Optimizing Resource Allocation in a Portfolio of Projects Related to Technology Infusion Using Heuristic and Meta-Heuristic Methods

Maximiliano S. Zuloaga, Bryan R. Moser
System Design and Management, Massachusetts Institute of Technology, Cambridge, MA, USA

Abstract—This paper proposes a method to address the planning and scheduling required to infuse technologies into a portfolio of product development projects. Definitive selection of technologies for infusion cannot be applied without taking into account available resources, time required to mature technologies and the interactions among them. Portfolio selection and the scheduling process have often been treated separately although they are interdependent. This research aims to bridge the gap between portfolio scheduling and technology infusion by considering both with realistic performance dynamics, in which the iterative nature of activities is included in the model. Given these improvements, methods for effectively allocating resources in a portfolio of projects related to technology infusion are recommended. Initially, a heuristic method is proposed based on priority rules. However, as the assumptions of the model are loosened a novel method is suggested that combines Genetic Algorithm (GA) and Artificial Bee Colony (ABC) approaches. Numerical results indicate that the hybrid meta-heuristic method based on GA-ABC is effective in finding good resource allocations while considering rework. At the same time, results confirm that rework can dramatically affect the projects that comprise the portfolio and therefore rework should be included in these analyses.

I. INTRODUCTION

Most of the time companies do not provide value to customers by means of breakthrough improvements that result in entirely new products or systems. The most common manner to add value to customers is through continuous improvements, of small to medium impact. New technologies are often at the core of both new and improved products. Current technology infusion frameworks assess the impact of introducing new technologies into existing products and results in a selection of one or more options. However, the selection of technologies should also consider operational issues, including potential scheduling and required resource allocation for an effective infusion. At the same time, the inclusion of a particular project into the portfolio should be decided, not only in terms of value and profitability, but also considering schedule and operational information.

This gap is the motivation for this paper: to find the most effective resource allocation, activity scheduling and project portfolio selection by combining different heuristic and meta-heuristics. Initially, the resource allocation of the portfolio is proposed based on different priority rules. Next, the initial model is expanded and many assumptions are relaxed in order to attain a flexible model that could fit a variety of situations. The initial method is tested with loosened assumptions and compared with a meta-heuristic method that combines evolutionary algorithm and swarm intelligence.

Additionally, the ability of firms to manage multiple projects effectively is of utmost importance. Projects have become an increasingly common structure for organizing work and dealing with many projects at the same time is common in most manufacturing and service companies, particularly for high-tech firms. Single-project settings have become rare in business today. In a study mentioned by [1], a survey was performed in which one of the main results indicated that 84% of the companies emphasized that it is common having to deal with multiple projects. Other research indicate that managers usually deal with up to four projects at the same time [2]. Moreover, a research by [3] revealed that 90% of projects take place in a multi-project context. Therefore, a relatively small improvement in their management could result in enormous benefit for the company.

A question that arises is whether difficulties are consistent in every multi-project environment. Lazlo et.al. [4] determined that independently of the nature of the portfolio of projects or particularities of the companies, a common quandary develops in the proper allocation of resources among simultaneous projects.

This paper is structured as follows. The remaining of Section I provides a brief overview of the importance of technology infusion as an essential mean that tech firms rely on to add value to their products, and discusses the basic characteristics of the static resource-constrained multi-project scheduling problem (RCMPS). Section II presents a comprehensive literature review of the methods and tools that have been used in the past years to target the RCMPS. Section III provides my personal stance about the suitability of the methods and tools, mentioned in the previous section, as applied in solving the current problem of technology infusion. Section IV provides the explanation of the basic model that is proposed to solve the most basic RCMPS; whereas in section V, many assumptions of the basic RCMPS are relaxed and the possibility of rework is introduced into the model. In section VI, the results of the basic and improved methods are provided as well as a comparison between both. Finally, section VII and VIII corresponds to the conclusion and limitations of the models, respectively.

A. Technology Infusion

The ever-increasing nature of competition is pressing hard on most manufacturing companies to either improve previous
products or release new ones. However, most products are not designed and built from scratch, they are based on previous versions of the product where new technologies are infused in order to add, or improve, features that allow companies to attain a competitive advantage.

Technology improvements may range from minor improvements of existing components of a particular system, new subsystem advancements or disruptive innovations that drastically modify the architecture of the host system or product. The objective of the current paper is to focus on continuous improvements where the level of invasiveness (related to the amount of design change) is small or medium, which are the most common advancements that companies have to deal with.

Sometimes, companies gradually develop new technologies in their corporate research labs. After, a certain level of maturity is attained, the technology is candidate for infusion. Another approach, is to acquire technologies from universities, suppliers or by the acquisition of smaller companies. This last approach will require working closely with the developers of the new technology in order to assure that the readiness level necessary for making a decision on infusion is reached [5].

Therefore, one key aspect that must be considered is the readiness of the technology. One common model used to explain the lifecycle of a particular technology is the “S-Curve” model, which explains the lifecycle of a particular technology in terms of phases, such as early development, fast adoption and stagnation. Another method is Technology Readiness Levels (TRL), in which the technology is categorized into one of nine different stages of development. While these methods allow comparison of technology options, the measures are not sufficient to decide which particular one should be selected. There are three basic pieces of information that are missing [6]:

- Level of difficulty in transitioning a specific technology from a laboratory environment to operational use.
- Effect that the technology might have on the current attributes of the product and associated manufacturing cost.
- Capture the expected value impact over time that the product with the infused technology might provide to the company.

There are many proposed ways to measure different technologies. Usually, one metric of return reflects the potential value that each technology may bring to the firm, and one metric of risk that should estimate the difficulty of integrating the new technology into the parent system. As an example in Figure 1, technology 3 provides the maximum return compared with the others but happens to be the most difficult to implement. Technology 1, on the other hand, would present less difficulty yet it provides the lowest return. Technologies 4 and 5 are less appealing options, based on this analysis, because there is another technology that delivers higher return with less associated risk (technology 2 is better that 5 in both dimensions and technology 1 is a better option that 4).

![Figure 1 – Risk vs. Return Among Candidate Technologies for Insertion](image)

For the analysis mentioned above, it is possible to define both the metrics of risk and return, and then perform a qualitative analysis based on experience or intuition. However, doing such analysis qualitatively and expecting to be accurate might be a difficult task. In this respect, [7] and [6] developed a quantitative framework for assessing the impact of technology infusion into current as well as future products and it was applied to the analysis of a hydrogen-enhanced internal combustion engine. Furthermore, [8] responded to different critics and suggestions based on the previously mentioned framework, applying the new methodology to a complex printing system.

Once a few technologies are selected for their expected return and potential associated risk, it becomes necessary to plan for their infusion. These technologies should be assessed for the interaction amongst them, estimating the required resources and verifying if each technology has the appropriate maturity level. Some technologies may require more maturation. Lastly, it is necessary to schedule the activities that will lead to the effective infusion into the host system.

Moreover, all the mentioned activities should be performed within a reasonable, limited schedule. If the adoption of new technologies is modeled as a diffusion process shaped as an “S-curve”, before the current product or system enters the stagnation phase, the novel technologies should have been incorporated into the host system to create a new version of the product.

In other words, it is expected that if the level of sales from a product are shown in a graph; normally, in an initial stage, the amount of sales would gradually increase until reaching a plateau and end up with a decrease in sales during a final stage. Ideally, the new version of the product should be launched before the sales of the old version enters the final phase. If a product is launched too early, it could negatively affect the level of sales of the previous version; whereas, launching too late could potentially cause competitive products to seize a greater market share.

B. RCMPS Characteristics

The resource-constraint project scheduling problem (RCPSP) is the problem of finding a way to allocate limited resources in order to complete a particular project while optimizing some predefined objective function.

The basic RCPSP deals with only one project, but there are many generalizations that can be performed in order to make a
Optimize: \[ f(F_{ij},...,F_{Nj};T_i,...,T_j) \quad \forall i \in N_i, \ j \in J \] (1)
Subject to: \[ F_{ij} - d_{ij} \geq F_{ij} \quad \forall i \in N_i, \ j \in J, \ i \in P_{ij} \] (2)
\[ \sum_{j \in O_i} P_{ijk} \leq R_k \quad \forall i, j \in O_i, \ k \in K, \ t \geq 0 \] (3)
\[ F_{ij} \geq 0 \quad \forall i \in N_i, \ j \in J \] (4)

1. The objective is to optimize a pre-specified performance measure.
2. Activities cannot start until all the preceding tasks are finished.
3. The amount of resource k utilized at every time period should always be less than or equal to the total availability \( R_k \).
4. The starting time of each activity should be positive.

More comprehensive problem. The multi-mode resource-constrained multi-project scheduling problem (MRCMSP) is a generalization of the more basic RCPSP in two dimensions: multiple projects have to be scheduled simultaneously and tasks can be executed in multiple modes (MRCPSP) [9].

Much research on RCPSP has been conducted over the last fifty years. Nevertheless, the literature available on the multi-project problem has only increased during the last two decades. When more than one project is considered, there are two general approaches:

- Combining all different projects into one “big” project by the addition of dummy activities at the beginning and at the end.
- Treating all projects separately, as a multi-project where each is independent from others.

The MRCPSP (resource constraint multi-project scheduling problem) where the objective is to maximize the performance, represented by some objective function, of a set of projects that take resources from a limited source. Each time concurrent activities of one or more projects of the portfolio exceed the amount of available resources, then a decision should be made about what activities to perform and which should be delayed as well as define how to best allocate the resources at hand. While aggregating many projects into one single “big” project has a considerable amount of research devoted to solve it, there are certain disadvantages to consider. To begin with, it implicitly considers that the returns, or penalties, associated with each project are the same; whereas in the MRCPSP it is possible to account for different utility functions and target dates associated with each project, which is more realistic. Second, there are cases in which different projects might be assigned to several project managers who are only interested in the success of those projects under their responsibility. Finally, considering projects separately allows for project termination if one in particular is restricting the overall utility of the portfolio.

Optimization methods impractical for large scale MRCPSP because they are NP-hard [10]. There are two basic ways of facing this issue:

- Using heuristic and meta-heuristic methods.
- Using exact methods but focusing on small scale problems or addressing a particular situation that allows simplification of the model.

The basic RCPSP can be characterized by a set of \( j=2,3,\ldots,J \) projects where each project contains \( i=1,2,\ldots,N \) activities. Task duration \( d_{ij} \) is deterministic. Tasks might be sequential or independent between each other. All projects and activities within the portfolio are known before scheduling them. The different projects consume resources from a common source, which means that resources might be shared among projects of the same portfolio. Each activity \( i \) requires exactly \( r_{ik} \) units of resource \( k \) during the whole period that the task is being done. Resources are always renewable so that the pool of available resources is \( R_k \) at every time period. \( P_i \) is the set of predecessors of task \( i \) in project \( j \). Let \( O_i \) be the set of on-going activities at time \( t \). The objective is to optimize a function of the finish dates \( (F_{ij},...,F_{Nj}) \) and target dates \( (T_1,...,T_J) \). The MRCPSP consists of the following points above.

II. LITERATURE REVIEW

As was addressed in the previous section, this paper will focus on the RCPSP, taking a multi-project approach by treating different projects of the same portfolio separately. Two general approaches that may be applied are exact methods and heuristics.

