

Article

Comparison of Heuristic Priority Rules in the Solution of the Resource-Constrained Project Scheduling Problem

Osman Hürol Türkakin ^{1,*} , David Arditi ²  and Ekrem Manisalı ³¹ Department of Civil Engineering Avcılar, Istanbul University-Cerrahpaşa, Istanbul 34320, Turkey² Department of Civil, Architectural, and Environmental Engineering, Illinois Institute of Technology, Chicago, IL 60616, USA; arditi@iit.edu³ Department of Civil Engineering, Nişantaşı University, Istanbul 34398, Turkey; ekrem.manisali@nisantasi.edu.tr

* Correspondence: turkakin@iuc.edu.tr; Tel.: +90-(212)-473-70-70

Featured Application: This study compares the existing heuristic rules using small/large networks, different resource types, relaxed/limited resource supply, and serial/parallel resource allocation procedures.

Abstract: Resource-constrained project scheduling (RCPS) aims to minimize project duration under limited resource availabilities. The heuristic methods that are often used to solve the RCPS problem make use of different priority rules. The comparative merits of different priority rules have not been discussed in the literature in sufficient detail. This study is a response to this research gap. It compares 17 heuristic priority rules and seeks the best performing heuristic priority rule. This is the first study ever that compares heuristic priority rules by considering combinations of variations in (1) resource allocation procedures, (2) number of activities, (3) number of resource constraints, and (4) resource supply levels. The objective is to understand the relative merits of heuristic rules used in solving the RCPS problem. The findings indicate that the “minimum late finish time” rule generates the shortest predicted project duration when used in parallel resource allocation, whereas the “minimum late start time”, “minimum late finish time”, and the “highest rank of positional weight 2” rules perform best in serial resource allocation. It was also found that parallel resource allocation is slightly superior to serial resource allocation in most instances.

Keywords: resource constrained project scheduling; heuristic methods; heuristic priority rules



Citation: Türkakin, O.H.; Arditi, D.; Manisalı, E. Comparison of Heuristic Priority Rules in the Solution of the Resource-Constrained Project Scheduling Problem. *Sustainability* **2021**, *13*, 9956. <https://doi.org/10.3390/su13179956>

Academic Editor: Amir Mosavi

Received: 5 August 2021

Accepted: 1 September 2021

Published: 5 September 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Work scheduling involves a set of decision-making processes that are modeled after a production system. Various types of scheduling methods were developed in the last century. Out of these, CPM is one of the most popular scheduling techniques. A work schedule includes several activities that need to be executed in a certain order and involves several resources that are used for the execution of these activities. Renewable resources such as construction equipment remain intact after production, whereas non-renewable resources such as construction materials are consumed and depleted during production.

Resource-constrained project scheduling (RCPS) aims to achieve minimum project duration when the work schedule is composed of a set of activities that are subject to not only precedence constraints but also to resources with limited availability [1,2]. In classical CPM, the main constraint in determining project duration is precedence relationships. However, resource constraints may cause delays in activities, which in turn may affect the project duration predicted by CPM. One of the outcomes of RCPS is activity start and finish times that conform to precedence relationships and satisfy resource constraints. RCPS techniques were used in various fields such as prefabricated construction [3], optimizing

time, cost and energy consumption [4], improving sustainability in construction [5], and financial optimization [6].

Several methods have been used to solve the RCPS problem. Heuristic methods are the most frequently used methods, especially in commercially available project management software packages. Even though different researchers have used different heuristic priority rules over the years, a detailed comparison including different resource conditions of these rules has never been made. In this study, a detailed comparison is presented after a brief literature review of RCPS problems and solutions. The results are tested by using cases generated by the Project Scheduling Problem Library (PSPLIB) developed by Kolisch and Sprecher [7]. These cases are modified by defining extra resource conditions to test the heuristic rules under varied conditions. The outcome is expected to bring clarity to the selection and use of the best performing heuristic rules in given situations, eliminating any doubts that existed in the past.

After the introduction to the study in Section 1, the organization of the paper involves Section 2—Materials and Methods, which includes a literature review of RCPS research and well as heuristics research (Section 2.1), followed by the mathematical formulation of the RCPS problem (Section 2.2), and a detailed overview of heuristics methods that includes not only the serial and parallel resource allocation methods but also the many heuristic priority rules (Section 2.3). Heuristic priority rules are compared using both serial and parallel resource allocation by solving single mode benchmark problems and the results are presented and discussed in Section 3—Results and Discussion. The conclusions are presented in Section 4.

2. Materials and Methods

2.1. Literature Review

Mathematical methods or exact algorithms constitute the most precise approaches for solving RCPS problems. These approaches are deterministic and exhaustive in nature. Mathematical methods were generally used in the earlier days of RCPS research. For example, the branch and bound method was used by Johnson [8] as early as in 1967. Branch and bound methods are still used for different kinds of RCPS problems (e.g., [9]). Mathematical methods are accurate, but in some cases fail to generate results. In addition, mathematical methods are not useful when tackling large problems. In such situations, the use of heuristic methods is recommended.

Heuristic and meta-heuristic methods do not depend on deterministic calculations and are faster than mathematical methods. Heuristic methods are problem-specific (i.e., they are designed to solve a particular problem and may not perform as well in other problems), whereas meta-heuristic methods are problem-independent (i.e., a meta-heuristic method such as tabu-search or genetic algorithms can be used to solve a variety of problems) [4]. Heuristic methods include specific rules that solve RCPS problems whereas meta-heuristic methods include computational procedures that gradually converge to the optimal solution. Neither heuristic nor meta-heuristic methods guarantee optimum results [10].

Various heuristic methods have been used to solve the RCPS problem. For example, lower bound methods can produce heuristic solutions to the RCPS problem [2]. On the other hand, meta-heuristic methods involve adaptive search algorithms [11], as they improve the solution gradually through several iterations with the hope of reaching the global optimum, but sometimes reaching only a local optimum that falls short of the global optimum. In contrast to other methods, heuristic and meta-heuristic methods involve randomness in calculations.

Even though heuristic and meta-heuristic methods do not guarantee an exact optimal solution, the results converge faster than mathematical methods. With the increased power and wider use of personal computers, meta-heuristic methods became popular and more frequently used to solve RCPS problems. The main advantage of heuristic and meta-heuristic methods is converging to a “reasonable” optimum solution within acceptable limits faster than deterministic methods, especially in solving large problems.

The main disadvantage is that the user may not always obtain the optimum solution. Since the early 1990s, genetic algorithms have been extensively used for solving RCPS problems (GA) [3,12–14]. Other meta-heuristic methods such as tabu-search [15], simulated annealing [16], ant colony algorithm [17] and the artificial bee colony with differential evolution [18] were applied. Different problem types such as multi-project cases can be solved using forward-backward genetic algorithm techniques [19]. Kannimuthu et al. [20] used Probabilistic Global Search Lausanne for multi-project multi-mode problem cases. Different techniques such as Entropy-based heuristics that are adapted for solution of RCPS problems, the main objective of entropy-based heuristics, is maximizing the total-entropy of the schedule [21,22].

