the iterations, the register of the best solution and the mean values of the five runs (instance: Network 12; iterations: 25000; population: 10), is presented in Figure 6 .

This “curve” is a typical fit in this kind of heuristic. In the first 20% of the iterations we obtain an improvement that is about 90% of the overall improvement that the method does over the 25000 iterations.

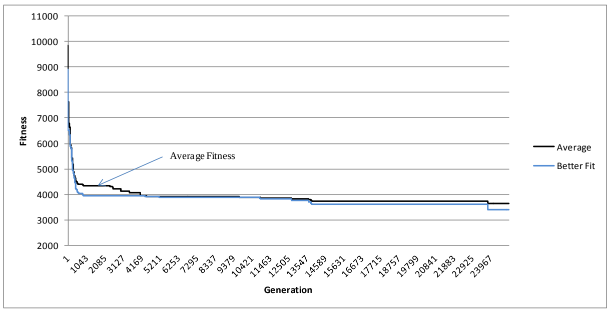
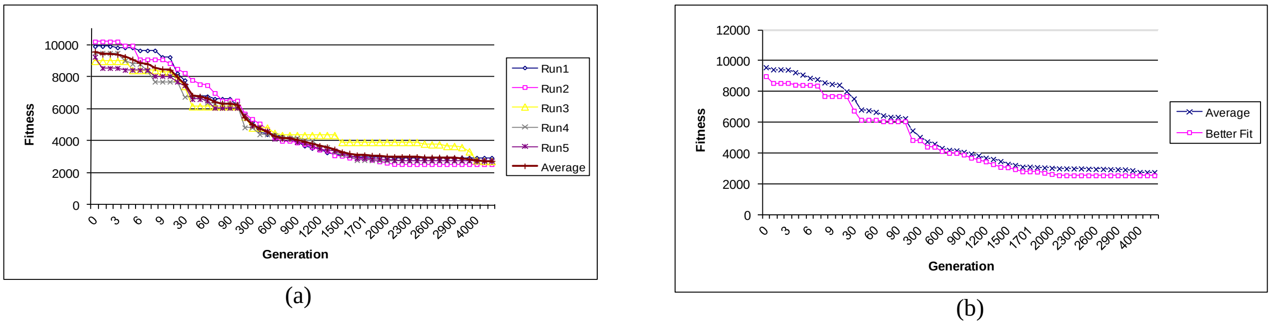


Figure 6: Evolution of the genetic algorithm performance

The summary of the values obtained for the five runs in different iterations between 1 and 5000 is presented in Figure 7(a) and (b).



(a)

(b)

Figure 7: Resume of the values for the 5 runs

The five runs in Figure 7 (a) present a similar and consistent behavior. Figure 7 (b) shows the average value and the best value obtained in five runs. It is possible to check for the consistency in the evolution of the average of the five runs.

Initially we developed a brute force algorithm and considered that all resources used in the processing of an activity are released only at the end of the larger work content of the activity. This algorithm was applied to the network 01 (3 activities), as a hypothetic project (Figure 5), and the better fit was 332.56.

If we observe the results provided by GA algorithm for the same instance, the value of the better fit was 214.00, which means the better fit is 27.53% under the expected value. We recall that in the GA, one or more P-resources can be released before the actual end of the activity (the work content is defined in Table 2).

This way, we can perceive an improvement of the better fit, just considering the earlier release of P-resources, which results in a lower cost and a better resource usage.

6. Conclusion

In this paper we studied the problem of project management with complementary resources. The importance of the problem lies in the opportunity to develop a system that would not only improve the allocation of often scarce resources, but also result in a reduction of uncertainties within the project, combined with increasing performance and reduced cost of the project.

We presented a mathematical model and a genetic algorithm developed to solve the problem. The genetic algorithm is based on random keys that allows for easy adaptation to complex models.

The method was tested with some activity networks and the results obtained allow us to demonstrate its validity, effectiveness and efficiency. Considering the feasibility of the proposed model, we believe that it can provide the user a new option of planning to determine the best combination of resources and lower cost of the project, improving firms capacity planning.