A. Exact Methods

Since it was demonstrated that the RSPSP were a generalization of the job-shop scheduling problem and therefore is non-deterministic polynomial-time hard (NP-hard) [11], most of the efforts shifted to the development of heuristic procedures, which allow attainment of “good” solution rather
than optimal ones. As mentioned before, RCPSP was proven to be NP-hard and because RCMPSP is a generalization of the more basic RCPSP, it is NP-hard as well. The reason is that exact algorithms lead to impractical execution time when the number of activities increase. Hence, optimal approaches are generally used for generating benchmark solutions, dealing with simplified problems or combined with heuristic methods.

The following studies are examples of methods that combine some form of optimization with heuristics. A solution methodology that combines deterministic dynamic programming, Lagrangian relaxation and heuristics was developed by [12] in order to design project schedules with special attention to varied task dependencies and communication activities. The authors state that even though the method was applied to a single project, it could be extended to manage multiple projects that share a common pool of resources.

A stochastic dynamic programming approach combined with Lagrangian relaxation and heuristics was used in [13] to study the scheduling of design projects with uncertain number of iterations, seeking to minimize weighted tardiness, earliness and risk penalties. Although the method was applied to handle uncertain number of iterations, it could be extended to include other kind of uncertainties.

Some research performed in project scheduling assumes information to be invariable even though uncertainties during project execution exist. Wang et al. [14] presented a method for the stochastic RCMPSP based on a Markov decision process (MDP) with an analysis of events that might arise during the execution of projects. The study includes an analysis of their proposed method against certain priority rules such as FCFS, EDD, COVERT, WSPT and ATC. Finally, the authors mentioned that a possible expansion of their work is the potential application to problems that contain multiple project networks where tasks might be executed in multiple modes.

B. Heuristic Methods

Heuristic procedures, on the other hand, are focused on finding a “good” solution rather than an optimal one. These methods could be sub-divided into three groups: priority rule (PR) based heuristics, other heuristics and meta-heuristics.

1) Priority Rule Based

The most common methods to solve RCMPSP are priority rule based; because they are intuitive, easy to implement, usually easy to understand and fast in term of computational effort [15]. A multitude of priority rules have been tested so far. They can be classified by the information they require in: (1) project-based, (2) activity-based, (3) resource-based and (4) composite [16]. Priority rule based heuristics are made up of two components: A schedule generation scheme (SGS) and a priority rule. The generation scheme creates a decision set for the tasks that remain to be scheduled. Next, a priority rule is used to choose from the decision set. Finally, a tie-breaker (TB) might be used if ties appear in the previous step.

In research by [17] many different PRs are compared. Following a similar line, [18] compared the performance of 20 different priority rules, using a parallel generation scheme (P-

SRS) based on the characteristics of the portfolio such as complexity, resource contention and resource distribution. The authors performed a full factorial experiment, using a generator by [19], measuring and analyzing the performance of the diverse PR standardizing tie-breakers (TB) to increase comparability. Two objective functions were considered, average percent delay and maximum percent delay, and the experiment was performed for both. The results for each objective function were summarized in tables where the practical use of them require managers to do a qualitative characterization of the portfolio in terms of complexity, resource contention and resource distribution.

A research has shown that the most suitable priority rules for iterative project portfolios differ significantly from those portfolios in which projects are acyclical [20]. In this study, the performance of the 20 different priority rules used for acyclical projects are compared in the case of iterative-activity projects and 11 new rules were added that account for some characteristics of iterative projects. The results were also summarized in two decision tables.

Reference [21] indicates that there are endless instances for the RCMPSP, but no single PR is best suited for every instance. Therefore, a learning process is proposed which will determine the most appropriate PR with the most fitting tie-breaker (TR).

Reference [22] studied the RCMPSP considering transfer time of resources. In every other research mentioned in this literature review, the authors assumed that resources could be transferred from one project to another without any expenses in time or cost. However, [22] included sequence-dependent and resource-dependent transfer times. This kind of multi-project problem with transfer times is identified as RCMPSPPTT. In this study, the objective function is to minimize the total average delay of the portfolio and the resource allocation is performed applying different priority rules.

2) Other Heuristics

Other heuristic methods include those that can neither be classified as PR based heuristics nor as meta-heuristics. This category, includes forward-backward improvement (FBI), sampling methods and others.

A multi-criteria heuristic method was developed by [1] that aims to improve resource allocation in multi-project scheduling. The method is divided into two phases. The first phase seeks to minimize a time-related variable such as average portfolio delay or maximum portfolio delay by iterative forward-backward passes. The second phase seeks to minimize a non-time related variable like project splitting, in-process inventory or idle resources; or maximize another non-time related variable - resource levelling. After both phases are finished, the final multi-project feasible schedule is presented. The authors showed that the proposed method improves the feasibility of multi-project schedule obtained from heuristic methods based on priority rules such as MINLFT and MAXTWK.

A hybrid heuristic that combines multi-pass random sampling and backward-forward improvement method was formulated in [23] directly addressing the RCMPSP. The best
possible configuration of the parameters was proposed. This configuration was obtained through a computational study, which targeted minimizing average project portfolio delay as well as maximum project duration.

3) Classical Meta-Heuristics

Many studies solve the RCPSP by applying a classical meta-heuristic approach. These approaches follow well-known paradigms that have been applied to solve many different problems in various fields, such as: genetic algorithms, simulated annealing and tabu search among others. Some of these methods are briefly described below with research related to the RCPSP.

A genetic algorithm (GA) is a problem solving technique based on the evolutionary ideas of natural selection which have been successfully applied to a number of project scheduling problems and extended to the multi-project case as well. GA belongs to a larger class of evolutionary algorithms (EA). Solution information is codified in a string called a chromosome. The algorithm tries to improve the chromosome’s potential “fitness function” by some operators such as mutation, crossover and selection. The more fit an individual is the more probable it is to be selected for the new generation.

A special case of the RCPSP was studied in [24]. This research considered a multi-project situation where activities have different usage modes. A particular budget is assigned for the portfolio and instead of a resource sharing (RS) policy for the diverse projects of the portfolio from a pool of resources (that is more common in this type of problems), it considers a resource dedication (RD) policy in which each project can only spend its assigned resources without the possibility of accessing resources from other projects of the portfolio. Considering the RD policy, the multi-project scheduling problem reduces to a multi-mode resource constrained project scheduling problem (MRCPSp) for every particular project in the portfolio. In order to solve this problem, the authors propose two approaches: a two-phase and a monolithic genetic algorithm.

A paper from [25] presents a GA suitable for the RCPSP where the scheduling is based on three things: priorities of the activities, delay times and release dates. The algorithm was tested on a set of 10, 20, 30, 40 and 50 projects composed of 1200, 2400, 3600, 4800 and 6000 tasks, respectively.

Simulated annealing algorithm (SA) is a stochastic method for combinatorial optimization problem. This algorithm tries to minimize the thermal energy of the system by cooling down a temperature parameter. When the thermal energy of the system is minimized, the solution is in a stable state and consequently is a good solution. Any particular solution that has a lower thermal energy, will always be accepted. However, if the thermal energy is higher (worse than the current solution), there is a probability of accepting the new solution and that probability decreases over time. SA uses the mentioned mechanism to avoid being trapped on a local optimum.

In a research performed by [26], a hybrid GA and SA is proposed for tackling the multi-project resource constraint scheduling problem. The GA, as it is a population based algorithm, provides a comprehensive exploration of the search space by recombining solutions to obtain new ones, whilst the SA algorithm focuses on the localized examination. The hybrid GA-SA algorithm was compared with other meta-heuristic methods like SA, GA and modified simulated annealing (MSA). The results were that the hybrid algorithm performed better in most cases, especially when the complexity of the multi-project scheduling increased.

A backward-forward hybrid GA with SA was applied to the RCPSP by [27]. This research tested with a set of 26 different portfolios formed by some projects contained in the PSPLIB. The effectiveness of the hybrid algorithm was verified by comparing the result with three popular priority rules and the outcome attained using Microsoft Project.

Another meta-heuristic that has been applied for solving scheduling problems belong to the so-called “swarm intelligence” group, which are nature-inspired meta-heuristics. Three common methods that have been used to solve the RCPSP as well as the RCPSP are particle swarm optimization (PSO), ant colony optimization (ACO) and artificial bee colony (ABC). In PSO, a population of possible solutions are treated as “particles” that move through the solution space, and are evaluated according to some fitness criterion in each time period. Over time, particles are accelerated towards those which have better fitness values. Ant colony optimization (ACO) emulates how ants direct each other to resources while exploring their environment by laying down pheromones. Those pheromones correlate with the probability of ants revisiting certain place and linger some time depending on the fitness value of the food source. Finally, ABC is an optimization algorithm, initially proposed by [28] to solve multidimensional optimization problems, inspired by royal honey-bee foraging behavior.

An example of the usage of ACO for solving the MRCPSP was proposed by [29], utilizing two types of pheromones regarding the solution in terms of sequence and mode selection of activities. The method was tested using projects from the project scheduling problem library (PSPLIB) and compared with other common meta-heuristics such as: GA, SA and PSO. The results for the ACO were usually better than those obtained with the other methods.

Chen and Ju [30] performed a comparative analysis of swarm intelligence and heuristic priority rules for solving a multi-project problem, specifically the MRCPSP where activities have multiple execution scenarios. The authors adopted the single-project approach instead of the multiple-project approach. Specifically, the authors used for swarm meta-heuristics; artificial bee colony (ABC) algorithm, ACO and PSO. For priority rules, they used both parallel generation scheduling (P-SGS) and serial generation scheduling (S-SGS). In addition, they used different priority rules for each scheme. The results of the simulation showed that swarm intelligence is superior to priority rule based heuristic for large scale problems, but present almost the same performance for small scale ones.

Differential evolution algorithms (DE) are a powerful technique that combine simple arithmetic operators with the classical crossover, mutation and acceptance operators. The
basic scheme in DE is generating trial parameter vectors. Mutation and crossover are used to generate new vectors (trial vectors). Selection then determines which of the vectors will survive the next generation. Research has been conducted using DE to solve project scheduling problems. Damak et al. (2009) solved the MRCPSP with a DE algorithm. In their approach a solution is represented by a mode assignment vector and a position vector. Neighbor solutions are generated using two mutation and crossover operators. A selection operator uses the an objective function that is penalized for infeasible solutions. The performance of this algorithm was evaluated against benchmark instances. The results were compared with the results obtained by two other approaches: simulated annealing and particle swarm optimization.

III. THIS RESEARCH

A. Purpose

Three main questions should be addressed before deciding on infusing certain technology into a system [5]: (1) Will the technology be ready for infusion? (2) Will the technology create additional value for the firm, customers and other stakeholders? (3) How will the technology be effectively transferred into the product?

The available frameworks related to technology infusion, only partially respond to the second and third interrogations. To answer the first as well as part of the third question requires accounting for the impact of each project on the existing portfolio, not only in terms of value and profitability, but also considering schedule and operational information. The current approach acts as a necessary complement of previous frameworks by contributing to fill the gap between the literature in portfolio project management; usually focused on project selection based on financial information, with multi-project management; mainly concerned with operational matters, activity scheduling and resource allocation.

B. Base Model

In order to solve the RCMPSP, PRs remain a popular method mainly for their speed and simplicity [21] and because of the following reasons stated in [20]:

- PRs have a low computational effort that makes them suitable for large scale problems common in RCMPSP.
- PRs are utilized together with other heuristic or exact methods.
- PRs are used as initial solutions for other, more complex, heuristic methods.
- PRs are common in project scheduling software.

