

# We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800

Open access books available

122,000

International authors and editors

135M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index  
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?  
Contact [book.department@intechopen.com](mailto:book.department@intechopen.com)

Numbers displayed above are based on latest data collected.  
For more information visit [www.intechopen.com](http://www.intechopen.com)



---

# Optimization Algorithms in Project Scheduling

---

Amer M. Fahmy

Additional information is available at the end of the chapter

<http://dx.doi.org/10.5772/63108>

---

## Abstract

Scheduling, or planning in a general perspective, is the backbone of project management; thus, the successful implementation of project scheduling is a key factor to projects' success. Due to its complexity and challenging nature, scheduling has become one of the most famous research topics within the operational research context, and it has been widely researched in practical applications within various industries, especially manufacturing, construction, and computer engineering. Accordingly, the literature is rich with many implementations of different optimization algorithms and their extensions within the project scheduling problem (PSP) analysis field. This study is intended to exhibit the general modelling of the PSP, and to survey the implementations of various optimization algorithms adopted for solving the different types of the PSP.

**Keywords:** project scheduling, project schedules optimization, resource-constrained scheduling, scheduling models, optimization algorithms

---

## 1. Introduction

The project scheduling problem (PSP) is one of the most challenging problems in the operations research (OR) field; thus, it has attracted large number of researchers within its modeling, solution methodologies, and optimization algorithms. The OR literature is intensely rich with researches focusing on different PSP problem types, from which the most famous and heavily researched problem type is the resource-constrained project scheduling problem (RCPSP).

PSP, especially RCPSP, has been considered as NP-Hard in the strong sense [1], and accordingly most researches within the last two decades concentrated on heuristics and meta-heuristics for solving different PSP types.

---

This study will start in Section 2 with reviewing the PSP modelling for various problem types. Then, PSPs solution approaches and architectures presented in literature will be demonstrated in Section 3. And finally, the last section of this study will survey the schedule optimization algorithms, and how each optimization technique was adopted and implemented within the project scheduling field; with a large focus on meta-heuristic optimization techniques, which are the base for most practical approaches for solving real-life/practical PSPs.

## 2. PSP modelling

According to the detailed surveys of the problem's developments and extensions [2–9], there is almost a common agreement on the concepts of how the problem should be mathematically modelled; however, the main differences were in the various problems classifications and the different mathematical notations used.

### 2.1. The general PSP model

In literature, the PSP was initially introduced within the manufacturing industry for job-shop scheduling, with a main concern of optimal allocation over time of the shop floor's scarce resources. Consequently, the PSP modelling has developed through the development of the basic modelling of the RCPSP.

Several problem notations and models were presented for the RCPSP, as surveyed by Brucker [5], from which the basic and most widely adopted notation can be summarized as follows: A project is represented by a set of activities  $V = \{1, \dots, n\}$ , where 1 and  $n$  are two dummy activities resembling the project network's start and end nodes. The characteristics defining each activity, that is,  $V$ : processing duration  $d_i$ , resource requirements  $r_{ik}$  (for each resource  $k$ ) along the activity's processing time, and a set of predecessors  $P_i$  logically tied to activity  $i$  with finish-to-start (FS) logic relations. The availability of each resource  $k$  is constrained throughout the project by the available resource units  $a_k$ . And finally, the problem's objective is to minimize the time span  $T$  of the project's schedule  $S$  without violating the resources constraints; where  $S$  is represented by a set of activities' start times  $S_1$  to  $S_n$ ; and  $T = S_n$ .

### 2.2. PSP model extensions

The basic RCPSP model has several short falls with respect to over-simplifying the characteristics of real-life scheduling problems. Consequently, several studies in the PSP context were focused on improving the problem's modelling by developing extensions which add missing practical aspects to the basic PSP model. The most important extensions can be categorized based on the short falls in the basic model which they addressed as follows:

- The basic PSP (and RCPSP) model considers only FS logic relations, and assumes all schedule relations without lag. Demeulemeester and Herroelen [10], Klein and Scholl [11], Chassiakos and Sakellariopoulos [12], and Vanhoucke [13] have researched this short fall and presented few model extensions to capture lag periods and other relation types available

in real applications. They accommodated these issues within the model by transforming all relations into start-to-start (SS) relations ( $S_{ij}$ ), and combined all sets of activities' predecessors into a single set of time lags  $l_{ij}$  corresponding to the schedule's logic; where the values of lag can be set to zero to represent SS relation,  $d_i$  to represent FS relation,  $d_j$  to represent FF relation, or any other necessary time lag. These model extensions are known in the scheduling literature as '*Generalized Precedence Relations*' or '*Minimum/Maximum Time Lags*'.

- Activities in reality are not necessarily processed in a single continuous run as assumed in the original PSP model. The need for presenting activities' splitting, or '*Pre-emption*', was introduced by allowing each schedule activity  $i$  to be split, or pre-empted, into  $j$  sections corresponding to the activity's duration  $d_i$ . For further details on pre-emption extensions, the reader is referred to [14–16]. Another alternative *pre-emption* modelling was presented and developed by Franck et al. [17], Lamothe et al. [18], and Kreter et al. [19], to resemble the non-linear availability of non-renewable resources through break-calendars.
- The '*Multi-Mode Scheduling*' is another practical extension in the project scheduling context, where activities can be executed in one of several alternative ways/methods. This has been modelled by Elmaghraby [20], Kolisch and Drexel [21], Hartmann [22], and Alcaraz et al. [23] through allowing each activity to have a set of execution modes  $M_i$ , where each activity mode is having a different duration and/or resource requirements.
- '*Cumulative Resources*' or '*Inventory Constraints*' is another form of resources irregularity. It was introduced by Neumann and Zimmermann [24] as a resource which is produced by some activities, cumulated in a storage area, and then used by other activities (e.g. precast elements in construction projects).

By combining most of these extensions to the basic model, the problem formulation can be summarized as follows:

$$\text{Objective: Minimize } S_{n,0} \quad (1)$$

$$\text{Subject to: } S_{1,1} = 0, \quad (2)$$

$$S_{i,0} + l_{ij} \leq S_{j,0} \quad \forall (i,j) \in L; i, j \in V \quad (3)$$

$$S_{i,j-1} + 1 \leq S_{i,j}, i = 1, 2, \dots, n; j = 1, 2, \dots, d_{im}; m \in M_i \quad (4)$$

$$\sum_{i \in S_t} r_{ikm} \leq a_k \quad \forall k \in K; j = 1, 2, \dots, d_{im}; m \in M_i; t = 1, 2, \dots, T \quad (5)$$

where  $V$  is the activities set;  $T$  is the schedule's time span;  $S_{ij}$  is the start time for section  $j$  of activity  $i$ ;  $d_{im}$  is the processing time of activity  $i$  under the execution mode  $m$ ;  $L$  is the set of schedule's logic (or precedence), including time lags represented in the form of activity pairs, and  $l_{ij}$  is the time lag between activities  $i$  and  $j$ ;  $S_t$  is a set which encapsulates the progress of all activities within time interval  $[t-1, t]$ ;  $K$  is the renewable resources set;  $r_{ijkm}$  is the resource requirement from resource  $k$  in the execution mode  $m$  for section  $j$  of activity  $i$ ; and finally,  $a_k$  represents the total available units for resource  $k$ .

### 2.3. PSP objectives extensions

The objective of the basic PSP model is minimizing time, while several practical applications involve other cost- or resource-related objectives. All above mentioned model extensions were also related to minimizing the project's make-span; however, other practical objectives were also introduced in literature as special cases of the PSP. Extensions to PSP objectives can be classified under two main categories: cost related and quality related.