Heuristic priority rules are the most common algorithms in solving RCPS problems [23] because of their simplicity and their high computation speed [24]. Heuristic priority rules reflect an intuitive background rather than optimality and are preferred in small size manufacturing. Heuristic priority rules are used for sorting activities and alleviating resource conflicts in the RCPS problem. The first heuristic rules were introduced by Kelley [25] in the early 1960s and later discussed by Davis and Patterson [26]. Tormos and Lova [27] introduced the parallel scheduling scheme for multi-project situations, and Klein [2] experimented with bi-directional planning in solving the RCPS problem. Buddhakulsomsiri and Kim [28] attempted to solve the multi-mode RCPS problem by using priority-based heuristic rules, considering labor vacations and activity splitting. Padman et al. [29] tested various heuristic rules to minimize the net present value of a project.

Davis and Patterson [26] conducted one of the earliest studies comparing heuristic priority rules. Their study involved comparing the performance of 8 priority rules on 83 different schedules. They found that the “minimum total slack” rule comes first among other rules. Özdamar and Ulusoy [30] used only parallel resource allocation along classical forward scheduling and iterative scheduling methods. They compared 5 heuristic priority rules by testing them on 205 different problems that were borrowed from studies conducted by Alvarez- Olaguibel and Goerlich [31]; Christofides et al., [32]; Li and Willis [33]; and Patterson [34]. Their proposed method, called “local constraint-based analysis”, was the best performing rule, while the “highest weighted resource utilization” rule was the runner up. Schirmer and Riesenberger [23] conducted a comparative analysis using sampling-based techniques. They used PSPLIB’s 480 schedules with 30 activities to test 10 priority rules and found that the “late start time” rule performed best with serial resource allocation, and the “worst case slack (WCS)” rule performed best with parallel resource allocation. In Schirmer’s [35] study, it is stated that performances of WCS and LFT rules are almost similar in parallel resource allocation. The WCS rule performs better only on PSPLIB’s 30 activity schedules and on larger size schedules, LFT (Late Finish Time) rule shows better performance. In Chen’s [36] study, it is discovered that LFT and LFTS (Late Finish Time Stochastic) rules perform better in stochastic case of RCPS problems.

The project size and resource constraint situations also affect the performance of heuristic priority rules [10]. Kanit et al. [37] investigated whether the size of the problem and the number of resource constraints affect the performance of priority rules on 10 real RCPS problems and found no significant differences for problem size. However, they observed a correlation between rule performance and the number of resource constraints. No analysis was found in the literature that considers combinations of variations in (1) resource allocation procedures, (2) number of activities, (3) number of resource constraints, and (4) resource supply levels. This is the first study that compares heuristic priority rules by considering combinations of variations in all these four conditions.

2.2. Problem Formulation

The RCPS problem is a NP-hard problem that involves achieving the shortest project duration given the availabilities of different types of resources [38]. The RCPS problem is described in detail by Brucker et al. [1] and Klein [10] for a single mode case. In the single mode case, activities can shift depending on the availability of float, but activity durations

and resource availabilities are fixed. The assumptions are that intermittent activities are disallowed, and the project network is not cyclical, nor is it activity-on-arrow. In this study, projects include four different renewable resources. The formulation of the single mode RCPS problem is presented in Equations (1) to (5) [39].

$$\min T = \sum_{t=EFT_j}^{LFT_j} t \cdot X_{jt}, \quad (1)$$

where:

$$X_{jt} = \begin{cases} 1, & \text{if } LFT_j < t \\ 0, & \text{otherwise} \end{cases}, \text{ for } j \in J \text{ and } t \in [EFT_j, LFT_j]. \quad (2)$$

In Equation (1), the objective of the analysis is the minimization of T , the project duration; LFT_j and EFT_j represent the late and early activity finish times respectively of activity j ; and t is any time between EFT_j and LFT_j . The set J includes all the activities in the project, and n is the total number of activities in the network. According to Equation (2), X_{jt} takes the value of 1 if activity j finishes at the end of time t , or the value of 0 otherwise.

The constraints are defined in Equations (3) to (5). Equation (3) is a constraint that makes sure that an activity may be shifted within its float, but no further. The predecessor constraints are defined in Equation (4) such that an activity cannot start before all its predecessor activities are completed. In Equation (5), resource constraints are defined such that all activities must be executed within resource limits. d_j is duration of activity j , r_{jr} is resource demand of activity j for resource r , P_j is activities preceding activity j , and R_r is supply of resource r .

Subject to:

$$\sum_{t=EFT_j}^{LFT_j} X_{jt} = 1 \quad j = 1, \dots, n, \quad (3)$$

$$\sum_{t=EFT_i}^{LFT_i} t \cdot X_{it} \leq \sum_{t=EFT_j}^{LFT_j} (t - d_j) \cdot X_{jt} \quad j = 2, \dots, n \quad i \in P_j, \quad (4)$$

$$\sum_{j=1}^n r_{jr} \sum_{\tau=t}^{t+d_j-1} X_{j\tau} \leq R_r. \quad (5)$$

2.3. Heuristic Methods

Heuristic methods are widely used for solving RCPS problems. Resource allocation procedures and a brief introduction to heuristic rules are presented in this section. Serial and parallel resource allocation are prominent procedures that use heuristic rules to sort the activities to be scheduled by their importance. In this study, these procedures are presented with flowcharts. Both methods are described with algebraic expressions in [18].

2.3.1. Serial Resource Allocation Procedure

The serial resource allocation procedure was proposed by Kelley [25]. It is widely used in heuristic/meta-heuristic methods. In the serial resource allocation procedure presented in Figure 1, the number of steps is equal to the number of activities in the network. In each step, one activity from the decision set (D_n) is selected and the earliest start time of this activity is determined without violating constraints. Two different activity sets are updated throughout the procedure, the scheduled set (S_n) and the decision set (D_n). The scheduled set consists of activities that have been scheduled according to precedence relationships, whereas the decision set includes activities that comply with not only precedence but also resource limitations. To be included in the decision set, an activity must satisfy two conditions, namely (1) the activity should not be in the scheduled set; and (2) all predecessors of the activity should have been scheduled. An activity is selected for inclusion in the decision set by using a heuristic priority rule. In the terminology of Klein's [10] study, the decision set is called "available activities", and the scheduled set is defined as a "partial schedule".

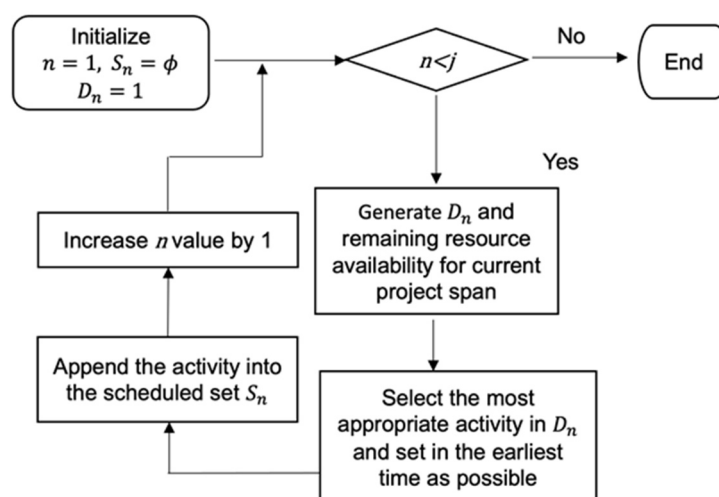


Figure 1. The serial resource allocation procedure.

As seen in Figure 1, at the beginning, by default, the first activity is inserted into the decision set. Because it is the only activity in the decision set, this activity is selected and scheduled. In the second step, the forthcoming decision set contains this scheduled activity's successors. By using priority rules, one of these activities is scheduled without violating resource and precedence constraints. After this activity is scheduled, this activity's start and finish times are determined without violating resource constraints, and the activity is included in the scheduled set (S_n). The process is completed after all activities are scheduled using this procedure.