In their research, [30] showed that diverse meta-heuristic models applying swarm intelligence tend to yield similar results as PR-based methods for small scale problems.
However, as the scale increases (either by adding more projects to the portfolio or having bigger projects) then swarm methods become more fitting. The main reason appears to be that due to its hybrid searching mechanism which combines local and global searching operations, swarm intelligence gets closer to the optimal solution. Therefore, for the basic model, we will use priority rules, but for the improved model which contains relaxed assumptions, and therefore as the space of possible solutions becomes bigger, we will provide a meta-heuristic algorithm that could potentially feed from PR-based solutions as initial input.

A study performed by [21] proposed a learning process to determine the most appropriate PR plus tie-breaker (TR) pair for each instance in a multi-project problem. The result obtained could be used directly as a solution or incorporated into another heuristic or exact method as an entrance to a later optimization process.

The method we propose in order to tackle the basic RCMPSP will be based on the work of [21], but instead of using a serial SGS, we will use a parallel one because a study carried out by [31] showed the superior performance of parallel SGS in a multi-project context. We believe this is a good fit because the results could be either used directly or as starting solutions for the posterior method that we will propose to target the situation where assumption are relaxed. However, the objective functions that are used in [21] correspond to minimizing the average percent delay and overall completion time; whereas, we intend to use a utility function that relates the overall return of each project depending on the actual finish date and a target date. Depending on the characteristics of the market, product involved, technologies to infuse and nature of competitors, there might be projects in which there is no benefit to finishing early, while for other projects there might be a premium for finishing early. We have not seen such kind of utility function applied in any of the works researched.

C. Improved Model

Previous research has already considered that a variation in resource allocation can alter its processing time. One way to model this is by using multi-modes per activity, where one task may have different resources allocated, associated cost and duration, depending on the mode. This allows diverse but discrete time-resource, time-cost and resource-resource trade-offs. This problem has been addressed in [30] or [24] among others.

In Figure 3, a decision tree is displayed that represents sequential and connected decisions. Each node represents decisions that can be controlled by the decision maker who has a finite number of possible assignments represented by the branches. Finally, the endpoints represent a complete assignment of all decisions. This diagram is a summary of the methods mentioned in the literature review and those that were selected for the base as well as the improved model. Therefore, a heuristic multi-pass method is proposed based on priority rules for the base model; whereas, the improved model might take the priority rule approach just as an initial solution but the proposed approach is a hybrid algorithm combining GA and swarm intelligence.

Finally, the possibility of rework will be taken into account in the improved model. Rework will turn the project into a cyclical one. In the literature review there are some works that consider iterative activities in the case of RCPSP [32], [33], [13] as well as the RCMPS [20], even though the literature is mostly devoid of studies that consider the multi-project problem with iterative tasks. New PRs will be considered based on the research made by [20] where they studied 11 new priority rules that account for some basic characteristics of iterative projects. Finally, system dynamic models show how work intensity and overtime affect the amount of rework [34]. However, that association is missing in project scheduling models. We will relate amount of the rework with scheduling models. We will relate amount of the rework with scheduling models.

IV. BASE MODEL: PRIORITY RULE-BASED MULTI-PROJECT ALGORITHM

Heuristic algorithms usually take one activity at a time, treating precedence among tasks as well as resource constraints. In order to schedule the activities, some criteria or priority rule are required, based on some metric, so that the order of activities is established.

A. General

There are two main schedule generation schemes (SGS) generally used to build feasible schedules: serial SGS which performs activity increments and parallel SGS that uses time increments. The first group corresponds to algorithms that sort all activities according to some priority rule and schedule tasks

---

![Algorithm Explanation](image-url)
to begin as soon as precedence restrictions and resource constraints allow. On the other hand, parallel SGS analyze period by period if resources are enough to schedule all permitted activities (those that have no precedence restrictions). If resources are not sufficient, then activities are chosen following certain priority rule and tasks that are left to program in the future are evaluated in another period. At the same time, scheduling with both SGS can be done forward or backward.

The current model is an algorithm based on time with forward scheduling where time is broken down into unitary time steps. As time progresses, tasks evolve being included in one of four different groups (R, S, O and C). All the activities of the diverse projects of the portfolio start in the R (remaining activity) group. When precedence relations allow activities go from R to S (selected activity) group. The afore-mentioned group is composed of all still un-started activities that could potentially be selected if there were infinite resources. However, as resources are limited it is necessary to choose from S activities based on some priority rule. Therefore, activities that belong to S are ordered based on PR and they are sequentially selected until resources are not enough to perform any more tasks. That way, for those activities where resources are sufficient, they are moved to O (on-going activity) group. There is an increment in time equal to the tasks with the minimum duration activity in the O group. Then, all tasks that were completed are moved to C (complete activity) group. Finally, as some activities were finished, it is possible that some tasks from group R might fulfill every precedence constraint and change to S group. This process is performed until all activities are part of the C group. Figure 4 illustrates the process that was mentioned, while the red rhomboid indicates the moment in which PRs could be selected for ordering tasks when resources do not suffice.

Previous research in technology infusion has used a probabilistic NPV analysis to assess the economic return due to the new technology [7] and [13]. A probabilistic simulation could be performed using Monte Carlo simulation (for example) in order to estimate the NPV values that might result as a consequence of infusing each technology. The diverse results of the simulation account for the uncertainties of the performance that the new technology may carry as well as the response of the market to it. Figure 5 illustrates the process that was mentioned, while the red rhomboid indicates the moment in which PRs could be selected for ordering tasks when resources do not suffice.

In Figure 5, basic utility functions are shown and explained. This utility functions reflect the expected return depending on the time. Utility functions do not have to follow the same pattern as the three cases shown below, these are simple examples that were included in the model but if another function is chosen the model could work all the same. The red square shown in the figures correspond to an infeasible finishing date. The line where the red figure ends marks the resource unconstrained finish date that cannot be improved if resources are limited. Therefore, with resource constraints the actual finishing date should be equal to or after the red figure ends.

1. The ultimate utility will be E[ΔNPV] if the project is finished at the target date or before it. As time goes beyond the target date, the utility will decrease until eventually it will equal the NPV if the current technology is maintained. At that moment, when the utility curve crosses the x axis, it will be the same weather the new technology is infused or not.
2. The utility is equal to E[ΔNPV] if the project is finished at the target date and is sequentially lower as the finish date goes beyond the target date. However, this function is different from the previous one in that there is a premium for finishing early. The earlier the project finishes the higher the utility.
3. The utility is again equal to E[ΔNPV] if the project finishes at the target date or during certain range that includes that date. Now, if the finishing date goes above or beyond the afore-mentioned range, then the utility is lower.

At a high level, it is possible to say that curve type 1 penalizes tardiness while earliness is not rewarded. Curve type 2 castigates tardiness but earliness is compensated. Finally, curve type 3 penalized tardiness as well as earliness.

B. Assumptions

1) General
   - Tasks are of fixed duration
   - Expected durations are known or can be estimated
   - Resources necessary for a given task are known and fixed
   - Maximum amount of resource types is known and fixed
• E(ΔNPV) for a target date are known and target dates are given
• Activities are done once (there is no rework)
• Project ends when all tasks are completed
• Portfolio of projects ends when all projects are finished
• Once an activity has started, resources assigned to that task cannot be reallocated until the activity is finished
• Activity preemption is not allowed
• Project preemption is not permitted

2) Control Variables
• Best Priority Rule for the portfolio
• Activity scheduling

C. Priority Rules

| Parameters: |
|-------------|
| i = activity index 1≤i≤N |
| j = project index 1≤j≤J |
| ESij = early start of activity i and project j |
| LSij = late start of activity i and project j |
| EFij = early finish of activity i and project j |
| LFij = late finish of activity i and project j |
| di = duration of activity i and project j |
| CPj = critical path duration of project j without resource constraints |
| ASj = set of activities already scheduled in project j |
| TSij = number of activities that follow activity i in project j |
| CSlj = number of activities that follow activity i in project j and belong to the critical path considering the case of unconstrained resources |
| ACTIMij = corresponds to the path of maximum duration starting from activity i of project j until the end of the project |

D. Tie-Breakers

When priority rules assign the same value to different activities, it is necessary to break the tie. TBs are priority rules used in a second step, once the actual PR has been applied. The list of TB is shorter than the list of PR. It is possible that more than one TB should be needed in order break all ties. The initial TB is mentioned with each PR in the previous list.

List of Tie-Breakers (TB)

| RAN: random |
|-------------|
| FCFS: first come first served |
| GRES: greatest resource requirement |
| SOF: shortest operation first |
| GRD: greatest resource demand |

In the morphological matrix shown in Table 1, the different decisions are listed with its associated alternatives. The alternatives chosen for each decision are indicated in red font and summarize the previous analysis of the current section. The parallel SGS was chosen for scheduling with forward direction. It was selected a multi-project approach, where the proposed method to solve the scheduling problem was a heuristic based on priority rules (PR) and tie-breakers (TB).

| Table 1 – Decisions for the base model (morphological matrix) |
|-------------|
| Decision | Alternative 1 | Alternative 2 | Alternative 3 |
| SGS for Scheduling | Serial | Parallel | Both |
| Scheduling Direction | Forward | Backward | Both |
| Portfolio Approach | Single “Big” Project | Multiple Projects | - |
| Method Applied | Exact | Heuristic | Combined |
| Heuristic Applied | Priority Rule Based | Meta-Heuristics | Other Heuristics |
### List of Priority Rules

| Rule       | Formula                                                                 |
|------------|-------------------------------------------------------------------------|
| WACRU:     | $\text{Max}(w \sum_{q=1}^{N_i}(1 + SLK ij)^{-\alpha} + (1 - w) \sum_{k=1}^{K} \frac{r_{ij}}{R_{Max,k}})$ |
| FCFS:      | $\text{Min}(ES ij)$                                                    |
| SOF:       | $\text{Min}(d ij)$                                                    |
| MOF:       | $\text{Max}(d ij)$                                                    |
| MINSLK:    | $\text{Max}(SLK ij)$ with $SLK ij = LS ij - \text{Max}(ES ij, t)$     |
| MAXSLK:    | $\text{Min}(f ij)$ with $f ij = CPI + d ij$                           |
| SASP:      | $\text{Max}(f ij)$ with $f ij = CPI + d ij$                           |
| LALP:      | $\text{Max}(f ij)$ with $f ij = CPI + d ij$                           |
| MINTWK:    | $\text{Min}\left(\sum_{k=1}^{K} \sum_{i \in AS} (d ij r_{ijk}) + d ij \sum_{k=1}^{K} r_{ijk}\right)$ |
| MAXTWK:    | $\text{Max}\left(\sum_{k=1}^{K} \sum_{i \in AS} (d ij r_{ijk}) + d ij \sum_{k=1}^{K} r_{ijk}\right)$ |
| GRES:      | $\text{Max}(\sum_{k=1}^{K} r_{ijk})$                                  |
| LRES:      | $\text{Min}(\sum_{k=1}^{K} r_{ijk})$                                  |
| SST:       | $ES ij + LS ij$                                                        |
| EDDF:      | $\text{Min}(LS ij)$                                                   |
| LCFS:      | $\text{Max}(ES ij)$                                                   |
| MAXSP:     | $\text{Max}(\frac{t - LF ij}{d ij})$                                  |
| MINLFT:    | $\text{Min}(LF ij)$                                                   |

### V. IMPROVED MODEL: HYBRID META-HEURISTIC ALGORITHM BASED ON GA AND ABC

#### A. Assumptions

Below are listed the assumptions for the improved model, where those that are in bold correspond to the ones that differ from the base model.