Cost-related objectives and/or constraints were introduced to overcome the over-simplification of the basic PSP models with respect to financial side effects of schedule changes. Various models were generated to present cost aspects of PSP, the most famous of which is the *trade-off problem*, while quality-related objectives were introduced to enhance the robustness of optimized schedules.

These model extensions (i.e. with different objective functions) were presented in as different PSP problem types, the most popular of which are:

- *The trade-off problems*: The 'Time/Resource Trade-off Problem (TRTP)' [25–27] and the 'Time-Cost Trade-off Problem (TCTP)' [28, 29] are the most famous objective-related extensions of the PSP model. In these problem types, shorter schedule time is exchanged with the increase in resources (or vice versa); and project cost can be either modelled as a constraint or as a non-renewable resource. In scheduling literature, *trade-off* problems are generally being used in one of two forms:

*Case 1*: Minimize make-span, and constraint total costs with a predefined budget:

*Objective Function*: Minimize  $F_n$

$$\text{New Constraint : } \sum_{i \in V} c_i \leq \bar{C} \quad (6)$$

where,  $c_i$  = Cost of activity  $i$ , and  $\bar{C}$  = Project maximum allowed cost (or project budget)

*Case 2*: Minimize total costs, and constraint total project make-span with a predefined target date or deadline date:

$$\text{Objective Function : Minimize } \sum_{i \in V} c_i \quad (7)$$

$$\text{New Constraint : } F_n \leq \bar{d} \quad (8)$$

where:  $c_i$  = Cost of executing activity  $i$ , and  $\bar{d}$  = Deadline for the project's completion

- *Liquidity constraints:* In practical applications, project's liquidity is usually constrained with a predefined value of cash allotted to the project; and accordingly, this value should not be exceeded by the negative side of the project's cash flow. To enable amending the scheduling model with cash flow constraints, several financial aspects will have to be taken into consideration such as selling price, advance payment, retention, and payment lag (the time lag between invoices applications and the actual receipt of payments); then a new constraint is to be added as follows:

$$\text{Cash flow constraint: } CF_t \geq -|NCF_{\max}| \quad \forall t = 1, 2, \dots, T \quad (9)$$

Where:  $CF_t$  = Cash flow at period  $t$ , and  $NCF_{\max}$  = Predefined maximum liquidity

- For self-financed projects, the scheduling literature contains few special case problems such as RCPSP with discounted cash-flow (RCPSP-DC) and maximum net present value (Max-NPV) where the time cost of financing money is being considered in the project's overall cost (refer to [30–32]). Representing this mathematically involves introducing a per period discounting rate ( $\alpha$ ), and adding it exponentially to represent its effect on successor periods; thus, the discounted cash flow for each activity  $i$  will be as shown in equation 10. And finally, the objective function will have to be modified to include the net present value for the per period cash flow values:

$$CF_i = \sum_{i=1}^{d_i} c_{it} e^{\alpha(d_i-t)} \quad (10)$$

$$\sum_{i \in V} q_t CF_i \quad (11)$$

$$\text{Where : } q_t = \exp(-\alpha t) \quad (12)$$

- *Quality indices:* Schedules quality measures were introduced in literature for the purpose of improving schedules robustness [33]. These indices were mainly researched within robust proactive scheduling approaches which focus on building predictive schedules for satisfying performance requirements predictably in a dynamic environment [34–36].

## 2.4. Multi-objective PSP (MOPSP) modelling

The above review of PSP problem types shows clearly that practical scheduling application involves multi-objective nature; which attracted few early scheduling researches (such as [37, 38]) to try to define how multi-objective concepts can be introduced to the scheduling field. Nevertheless, to date, there are only few researches in the scheduling context which adopted multi-objective approaches [39].



In general, the approaches presented for MOPSP modelling (as defined and/or surveyed by Hwang [37], Slowinski [38] and Ballestín and Branco [39]) can be summarized under the following main categories:

1. Non-interactive approach: Where the optimality decision is left to the optimization algorithm after feeding it with either a predefined '*Weighted Objectives Function*', or an '*Objectives Priority List*' through which objectives are being ordered and optimized based on predefined priorities.
2. Semi-interactive approach: The optimization algorithm defines the solutions '*Pareto Front*', or a group of optimal/near-optimal solutions for single/multiple objectives; then the decision for selecting an optimum solution is left to the decision maker.
3. Interactive approach: In this approach, the algorithm interacts with the decision maker throughout the optimization steps. In each step, good quality solutions are proposed to the decision maker to select the most effective solutions, which will then be used by the algorithm in the next iteration to generate further improved solutions.

For further details and applications of the MOPSP, the reader is referred to the research of Tung et al. [40], Hsu et al. [41], Kacem et al. [42, 43], Loukil et al. [44], Xia & Wu [45], and Fahmy et al. [9].

### 3. PSP solution approach and architecture

#### 3.1. Solution approaches

The approach to be used for solving the PSP is dependent on the problem's conditions and the application's environment. In literature, the solution approaches can be classified under two main categories: *static scheduling* and *dynamic scheduling*.

*Static scheduling* (or *predictive scheduling*) is the process of identifying how and when each activity in schedule should be executed. It involves the generation of a good quality optimized initial (or baseline) schedule. This can be in a *deterministic* approach (like the vast majority of researches within the scheduling context [4]), where the durations of all schedule activities (or activity modes) are initially available; or can be in a *stochastic* project scheduling approach, which aimed to close gaps within the initial available information to enable scheduling under uncertainties. The stochastic activity durations method is a probabilistic modelling approach for scheduling project activities with uncertain durations [46]. The duration of activities is defined in the problem model by random vectors of durations, which are distributed according to a deterministic probability distribution [5].

In practical scheduling, real-time events extremely disrupt schedules integrity, and causes optimized schedules to become neither optimized nor realistic. The *dynamic scheduling* concept was introduced to enable, dynamically, the mitigation of the impacts of real-time events on project schedules. A *dynamic scheduling* solution involves the selection of a scheduling approach, a rescheduling strategy, and a rescheduling policy; or in simple terms, the process

of how to generate the original baseline and the strategy of how to respond to real-time events. For further details on dynamic scheduling, the reader is referred to the surveys of Herroelen and Leus [46], Aytug et al. [47] and Ouelhadj and Petrovic [48].

### 3.2. Solution architectures

The *dynamic scheduling* solution architecture involves several cycles of *static scheduling* to be executed based on certain timing and/or criteria (based on a predefined rescheduling policy), and to involve partial or full schedule optimization (based on a predefined rescheduling strategy). The solution architecture can also involve one optimization level (*single-agent architecture*) or several optimization levels (*multi-level autonomous or mediator architecture*).

Although approaches and architectures of scheduling solutions seem to contain various analysis concepts, the basic outlines of the underlying analysis engine (or the optimization algorithm) remain the same. Whether the optimization process is performed once or at different stages, or the process is executed on single or multiple levels, the core algorithm will be focusing within each optimization cycle on a static schedule snapshot (full or partial). And accordingly, the final section of this study will review optimization algorithms from a generic perspective, regardless of the approach or architecture of the scheduling solution to be integrated with.