Increasing the size of the network as in Network 12, we observed some distortions caused by false better fitness and premature convergence (that occurs after 20% of the iterations).

We have discussed about the earlier resource release and how can we save more time and money (depending of the network), using this modified GA algorithm, which presented an improvement of network 01 better fit of 27.53%.

For the next steps, we intend to develop some specific genetic operators that will be implemented mainly in the form of mutation, to increase the diversity of the population and reduce the impact of the premature convergence providing a way to escape of the better fitness traps.

7. References

1. Arroub, Y. and Kadrou, N. An efficient algorithm for the multi-mode resource constrained project scheduling problem with resource flexibility. International Conference on Industrial Engineering and Systems Management, 2009, 13-15, Montreal

2. Li, H. and Womer, K. Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm, Journal of Scheduling, 2009, 12(3), 281-298

3. Mulcahy, Rita. PMP, Exam Prep – Rita's Coruse in a Book for Passing the PMP Exam Fifth Edition – For PMBOK Guide – Third Edition. 5a ed. s.l.: RMC Publications, 2005

4. Demeulemeester, E. and Herroelen, W. A Branch-and-Bound Procedure for the Multiple Resource-Constrained Project Scheduling Problem. Management Science, 1992, 38(12), 1803-1818.

5. Patterson, J. H. A comparison of exact approaches for solving the multiple constrained resource project scheduling problem. Management Science, 1984, 30(7), 854–867

6. Kremer, M., The O-Ring Theory of Economic Development, The Quarterly Journal of Economics, 1993, 108(3), 551-575

7. Rudolph, Adam J. and Elmaghraby, Salah E. The Optimal Resource Allocation in Stochastic Activity Networks via Continuous Time Markov Chains. International Conference on Industrial Engineering and Systems Management, 2009

8. Tereso, A., Araújo, M. and Elmaghraby, S., Project management: multiple resources allocation, 2008

9. Tereso, A. P., Araújo, M. and Moutinho, R. Duration Oriented Resource Allocation Strategy on Multiple Resources Projects under Stochastic Conditions. International Conference on Industrial Engineering and Systems Management. 2009a.

10. Tereso, A., Araújo, M., Moutinho, R., and Elmaghraby, S. Quantity Oriented Resource Allocation Strategy on Multiple Resources Projects under Stochastic Conditions. International Conference on Industrial Engineering and Systems Management, 2009b

11. Tereso, A. P., Araujo, M. M. and Elmaghraby, S. E., Adaptive resource allocation in multimodal activity networks, Int. J. Production Economics, 2004, 92, 1-10

12. Oliveira, J., Dias, L. and Pereira, G. Solving the Job Shop Problem with a random keys genetic algorithm with instance parameters. 2nd International Conference on Engineering Optimization, 2010, Lisbon – Portugal.

13. Oliveira, J., Scheduling the truckload operations in automatic warehouses, European Journal of Operational Research, 2007

14. Oliveira, J. A genetic algorithm with a quasi-local search for the job shop problem with recirculation. Springer, Berlin / Heidelberg: Applied Soft Computing Technologies: The Challenge of Complexity, 2006, 221-234, ISBN 978-3-540-31649-7, DOI 10.1007.

15. Goldberg, D. E., Genetic Algorithms in Search. Optimization and Machine Learning, 1989, Addi-son-Wesley, Reading, MA

16. Giffler, B and Thompson, G L, Algorithms for solving production scheduling problems, Operations Research, 1960, 8, 487-503

17. Cheng, R, Gen, M. and Tsujimura, Y. A tutorial survey of job-shop scheduling problems using genetic algorithms. Representation, Computers & Industrial Engineering, 1996, 983-997

18. Tereso, A. P. Adaptive Resource Allocation in Multimodal Activity Networks. PhD Thesis, 2002, (D. d. Sistemas, Éd.), Guimaraes, Portugal: Universidade do Minho