- Task duration depends on the resource allocation
- Resources assigned to a particular task might vary
- Maximum amount of resource types is known and fixed
- $E(\Delta NPV)$ for a target date are known and target dates are given
- Activities might be done more than once (rework)
- Project ends when all tasks are completed
- Portfolio of projects ends when all projects are finished
- Resources might be reassigned once an activity has started
- Activity preemption is not allowed
- A particular project might be eliminated of the portfolio

#### B. Project Cancellation Is Allowed

There are situations in which the firm does not possess the necessary resources to manage the entire portfolio of projects. In such cases, one solution could be to focus available resources in those projects that will yield important benefit or the ones that will provide a strategic advantage, postponing the others. However, the associated outcome of a particular project varies though time. This effect originates because competitors might react, customer expectations might change or new technologies may arrive. Given penalties from a late time to market, continuing with the current product ends up being better than infusing a technology in an unsuitable moment. Cancellation of projects would leave resources available for the remaining ones. Therefore, it is important to incorporate, in the model, the possibility of calling off projects if that yields a greater overall benefit for the company.

To incorporate project cancellations into the model, it is necessary to compare the results attained when each project is abandoned, leaving the combination that yields the best result.
The base model starts with all projects within the portfolio. However, cancelling one or more projects could potentially generate a better result; then, the combinations are analyzed and the best portfolio will remain regardless of the initial number of projects.

Another consideration that should be taken into account is the interactions among the different projects of the portfolio. Each project might correspond to the infusion of a particular technology into a product. It is also possible that one technology is being infused into many products. Finally, it could be that many technologies are infused into many products. In any case, various types of interactions could be possible among projects. Therefore, if two projects A and B are considered, the interactions could be as follows:

- Independent: project A is independent of project B.
- Enabling: project A must be done in order to do project B.
- Inclusive: project A must be done in order to do project B and vice-versa.
- Exclusive: if project A is done, then project B cannot be executed, and vice-versa.

These interactions can be included into a matrix. Those interactions could be coded in many different ways. With this information, the combinations of projects could be reduced eliminating those that are infeasible. If $a_{ij}$ is the element of a matrix in row $i$ and column $j$, then:

- If $a_{ij} = a_{ji} = 1$, it corresponds to an independent interaction.
- If $a_{ij} = a_{ji} = 0$, it relates to inclusive interaction.
- If $a_{ij} = a_{ji} = -1$, it corresponds to exclusive interaction.
- If $a_{ij} = 1$ & $a_{ji} = 0$, it relates to an enabling interaction.

C. Multi-Mode

The multi-mode resource-constrained multi-project scheduling problem (MRCMPSP), similar to the previous RCMSP that have been covered so far, also consists of a portfolio of $i$ projects ($i=1, \ldots, N$), where each one contains a set of $j$ activities ($j=1, \ldots, J$) that have to be scheduled under precedence and resource constraints. Nonetheless, when multiple execution modes are considered, each activity $j$ ($j=1, \ldots, J$) can be executed in one of diverse $M_{ij}$ modes, which represents a combination between its resource requirements and its duration. When multiple modes are considered, it is possible to include non-renewable resources apart from the renewable ones. The resource constraints mean that the available amount $R_k$ for every renewable resource $k$ ($k=1, \ldots, K$) is limited per period of time for every resource and the amount available of each nonrenewable $W_k$ resource $w$ ($w=1, \ldots, W$) is limited for the entire project duration. Each activity $j$ ($j=1, \ldots, N$) executed in mode $m$ ($m=1, \ldots, M_j$) has duration $d_{jnm}$ and requires $r_{jnmk}$ units of renewable resource $k$ ($k=1, \ldots, K$) and $W_{jinwk}$ units of nonrenewable resource $w$ ($w=1, \ldots, W$). The goal of solving the MRCMPSP is to find sequence and mode selection for each activity as well as the resultant schedule, including start times and resource allocation policies that leads to a maximization of the return of the project portfolio.

Adding multiple modes to the previous scheduling problem, allow to incorporate many tradeoffs such as:

- Duration-Resource: changing the amount of resource affect the duration of the activities.
- Resource-Resource: modifying one type of resource might affect the required amount of another class of resource.

Now that the multi-mode is studied, it is possible to define different types of resources that where not possible to consider before. In this paper, the following definitions for each kind of resource are considered:

- Renewable resources
- Non-renewable resources
- Doubly constraint resources

Before the hybrid meta-heuristic method is started, [35] proposed a pre-processing procedure to reduce the search space whenever there are activities with multiple modes involved as well as non-renewable resources. This pre-procedure consists on eliminating inefficient modes, non-executable modes and redundant non-renewable resources. It should be considered that deleting with a non-executable mode might cause that another mode becomes non-executable.

D. Representation

In order to apply a meta-heuristic method (such as GA, TS or PSO) to the scheduling problem, first it is necessary to start defining a representation for the solution space. In [36] are mentioned five different representations. The first two representations that are listed are the most used:

- Activity List (AL)
- Random Key (RK)
- Priority Rule (PR)
- Shift Vector (SV)
- Schedule Scheme (SS)

In addition, in order to convert the representation into a schedule, a decoding procedure is required, which is related with the selected representation. Finally, operators should be determined to produce new solutions based on previous ones. Two groups of operators can be established:

- Unary Operators: create a new solution based on an existing one. They are usually used for local search procedures. In GA it is used for mutation.
- Binary Operators: define a new solution based on two different solutions already in existence. They are used in GA for crossover.
the use of the meta-heuristic, into a discrete value vector. This is in order to convert a continuous value vector obtained through type of representation and decoding procedure should be used for any project scheduling problem, the search space is discrete. Therefore, a multi-dimensional search space. However, for any project scheduling problem, it is possible that a new vector turns out to be invalid because it is violating precedence constraints. Furthermore, it is conceivable that a set of mode selections end up being infeasible because the non-renewable resource constraint might be infringed. Whenever a situation like the one mentioned takes place, there are five paths to follow:

- Keep the infeasible solution: it will affect negatively the objective function making less probable that the infeasible solution survives.
- Iterate for a predetermined number of cycles: iterate for a limited number of cycles trying to turn the solution into a feasible one, the final result is kept whatever the outcome.
- Iterate until the solution becomes feasible: iterate until it becomes feasible or until all options were explored.
- Discard infeasible solution: directly the solution is eliminated.
- Guide the search: so that no infeasible solution is chosen in the first place.

E. Validation Generator

Many meta-heuristic methods are suitable for a continuous multi-dimensional search space. However, for any project scheduling problem, the search space is discrete. Therefore, a type of representation and decoding procedure should be used in order to convert a continuous value vector obtained thought the use of the meta-heuristic, into a discrete value vector. This procedure was covered in the previous section. Nevertheless, since the MRCMPSP is a precedence constraint optimization problem, it is possible that a new vector turns out to be invalid because it is violating precedence constraints. Furthermore, it is conceivable that a set of mode selections end up being infeasible because the non-renewable resource constraint might be infringed. Whenever a situation like the one mentioned takes place, there are five paths to follow:

- Keep the infeasible solution: it will affect negatively the objective function making less probable that the infeasible solution survives.
- Iterate for a predetermined number of cycles: iterate for a limited number of cycles trying to turn the solution into a feasible one, the final result is kept whatever the outcome.
- Iterate until the solution becomes feasible: iterate until it becomes feasible or until all options were explored.
- Discard infeasible solution: directly the solution is eliminated.
- Guide the search: so that no infeasible solution is chosen in the first place.

F. Schedule Generation Scheme

The core difference between serial and parallel SGS is that the former is activity-incremented; whereas the latter proceeds in time increments.

The serial SGS generates feasible and, as [16] showed, active schedules. These are schedules where none of the activities may start earlier without delaying some other activity. The optimal solution will always be part of the active schedules.

The parallel SGS always generates feasible schedules, as the serial SGS does. However, it has been shown by [16] that the parallel SGS constructs non-delay schedules, which is a schedule in which no resource is kept idle while an activity is waiting to be executed. Non-delay schedules are included in the group of active schedules. Therefore, the parallel SGS searches in a smaller solution space that the serial SGS. Notwithstanding, there is an important shortcoming with parallel SGS because [16] exposed that often non-delay schedules do not contain optimal ones, while optimal schedules are always in the active set. Thus, if the parallel SGS is applied, it is necessary to perform some procedure, such as adding delay between activities, to explore the solution space that belongs to the active set but is outside the non-delay schedules. To reduce the solution space, parameterized active schedules, introduced by [25] are used. The idea behind parameterized active schedules is to increase or reduce the search space by controlling the maximum delay allowed.

In the morphological matrix displayed in Table 2, the different decisions are listed with associated alternatives. All this information was covered in previous paragraphs, but in this table the information is summarized and the alternatives chosen for each decision are indicated in red font.

G. Genetic Algorithm

A genetic algorithm is a meta-heuristic optimization technique developed by Holland, based on natural evolution and the survival of the fittest. It is considered a population-based method, because it improves a set of solutions from one generation to the next. There are three common genetic operators: selection, crossover and mutation. Many variation of GAs can be achieved by varying the mentioned three operators.

Selection: this operator is responsible for finding those individuals that are more “fitted for survival”. In nature, it is usually seen that individuals from a particular species compete for scarce resources and survival. Ultimately, those individuals that prove to be more fitted are the ones that prevail over the less fitted ones. A fitter individual has more chances of producing higher number of offspring and consequently the genetic material is carried on through multiple generations. There are several selection operators, two of the most common are: ranking-based and proportional.

In this paper, selection operator is implemented, initially, by ranking the individuals of a generation according to their fitness value. The most fitted at the top, while the less fitted are located at the bottom. Then, the entire generation is divided into two groups, “the good” representing the top 40% and “the fair” that correspond to the 60% of the remaining individuals. Within the former group, a small percentage at the “top” that account for the best individuals are copied from the current generation into the following one. This strategy is known as

| Decision                  | Alternative 1 | Alternative 2       | Alternative 3 | Alternative 4 |
|---------------------------|---------------|----------------------|---------------|---------------|
| Representation            | Activity List | Random Key           | Priority Rule | -             |
| SGS for Scheduling        | Serial        | Parallel             | Both          | -             |
| Unary Operator            | Swap Moves    | Random Changes       | None          | -             |
| Binary Operator           | One-Point Crossover | Two-Point Crossover | Uniform Crossover | Shift        |

TABLE 2 – CONFIGURATION CHOICES FOR GA (MORPHOLOGICAL MATRIX)
“elitist” and guarantees that the genetic material of the most fitted individuals survives through generations. In Figure 6 it is displayed the process between two generations.

![Selection Operator](image)

**Figure 6 – Selection Operator**

The remaining entities of the current generation are selected for crossover with a probability that goes associated with the fitness function, where the probability value is calculated as:

\[ pv = \frac{f_{itv}}{\sum_{i=1}^{n} f_{itl}} \]

Pop = Total population in each generation.
fit, = Fitness value of the v member of a particular generation.

Therefore, the better fitter the individual is, higher are the chances of being selected for crossover. Finally, a minor percentage at the end of the new generation correspond to totally new individuals that are randomly generated, adding variety to the genetic material present in each generation and prevents a rapid convergence to a local minimum.

**Crossover:** this is a binary operator because it combines the information of two parent chromosomes to spawn one or more new individuals that inherit characteristics of both parents. There are several ways of implementing this mechanism, being the most common: one-point crossover, two-point crossover and uniform crossover.