## 4. PSP optimization algorithms

### 4.1. Heuristic algorithms

Scheduling is one of the most researched topics within the field of operations research; thus, it will be very difficult to present a survey which covers all heuristic approaches presented for schedules optimization. But in general terms, scheduling heuristic algorithms, or the majority of them, are having a common procedural approach which can be generalized as follows: (1) Initialize an ordered activity list, (2) generate schedule, and (3) improve the schedule quality. And accordingly, these heuristics consist of three main components:

1. A predefined criteria for initial ordering of activities [or a *priority rule*],
2. A process for creating the schedule from the ordered activity list [or a schedule generation scheme (SGS)], and optionally
3. The application of one or more processes to improve the generated schedule's quality.

#### 4.1.1. Priority rules

The basic function of *priority rules* (PRs) is to define an initial arrangement for the activities list in a logical way which will produce a solution with good quality. Kelley [49] introduced the concepts and the first set of PRs; then, several other researches followed Kelly's merits by introducing additional PRs and comparing their results (as surveyed by Kolisch [50]). PRs



provide simple and speedy way to obtain solutions, and that is why they are widely used by commercial scheduling software [51]. **Table 1** lists the most commonly adopted *PRs* within scheduling literature, and their criteria for activities ordering.

Basis	Priority rule	Ordering criteria
Time	Earliest Start Time (EST)	Ascending based on activities earliest start
	Earliest Finish Time (EFT)	Ascending based on activities earliest finish
	Latest Start Time (LST)	Ascending based on activities latest start
	Latest Finish Time (LFT)	Ascending based on activities latest finish
Duration	Shortest Processing Time (SPT)	Ascending based on activities shortest processing mode duration
	Longest Processing Time (LPT)	Ascending based on activities largest processing mode duration
	Greatest Rank Positional Weight (GRPW)	Descending based on the total duration of the activity and all its direct successors
Float	Minimum Slack (MSLK)	Ascending based on activities slack
Resources	Greatest Resource Work Content (GRWC)	Descending based on the total resource requests of the activity
	Greatest Cumulative Resource Work Content (GCRWC )	Descending based on the total resource requests of the activity and all its direct successors
Logic relations	Most Immediate Successors (MIS)	Descending based on the number of their direct successors
	Most Total Successors (MTS)	Descending based on the number of their direct and indirect successors
	Least Non-Related Jobs (LNRJ)	Activities are sorted in ascending order based on the number of activities which are not directly or indirectly inter-related

**Table 1.** Priority rules and related ordering criteria.

4.1.2. *Schedule generation scheme (SGS)*

Ordered activity lists are then passed to a SGS to produce the output schedules. As per the survey presented by Kolisch [52], the first versions of the *serial* (SSGS) and the parallel (PSGS) were presented Kelley [49]; then later Bedworth and Bailey [53] introduced another approach for the PSGS which they titled as the “*Brooks Algorithm*”.

PSGS has been verified that it can only generate non-delay schedules, and the set of non-delay schedules is just a subset of all schedules, hence the SSGS is suggested for RCPSP [54].

#### 4.1.3. Forward-backward scheduling (FBS)

The FBS is one of the most famous schedule improvement techniques. It was proposed by Li and Willis [55], and was found to significantly improve the results. Its procedures involve applying SGS in a forward direction and performing another cycle in reverse order and backward scheduling (reversed precedence network).

#### 4.1.4. Justification schemes

Valls et al. [56] introduced another process for improving schedules quality using a technique they called *justification scheme*. This process involves manipulating the activities positions in the project's time frame without violating the resource constraints, through two cycles, one forward (to the right) and another backward (to the left); which eventually guarantees an overall project duration either shorter or at least the same. Later, Fahmy et al. [57] presented the *stacking justification*, a variation to the original technique; within which the activities selection criteria in each justification cycle was modified in a way to minimize the gaps within resource usage profiles.

### 4.2. Meta-heuristic algorithms

The most common use of Meta-heuristics in PSP solving involves the generation of activities order list which can produce better solutions based on experience gained in previous generation cycles.

Heuristics can solve scheduling problems in short time, but because these procedures cannot adapt dynamically to the problems constraints, so the resulting solutions cannot be guaranteed to be neither optimum nor of good quality.

Due to the overwhelming complexity of scheduling problems within real-life applications, meta-heuristics techniques have been implemented in the development of most practical applications presented in literature during the last two decades.

Various meta-heuristic techniques were adopted in the PSP field, the following sections of this study will exhibit a non-exhaustive review for how each of the commonly adopted meta-heuristic optimization techniques was conceptually implemented within scheduling applications, whilst the details/variations of each of the optimization techniques were considered beyond the scope of this study, and accordingly will only be reviewed to the level needed to support the scheduling algorithms survey perspective.

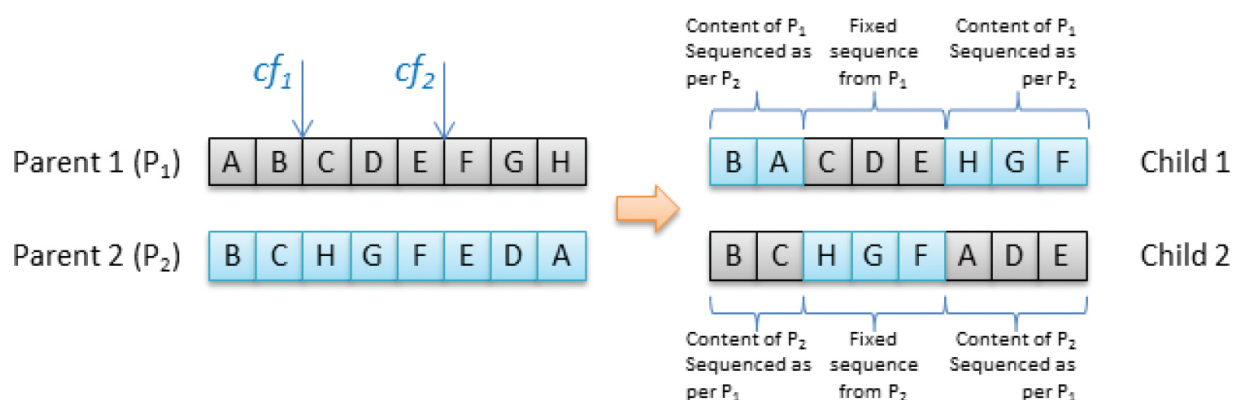
### 4.3. Genetic algorithms

Genetic algorithms (GA) technique is one of the most popular meta-heuristics in the field of scheduling. Holland [58] developed this method as a simplification of evolutionary processes occurring in nature. GA is basically an iterative evolutionary method through which the overall quality of solutions (or *genomes*) population is improved from one generation to the next through three nature resembled mechanisms: *selection*, *crossover*, and *mutation* [59].

In scheduling, the solutions population is selected either randomly or based on a predefined priority rule. The quality of schedules is calculated using an objective function based on the optimization goals, either for a single objective (minimizing time, cost, resource levelling ..., etc.), or combined objectives. Then iteratively, the solutions with higher quality are *selected* for *mating*; and then, using a *crossover* process, a new group of individuals are generated, and added to the best solutions reached previously to form the new generation. Finally, a *mutation* mechanism is applied in each generation to ensure further exploration of solutions-space.

There are several approaches adopted in scheduling literature for the presentation of schedules for GA implementation. The most common approach involves setting the *genomes* as a presentation of the activities priority/sequence list (*S*); where the sequence of activities within the *genome* will be used by the *SGS* to generate the solution's schedule.

For this presentation, the optimization process starts with population generation (randomly or using *PRs*). The *crossover* mechanism will then involve *mating* high fitness solutions, where each pair of *parent* solutions will generate a pair of *child* solutions using a predefined number of *crossing-points* (*cp*) (such as the two-point approach adopted by Shadrokh and Kianfar [60] as shown in **Figure 1**). And finally, a *mutation* mechanism is applied using a predefined probability  $P_{mut}$ ; where a random activity *i* is exchanged in the sequence list (*S*) with another random activity *j* ( $i \neq j$ ), and the same can be performed for the mutation of activity modes list if a multi-mode schedules was used [27].