Figure 2 shows a sample 8-activity network. Figure 3 shows the outputs of the serial resource allocation procedure's six steps in rearranging the activity start-finish times. In each step, the scheduled set (S_n) is augmented by one activity by selecting the most appropriate activity from the decision set (D_n), as explained previously. In this case, the activity with the smallest duration is the most eligible activity in the decision set. Each step in Figure 3 represents one calculation cycle as displayed in Figure 1.

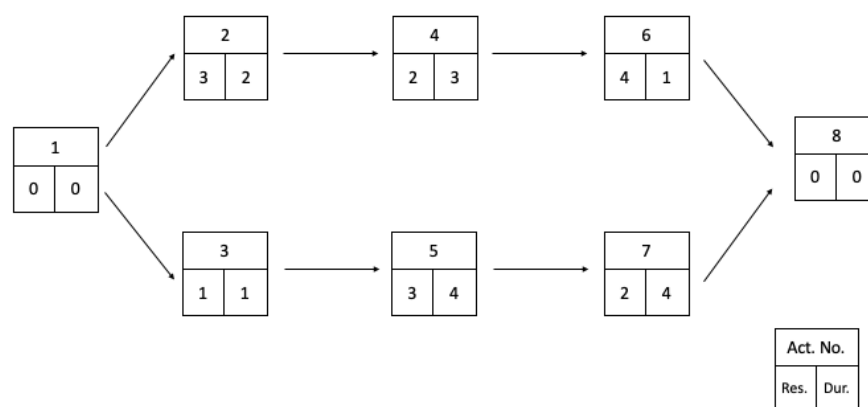


Figure 2. A sample network.

- Activity 1, which has no predecessors, is automatically added to the scheduled set (S_n) but does not appear in the resource diagram in Step 1 because it does not consume any resources.
- In the first step, the successor of Activity 1 that has the smallest duration is selected for inclusion in the scheduled set (S_n). In other words, Activity 3 is the most eligible activity out of Activities 2 and 3 that are connected to Activity 1 in the decision set (D_n) and that have respective durations of 2 and 1 days. So, Activity 3 is added to the scheduled set in the first step.

- In the second step, the successor of Activities 1 or 3 that has the smallest duration is selected for inclusion in the scheduled set (S_n). In other words, Activity 2 is the most eligible activity out of Activities 2 and 5 that are connected to Activities 1 and 3 in the decision set (D_n), and that have respective durations of 2 and 4 days. So, Activity 2 is added to the schedule set in the second step.
- In the third step, the successor of Activities 1, 2, or 3 that has the smallest duration is selected for inclusion in the scheduled set (S_n). In other words, Activity 4 is the most eligible activity out of Activities 4 and 5 that are connected to Activities 1, 2, and 3 in the decision set, and that have respective durations of 3 and 4 days. So, Activity 4 is added to the schedule set in the third step.
- This procedure is continued until no activity remains in the decision set.

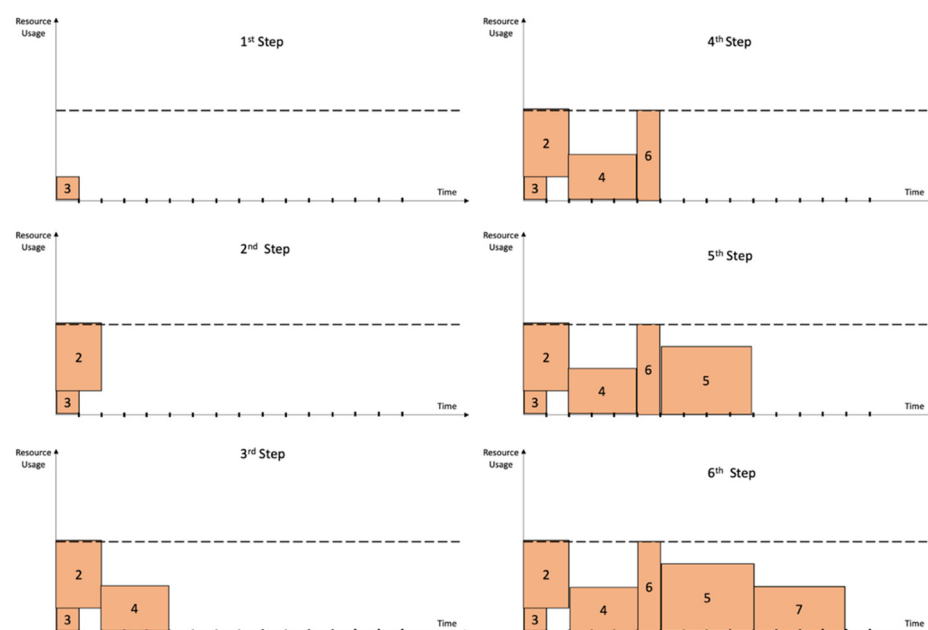


Figure 3. The first five steps in serial resource allocation.

2.3.2. Parallel Resource Allocation Procedure

The flow chart of the parallel resource allocation procedure is presented in Figure 4. There are two known algorithms that are related to parallel resource allocation, the Kelley Algorithm proposed by Kelley [25], and the Brooks Algorithm proposed by Bedworth and Bailey [40]. Inspired from Kolisch's [24] study, the Brooks Algorithm is used to perform parallel resource allocation in this study. There are several similarities and differences between parallel and serial resource allocation procedures. Similar to the procedure in serial resource allocation, the total number of steps in the parallel resource allocation procedure is equal to the total number of activities in the schedule. The main additional variable that is used in parallel resource allocation is the time of the allocation (t_n), which displays the current time of the resource allocation step. In addition, the scheduled set (S_n) used in serial resource allocation is replaced by the active set (A_n) and the completed set (C_n) in parallel resource allocation. The active set (A_n) includes the processed activities at time (t_n), which means that the start and finish times of these activities are within (t_n). Once the activities' start and finish times fall behind time (t_n), these activities are inserted to the completed set (C_n). The decision set in parallel resource allocation has the same meaning as the decision set in serial resource allocation, the only difference being that the activities must begin at time (t_n) to qualify for a place in the decision set (D_n). Similar to serial resource allocation, the most appropriate activity is selected at each step of the parallel resource allocation procedure by using heuristic priority rules.

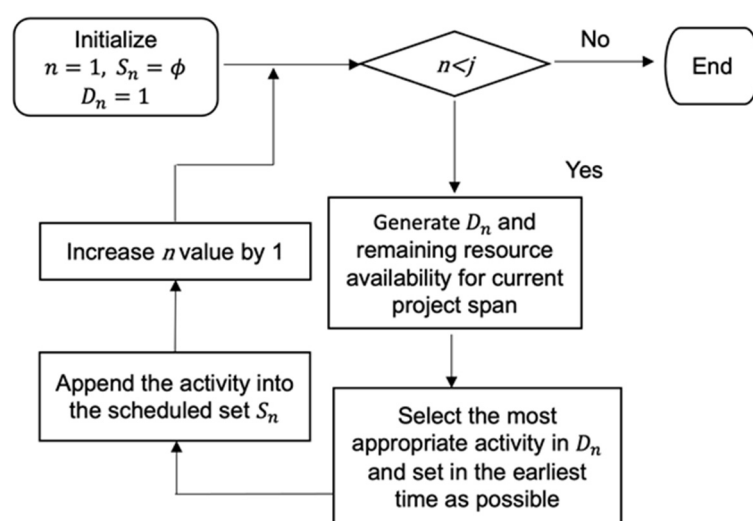


Figure 4. The parallel resource allocation procedure.