The crossover operator selected depends if the two individuals chosen were part of the same or different groups. If both belong to “the good” (G) or “the fair” (F), then a two-point crossover with P_{cross}=0.9 is applied. Two non-negative integers are generated, n1 and n2, where n1 > n2. Given a pair of selected individual, “mother” and “father”, two offspring are generated, a “son” (S) and a “daughter” (D). The S receives the first n1 spots in the chromosome as well as those above n1+1 from the “father”; whilst from the “mother”, the S inherits the sequence between n1+1 and n2. The D is generated in the same manner, but in those genes where the S receives from the “father”, the D inherits from the “mother”, and vice versa. An example of this type of crossover operator is shown in Figure 7.

![Crossover of Different Groups](image)

**Figure 7 – Crossover of Different Groups**

On the other hand, when the selected individuals belong to different groups, a uniform crossover is proposed. For each gene, a random number between 0 and 1 is randomly generated and if the value is below a certain threshold that in this case is P_{cross}=0.7, the gene corresponding to G is selected; otherwise the genetic material from F is inherited. With this procedure, the probability of receive genes from the most fitted individual is increased. In Figure 8, it can be seen an example of the procedure mentioned.

![Crossover Within the Same Group](image)

**Figure 8 – Crossover Within the Same Group**

**Mutation:** After the crossover operator has been applied, the offspring might still resemble the parents. Mutation modify the chromosomes adding some extra variability that allows to increase the variation between one generation to the next and helps to prevent rapid convergence produced when a generation is trapped in a local optimum. There are different manners to implement this operator as well.

Mutation could be considered as a unary operator that creates a new chromosome based on an existing one. After crossover has been applied, then mutation is considered. The approach taken to implement this operator was that each position in the chromosome has a probability P_{mut} of being selected for mutation. The probability is zero for the “top” individuals of the new generation and increases as it goes down, until it reaches the individuals at the “bottom” for whom the probability P_{mut}=1 and thereby are randomly generated individuals. Whenever an entity is selected for mutation, it randomly reselects a value n_{new} within the interval [n_{old} – n_{old}; n_{old} + n_{old}] from a uniform distribution, where n is a value that decreases through the generations.

**Immigration:** these correspond to totally new individuals that are randomly generated and might not have similar genetic material as the other members of the new generation. These individuals can be thought as immigrant or new settlers that bring their own genetic characteristics to the population. This process is performed to prevent premature convergence.
**Chromosome codification:** as the representation used for the proposed method corresponds to random key representation, every gene that composes the chromosome is a number between 0 and 1. A chromosome represents a particular solution of the scheduling problem that is encoded as a vector of random keys and the purpose of the meta-heuristic method proposed is to attain the chromosome that yields the best possible value of the objective function. In an indirect representation, such as the one used, it is required a special procedure to decode the solution contained in the chromosome, which is called phenotype. Each chromosome is made of \(3n^*+m\) genes, where \(n^*\) represents the number of activities in the whole portfolio and \(m\) signifies the number of projects in the portfolio.

The initial \(n^*\) genes define the priorities. Genes between \(n^*+1\) and \(2n^*\) are used to determine the mode in which the activity is executed. The following \(n^*\) genes represent the delay before starting each activity and the last \(m\) genes define if the project is completed or cancelled.

**Decoding:** in order to obtain the solution (phenotype) from the chromosome it is necessary to decode it. Therefore, it is necessary to decode priorities, modes, delays and cancelled projects.

- **Priority:** the initial \(n^*\) genes of the chromosome represent the priority of the activities \(i\) \((i=1, \ldots, n^*)\) that belong to the portfolio of projects.

\[ Priority_i = Gene_i \]

- **Mode:** the genes between \(n^*+1\) and \(2n^*\) correspond to the mode. For each activity \(i\) there are certain modes in which the activity might be performed. Then, the space between 0 and 1 is divided in as many segments as modes for a particular activity. The mode will depend on the segment in which the gene is included. In the equation below \(Nm_i\) corresponds to the number of modes of activity \(i\).

\[ Mode_i = RoundUp\left(\frac{Gene_{i+n^*+1}}{Nm_i}\right) \]

- **Delay:** genes between \(2n^*+1\) and \(3n^*\) are used to define the delay times. In the equation below \(MDur\) correspond to the maximum duration amongst all task durations. The factor 1.2 is proposed after trying with values between 1 and 2.

\[ Delay_i = Gene_{2n^*+i} \times MDur \times 1.2 \]

**Cancellation:** the last \(m\) genes indicate which projects of the portfolio are going to be performed and which ones are going to be cancelled. If the value of the gene is closer to 1 then the project \(j\) \((j=1, \ldots, m)\) is performed, otherwise project \(j\) is cancelled. It should be mentioned that if the result is infeasible when taking into account the interaction among technologies, the genes corresponding to the cancellation are modified until a feasible outcome is attained.

\[ Cancellation_j = Round(Gene_{3n^*+j}) \]

**Population Size:** the size could be proportional to the number of activities within the portfolio multiplied by a number between 2 and 20. This policy is not adequate because the number of activities might differ depending on the portfolio. Another possibility is when the size is proportional to the number of genes in the chromosome multiplied by a number between 2 and 8.

\[ Pop\ Size = 4 \times \#Genes \]

**H. Artificial Bee Colony**

Artificial bee colony (ABC) belongs to the group of “swarm intelligence”, which refers to the collective behavior of decentralized and self-organized systems; usually composed by agents that follow simple rules, where their interactions lead to the emergence of intelligent behavior. The most common algorithms of this group that were applied to the project scheduling and planning problem are ABC, ACO and PSO [30].

The ABC is a population-based algorithm inspired by the intelligent foraging behavior observed in real honey bees. ABC considers three different kinds of honeybees: employed, onlookers and scouts. Each solution in the search space is treated as a food source, where the fitness value of the solution is represented by the amount of nectar contained in each food source. The number of employed bees equals the food sources around the hive. Employed bees share their information with onlookers, while onlookers select one of the food sources according to this information. Finally, a scout is a bee performing random search around the hive in order to find new food sources.

**The initial population:** correspond to SN number of randomly generated food sources that are created applying the following formula:

\[ x_i^{j} = x_{min}^{j} + \alpha(x_{max}^{j} - x_{min}^{j}) \]

where \(\alpha\) is a random number within the interval \((0,1)\), \(i=1, 2, \ldots, SN\) and \(j=1, 2, \ldots, D\) are the lower and upper bounds for dimension \(j\), respectively. Each food source is assigned to SN number of employed bees.

**The search phase:** each employed bee explores the neighborhood of every food source \(x_i^{j}\) by modifying one parameter obtaining \(v_i^{j}\).

\[ v_i^{j} = x_i^{j} + \gamma_{i}^{j}(x_i^{j} - x_{k}^{j}) \]

Where \(\gamma_{i}^{j}\) is a random number in the range \([-1, 1]\), \(k=1, 2, \ldots, SN\) and \(j=1, 2, \ldots, D\). Afterwards, \(x_i^{j}\) is compared with \(v_i^{j}\) and greedy selection is applied depending on the amount of nectar.
in each location. If \( v^t_i \geq x^t_i \), then \( v^t_i \) replace \( x^t_i \), otherwise \( x^t_i \) remains as the food source.

**Selection phase:** every employed bee shares the information about the diverse food sources with the onlooker bees which depending on the amount of nectar in each source execute some kind of selection scheme like ranking based, tournament selection and proportional selection, among others. In the original ABC algorithm, roulette selection is applied:

\[
pi = \frac{fit_i}{\sum_{l=1}^{SN} fit_l}
\]

where \( fit_i \) is the fitness value of each food source and the higher the value, the more likely it is that the food source is selected by the onlooker bees. Then, onlooker bees will continue to search the neighboring area by using the equation previously indicated for the search phase.

\[
v^t_i = x^t_i + \gamma_i (x^t_i - x^t_j)
\]

**Scout bee phase:** this is the last phase and it takes place if any food source cannot be improved further after searching the neighborhood for a pre-established number of cycles. In that case, the food source is abandoned and scout bees are sent to explore the region around the hive for brand new food sources to exploit. The formula corresponds to the one utilized for the initial population.

I. **Hybrid Algorithm (GA-ABC)**

This algorithm takes the strategy that was indicated in the subsection: Genetic Algorithm. However, it adds components of ABC to attain a nice balance between exploration and exploitation of the search space as well as reaching a suitable convergence speed.

The initial population is set with a number of chromosomes that depends on the genes contained in each one of them as it was explained in previous sections. The population of solution is created using the priority rules available from the base model and then it is populated with randomly created priorities to guarantee a varied population. This procedure is explained in detail in [37]. The initial mode assignment is performed by following a procedure called Minimum Normalized Resources (MNR) that reduces the probability of starting with infeasible solutions with respect to non-renewable resources. The MNR implies selecting the mode that has the minimum resource requirement of non-renewable resources (NW).

\[
NW_{ijm} = \sum_{k \in NW} \frac{wij_{jmk}}{W_k}
\]

The modes are initially selected following the MNR criterion, but in order to add diversity to the initial population, the modes of 50% of the total individuals are randomly changed, if the solutions are feasible then solutions are kept. Otherwise, for those assignments that turned out to be infeasible it will change the value randomly to another set of modes as long as the NW value is reduced. The procedure is performed for up to 100 iterations. For the genes that correspond to project cancellation, random key numbers are created and if the results are compared with the infeasible technology interactions so as to maintain a population of feasible solutions for selection. Afterwards, the individuals of the population are selected as it was mentioned in the section related to GA, and the crossover operator as well as mutation are applied as explained. Then, each solution enters the employee bee phase where one by one, each food source is compared with one neighbor solution that is obtained by modifying one gene. Greedy selection is applied and the best solution remains (if solutions have the same fitness value, the new solution remains). Subsequently, a probability vector is calculated according to the fitness value of each solution in order to decide which ones enter the onlooker bee phase. With the selected solutions, the neighborhood is analyzed by selecting at least three solutions changing one gene, if the solutions are better it increases the probability of continuing with the search up to certain iterations. However, if any of the three solutions is better it goes to the next food source. Finally, the algorithm examines which solutions did not improved for the last 50 iterations and it send scout bees to find new solutions unless it is a member of the “top” group in which case the solution remain for the potential of creating fitted individuals after crossover or mutation.

To maintain diversity in the population, the mechanism of immigration as well as directed mutating is introduced. The differences between solutions are measured with a population affinity index (PAI).

\[
PAI = \frac{\sum_{j=1}^{SN} \left( priority_j - priority_{j'} \right)^2 + \left( delay_j - delay_{j'} \right)^2 + \left( mod_j - mod_{j'} \right)^2 + \left( cancel_j - cancel_{j'} \right)^2}{\sum_{j=1}^{SN} \left( priority_j - priority_{j'} \right)^2 + \sum_{j=1}^{SN} \left( delay_j - delay_{j'} \right)^2 + \sum_{j=1}^{SN} \left( mod_j - mod_{j'} \right)^2 + \sum_{j=1}^{SN} \left( cancel_j - cancel_{j'} \right)^2}
\]

Whenever, the PAI is close to 0 then it means that the difference between two solutions is minimal. In addition, it is used as a global indicator of the variation in the entire population. In case that the population is not varied enough, a procedure is applied with certain probability \( P_{rep} \) that consist on exchanging one individual of the population that does not belong to the “top” group for a randomly generated individual with a probability \( P_{change} \) using the MNR procedure previously described. This procedure helps to maintain a diverse class of individuals without converging prematurely. The values of \( P_{rep} \) and \( P_{change} \) are set to 0.5 and 0.2 respectively.