**Figure 1.** Crossover mechanism for activities priority lists.

Several other priority-based presentations were introduced in literature, most of which were surveyed by Cheng et al. [61, 62]. Few other conceptually different presentations were also available in literature, such as the presentation of Wall [63], where he presented the schedule's *chromosome* as a binary string corresponding to each activity's lag from its normal scheduled position (i.e. a solution scheduled normally based on its precedence will have all its *chromosome* digits set to '0').

According to the detailed performance survey of Kolisch and Hartmann [64], the GA (and its variants/combinations with other techniques) is ranked as the best performing algorithm (as well as most of the top 10 performances) for most problem sizes of the single-mode RCPSP

(SRCPSP). For further review of GA implementations within RCPSP, refer to the publications of Hartmann [65, 66], Alcaraz and Maroto [67], Alcaraz et al. [68] and Valls et al. [56, 69].

#### 4.4. Particle swarm optimization (PSO)

Particle swarm optimization, in comparison with other commonly used optimization techniques, is one of the most recent optimization meta-heuristics. PSO was introduced by Kennedy and Eberhart [70] as a mathematical presentation for the swarming behaviour of flocking birds. PSO is an evolutionary algorithm, where a population of candidate solutions, resembled by particles, and the optimization process occurs by iteratively adjusting the particles' position and velocity within the search space through assessing solutions' quality through predefined measuring criteria.

With a simple mathematical presentation, PSO operates with the following two formulae, where each solution is presented as particle  $i$  with  $n$  number of components,  $V_{ij}^t$  is the velocity of component  $j$  of particle  $i$  in iterations  $t$  (and similarly for  $V_{ij}^{t-1}$  in iteration  $t-1$ );  $X_{ij}^t$  is the positions of component  $j$  of particle  $i$  in iterations  $t$ ; the positions vectors of the best solutions found up to iteration  $t-1$  locally for particle  $i$  and globally in the swarm are stored in  $L_{ij}^{t-1}$  and  $G_j^{t-1}$ , respectively;  $r_1$  and  $r_2$  are two random numbers (from 0 to 1);  $c_1$  and  $c_2$  are two learning coefficients ( $c_1$  defines the influence of the local best solution on the new velocities, while  $c_2$  applies a similar approach for the global best solution).

$$V_{ij}^t = wV_{ij}^{t-1} + r_1c_1(L_{ij}^{t-1} - X_{ij}^{t-1}) + r_2c_2(G_j^{t-1} - X_{ij}^{t-1}) \quad (13)$$

$$X_{ij}^t = X_{ij}^{t-1} + V_{ij}^t \quad (14)$$

Variations of PSO is beyond the scope of this study, but in general most PSO variations focused on improving PSO main drawbacks such as early convergence, parameter dependency, and loss of diversity. The most popular PSO variations are:

- Shi and Eberhart [71] introduced an inertia weight ( $w$ ) to enable the control of iterations velocity influence on succeeding iterations.

$$V_{ij}^t = w \times V_{ij}^{t-1} + c_1 \times r_1 \times (L_{ij}^{t-1} - X_{ij}^{t-1}) + c_2 \times r_2 \times (G_j^{t-1} - X_{ij}^{t-1}) \quad (15)$$

- Bratton and Kennedy [72] presented the 'Standard PSO' in which they introduced the *constriction factor* ( $\gamma$ ) as a multiplier to the equation of velocity.

$$V_{ij}^t = \gamma \times (V_{ij}^{t-1} + c_1 \times r_1 \times (L_{ij}^{t-1} - X_{ij}^{t-1}) + c_2 \times r_2 \times (G_j^{t-1} - X_{ij}^{t-1})) \quad (16)$$

In the scheduling field, PSO is conceptually implemented by resembling each schedule (or solution) as a particle, with the activities' priorities as the components of this particle [73, 74]. The quality of schedules (or solutions' fitness) is calculated via a predefined objective function. Initially, activities priorities can be initialized using a single priority rule, or a combined rule [57]. Then, iteratively, positions and velocities vectors of particles/components are adjusted, then the quality rating of each solution is assessed, the global best solution and the local best solution for each particle are logged. The stopping position can be set to a maximum analysis duration, a certain number of schedules to be generated, or a specific quality to be reached (Figure 2).

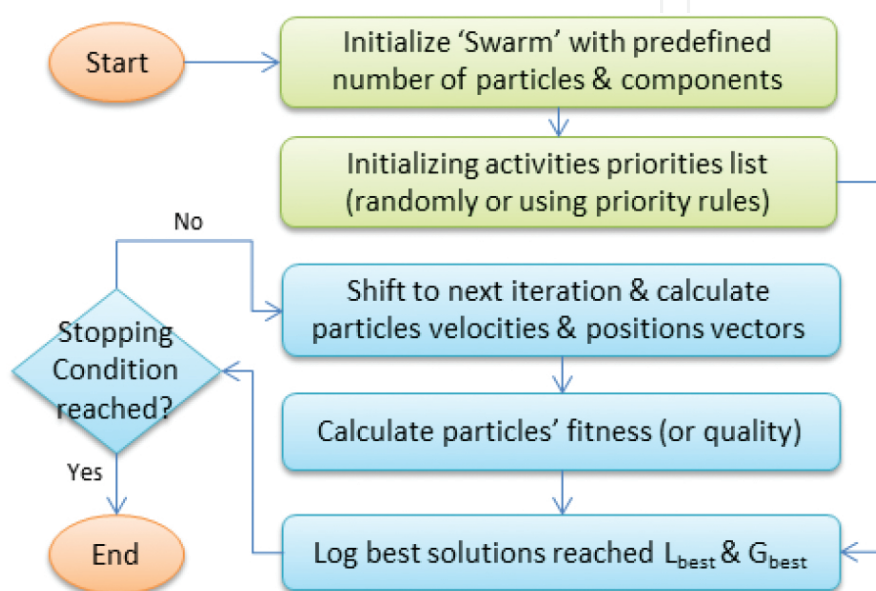


Figure 2. General PSO flow chart for scheduling.

Albeit that the number of application of PSO in the scheduling context is not large as GA, the results of its application, especially for SRCPSP, is highly ranked in general comparisons with all techniques (*c.f.* [54, 57]).