Figure 5 shows the outputs of the parallel resource allocation procedure's first five steps in rearranging the activity start-finish times. As mentioned above, the time of the allocation for the n th step (t_n) is an additional parameter that is used in parallel resource allocation. At the beginning, t_n is zero (time now) and is incremented in each step until the conclusion of the resource allocation processes. The selection of the activity with the smallest duration as the most eligible activity in the decision set is used as a priority rule at every allocation time (t_n). Each step in Figure 5 represents one calculation cycle as displayed in Figure 1.

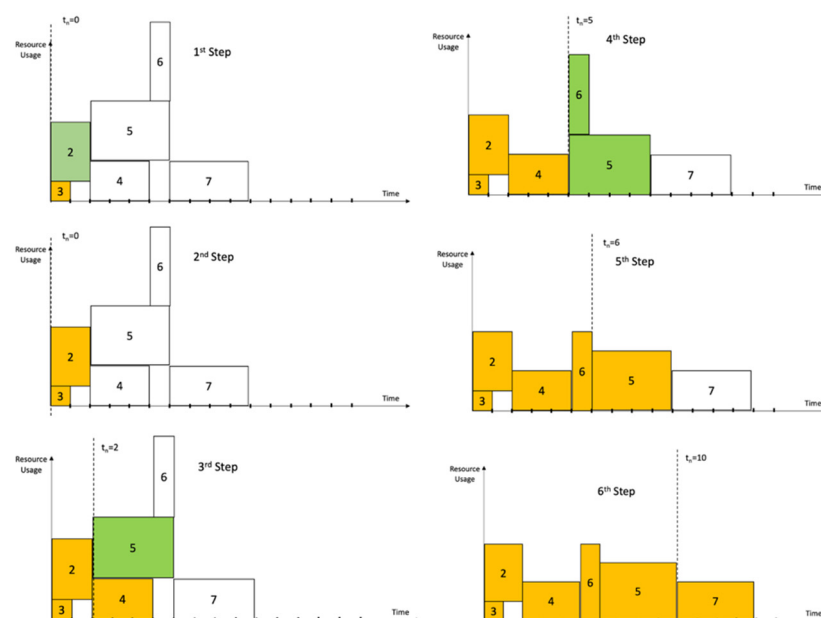


Figure 5. The first five steps in parallel resource allocation.

- Activity 1, which has no predecessors, is automatically added to the active set (A_n) but does not appear in the resource diagram in Step 1 because it does not consume any resources. Neither does Activity 8 for the same reason.
- In the first step, i.e., when $t_1 = 0$, there are four activities (Activities 2 and 3) in the decision set (D_1) that can start at time zero. Activity 3 is the most eligible as it has the smallest duration out of these four activities and is plotted in the first step.

- In the second step, since Activity 3 has been scheduled in the preceding step, there is one remaining activity (the second activity) in the decision set (D_2) that can start at time zero ($t_2 = \text{Day } 0$) that is directly started on day 0.
- In the third step, since Activities 2 and 3 have been scheduled in the previous steps, there are two activities (Activities 4 and 5) in the decision set (D_3) that can start at time zero ($t_3 = \text{Day } 2$). Activity 4, which has the smallest duration out of these two activities is the most eligible and is scheduled in the third step.
- In the fourth step, since Activities 2, 3, and 4 have been scheduled in the previous steps, in this case, there are two activities (activity 6 and 5) in the decision set (D_4) that can start at time $t_4 = \text{Day } 5$. Activity 6 is selected to be scheduled since its duration is lower than the duration of Activity 5. This procedure is continued until no activity remains in the decision set. As seen in Figure 5, the remaining activities, all of which use resources, are scheduled without exceeding the maximum supply.

2.3.3. Heuristic Priority Rules

Browning and Yassine [41] tested 20 heuristic priority rules on 12,320 problems. Özkan and Gülçiçek [42] found the optimum heuristic rules for specific project types using artificial neural networks. Heuristic priority rules can be classified as (1) rules that use network characteristics, (2) rules that use time values, (3) rules that use resource parameters, and (4) rules that use a combination of the previous three rules [31,43].

The 17 rules presented in Table 1 are borrowed from Klein's work [10]. This shortened version of Klein's list [10] of heuristic priority rules excludes the rules that were found by Klein [10] to be irrelevant in resource allocation. The first column represents the abbreviations of the rules, the second column represents the formula of each rule. The terms "max" and "min" represent the importance factor. As an example, in the SAD rule, activities with shorter durations are more important than activities with longer durations.

In the SAD rule, the activity that is selected from the decision set is the activity that has the minimum activity duration (d_j); in the MIS rule it is the maximum number of immediate successors (F_j); and in the MTS rule it is the maximum number of successor activities (F_j^*).

The HRPW rule uses the "positional weight 1" rule that is calculated by adding the duration of the related activity (d_j) to the sum of the durations of all immediate successors ($\sum_{i \in F_j} d_i$). According to the HRPW priority rule, the activity with the highest positional weight 1 is selected from the decision set. The HRPW* rule uses "positional weight 2" formulated by adding the duration of the relevant activity (d_j) to the sum of the durations of all successor activities ($\sum_{i \in F_j^*} d_i$), not only the immediate successors. The HRPW* priority rule requires that the activity with the highest positional weight 2 should be selected from the decision set.

The EFT and EFTD rules are the static early finish and the dynamic early finish heuristic rules, respectively. The static early finish rule uses the initial CPM's early finish values before resource allocation calculations started. On the other hand, in the dynamic early finish rule, the CPM values are updated in each step of the resource allocation calculations. These updated early finish values are used for the selection of the eligible activity.

The LST and LFT are the late start and finish times of an activity. These rules stipulate that the activities with the earliest LST and LFT are selected from the decision set.

The STFS and STFD rules require that activities with the smallest static total float (i.e., float that exists in the original schedule before resource allocation), and dynamic total float (i.e., float that is created in every step of the resource allocation process), respectively be picked from the decision set.

Table 1. Heuristic priority rules.

Priority Rule	Formula	Process for Picking Activities at Each Iteration
SAD	Min d_j	The activity that has the minimum activity duration (d_j) is picked.
MIS	Max F_j	The activity that has the maximum number of immediate successors (F_j) is picked.
MTS	Max F_j^*	The activity that has the maximum number of total successors is picked.
HRPW	Max $(d_j + \sum_{i \in F_j} d_i)$	The activity that has the highest rank positional Weight 1 (see explanation in the text above Table 1) is picked.
HRPW*	Max $(d_j + \sum_{i \in F_j^*} d_i)$	The activity that has the highest rank positional Weight 2 (see explanation in the text above Table 1) is picked.
EFT	Min EFT_j	The activity with the earliest finish time is picked.
EFTD	Min $EFTD_j$	The activity with the earliest dynamic finish time is picked.
LST	Min LS_j	The activity with the earliest late start time is picked.
LFT	Min LF_j	The activity with the earliest late finish time is picked.
STFS	Min LS_j-ES	The activity with the smallest static total float is picked.
STFD	Min LS_j-ES_j	The activity with the smallest dynamic total float is picked.
HRD	Max $(d_j \times \sum_{r=1}^m u_{jr})$	The activity with the highest resource demand is picked.
HWRU	Max $[(\omega \times F_j) + (1 + \omega) \times \sum_{r=1}^m (\frac{u_{jr}}{a_r})]$	The activity with the highest weighted resource utilization and number of successor activities is picked.
HRU1	Max $\sum_{i \in \Pi_{jh}} \sum_{r=1}^m (\frac{u_{ir}}{a_r}) h = 1, \dots, \pi_j$	The activity with the highest resource utilization in all paths succeeding activity j is picked.
HRU2	Max $\sum_{i \in \Pi_{jh}} \sum_{r=1}^m (\frac{u_{ir} d_i}{a_r}) h = 1, \dots, \pi_j$	The activity with the highest resource utilization in all paths succeeding activity j (including the impact of the duration of activity i) is picked.
TIMROS	$\frac{\omega(LS_n - LS_j)}{LS_n - LS_1} + (1 - \omega) \times acr_j / acr_1$	The activity is picked according to Bedworth's [42] heuristic rule HRU1 (see clarification in the text below).
TIMRES	$\frac{\omega(LS_n - LS_j)}{LS_n - LS_1} + (1 - \omega) \times act_j / act_1$	The activity is picked according to Elsayed and Nasr's [43] heuristic rule HRU2 (see clarification in the text below).