J. **Reason for Choosing the Proposed Methods**

Every meta-heuristic should be designed with the objective of exploring the search space as thoroughly as possible. With that in mind, two concepts arise that lead meta-heuristic applications to high performance: exploration and exploitation. Even though the afore-mentioned terms are common in scientific literature related to meta-heuristics, there have been researchers that refer to them as diversification and intensification, respectively. These expressions might be used as synonyms. In any case, the main idea is that search should be diversified enough so that different regions of the search space are explored, avoiding staying in a particular and limited region alone because other areas might contain better solutions. At the same time, some effort should be spent in searching the neighborhood around good solutions because better points might be located close to them. Many other definitions can be

2017 Proceedings of PICMET '17: Technology Management for Interconnected World
found in literature, but they all seem to share similar principles. “The search performed by a meta-heuristic approach should be “clever” enough to both intensively explore areas of the search space with high quality solutions, and to move to unexplored areas of the search space when necessary” [38].

**Exploration:** refers to the broad examination of unvisited regions of the search space. In some literature, the term is stated as diversification. Figure 9 displays an example.

**Exploitation:** is the thorough search around good solutions found in the past. This term is also indicated as intensification. Figure 10 shows an example.

![Figure 9 – Exploring the Search Space](image)

![Figure 10 – Exploiting a Region](image)

In order to achieve a successful search with a meta-heuristic method, both strategies must be contemplated and balanced. In other words, it is necessary to perform a good job in exploring and exploiting the search space. “A meta-heuristic will be successful on a given optimization problem if it can provide a balance between the exploitation of the accumulated search experience and the exploration of the search space to identify regions with high quality solutions in a problem specific, near optimal way.” [39].

Every method, either population-based or trajectory based, possess both exploration and exploitation components. It could be possible that some methods provide a natural way of exploring or exploiting the solution space, and in those cases, they should be balanced with components that provide the dimension in which the selected method shows weakness. All meta-heuristic should have components that allow exploring the whole search space while each region is exploited so that there is a good probability that the best solution was chosen. For instance, in a trajectory-based method such as SA, the decrease of the temperature parameter drives the system from exploration to exploitation. Another example is that, previously in this section, it has been indicated that for selection in GA it is proposed elitism as a mechanism, which means that the top individuals are copied unchanged from one generation to the next. This mechanism guarantees that elite individuals survive through generations, but could lead to rapid convergence and prevents exploring other regions of the search space. However, that procedure is counterbalanced by another mechanism such as immigration, which guarantees that random individuals are being created, thereby favoring the exploration of new regions of the search space.

In addition, two other dimensions to consider when selecting and designing a meta-heuristic method are convergence and memory. The former relates to the time that takes to arrive to a situation in which solutions from successive generations are fairly similar. The latter, refers to the how the information about previous generations affects the new one.

**K. Considering Rework**

Every innovative and developmental process is inherently cyclical. Usually, iterations represent the rework of a particular activity due to discrepancies between expected and actual results, where the differences are found due to feedback information discovered later in the project. Furthermore, with “concurrent engineering” where design and engineering work in parallel, the increase in iterations became a huge challenge. Even though the process of infusing technology into a product or system is not as iterative as building a product from scratch, it has certain amount of iterations in it. Therefore, it becomes important to incorporate this feature into the model to make it more realistic of a technology infusion process. Of course, the amount of iteration will depend on the nature of the projects, where software projects will probably have a lot of iterations. On the other hand, infusing technology into an aircraft will require most of the work to be done upfront. In the middle, is that high tech consumer products will fall and that is the focus of the current paper. It is included the possibility of rework with the following assumptions:

- The probability of discovering rework after each activity is previously known
- It is possible to either encounter rework immediately after the same task is finished or during other activities with a pre-established probability.
- It is not possible to do, at the same time, activities that depend on the activity that requires rework.
- If certain activity needs rework at some time, the probability of redoing posterior activities remains unaffected.
- Resources necessary for redoing an activity are previously known

To indicate the activities that necessitate rework, the DSM (Design Structure Matrix) was used. Cycles in the network can be represented by feedback arcs, which represent the potential yet not the certainty of having to return to an activity previously done. Feedback arcs have a probability associated with them. Hence, they will be indicated with numbers between 0 and 1 located in the matrix supra-diagonal, which denote the probability that the activity of the corresponding row i requires rework while task j, indicated by the column, is being performed.

Regarding the time required for each iteration once rework is included, two elements are considered: learning effect and
degree of connectivity. These elements determine the duration of a particular activity when it is not being done for the first time. The learning effect considers the fact that people who complete an activity gain experience through successive iterations. Learning reduces the time required to complete certain task or increases the performance of workers. Degree of connectivity is related with the level of dependency among successive tasks. A higher dependency means that more time is necessary to propagate the change thought all dependent activities. The parameters $\varphi_{ij}$ (0 ≤ $\varphi_{ij}$ ≤ 1) and $\gamma_{ij}$ ($\gamma_{ij}$ ≥ 0) capture both the degree of connectivity and learning effect for task i from project j, respectively. The resources necessary after successive iterations can be calculated with the following formula:

$$r_{ij}^{*} = t_{ijm} + (t_{ijm} - t_{ijm}^0) \times e^{-\gamma_{ij}(k-1)} \times \varphi_{ij}$$

where $t_{ijm}$ is the time required to finish task i from project j in mode m when is done for the first time. $t_{ijm}^0$ is the minimum time possible and k represents the number of iterations. The parameter that represents the degree of connectivity is calculated as follows:

$$\gamma_{ij} = \sum_{i=1}^{N} l_{ij}$$

where $l_{ij}$ is the dependency intensity associated with task i from project j. $r_{ijm}$ are the resources required for activity i from project j in mode m. As the number of iterations increase, the factor $e^{-\gamma_{ij}(k-1)}$ becomes closer to zero and the necessary resources decrease until reaching the minimum value ($t_{ijm}^0$).

L. Including Rework in the GA-ABC Algorithm

Once rework is included in the model and considering the fact that activities might modify duration depending on the resources that were assigned to them, the optimal scheduling depends on what happens in reality. Probably, it will not be possible to optimize for every probabilistic scenario. The optimal solution will be different in each case and reality will collapse into one of them, but it is not possible to know in which particular one upfront. However, it is possible to find resource allocations that are robust, leading to consistently better results than other resource allocations.

Monte Carlo simulation is used to evaluate how the fitness function is impacted by different scenarios driven by rework activities as well as the resource allocation. Stochastic simulation provides a varying outcome because the fitness value may change with identical chromosomes due to the differences in rework as well as in duration. Different fitness values will be related with one chromosome where it is possible to calculate the expected value and the variation from that expected value leaving the outcome in a location within the graph that relates expected return with variation.

The characteristics of trade-spaces were already covered in section I.A, but the main feature is that it is possible to identify the best tradeoffs, which consist of elements that are superior to other in at least one dimension while inferior with regard to others. The concept of $\epsilon$-dominance consists on dividing the objective space into discrete hyper-boxes, as it is shown in Figure 11. This procedure allows reducing cases where insignificant differences in the fitness value lead to situations in which elements near the Pareto-frontier are categorized as dominated solutions. Those elements that are located in a narrow swatch near the frontier are guaranteed to be incorporated into the Pareto-front with this procedure.

It should be noted that this approach has a major downside, as evaluation of the fitness value of each chromosome though simulation can be time consuming. However, the $\epsilon$-dominance criterion leads to a reduced number of different outcomes, which limits the varying results of the simulation.

VI. RESULTS

A. Introduction

This section is devoted to showing the results obtained while running the proposed methods for the base and improved models. The methods are compared in the following manners:

- **Using standard benchmark instances:** the PSPLIB contains diverse data sets for several types of resource constrained project scheduling problems, together with the best found solutions for each instance.

- **Comparing the methods proposed for the base model against the improved one:** as the assumptions of the base model are loosened, the necessity of a new method becomes clear.

- **Comparing with results obtained using other methods from literature:** when there are results available
regarding the performance of different methods using similar models.

After defining how the methods are going to be compared, it is necessary to determine the dimensions for comparison and how they are measured. The goal will be to establish the usefulness of a particular method when a particular schedule of activities, resource allocation and portfolio of projects is proposed.

The usefulness of the method will depend on two factors:

**Credibility:** according to the definition adopted in this document, a method is credible if it can achieve a feasible solution, that is “good” and within a reasonable time. Therefore, credibility depends upon:

- **Feasibility:** no constraint should be broken
- **Computational Time:** reasonable when compared to other methods
- **Effectivity:** in searching the space of solutions for those with “good” fitness values

**Inclusiveness:** it depends on the capabilities of absorbing the relaxed assumptions of the models. Those points were covered in the previous section and they have to do with the characteristics that the proposed method should be capable of incorporating, such as:

- **Multiple target dates:** different projects may have diverse target dates and associated return
- **Technology Interaction:** consider the relationship among projects
- **Portfolio selection:** capacity of selecting those projects that will maximize return
- **Multiple resource types:** consider both renewable and non-renewable resources that might be either local or global
- **Multiple modes:** duration of activities can vary depending on resource allocation
- **Differences in performance:** resources may possess different suitability for doing a particular task
- **Rework:** certain activities may have to be redone and resource allocation may affect that probability

### B. Base Model

The base model shares the assumptions of the static RCMPSP, which were explained in section 1.B. The proposed method to deal with it is based on parallel SGS combined with different PRs and TBs. The details of the method were covered in section 1.Initially, the method was utilized to solving the RCPSP, which is similar to the multi-project case. However, the Project Scheduling Problem Library (PSPLIB) contains numerous instances for the case with single projects, providing a standardized benchmark for comparing the credibility of the method. The instances of the PSPLIB are available on a public web site [http://www.om-db.wi.tum.de/psplib/](http://www.om-db.wi.tum.de/psplib/) (last verification of address: 10-10-2016) which contains different problem sets for various types of resource constrained project scheduling problems. All the instances were generated using the project generator ProGen. In case of the RCMPSP, the instances available in the MPSPLIB will be used. These test problems were generated by [40] and can be used as a benchmark for problems with multiple projects. All the test problems consist of 2, 5, 10 or 20 project instances, composed of 30, 90 or 120 activities each, obtained from the PSPLIB that were combined into diverse portfolios which are publicly available, together with the best found solutions, from the web site [http://mpsplib.com](http://mpsplib.com) (last corroboration of address: 10-10-2016).

Several data sets were selected with diverse number of activities. The study included instances of 30, 60, 90 and 120 activities. Nevertheless, in the following paragraphs it will be shown the cases of instances j30_2_2 and J30_45_8 (composed of 30 tasks), and J120_32_4 (composed of 120 activities).

By applying the method based on PR and TB, the results indicate that for instance J30_2_2, it reaches the best possible solution using many different PR, which makes unnecessary to use TB. On the other hand, for instance J120_32_4 the best PR is notoriously deviated from the best possible solution. The deviation is measured with the ensuing formula:

\[
\text{Deviation} = \frac{(\text{Solution Best Possible} - \text{Solution Found})}{\text{Solution Best Possible}}
\]

| TABLE 3 – FOUND SOLUTIONS APPLYING THE METHOD BASED ON PR |
|---------------------------------|----------------|----------------|
| Best solution                   | 51             | 94             | 136           |
| Best found solution             | 51             | 100            | 169           |
| Deviation                       | 0%             | 4%             | 24%           |
| Best PR                         | FCFS / MINSLK / etc. | EDDF / TWK-LST | MAXSP         |

Table 3 shows, for the three selected instances, the deviation from the best possible solution. In case of the instance J120_32_4, the best PR turned out to be MAXSP. For the instance J30_45_8 there are two PR that produced the same result: EDDF and TWK-LST. Finally, J30_2_2 had 13 PR that reached the best possible solution, such as FCFS, MINSLK, EDDF, MINLFT, MAXSP, TWK-LST, MCS and others.