#### 4.5. Ant colony optimization (ACO)

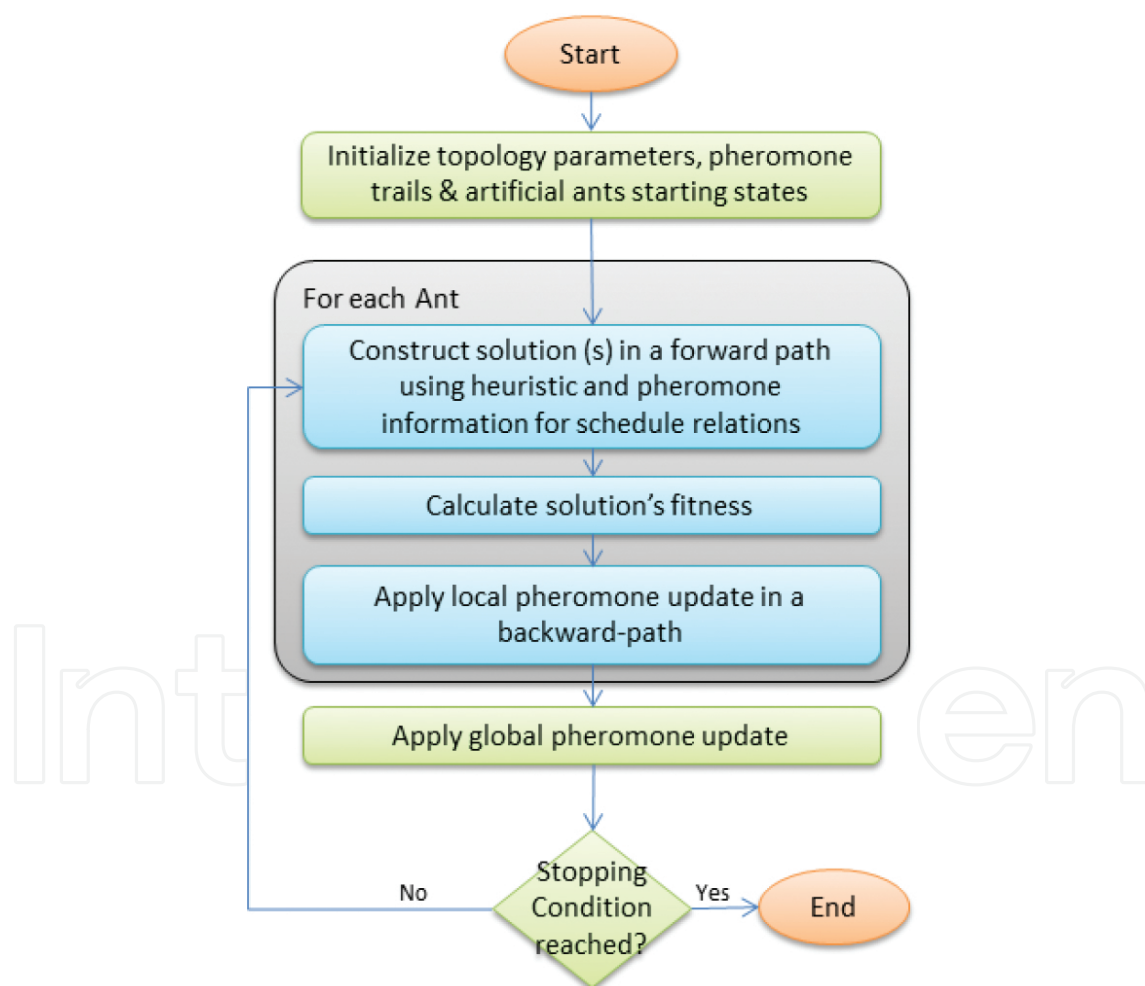
ACO is a population based, multi-agent, meta-heuristic technique within the broader family of *swarm intelligence* optimization methods; initially proposed by Dorigo [75]. ACO uses the concepts of food seeking in ant colonies as a basis for an optimization algorithm for seeking optimal solutions within solutions space.

In nature, ants use *pheromone* as the means of communication when searching for food. Upon finding food, after random wandering in the colony's neighbourhood, ants lay *pheromone* in their way back to the colony; thus, other ants can use these traces to guide their movements from colony to food sources. These *pheromone* traces decays/evaporates with time, which leads to traces for shorter paths to be higher in strength than others, and accordingly guides ants to shorter paths to food.



Application of ACO in scheduling started with a proposed ant system for job-shop scheduling by Colorni et al. [76]; followed by several other applications in various scheduling problems, such as flow-shop scheduling [77–79], flexible job-shop scheduling [80], resource-constrained scheduling [81, 82], and total tardiness problems [83, 84].

The most common presentation of ACO in the scheduling literature was outlined by *Stützle and Dorigo* [85] as follows: set of network nodes ( $C=\{c_1, c_2, c_3, \dots, c_N\}$ ), set of problem states (sequence or relationships) of element  $C$  ( $x=\{c_i, c_j, c_k, \dots\}$ ), set of constraints ( $\Omega$ ), set of all states ( $X$ ), set of all feasible states ( $\bar{X} \subseteq X$ ), objective function  $f(s)$  (where  $s$  is a candidate solution,  $s \in S$ ), set of all feasible solutions ( $S^* \subseteq \bar{X}$ ), a pheromone trail ( $\tau_{ij}$ ) representing the desirability of relation ( $r_{ij}$ ), and finally heuristic problem-specific information can be defined within ( $\eta_{ij}$ ).



**Figure 3.** Simplified ACO flow chart for scheduling.

Then, the problem topology and the simulation behaviour can be simplified as shown in the flow chart in **Figure 3**. The algorithm starts with initializing the problem's topology (or the



network's details), the pheromone trails ( $\tau_0$ ) either randomly or using a priority rule, and the artificial ants starting states.

Then, within each iteration cycle, each ant moves a two-way path seeking a food source (or a feasible solution). In the forward path, the ant's movements (or selected network relations) is defined using heuristic information ( $\eta$ ) and current pheromone information ( $\tau$ ). After solution ( $s$ ) is constructed, the solution's fitness is calculated using the objective function ( $f$ ). The ant then returns in a backward path applying local pheromone based on the quality achieved for solution ( $s$ ). Finally, the global pheromone is updated before the stopping condition is checked.

Although, as per the detailed performance surveys mentioned earlier, ACO did not demonstrate a competitive performance to other meta-heuristics (GA, PSO, and TS), except for large size problems; this can be positively inferred that ACO's performance increases with the increase in problem's size due to its high exploration capabilities.

#### 4.6. Other methods

Due to the complicated and challenging nature of scheduling problems, the scheduling field has been usually one of the first testing fields for any meta-heuristic optimization technique being introduced within the *operational research* literature. It would be a very exhaustive task to summarize all meta-heuristics adopted for solving different scheduling problem types; so, beside the techniques mentioned in the previous sections, the following paragraphs briefly summarize other meta-heuristics widely adopted in the scheduling research context.

Tabu search (TS) is one of the techniques with high performance results within the scheduling literature. The TS is a local search technique, initially proposed by Glover [86]. It involves exploring the search space through searching neighbourhoods of potential solution(s). The following are some examples of TS applications in scheduling: resource-constrained scheduling [87–89], flow-shop scheduling [90, 91], and flexible job-shop scheduling [92].

The simulated annealing (SA) was presented by Kirkpatrick et al. [93], as an optimization method which mimics the behaviour of a system in thermodynamic equilibrium at certain temperature. It is a probabilistic meta-heuristic which focus on finding a quick approximated global optimum of a search space. SA showed moderate performance with scheduling applications, such as Rutenbar [94], Bouleimen and Lecocq [95], and Dai et al. [96].

Other meta-heuristics adopted in scheduling include, Neural networks [97–99], Scatter Search [100], Electromagnetism-like method [101, 102], Sampling method [103, 104], and Bees Algorithm [105, 106].

In addition, scheduling literature contains vast amount of meta-heuristics with combinations of the various optimization techniques, such as Kochetov and Stolyar [107] for GA and TS, Niu et al. [108] for PSO and GA, Chen and Shahandashti [109] for GA and SA, Moslehi and Mahnam [110] for PSO and LS, and Deane [111] for GA and NN.

## 5. Conclusion

The design and implementation of a robust scheduling system are essential for the successful use of planning and scheduling practices within projects. A scheduling system involves modelling the problem, selecting a solution approach to be used in a static and/or dynamic analysis for optimizing schedules, and finally the selection of an optimization technique which suits most the characteristics and conditions of the project type under analysis.