In the HRD rule, the activity with the greatest resource demand is selected from the decision set. Resource demand is formulated as the product of the duration of an activity j (d_j) with the demand in resource (r) in activity (j) ($\sum_{r=1}^m u_{jr}$) where m is defined as the total number of resource types. The HWRU rule was proposed by Ulusoy and Özdamar [44] and makes use of the sum of the weighted resource demand $[(1 + \omega) \times \sum_{r=1}^m (\frac{u_{jr}}{a_r})]$ and the total number of immediate successor activities $(\omega \times |F_j|)$ where (ω) is the weight, (u_{jr}) is the demand of resource (r) in activity (j), and (a_r) is the availability of resource (r). The activity that displays the largest $[(\omega \times |F_j|) + (1 + \omega) \times \sum_{r=1}^m (\frac{u_{jr}}{a_r})]$ is selected from the decision set.

The HRU1 rule was introduced by Bedworth [45] and makes use of the relationship $\sum_{i \in \Pi_{jh}} \sum_{r=1}^m (\frac{u_{ir}}{a_r}) | h = 1, \dots, \pi_j$. Where π_{jh} is a set of activities on path (h) placed after activity (j). The sets $(1, \dots, \pi_j)$ represent all paths placed after activity (j) in the network. The HRU2 rule was proposed by Elsayed and Nasr [46] and is a modified version of the HRU1 rule as it makes use of the relationship $\sum_{i \in \Pi_{jh}} \sum_{r=1}^m (\frac{u_{ir} d_i}{a_r}) | h = 1, \dots, \pi_j$.

The TIMROS and TIMRES rules use the HRU1 and HRU2 rules, respectively. While LS_n represents the late start time of the last activity, acr_j represents the HRU1 value of activity j , and act_j represents activity j 's HRU2 value. In the many studies that have compared heuristic rules, there is no consensus on how superior a specific rule is to other rules in different types of problems [41,47–50]. Franco-Duran and de la Garza [51] investigated the selection of appropriate heuristic rules but only for specific types of projects. Messelis and de Causmaecker [48] developed a super algorithm that selects an appropriate algorithm but only for individual multi-mode resource-constraint project scheduling cases. Vázquez et al. [49] developed a genetic algorithm-based auto selection tool to determine the most appropriate priority rule but only for a multi-project resource

constrained project scheduling problem; they also proposed a tie breaker mechanisms in case pairs of rules get the same priority. Franco-Duran and de la Garza [52] also used tie-breakers to enhance heuristic rule performance and compared the heuristic rules on 142 different networks. The identification of appropriate heuristic rules is an open research question [53].

3. Test Results and Discussion

In this study, 17 heuristic priority rules are compared using both serial and parallel resource allocation by solving single mode benchmark problems obtained from PSPLIB, a library that was introduced by Kolisch and Sprecher [7]. A random problem generator called ProGen is used for generating the problems [54].

A sample network generated by PSPLIB is presented in Figure 6. There are 32 activities in the network. The first and last activities are dummy activities, these activities do not need resources and their durations are equal to zero.

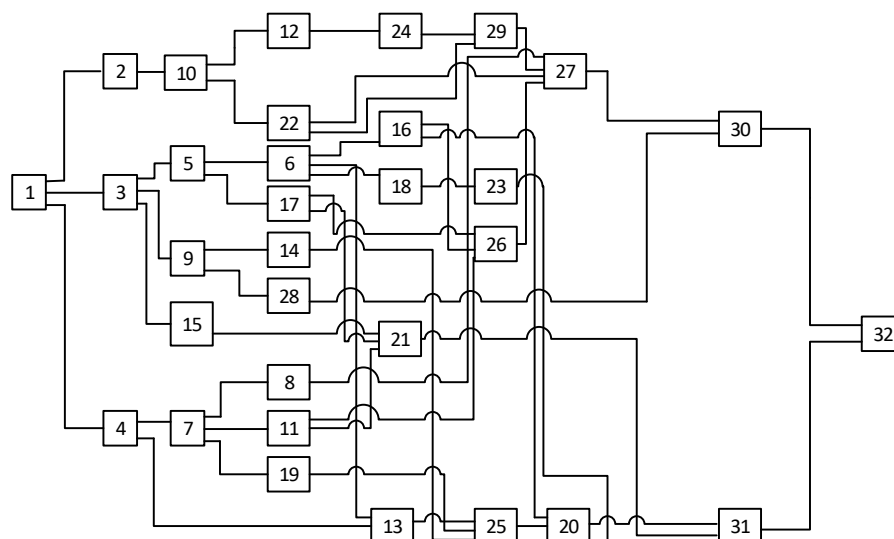


Figure 6. Sample network diagram from benchmark networks.

The benchmark problems in PSPLIB consist of four sets of different size problems. These 4 problem sets have 480, 480, 480, and 600 schedules that include 30, 60, 90, and 120 activities, respectively. In this study, all these problems are solved using the 17 different heuristic rules presented in Table 1 using both the serial and parallel resource allocation procedures.

To expand the testing to reflect varying conditions, six combinations of resource types and resource supply levels were also considered (Table 2).

Table 2. Test combinations and descriptions.

Combination No.	Number of Resource Types	Resource Supply Level
1	1	Relaxed supply level
2	1	Minimum supply level
3	4	Relaxed supply level
4	4	Minimum supply level
5	10	Relaxed supply level
6	10	Minimum supply level

- To increase the variety of problem sets, situations where only one type of resource is used (Combinations 1 and 2 in Table 2), 4 types of resources are used (Combinations 3 and 4 in Table 2), and 10 types of resources are used (Combinations 5 and 6 in Table 2) are also considered. In PSPLIB's default case, the scheduled activities use four different

types of resources. So, looking into the use of fewer or more numerous resource types widens the research field and helps with the interpretation of the results.

- To reflect the varying conditions in construction projects, two resource supply levels are defined in the last column of Table 2. The supply in the “minimum supply level” satisfies an activity’s demand for a specific resource. The supply in the “relaxed supply level” exceeds demand by a factor of 2.5.

All calculations were programmed in Matlab for all 17 heuristic priority rules that were presented in Table 1 using both serial and parallel resource allocation procedures, and the variations expressed in the 6 combinations presented in Table 2. In other words, 12 tests were conducted to test each of the 17 priority rules on each of the 2040 benchmark problems, a total of $12 \times 17 \times 2040 = 416,160$ tests.