One question that is important to respond is how credible are the result. From the small sample provided in Table 3, it is noticeable that deviation may range extensively. Therefore, it will be analyzed if the deviation is associated with some project characteristics. The different instances that were studied had been categorized according to the number of activities, degree of complexity and utilization factor. The procedure was similar to the one explained for the previous three cases. For each one of them the deviation was calculated and the results are shown in the following paragraphs.
When the method is applied to the multi-project case, results follow similar pattern depending on the three characteristics covered: number of activities, network complexity and utilization factor. It will be considered the case of the following portfolio of projects:

| Portfolio | Instances | Projects | Activities | R1 | R2 | R3 | R4 |
|-----------|-----------|----------|------------|----|----|----|----|
| J30_2_2   | 2         | 60       | 13         | 11 | 13 | 16 |
| J30_45_8  | 1         | 60       | 13         | 11 | 13 | 16 |

The characteristics of both instances that compose the portfolio are shown in Table 3. If it is considered that the objective is to reduce the total make-span of the portfolio, which is measured from the earliest activity to the end of the last task (independently of the project that belongs to). In that case, the PR that yields the best result is the maximum total successors (MTS). However, in [27] it is shown that applying backward-forward hybrid genetic algorithm for solving the same portfolio, the result is better than any achievable with the proposed priority rules.

Nonetheless, minimization of total portfolio delay as an objective function has important limitations in the case of technology infusion projects because time-to-market and the maturity level of technologies and their interactions lead to consideration of separate target dates for each project. It has been emphasized in section I.A the importance of considering different target dates for each project. Consequently, the proposed objective function that was covered in Section IV.A. is used.

Table 5 below displays variations of the parameters of the objective function; it will be shown how these settings affect the results of the proposed method. Considering scenario 1, the maximum return possible is $2.500 if both projects end within 120 days. The result will decrease if any of the projects finish beyond the target date. So far, project cancellation is not allowed.

| Scenario 1 | Scenario 2 | Scenario 3 |
|-------------|-------------|-------------|
| Project J30_2_2 |             |             |
| Target Date | 120 days    | 90 days     | 140 days |
| MDL         | 200 days    | 200 days    | 200 days |
| Curve Type  | 1           | 1           | 3        |
| E[ANPV]     | 1.000       | 1.500       | 1.500    |

| Project J30_45_8 |             |             |
| Target Date      | 120 days    | 140 days    | 140 days |
| MDL              | 200 days    | 200 days    | 200 days |
| Curve Type       | 1           | 1           | 3        |
| E[ANPV]          | 1.500       | 1.500       | 1.500    |

After running the method, the result indicates that the best PR turns out to be MTS, giving a total return of $2.388. Project J30_2_2 finishes within the 120 days, but projects J30_45_8 ends after 129 days. Regarding scenario 2, the maximum possible return is $3.000 and after applying the method 5 different PRs reach that sum, none of them is MTS though. The 5 PRs are MAXTWK, EDDF, MINLFT, TWK-LST and TWK-EST. Finally, for scenario 3 the maximum return is again $3.000. The main difference in this scenario compared to the preceding ones is that scenario 1 and 2 only penalized tardiness, whereas scenario 3 castigates both tardiness and earliness. The PR that attains the best result is LCFS yielding $2.765, where project J30_2_2 has an estimated end date in 111 days and project J30_45_8 finishes exactly in 140 days.

In general, the results for the base method reveal that PR performance depends on the chosen objective function, or in this case, the parameters selected for the objective function. In the paragraph above three scenarios are shown; by modifying parameters of the objective function slightly, the PR that provides the best outcome varies in each case. In addition, as the parallel SGS that is used to create the project plan generates only non-delay schedules and there are cases in which earliness is penalized, it could be beneficial delaying some activities in order to finish the project as close as possible from the target date. Therefore, the method should include the possibility of delaying tasks. Finally, it has been shown that solutions obtained though PRs deviate from the best possible schedules as: (1) number of activities increase, (2) complexity decreases and (3) utilization factor gets closer to the interval [1.3-1.5]. The fact that in this document the focus is a project portfolio makes probable that the number of activities will be elevated. Hence, when the objective is minimization of make-span, best possible solutions are usually not attained with PRs.

C. Improved Model

From the previous sub-section, it became clear that it was necessary to create a more effective method for finding solutions to the RCMPSP. The proposed approach consists of a hybrid meta-heuristic method combining GA and ABC, and from now on, it will referred to as “improved method”. Initially, the improved method is compared for the same 40 instances considered for the method based on PR. The results indicate that after 5000 iterations the improved method is able to find the best possible solutions in each case. The computational time never exceeded 2 seconds. In Table 6 it is possible to appreciate the outcome obtained for the instances shown in the previous sub-section. As it is noted, the deviation from the best possible solution result in these cases is 0%.

| Table 6 – Found Solutions Applying the Meta-heuristic Method |
|---------------|-----------------|-----------------|-----------------|
| J30_2_2       | J30_45_8        | J120_32_4       |
| Best Solution | 51              | 94              | 136             |
| Best Found Solution | 51              | 94              | 136             |
| Deviation     | 0%              | 0%              | 0%              |
| Iterations    | 5000            | 5000            | 5000            |
| Computational Time (sec) | 0.6            | 0.9             | 1.6             |
Regarding multi-projects, in a study performed by [26] there are two multi-project instances, where one corresponds to a test portfolio and the other to a real portfolio. In both portfolios, it has been compared the performance of different meta-heuristic methods:

- A genetic algorithm
- Simulated annealing
- Hybrid GA – SA algorithm
- Modified simulated annealing (arithmetically improved)
- Modified simulated annealing (logarithmically improved)

In the current document it is compared the performance obtained with the improved method against the mentioned meta-heuristics only for the test portfolio, which is composed of 74 activities and 2 renewable resources.

**Table 7 – Numerical Comparison of Methods**

| Method   | Average | Runs |
|----------|---------|------|
| GA       | 135.5   | 10   |
| SA       | 135.4   | 10   |
| GA+SA    | 134.5   | 10   |
| MSA(1)   | 134.2   | 10   |
| MSA(2)   | 133     | 10   |
| Base Model | 154.7  | 10   |
| Improved Model | 126.1 | 10   |

In Table 7 are shown the numerical result of applying the improved method covered in section V to the test portfolio. As it can be appreciated, the results indicate that after 10 runs outperforms on average the other five methods. The method was stopped in each run after 5,000 iterations, which took an average of 16 seconds each.

In addition, section V.I covered the concept of hybridization between evolutionary algorithm and swarm intelligence. As was indicated in section V.I, simple GA usually suffer from the problem of rapid convergence, a fact that signify that without mechanisms that assure a varied individuals in each generation, it could lead to exploring only a sub-section of the whole solution space potentially leading to a local optimum instead of the global one. The mechanism that was proposed, in this document, to address the suggested issue was the hybridization with ABC, which provides very good exploratory characteristics. In order to test the suitability of the hybridization, an experiment was performed in which the results for a simple GA and the GA-ABC algorithm are compared for different portfolios. In each case, 10 runs were executed for each portfolio and the best feasible solution was documented.

The results, for the same portfolio utilized in the previous paragraph, are shown in Figure 12, where the hybrid GA-ABC (blue line) proved to be slightly better than the simple GA (red line) after 1000, 3000 and 5000 generations. Nonetheless, there is another conclusion that could be drawn from the graph which corresponds to the variability of the results during different runs. It is notable that the areas corresponding to the red ellipses are greater than blue ones. This means that simple GA might yield quite different results from one experiment to the other, where sometimes it could lead to solutions that are far from optimal.

![Figure 12 – Comparison of Performance Simple GA vs. Hybrid GA-ABC](image)

Furthermore, in some cases the approach stops making improvements displaying premature convergence. On the other hand, the hybrid algorithm has mechanisms than ensure the variability of the population which guarantees that the results are better and do not change from one run to another. In addition, the hybrid approach shows continuous improvements indicating that the algorithm does not converge prematurely.

Figure 13 displays the expected return when rework is not considered (first column) and adding the average effect of it (third column). The best schedule and resource allocation achieved by applying the improved model to a portfolio of projects, simulated 5000 times with the rework conditions established. It is noticeable, for the example considered, the impact that rework has on the duration of each project and consequently in the expected return, which was 61% lower. Furthermore, if one of the projects is cancelled (j20_11_8) the expected return increases a 9%. Even though it is completely lost the return from the project that was cancelled, it liberates global resources that could be allocated in the remaining projects achieving a greater overall return.

![Figure 13 – Example of Effect of Rework](image)

The effect of rework was tested with different portfolios, but it should be mentioned that the probabilities of redoing activities as well as the number of feedback arcs could
dramatically affect the duration of projects and the expected return.

For different portfolios, when iterations due to rework are included, it is interesting to verify if the best solution when rework is zero still stands. One caveat, is that feedback arcs represent the possibility but not the certainty of redoing certain task. Therefore, experimenting with the same solution might yield diverse results over time. Thus, the results will be analyzed in a trade-space where the x-axis represents the average value obtained though many runs and the y-axis depict the variation over the expected value. Non-dominated solutions are part of the Pareto Frontier, which means that there might be more than one solution that is equally suitable and the final decision depends on a trade-off between expected return and its variation.

The goal is to find robust activity schedules and resource allocations that yield a great expected value while keeping the variation at a minimum. The best case scenario (utopia point), would be to achieve maximum expected return with no variation. The question that arises is if the best solution when rework is not considered belongs to the Pareto Frontier when it is incorporated. Figure 14 shows the trade-space with the feasible solutions of the last generation attained when the improved method was executed.

![Figure 14](image1)

Figure 14 – Trade-Space for the Example Portfolio and Pareto Frontier

Furthermore, once the improved method is executed considering iterations, the Pareto Frontier is shown in red in Figure 14 (b) and Figure 15. The former graph makes clear that the new Pareto Frontier is closer to the utopia point than the previous frontier; whereas, the latter diagram shows the comparison between the best solution without considering rework and the Pareto Frontier (red line) formed by the best solutions when iterations are included. It stands out, from Figure 15, that it is possible to achieve the same expected value with approximately 54% less variation, or the same variation but with an increase in the expected value of about 25%. This result signifies that the best solutions found considering no rework might not be part of the Pareto Frontier when the effect of iterations is included. In addition, the same analysis was performed in 29 other portfolios and the outcome was that in only 12% of the cases the best solution without rework ended up being part of the Pareto Frontier once iterations were incorporated. Consequently, it is important to plan for rework because the best solution will probably be different.