This study reviewed the concepts and researches presented for these three factors of building a scheduling system, with a more detailed focus on meta-heuristic optimization algorithms adopted in the project-scheduling context.

## Author details

Amer M. Fahmy

Address all correspondence to: [amer.fahmy@dynamicsscheduling.net](mailto:amer.fahmy@dynamicsscheduling.net)

CCC/TAV JV, Development of Muscat International Airport Project, Muscat, Oman

## References

- [1] Blazewicz, J., Lenstra, J.K., Rinnooy Kan, A. H. G. (1983). "Scheduling subject to resource constraints: Classification and complexity". *Discrete Applied Mathematics* 5, 11-24.
- [2] Icmeli, O., Erenguc, S.S., & Zappe, C.J. (1993). "Project scheduling problems: A survey". *International Journal of Operations & Production Management* 13 (11), 80-91.
- [3] Elmaghraby, S.E. (1995). "Activity nets: A guided tour through some recent developments". *European Journal of Operational Research* 82, 383-408.
- [4] Herroelen, W., Reyck B.D., & Demeulemeester E. (1998). "Resource-constrained project scheduling: A survey of recent developments". *Computers and Operations Research* 25(4), 279-302.
- [5] Brucker, P., Drexl, A., Mohring, R., Neumann, K., & Pesch, E. (1999). "Resource-constrained project scheduling: Notation, classification, models, and methods", *European Journal of Operational Research* 112(1), 3-41.
- [6] Kolisch, R., & Padman, R. (2001). "An integrated survey of deterministic project scheduling". *Omega, the International Journal of Management Science* 29, 249-272.
- [7] Neumann, K., Schwindt[p3], C., & Zimmermann, J. (2003). "Project Scheduling with Time Windows & Scarce Resources". Springer.

- [8] Hartmann, S., & Briskorn, D. (2010). "A survey of variants and extensions of the resource-constrained project scheduling problem". *European Journal of Operational Research* 207(1), 1-14.
- [9] Fahmy, A.M., Hassan, T.M., & Bassioni, H. (2014). "A Dynamic Scheduling Model for Construction Enterprises". PhD thesis, School of Civil & Building Engineering, Loughborough University, UK.
- [10] Demeulemeester, E., & Herroelen, W. (1996). "Modelling setup times, process batches and transfer batches using activity network logic". *European Journal of Operational Research* 89, 355-365.
- [11] Klein, R., & Scholl, A. (1999). "Progress: Optimally solving the generalized resource constrained project scheduling problem". *Mathematical Methods of Operations Research* 52(3), 467-488.
- [12] Chassiakos, A., Sakellaropoulos, S. (2005). "Time-cost optimization of construction projects with generalized activity constraints". *Journal of Construction Engineering and Management* 131, 1115-1124.
- [13] Vanhoucke, M. (2006). "Work continuity constraints in project scheduling". *Journal of Construction Engineering and Management* 132, 14-25.
- [14] Demeulemeester, E., & Herroelen, W. (1996). "An efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem". *European Journal of Operational Research* 90(2), 334-348.
- [15] Brucker, P., Knust, S. (2001). "Resource-constrained project scheduling and timetabling". *Lecture Notes in Computer Science* 2079, 277-293.
- [16] Ballestín, F., Valls, V., & Quintanilla, S. (2008). "Pre-emption in resource-constrained project scheduling". *European Journal of Operational Research* 189(3), 1136-1152.
- [17] Franck, B., Neumann, K., & Schwindt, C. (2001). "Project scheduling with calendars". *OR Spectrum*, 23(3), 325-334.
- [18] Lamothe, J., Marmier, F., Dupuy, M., Gaborit, P., & Dupont, L. (2016). "Scheduling rules to minimize total tardiness in a parallel machine problem with setup and calendar constraints". *Computers & Operations Research* 39 (6), 1236-1244.
- [19] Kreter, S., Rieck, J., & Zimmermann, J. (2016). "Models and solution procedures for the resource-constrained project scheduling problem with general temporal constraints and calendars". *European Journal of Operational Research* 251 (2), 387-403.
- [20] Elmaghraby, S. (1977). "Activity networks: Project Planning and Control by Network Models". Wiley, New York.
- [21] Kolisch, R., & Drexel, A. (1997). "Local search for non-preemptive multi-mode resource constrained project scheduling". *IIE Transactions* 29, 987-999.

- [22] Hartmann, S. (2001). *"Project scheduling with multiple modes: A genetic algorithm"*. Annals of Operations Research 102, 111-135.
- [23] Alcaraz, J., Maroto, C., & Ruiz, R. (2003). *"Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms"*. Journal of the Operational Research Society 54, 614-626.
- [24] Neumann, K., & Zimmermann, J. (2002). *"Project scheduling with inventory constraints"*. Mathematical Methods of Operations Research 56, 513-533.
- [25] Deckro, R., Hebert, J., Verdini, W., Grimsrud, P., & Venkateshwar, S. (1995). *"Nonlinear time/cost tradeoff models in project management"*. Computers and Industrial Engineering 28 (2), 219-229.
- [26] Demeulemeester, E., de Reyck, B., & Herroelen, W. (2000). *"The discrete time/resource trade-off problem in project networks – A branch-and-bound approach"*. IIE Transactions 32, 1059-1069.
- [27] Ranjbar, M., & Kianfar, F. (2007). *"Solving the discrete time/resource trade-off problem in project scheduling with genetic algorithms"*. Applied Mathematics and Computation 191(2), 2007 451-456.
- [28] Demeulemeester, E., de Reyck, B., Foubert, B., Herroelen, W., & Vanhoucke, M. (1998). *"New computational results on the discrete time/cost trade-off problem in project networks"*. Journal of the Operational Research Society 49, 614-626.
- [29] Ranjbar, M., Kianfar, F., & Shadrokh, S. (2008). *"Solving the resource availability cost problem in project scheduling by path relinking and genetic algorithm"*. Applied Mathematics and Computation 196(2), 879-888.
- [30] Kimms, A. (2001). *"Maximizing the net present value of a project under resource constraints using a lagrangian relaxation based heuristic with tight upper bounds"*. Annals of Operations Research 102 (1-4), 221-236.
- [31] Mika, M., Waligóra, G., & Weglvarz, J. (2005). *"Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models"*, European Journal of Operational Research 164 (3), 639-668.
- [32] Padman, R., & Zhu, D. (2006). *"Knowledge integration using problem spaces: A study in resource-constrained project scheduling"*. Journal of Scheduling 9 (2), 133-152.
- [33] Icmeli-Tukel, O., & Rom, W.O. (1997). *"Ensuring quality in resource constrained project scheduling"*. European Journal of Operational Research 103 (3), 483-496.
- [34] Mehta, S. V., & Uzsoy, R. (1999). *"Predictable scheduling of a single machine subject to breakdowns"*. International Journal of Computer Integrated Manufacturing 12(1), 15-38.
- [35] Vieira, G.E., Hermann, J.W., & Lin, E. (2003). *"Rescheduling manufacturing systems: a framework of strategies, policies and methods"*. Journal of Scheduling 6(1), 36-92.