Tables 3 and 4 show the average increases in project duration after RCPS procedures are implemented using serial and parallel resource allocation procedures, respectively, for each heuristic rule. The rules that achieve the lowest increase in project duration are underlined and bolded. The following are observed in Tables 3 and 4.

Table 3. Average increase in project duration after serial resource allocation, and heuristic rule performance rankings (in parentheses).

Heuristic Rules	1 Resource Type		4 Resource Types		10 Resource Types	
	Relaxed Supply	Limited Supply	Relaxed Supply	Limited Supply	Relaxed Supply	Limited Supply
SAD	27.05% (4)	165.09% (4)	48.64% (4)	273.07% (4)	74.42% (4)	345.80% (4)
MIS	19.35% (2)	142.79% (2)	35.94% (3)	241.96% (2)	56.47% (2)	319.51% (2)
MTS	15.31% (2)	135.89% (2)	29.73% (2)	232.42% (2)	48.86% (2)	312.08% (1)
HRPW	21.60% (3)	151.63% (3)	39.20% (3)	253.21% (3)	60.59% (3)	327.77% (3)
HRPW*	14.25% (1)	135.14% (1)	28.19% (2)	231.59% (1)	46.71% (1)	311.13% (1)
EFT	21.41% (3)	145.38% (2)	39.62% (3)	249.37% (3)	61.61% (3)	327.30% (3)
EFTD	20.92% (3)	143.45% (2)	38.22% (3)	245.33% (2)	59.71% (3)	324.26% (3)
LST	13.61% (1)	135.12% (1)	27.50% (1)	232.51% (2)	46.22% (1)	311.99% (1)
LFT	14.05% (1)	135.02% (1)	28.39% (1)	232.47% (2)	47.75% (1)	312.48% (2)
STFS	34.38% (4)	177.17% (4)	59.41% (4)	289.40% (4)	86.69% (4)	356.53% (4)
STFD	19.27% (2)	154.48% (3)	36.58% (2)	257.02% (3)	58.37% (3)	331.37% (3)
HRD	22.31% (3)	155.36% (3)	41.51% (3)	256.40% (3)	64.32% (3)	332.48% (3)
HWRU	21.96% (3)	144.88% (2)	40.62% (3)	244.82% (2)	62.43% (3)	322.53% (2)
HRU1	20.78% (3)	144.90% (2)	36.49% (2)	241.74% (2)	56.93% (2)	320.16% (2)
HRU2	20.61% (3)	146.32% (2)	36.41% (2)	244.25% (2)	56.60% (2)	321.15% (2)
TIMROS	16.97% (2)	140.48% (2)	30.45% (2)	235.68% (2)	49.68% (2)	315.43% (2)
TIMRES	17.32% (2)	142.30% (2)	31.01% (2)	238.82% (2)	50.35% (2)	317.04% (2)

Note: The minimum value in the problem set in each column is underlined.

As expected, when the supply is larger than the demand (“relaxed supply” regardless of number of “resource types”), the project duration does not suffer from as much time extension as when resource supply is strictly limited to demand.

- Assuming the same supply level in all cases, projects with a larger variety of resources suffer more time extension compared to projects that use only few types of resources (e.g., 10 “resource types” vs. 1 “resource type”, regardless of supply level).
- The increases in project duration vary dramatically for each of the six combinations regardless of whether one uses serial or parallel resource allocation, but there is only a slight difference between the performance of the serial and parallel resource allocation procedures. The increases in project duration are calculated by dividing the delayed duration by the critical path length, very much like Franco-Duran and de la Garza [51] did in their study.
- To assess the performance of heuristic rules relative to each other’s, they are grouped in Figure 7 by their distance to the mean increase of project duration assuming a normal distribution. The worst performing heuristics fall into the fourth slice that represents a performance that surpasses the mean increase in project duration (μ) by more than one standard deviation ($+\sigma$), whereas the ones that fall into the first slice are considered the best performing heuristic rules since the first slice represents a performance that falls behind the mean increase in project duration (μ) by more than one standard deviation ($-\sigma$).

- In serial resource allocation (Table 3), while the LST priority rule consistently results in a desirable outcome, i.e., shorter project durations when ample resources are available, the HRPW* rule causes the largest time extensions when resources are limited. On the other hand, in parallel resource allocation (Table 4), the LFT rule stands out as the best performing priority rule in almost all cases.

Table 4. Average increase in project duration after parallel resource allocation, and heuristic rule performance rankings (in parentheses).

Heuristic Rules	1 Resource Type		4 Resource Types		10 Resource Types	
	Relaxed Supply	Limited Supply	Relaxed Supply	Limited Supply	Relaxed Supply	Limited Supply
SAD	22.69% (3)	154.64% (3)	37.95% (3)	249.08% (3)	58.28% (3)	322.58% (2)
MIS	21.89% (3)	154.78% (3)	37.12% (3)	248.00% (3)	56.72% (2)	321.30% (2)
MTS	15.29% (2)	134.52% (1)	27.79% (2)	223.10% (1)	44.64% (1)	302.20% (1)
HRPW	20.11% (3)	150.37% (3)	35.16% (3)	246.50% (2)	54.40% (2)	320.54% (2)
HRPW*	14.58% (1)	134.07% (1)	26.93% (2)	223.26% (1)	43.54% (1)	302.21% (1)
EFT	19.72% (2)	140.85% (2)	34.60% (2)	234.02% (2)	53.41% (2)	312.45% (2)
EFTD	22.69% (3)	154.64% (3)	37.95% (3)	249.08% (3)	58.28% (3)	322.58% (2)
LST	14.02% (1)	134.43% (1)	26.56% (2)	225.59% (1)	43.40% (1)	304.40% (1)
LFT	14.19% (1)	132.90% (1)	26.28% (2)	221.78% (1)	43.06% (1)	301.15% (1)
STFS	17.58% (2)	150.36% (3)	31.63% (2)	244.00% (2)	50.20% (2)	318.46% (2)
STFD	14.02% (1)	134.43% (1)	26.56% (2)	225.59% (1)	43.40% (1)	304.40% (1)
HRD	21.97% (3)	158.63% (4)	36.87% (3)	250.31% (3)	57.04% (2)	326.48% (3)
HWRU	19.19% (2)	138.17% (2)	32.64% (2)	227.78% (1)	51.06% (2)	305.83% (1)
HRU1	18.52% (2)	139.90% (2)	30.77% (2)	228.52% (1)	48.45% (2)	307.39% (1)
HRU2	18.96% (2)	144.39% (2)	31.37% (2)	233.99% (2)	49.59% (2)	311.80% (1)
TIMROS	16.06% (2)	136.30% (2)	27.91% (2)	225.28% (1)	44.94% (1)	304.64% (1)
TIMRES	17.12% (2)	141.36% (2)	28.95% (2)	230.27% (1)	46.33% (1)	308.63% (1)

Note: The minimum value in the problem set in each column is underlined.

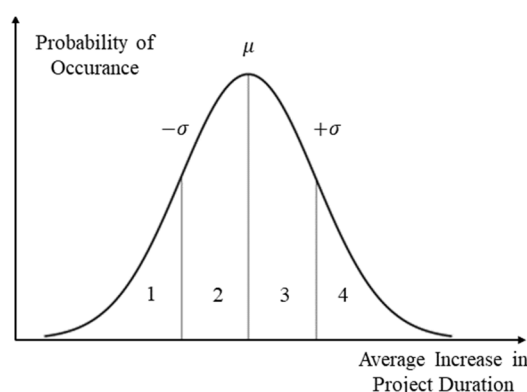


Figure 7. Performance ranking of heuristic rules.