![Figure 15](image2)

Figure 15 – Comparison of the Best Solution without Rework and the Best Solutions w/ Rework Included

In conclusion, the current section initiated by specifying the manner for measuring the usefulness of each method on two dimensions: credibility and inclusiveness. Section VLB explained the results obtained applying the base method, which was founded on a rigid model (static RCMPS). Once the assumptions of the mentioned model were relaxed, another method was required. Thereby, the improved method was proposed and tested in section VLC. Its outcomes were compared against the base method and results obtained from other algorithms available in literature. It is shown that the new method is able to find feasible, optimal or near optimal solutions in a variety of situations within acceptable computational time. For comparison against a common benchmark, results were compared using publicly available instances. The improved method is credible for the single project, multi-project and multi-mode cases. Finally, the method was tested for conditions that are not commonly treated in literature, the addition of rework, in order to demonstrate the model’s representativeness and the ability of the improved method to deal with rework. Since rework can contribute a significant increase in multi-project duration, consideration of these factors are critical when forecasting the allocation of resources and schedule of activities.
VII. CONCLUSION

In order to design portfolio through allocation and scheduling which includes optional insertion of technologies, initially a heuristic method was proposed that uses parallel SGS and a combination of priority rules and tie-breakers (PR-TB). However, numerical results indicate that the base method failed to provide flexible schedules due to the non-delay nature of the generated schedules.

Later, a hybrid meta-heuristic method was proposed based on swarm intelligence and an evolutionary algorithm to effectively schedule activities, allocate resources, and manage the project portfolio. Regarding the performance of this new method, numerical results show that high-quality schedules are generated and resources are allocated efficiently.

Existing meta-heuristic approaches for task scheduling rarely incorporate rework into the model. The improved method accounts for the interaction among the projects that comprise the portfolio as well as the possibility of rework. To evaluate the different scenarios that arise depending on how rework loops are distributed and their associated probabilities, the ε-dominance criterion leads to a reduced number of outcomes. By varying the size of the hyper-boxes, it is possible to modify the time required for the simulation. Although consideration of rework increases significantly the complexity of the problem, the results show that the best solution attained before rework is considered can be far from the pareto front of the best solution once rework is incorporated. The proposed method allows locating the set of best tradeoffs in terms of expected return and risk associated.

The results demonstrate that adding rework, and relating the performance of resources with some variation in the probability of redoing tasks, produces in some cases a notable increase in multi-project duration that depends on the probability of rework, amount of feedback arcs and how performance affects activity duration.

In addition, results reveal that inclusion of rework attributes should influence the most effective manners to allocate resources and schedule activities. Therefore, project managers should include rework attributes of activities and their dependence across project architecture in the planning phase. Furthermore, the inclusion of rework can alter the composition of the portfolio that achieves the higher return, as was shown with a numerical example.

VIII. LIMITATIONS AND FUTURE WORK

Several directions for future work could be followed. One possibility arises because technologies are rarely ready for infusion. It is probable that they require certain degree of maturation before incorporating them into the host product or system. However, if the required readiness level (TRL) is not achieved, it might not be beneficial to infuse them. Therefore, reaching different TRls might be associated with diverse returns that could affect the attractiveness of including a particular technology. The current method might be expanded by investigating how the TRL might affect portfolio selection.

Another possibility related to technology infusion is due to interaction among more than two technologies. For instance, the inclusion of two independent technologies simultaneously might affect the inclusion of a third, while the two technologies taken separately might not influence that third one. The current method takes into account a relation between pair of technologies but does not consider more complex interactions. Investigating complex technology interaction in the performance space would be an interesting way to extend the current approach.

Another topic of interest is to study the dynamic arrival of a project to the portfolio in real time. The current model assumes a static scenario, where it is necessary to decide upfront the projects that will comprise the portfolio. It has been shown that in certain occasions cancelling a project might benefit the overall return of the portfolio. Therefore, it could be important to establish how each incoming project could potentially affect the schedule, project selection and resource allocation in an existing portfolio.

Other lines of improvement include the incorporation of resource transfer time and the related cost for project execution. Transfer of resources may take time mainly when certain resources are physically moved from one location to another. In addition, when resources need to be adjusted before starting another activity, setup and shut-down may be required. People might require training before initiating another activity. The literature available on resource transfer time is scarce but studies have shown that it affects both project delay and portfolio overall duration [22].

REFERENCES

[1] A. Lova, C. Maroto & P. Tormos, “A multicriteria heuristic method to improve resource allocation in multiproject scheduling”, European Journal of Operational Research, 127, pp. 408-424, 2000.
[2] C. Maroto, P. Tormos, A. Lova, “The evolution of software quality in project scheduling”, In: Weglarz, J. (Ed.), Project Scheduling. Kluwer Academic Publishers, Boston, pp. 239-259, 1999.
[3] J.H. Payne, “Management of multiple simultaneous projects: A state-of-the-art review”, International Journal of Project Management, 13(3), pp. 163-168, 1995.
[4] Z. Laslo & A.I. Goldberg, “Resource allocation under uncertainty in a multi-project matrix environment: Is organization conflict inevitable?”, International Journal of Project Management, 26, pp. 773-788, 2008.
[5] E. Crawley, B. Cameron & D. Selva, “System Architecture: Strategy and Product Development for Complex Systems”, 1 ed. Pearson Higher Education, 2016.
[6] R. Smaling & O. de Weck, “Assessing risks and opportunities of technology infusion in system design”, Systems Engineering 10(1), pp. 1-25, 2007.
[7] R. Smaling, “System architecture selection under uncertainty”, Ph.D. Thesis, Engineering System Division, Massachusetts Institute of Technology, Cambridge, MA, 2005.
[8] E. Suh, M.R. Furst, K.J. Mihalyov & O.L. de Weck, “Technology infusion: An assessment framework and case study”, Systems Engineering 13(2), pp. 186-203, 2010.
[9] T. Wauters, J. Kinable, P. Smet, W. Vancroonenburg, G. Vanden Bergh & J. Verstichel, “The multi-mode resource-constrained multi-project scheduling problem”, J Sched, 19, pp. 271-283, 2014.
[10] J.K. Lenstra & A.H. Kan, “Complexity of scheduling under precedence constraints”, Operations Research, 26(1), pp. 22-35, 1978.
[11] J. Blazewicz, J.K. Lenstra & A.H.G. Rinnooy Kan, “Scheduling subject to resource constraints”, Discrete Applied Mathematics 5, 11-24, 1983.
[12] M. Ni, P.B. Luh & B.R. Moser, “An optimization-based approach for design project scheduling”, IIE Transactions on Automation Science and Engineering, 5(3), 2008.

[13] P.B. Luh, F. Liu & B.R. Moser, “Scheduling of design projects with uncertain number of iterations”, European Journal of Operational Research, 113(3), pp. 575-592, 1999.

[14] X. Wang, Q. Chen, N. Mao, X. Chen & Z. Li, “Proactive approach for stochastic RCMPSP based on multi-priority rule combinations”, International Journal of Production Research, 53(4), pp. 1098-1110, 2015.

[15] P. Brucker, A. Drexl, R. Möhring, K. Neumann & E. Pesch, “Resource-constrained project scheduling: Notation, classification, models and methods”, European Journal of Operational Research, 112, pp. 3-41, 1999.

[16] R. Kolisch, “Efficient priority rules for the resource-constrained project scheduling problem”, Journal of Operations Management, 14(3), pp. 179-192, 1996.

[17] I. Kurtulus & E.W. Davis, “Multi-project scheduling: categorization of heuristic rules performance”, Management Science, 28(2), pp. 161-172, 1982.

[18] T.R. Browning & A.A. Yassine, “Resource-constrained multi-project scheduling: Priority rule performance revisited”, Int. J. Production Economics, 2010a.

[19] T.R. Browning & A.A. Yassine, “A random generator of resource-constrained multi-project network problems”, Journal of Scheduling, 13(2), p143-161, 2010b.

[20] T.R. Browning & A.A. Yassine, “Managing a portfolio of product development project under resource constraints”, Journal of the Decision Science Institute, 2015.

[21] E. Pérez Vázquez, M. Posada Calvo & P. Martín Ordóñez, “Learning process on priority rules to solve the RCMPSP”, Journal Intell Manuf, 26, pp. 123-138, 2015.

[22] D. Krüger & A. Scholl, “A heuristic solution framework for the resource constrained (multi-)project scheduling problem with sequence-dependent transfer times”, European Journal of Operational Research, 2008.

[23] A. Lova, P. Tornos, “Combining random sampling and backward-forward heuristics for resource-constrained multi-project scheduling”, In: Proceedings of the 8th International Workshop on Project Management and Scheduling, Valencia, Spain, April 3-5, pp. 244-248, 2002.

[24] U. Besikci, Ü. Bilge & G. Ulusoy, “Multi-mode resource constrained multi-project scheduling and resource portfolio problem”, European Journal of Operational Research, 2014.

[25] J.F. Gonçalves, J.J.M. Mendes & M.G.C. Resende, “A genetic algorithm for the resource constrained multi-project scheduling problem”, AT&T Labs Research Technical Report, 2006.

[26] P. Chen & S.M. Shahandashi, “Hybrid of genetic algorithm and simulated annealing for multiple project scheduling with multiple resource constraints”, Automation in Construction, 18, pp. 434-443, 2009.

[27] R. Sonmez & F. Uysal, “Backward-forward hybrid genetic algorithm for resource-constrained multiproject scheduling problem”, Journal Comput. Civ. Eng., 29(5), 2015.

[28] D. Karaboga & B. Bastuk, “On the performance of artificial bee colony (ABC) algorithm”, Applied Soft Computing, 8(1), pp. 687-697, 2008.

[29] H. Li & H. Zhang, “Ant colony optimization-based multi-mode scheduling under renewable and nonrenewable resource constraints”, Automation in Construction, 35, pp. 431-438, 2013.

[30] T. Chen & C. Ju, “Comparative analysis of swarm intelligence and heuristic priority rules for solving multi-project scheduling problem”, International journal of computing science and mathematics, 6(6), pp. 581-599, 2015.

[31] A. Lova & P. Tornos, “Analysis of scheduling schemes and heuristic rules performance in resource-constrained multi-project scheduling”, Annals of Operations Research, 102(1-4), pp. 263-286, 2001.

[32] C. Meier, A.A. Yassine, T.R. Browning & U. Walter, “Optimizing time-cost trade-offs in product development projects with multi-objective evolutionary algorithm”, Research Engineering Design, 27, pp. 347-366, 2016.

[33] T. Chen & C. Ju, “Characteristic description of coupled task sets based on design structure matrix and its optimal time and efforts solution by genetic algorithm”, Journal of Convergence Information Technology, 5(7), pp. 141-147, 2010.

[34] J.M. Lynes & D.N. Ford, “System dynamics applied to project management: a survey, assessment, and directions for future research”, Syst. Dyn. Rev., vol. 23, no. 2–3, pp. 157–189, 2007.

[35] A. Sprecher, S. Hartmann & A. Drexl, “An exact algorithm for project scheduling with multiple modes”, OR Spectrum 19, pp. 195-203, 1997.

[36] R. Kolisch & S. Hartmann, “Heuristic algorithms for the resource-constrained project scheduling problem: classification and Computational analysis”, in: J.Wessglarz(Ed.), Project Scheduling - Recent Models, Algorithms and Applications, Kluwer Academic Publishers, pp. 147-178, 1999.

[37] A. Lova, P. Tornos, M. Cervantes & F. Barber, “An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes”, International Journal of Production Economics, 117, pp. 302-316, 2009.

[38] C. Blum & A. Roli, “Meta-heuristics in combinatorial optimization: Overview and conceptual comparison”, ACM Computing Survey, 35(3), pp. 268-308, 2003.

[39] T. Stützle, “Local search algorithm for combinatorial problems - Analysis, algorithms and new applications”, DISKI-Dissertationen zur Künstlichen Intelligenz, 1999.

[40] J. Homberger, “A multi-agent system for the decentralized resource-constrained multi-project scheduling problem”, International Transactions in Operational Research, 14(6), pp. 565-589, 2007.