- [36] Frederickson, G. N., & Solis-Oba, R. (2006). *"Efficient algorithms for robustness in resource allocation and scheduling problems"*. Theoretical Computer Science 352(1-3), 250-265.
- [37] Hwang, C., & Masud[p4], A. (1979). *"Multi-objective decision making, methods and applications. A state of the art survey"*. Lecture notes in economics and mathematical systems, Springer-Verlag.
- [38] Slowinski, R. (1981). *"Multi-objective network scheduling with efficient use of renewable and nonrenewable resources"*. European Journal of Operational Research 7, 265-273.
- [39] Ballestín, F., & Blanco, R. (2011). *"Theoretical and practical fundamentals for multi-objective optimisation in resource-constrained project scheduling problems"*. Computers & Operations Research 38, 51-62.
- [40] Tung, L., Li, L., & Nagi, R. (1999). *"Multi-objective scheduling for the hierarchical control of flexible manufacturing systems"*. The International Journal of Flexible Manufacturing Systems, 11, 379-409.
- [41] Hsu, T., Dupas, R., & Jolly, D., Goncalves, G. (2002). *"Evaluation of mutation heuristics for the solving of multiobjective flexible job shop by an evolutionary algorithm"*. In: Proceedings of the 2002 IEEE international conference on systems, man and cybernetics (pp. 6-9).
- [42] Kacem, I., Hammadi, S., & Borne, P. (2002). *"Approach by localization and multi-objective evolutionary optimization for flexible job-shop scheduling problems"*. IEEE Transactions on Systems, Man, and Cybernetics, Part C, 32(1), 1-13.
- [43] Kacem, I., Hammadi, S., & Borne, P.,(2002). *"Pareto-optimality approach for flexible job-shop scheduling problems: Hybridization of evolutionary algorithms and fuzzy logic"*. Mathematics and Computers in Simulation 60, 245-276.
- [44] Loukil, T., Teghem, J., & Tuytens, D. (2005). *"Solving multi-objective production scheduling problems using metaheuristics"*. European Journal of Operational Research 161, 42-61.
- [45] Xia, W., & Wu, Z. (2005). *"An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems"*. Computers & Industrial Engineering 48, 409-425.
- [46] Herroelen, W., & Leus, R. (2005). *"Project scheduling under uncertainty: Survey and research potentials"*. European Journal of Operational Research 165(2), 289-306.
- [47] Aytug, H., Lawley, M.A., McKay, K., Mohan, S. & Uzsoy, R. (2005). *"Executing production schedules in the face of uncertainties: A review and some future directions"*. European Journal of Operational Research, 161(1), 86-110.
- [48] Ouelhadj, D., & Petrovic, S. (2009). *"A survey of dynamic scheduling in manufacturing systems"*. Journal of Scheduling 12, 417-431.



- [49] Kelley, J.E., Jr. (1963). *"The critical-path method: Resources planning and scheduling"*. In: J.F. Muth and G.L. Thompson (Eds.). Industrial Scheduling, Prentice-Hall, Englewood Cliffs. NJ, 347-365.
- [50] Kolisch, R. (1996). *"Efficient priority rules for the resource-constrained project scheduling problem"*. Journal of Operations Management 14 (3), 179-192.
- [51] Herroelen, W.S. (2005). *"Project scheduling-theory and practice"*. Production and Operations Management 14(4), 413-432.
- [52] Kolisch, R. (1996). *"Serial and parallel resource-constrained project scheduling methods revisited: theory and computation"*. European Journal of Operational Research 90 (2), 320-333.
- [53] Bedworth, D.D., & Bailey, J.E. (1982), *"Integrated Production Control Systems - Management, Analysis, Design"*. Wiley, New York.
- [54] Chen, R.M. (2011). *"Particle swarm optimization with justification and designed mechanisms for resource-constrained project scheduling problem"*. Expert Systems with Applications 38, 7102-7111.
- [55] Li, K. Y., & Willis, R. J. (1992). *"An iterative scheduling technique for resource constrained project scheduling"*. European Journal of Operational Research 56, 370-379.
- [56] Valls, V., Ballestin, F., & Quintanilla, M.S. (2005). *"Justification and RCPSP: A technique that pays"*. European Journal of Operational Research 165, 375-386.
- [57] Fahmy, A.M., Hassan, T.M., & Bassioni, H. (2014). *"Improving RCPSP solutions quality with Stacking Justification – Application with particle swarm optimization"*. Expert Systems with Applications 41(1), 5870-5881.
- [58] Holland, H.J. (1975). *"Adaptation in Natural and Artificial Systems"*. The University of Michigan Press, Ann Arbor.
- [59] Goldberg, David (1989). *"Genetic Algorithms in Search, Optimization and Machine Learning"*. Addison-Wesley Professional, Boston, MA.
- [60] Shadrokh, S., & Kianfar, F. (2007). *"A genetic algorithm for resource investment project scheduling problem, tardiness permitted with penalty"*. European Journal of Operational Research 181, 86-101.
- [61] Cheng, R., Gen, M., & Tsujimura Y. (1996). *"A tutorial survey of job-shop scheduling problems using genetic algorithms, part II: hybrid genetic search strategies"*. Computers & Industrial Engineering 30 (4), 983-997.
- [62] Cheng, R., Gen, M., & Tsujimura Y. (1999). *"A tutorial survey of job-shop scheduling problems using genetic algorithms, part II: hybrid genetic search strategies"*. Computers & Industrial Engineering 36 (2), 343-364.



- [63] Wall, M. B. (1996). "A Genetic Algorithm for Resource-Constrained Scheduling". PhD thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, USA.
- [64] Kolisch, R., & Hartmann, S. (2006). "Experimental investigation of heuristics for resource-constrained project scheduling: An update". *European Journal of Operational Research* 174, 23-37.
- [65] Hartmann, S. (1998). "A competitive genetic algorithm for resource-constrained project scheduling". *Naval Research Logistics* 45, 733-750.
- [66] Hartmann, S. (2002). "A self-adapting genetic algorithm for project scheduling under resource constraints". *Naval research Logistics* 49, 433-448.
- [67] Alcaraz, J., & Maroto, C. (2001). "A robust genetic algorithm for resource allocation in project scheduling". *Annals of Operations Research* 102, 83-109.
- [68] Alcaraz, J., Maroto, C., & Ruiz, R. (2004). "Improving the performance of genetic algorithms for the RCPS problem". In: *Proceedings of the Ninth International Workshop on Project Management and Scheduling*, Nancy, 40-43.
- [69] Valls, V., Ballestin, F., & Quintanilla, M.S. (2008). "A hybrid genetic algorithm for the resource-constrained project scheduling problem". *European Journal of Operational Research* 185(2), 495-508.
- [70] Kennedy, J., & Eberhart, R. (1995). "Particle swarm optimization". *Proceedings of the 1995 IEEE international conference on neural networks*, 4, 1942-1948.
- [71] Shi, Y., & Eberhart, R.C. (1998). "A modified particle swarm optimizer". *Proceedings of IEEE Congress on Evolutionary Computation* 1998, 69-73.
- [72] Bratton, D., & Kennedy, J. (2007). "Defining a standard for particle swarm optimization". *Proceedings of IEEE swarm intelligence symposium, SIS 2007*, 120-127.
- [73] Zhang, C., Sun, J., Zhu, X., & Yang, Q. (2008). "An improved particle swarm optimization algorithm for flowshop scheduling problem". *Information Processing Letters* 108(4), 204-209.
- [74] Zhang, H., Li, H., & Tam, C. M. (2005). "Particle swarm optimization-based schemes for resource-constrained project scheduling". *Automation in Construction* 14, 393-404.
- [75] Dorigo, M. (1992). "Optimization, Learning and Natural Algorithms". PhD thesis, Politecnico di Milano, Italy.
- [76] Colorni, A., Dorigo, M., Maniezzo, V., & Trubian, M. (1994). "Ant system for job-shop scheduling". *Belgian Journal of Operations Research, Statistics and Computer Science* 34 (1), 39-53.
- [77] Stützle, T., & Hoos, H.H. (1997). "The max-min ant system and local search for the traveling salesman problem". *Proceedings of ICEC&97*, 309-314.