In serial resource allocation (Table 3), the SAD and STFS rules rank fourth in every combination of resource type and relaxed/limited supply and the HRPW, EFT, EFTD, STFD, HRD, HWRU, HRU1, and HRU2 rules rank third at least once. These rules consistently failed to perform better than the other rules in serial resource allocation as they resulted in longer extensions of the project duration. On the other hand, the MTS, HRPW*, LST, LFT, TIMROS, and TIMRES rules consistently ranked first or second.

In parallel resource allocation (Table 4), only the HRD rule is ranked fourth, and the SAD, MIS, HRPW, EFTD, and STFS rules are ranked third at least once. In parallel resource allocation, these rules are considered as low performance rules, whereas the MTS, HRPW*, EFT, LST, LFT, STFD, HWRU, HRU1, HRU2, TIMROS, and TIMRES rules ranked first and second.

When one considers both serial and parallel resource allocation combined, Tables 3 and 4 indicate that the MTS, HRPW*, LST, LFT, TIMROS, and TIMRES rules consistently rank first and second (i.e., they resulted in shorter extensions of the project duration) and can be considered as the best performing rules regardless of the resource allocation procedure used.

Table 5 shows the average increases in project duration by size of network after RCPS procedures are implemented using serial and parallel resource allocation under the six combinations of circumstances defined in Table 2. After comparing the serial and parallel resource allocation procedures, the better results (i.e., the lowest increases in project duration) are underlined and bolded in Table 5.

Table 5. Average increases in project duration after RCPS procedures.

Combinations as of Table 2		30-Activity Networks (480 Cases)		60-Activity Networks (480 Cases)		90-Activity Networks (480 Cases)		120-Activity Networks (600 Cases)	
Number of Resource Types	Resource Supply Levels	Serial Allocation	Parallel Allocation	Serial Allocation	Parallel Allocation	Serial Allocation	Parallel Allocation	Serial Allocation	Parallel Allocation
1 resource type	Relaxed supply Limited supply	<u>7.02%</u>	7.15%	<u>7.46%</u>	7.89%	<u>7.61%</u>	7.79%	<u>24.66%</u>	25.35%
		57.34%	55.17%	103.76%	101.73%	140.54%	138.4%	184.17%	182.17%
4 resource types	Relaxed supply Limited supply	19.18%	<u>18.12%</u>	16.88%	<u>16.71%</u>	15.88%	<u>15.61%</u>	46.17%	<u>43.50%</u>
		117.03%	112.25%	185.19%	177.04%	241.40%	230.99%	303.23%	290.64%
10 resource types	Relaxed supply Limited supply	37.33%	<u>35.05%</u>	30.33%	<u>29.05%</u>	27.76%	<u>26.56%</u>	73.37%	<u>67.17%</u>
		158.46%	155.08%	249.93%	242.01%	325.45%	314.88%	405.31%	391.59%

Note: The lowest increases in project duration are underlined and bolded.

As seen in Table 5, parallel resource allocation performs better than serial resource allocation in most of the cases, the only exception occurring in the case of one resource type with a relaxed supply level. This finding can be interpreted to mean that in simple cases where there is only one resource that is available in abundance relative to demand, it is quite sufficient to perform simple serial resource allocation rather than the more complex parallel resource allocation. It can also be inferred from this finding that parallel resource allocation is slightly more advantageous than serial resource allocation in situations that involve multiple resource types, a situation that occurs quite often in construction projects. The results do not agree with Kim and Ellis Jr.'s [55] study that used an elitist genetic algorithm to compare these resource allocation techniques and that found that serial allocation performed better than parallel allocation. In that article it is said that the search space of serial resource allocation is larger than the search space of parallel resource allocation. As in Kolisch [24]'s study, serial resource allocation outperforms parallel resource allocation in case of multi-pass techniques. However, in the single-pass case, parallel resource allocation performs better than serial resource allocation.

4. Conclusions

The main advantage of heuristic methods in solving RCPS problems is that they operate with high computational speed because they use simple priority rules. Using the best-performing priority rules enhances the advantages of heuristic RCPS. Most scheduling software packages use heuristic priority rules. In this study, the most common heuristic priority rules are tested under a variety of conditions defined by network size, resource allocation procedure, number of types of resources, and resource supply levels.

It is not easy to determine by surveying the literature which priority rule exhibits the best performance in solving the RCPS problem. While some researchers consider few real projects, others compare few (but not all) priority rules, and others use only one size (often a small size) network to compare the rules. To fill this gap, 17 priority rules were investigated in this study in a total of 2040 projects, i.e., all the projects in PSPLIB organized in four sets of work schedules of 30, 60, 90, and 120 activities, using different types of resources and different resource supply levels.

Overall, it was found that the MTS, HRPW*, LST, LFT, TIMROS, and TIMRES rules consistently rank first and second as they resulted in shorter extensions of the project

duration regardless of the resource allocation procedure used. It was also found that in most cases, parallel resource allocation performs better than serial resource allocation, and that the “minimum late finish” rule (LFT) performs better than the other priority rules in parallel resource allocation. In serial resource allocation, the “minimum late start time” (LST) and the “highest rank of positional weight 2” (HRPW*) rules share the first rank depending on the combination of resource supply levels and number of resource types. All three rules are simple and tend to demonstrate that simple heuristic priority rules perform better (i.e., result in shorter extensions of the project duration) than complicated rules.

Even though the study presents a thorough comparison of priority rules used in heuristic methods, it also has limitations. Multi-mode and multi-project applications, the impact of project type and project scope need to be investigated in future research. In addition, 1, 4, and 10 resource types and minimum and relaxed supply levels are considered in Table 2. The use of more sophisticated measures such as the Resource Factor (RF) defined by Pascoe [56] and the Resource Strength (RS) proposed by Kolisch [43] can be explored in future research.

Author Contributions: Conceptualization, O.H.T.; Formal analysis, O.H.T. and D.A.; Investigation, O.H.T.; Methodology, O.H.T. and D.A.; Project Administration, O.H.T. and E.M.; Software, O.H.T.; Supervision, D.A. and E.M.; Validation, O.H.T.; Writing—original draft, O.H.T.; Writing—review & editing, D.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available on request from corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

Table of Acronyms

CPM	Critical Path Method
RCPS	Resource Constrained Project Scheduling
PSPLIB	Project Scheduling Problem Library
WCS	Worst Case Slack
LFT	Late Finish Time
LST	Late Start Time
EFT	Early Finish Time
SAD	Shortest Activity Duration
MIS	Maximum Immediate Successors
MTS	Maximum Total Successors
HRPW	Highest Rank Positional Weight
EFT	Early Finish Time
EFTD	Early Start Time Dynamical
LST	Late Start Time
LFT	Late Finish Time
STFS	Smallest Static Total Float
STFD	Smallest Dynamic Total Float
HRD	Highest Resource Demand
HWRU	Highest Weight Resource Utilization
HRU	Highest Resource Utilization

References

1. Brucker, P.; Drexler, A.; Möhring, R.; Neumann, K.; Pesch, E. Resource-constrained project scheduling: Notation, classification, models, and methods. *Eur. J. Oper. Res.* **1999**, *112*, 3–41. [\[CrossRef\]](#)
2. Klein, R. Bidirectional planning: Improving priority rule-based heuristics for scheduling resource-constrained projects. *Eur. J. Oper. Res.* **2000**, *127*, 619–638. [\[CrossRef\]](#)