- [78] Stützle, T., & Hoos, H.H. (2000). "MAX MIN Ant System". *Future Generation Computer Systems* 16, 889-914.
- [79] Shyu, S.J., Lin, B.M.T., & Hsiao, T.S. (2004). "Ant colony optimization algorithm for the cell assignment problem in PCS networks". *Computers & Operations Research* 33 (6), 1713-1740.
- [80] Xing, L.N., Chen, Y.W., Wang, P., Zhao, Q.S., & Xiong, J. (2010). "A knowledge-based ant colony optimization for flexible job shop scheduling problems". *Applied soft Computing*, 10 (3), 888-896.
- [81] Merkle, D., Middendorf, M., & Schmeck, H. (2002). "Ant colony optimization for resource-constrained project scheduling". *IEEE Transactions on Evolutionary Computation* 6, 333-346.
- [82] Lo, S. T., Chen, R. M., Huang, Y. M., & Wu, C. L. (2008). "Multiprocessor system scheduling with precedence and resource constraints using an enhanced ant colony system". *Expert Systems with Applications* 34(3), 2071-2081.
- [83] Bauer, A., Bullnhermer, B., Hartl, R.F., & Strauss, C. (1999). "An ant colony optimization approach for the single machine total tardiness problem". *Proceedings of the 1999 Congress on Evolutionary Computation*. Washington, DC, USA, 1445-1450.
- [84] Merkle, D., & Middendorf, M. (2003). "An ant algorithm with a new pheromone evaluation rule for total tardiness problems". *Applied Intelligence* 18 (1), 105-111.
- [85] Stützle, T., & Dorigo, M. (2002). "The Ant Colony Optimization metaheuristic: algorithms, applications, and advances". In: F. Glover, G. Kochenberger (Eds.), *Handbook of Metaheuristics*, Kluwer Academic Publishers, Norwell, MA.
- [86] Glover, F. (1986). "Future Paths for Integer Programming and Links to Artificial Intelligence". *Computers and Operations Research*, 13 (5), 533-549.
- [87] Thomas, P. R., & Salhi, S. (1998). "A tabu search approach for the resource constrained project scheduling problem". *Journal of Heuristics*, 4(2), 123-139.
- [88] Baar, T., Brucker, P., & Knust, S. (1998). "Tabu-search algorithms and lower bounds for the resource-constrained project scheduling problem". In: S. Voss, S. Martello, I. Osman, C. Roucairol (Eds.), *Meta-heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers, Dordrecht, 1-8.
- [89] Nonobe, K., & Ibaraki, T. (2002). "Formulation and Tabu search algorithm for the resource constrained project scheduling problem (RCPSP)". In: Ribeiro, C.C., Hansen, P. (Eds.), *Essays and Surveys in Metaheuristics. Operations Research/Computer Science Interfaces Series*, Kluwer Academic Publishers 15, 557-588
- [90] Ben-Daya, M., & Al-Fawzan, M. (1998). "A tabu search approach for the flow shop scheduling problem". *European Journal of Operational Research*, 106 (2-3), 226-253.

- [91] Wojciech, B., Pempera, J., & Smutnicki, C. (2013). "Parallel tabu search algorithm for the hybrid flow shop problem". *Computers & Industrial Engineering* 65 (3), 466-474.
- [92] Jia, S., & Hu, Z.H. (2014). "Path-relinking Tabu search for the multi-objective flexible job shop scheduling problem". *Computers & Operations Research* 47, 11-26.
- [93] Kirkpatrick, S, Gelatt, C. D, & Vecchi, M. P. (1983). "Optimization by Simulated Annealing". *Science*. 220(4598), 671-680.
- [94] Rutenbar, R. A. (1989). "Simulated annealing algorithms: An overview". *Circuits and Devices Magazine, IEEE* 5, 19-26.
- [95] Bouleimen, K., & Lecocq, H. (2003). "A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode versions". *European Journal of Operational Research* 140(2), 268-281.
- [96] Dai, M., Tang, D., Giret, A., Salido, M.A., & Li, W.D. (2013). "Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm". *Robotics and Computer-Integrated Manufacturing* 29 (5), 418-429.
- [97] Foo, S. Y, & Takefuji, Y. (1988). "Stochastic neural networks for solving job-shop scheduling: Part 1". In: *IEEE International Conference on Neural Networks 1988, San Diego, CA, USA*, 275-282.
- [98] Cedimoglu, I. H. (1993) "Neural networks in shop floor scheduling". Ph.D. Thesis, School of Industrial and Manufacturing Science, Cranfield University, UK.
- [99] Sim, S. K, Yeo, K. T, & Lee, W. H. (1994). "An expert neural network system for dynamic job-shop scheduling". *International Journal of Production Research* 32(8), 1759-1773.
- [100] Debels, D., De Reyck, B., Leus, R., & Vanhoucke, M. (2006). "A hybrid scatter search/electromagnetism meta-heuristic for project scheduling". *European Journal of Operational Research* 169, 638-653.
- [101] Chang, P.C., Chen, S.H., & Fan, C.Y. (2009). "A hybrid electromagnetism-like algorithm for single machine scheduling problem". *Expert Systems with Applications* 36 (2-1), 1259-1267.
- [102] Khalili, M., & Tavakkoli-Moghaddam, R. (2012). "A multi-objective electromagnetism algorithm for a bi-objective flowshop scheduling problem". *Journal of Manufacturing Systems* 31 (2), 232-239.
- [103] Tormos, P., & Lova, A. (2001). "A competitive heuristic solution technique for resource-constrained project scheduling". *Annals of Operations Research* 102, 65-81.
- [104] Tormos, P., & Lova, A. (2003). "An efficient multi-pass heuristic for project scheduling with constrained resources". *International Journal of Production Research* 41(5), 1071-1086.
- [105] Low, C.S., Sivakumar, M,A, & Gay, K.L. (2007). "Using a Bee Colony Algorithm for Neighborhood Search in Job Shop Scheduling Problems". In: *21st European Conference on Modelling and Simulation ECMS*.

- [106] Wong, L.P., Puan, C.Y., Low, M. Y., & Chong C. S. (2008). *"Bee Colony Optimization algorithm with Big Valley landscape exploitation for Job Shop Scheduling problems"*. In: Simulation Conference, 2008, WSC.
- [107] Kochetov, Y., & Stolyar, A. (2003). *"Evolutionary local search with variable neighborhood for the resource constrained project scheduling problem"*. In: Proceedings of the 3rd International Workshop of Computer Science and Information Technologies, Russia.
- [108] Niu, Q., Jiao, B., & Gu, X. (2008). *"Particle swarm optimization combined with genetic operators for job shop scheduling problem with fuzzy processing time"*. Applied Mathematics and Computation 205 (1), 148-158.
- [109] Chen, P.H., & Shahandashti, S. M. (2009). *"Hybrid of genetic algorithm and simulated annealing for multiple project scheduling with multiple resource constraints"*. Automation in Construction 18 (4), 434-443.
- [110] Moslehi, G., & Mahnam, M. (2011). *"A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search"*. International Journal of Production Economics, 129 (1), 14-22.
- [111] Deane, J. (2012). *"Hybrid genetic algorithm and augmented neural network application for solving the online advertisement scheduling problem with contextual targeting"*. Expert Systems with Applications 39 (5), 5168-5177.