3. Xie, L.; Chen, Y.; Chang, R. Scheduling Optimization of Prefabricated Construction Projects by Genetic Algorithm. *Appl. Sci.* **2021**, *11*, 5531. [\[CrossRef\]](#)
4. He, W.; Li, W.; Wang, W. Developing a Resource Allocation Approach for Resource-Constrained Construction Operation under Multi-Objective Operation. *Sustainability* **2021**, *13*, 7318. [\[CrossRef\]](#)
5. Kim, K. Generalized Resource-Constrained Critical Path Method to Improve Sustainability in Construction Project Scheduling. *Sustainability* **2020**, *12*, 8918. [\[CrossRef\]](#)
6. Klimek, M. Financial Optimization of the Resource-Constrained Project Scheduling Problem with Milestones Payments. *Appl. Sci.* **2021**, *11*, 661. [\[CrossRef\]](#)
7. Kolisch, R.; Sprecher, A. PSPLIB—a project scheduling problem library: OR software-ORSEP operations research software exchange program. *Eur. J. Oper. Res.* **1997**, *96*, 205–216. [\[CrossRef\]](#)
8. Johnson, T.J.R. An Algorithm for the Resource Constrained Project Scheduling Problem. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1967.
9. Gutjahr, W.J. Bi-objective multi-mode project scheduling under risk aversion. *Eur. J. Oper. Res.* **2015**, *246*, 421–434. [\[CrossRef\]](#)
10. Klein, R. *Scheduling of Resource-Constrained Projects*; Springer Science & Business Media: New York, NY, USA, 1999; Volume 10, ISBN 978-1-4613-7093-2.
11. Kolisch, R.; Drexel, A. Adaptive search for solving hard project scheduling problems. *Nav. Res. Logist.* **1996**, *43*, 23–40. [\[CrossRef\]](#)
12. Bean, J.C. Genetic algorithms and random keys for sequencing and optimization. *ORSA J. Comput.* **1994**, *6*, 154–160. [\[CrossRef\]](#)
13. Wall, M.B. A Genetic Algorithm for Resource-Constrained Scheduling. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1996.
14. Jia, L.; Yisheng, L.; Ying, S.; Jian, L. Solving Resource-Constrained Project Scheduling Problem via Genetic Algorithm. *J. Comput. Civ. Eng.* **2020**, *34*, 4019055. [\[CrossRef\]](#)
15. Nonobe, K.; Ibaraki, T. Formulation and tabu search algorithm for the resource constrained project scheduling problem. In *Essays and Surveys in Metaheuristics*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 557–588.
16. Bouleimen, K.; Lecocq, H. A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version. *Eur. J. Oper. Res.* **2003**, *149*, 268–281. [\[CrossRef\]](#)
17. Fink, A.; Homberger, J. An ant-based coordination mechanism for resource-constrained project scheduling with multiple agents and cash flow objectives. *Flex. Serv. Manuf. J.* **2013**, *25*, 94–121. [\[CrossRef\]](#)
18. Tran, D.-H.; Cheng, M.-Y.; Cao, M.-T. Solving Resource-Constrained Project Scheduling Problems Using Hybrid Artificial Bee Colony with Differential Evolution. *J. Comput. Civ. Eng.* **2016**, *30*, 4015065. [\[CrossRef\]](#)
19. Sonmez, R.; Uysal, F. Backward-Forward Hybrid Genetic Algorithm for Resource-Constrained Multiproject Scheduling Problem. *J. Comput. Civ. Eng.* **2015**, *29*, 4014072. [\[CrossRef\]](#)
20. Kannimuthu, M.; Raphael, B.; Ekambaram, P.; Kuppuswamy, A. Comparing optimization modeling approaches for the multi-mode resource-constrained multi-project scheduling problem. *Eng. Constr. Archit. Manag.* **2020**, *27*, 893–916. [\[CrossRef\]](#)
21. Christodoulou, S.; Ellinas, G.; Aslani, P. Entropy-based scheduling of resource-constrained construction projects. *Autom. Constr.* **2009**, *18*, 919–928. [\[CrossRef\]](#)
22. Christodoulou, S.E. Entropy-Based Heuristic for Resource-Constrained Project Scheduling. *J. Comput. Civ. Eng.* **2017**, *31*, 4016068. [\[CrossRef\]](#)
23. Schirmer, A.; Riesenberger, S. *Parameterized Heuristics for Project Scheduling: Biased Random Sampling Methods*; Manuskripte aus den Instituten für Betriebswirtschaftslehre der Universität Kiel: Kiel, Germany, 1997.
24. Kolisch, R. Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. *Eur. J. Oper. Res.* **1996**, *90*, 320–333. [\[CrossRef\]](#)
25. Kelley, J.E. The critical-path method: Resources planning and scheduling. *Ind. Sched.* **1963**, *13*, 347–365.
26. Davis, E.W.; Patterson, J.H. A comparison of heuristic and optimum solutions in resource-constrained project scheduling. *Manage. Sci.* **1975**, *21*, 944–955. [\[CrossRef\]](#)
27. Tormos, P.; Lova, A. A competitive heuristic solution technique for resource-constrained project scheduling. *Ann. Oper. Res.* **2001**, *102*, 65–81. [\[CrossRef\]](#)
28. Buddhakulsomsiri, J.; Kim, D.S. Priority rule-based heuristic for multi-mode resource-constrained project scheduling problems with resource vacations and activity splitting. *Eur. J. Oper. Res.* **2007**, *178*, 374–390. [\[CrossRef\]](#)
29. Padman, R.; Smith-Daniels, D.E.; Smith-Daniels, V.L. Heuristic Scheduling of Resource-Constrained Projects with Cash Flows. *Nav. Res. Logist.* **1997**, *44*, 365–381. [\[CrossRef\]](#)
30. Özdamar, L.; Ulusoy, G. A note on an iterative forward/backward scheduling technique with reference to a procedure by Li and Willis. *Eur. J. Oper. Res.* **1996**, *89*, 400–407. [\[CrossRef\]](#)
31. Olaguibel, R.A.-V.; Goerlich, J.M.T. Heuristic algorithms for resource-constrained project scheduling: A review and an empirical analysis. In *Advances in Project Scheduling*; Elsevier: Amsterdam, The Netherlands, 1989; pp. 113–134.
32. Christofides, N.; Alvarez-Valdés, R.; Tamarit, J.M. Project scheduling with resource constraints: A branch and bound approach. *Eur. J. Oper. Res.* **1987**, *29*, 262–273. [\[CrossRef\]](#)
33. Li, K.Y.; Willis, R.J. An iterative scheduling technique for resource-constrained project scheduling. *Eur. J. Oper. Res.* **1992**, *56*, 370–379. [\[CrossRef\]](#)

34. Patterson, J.H. A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem. *Manag. Sci.* **1984**, *30*, 854–867. [\[CrossRef\]](#)
35. Schirmer, A. Case-based reasoning and improved adaptive search for project scheduling. *Nav. Res. Logist.* **2000**, *47*, 201–222. [\[CrossRef\]](#)
36. Chen, Z.; Demeulemeester, E.; Bai, S.; Guo, Y. Efficient priority rules for the stochastic resource-constrained project scheduling problem. *Eur. J. Oper. Res.* **2018**, *270*, 957–967. [\[CrossRef\]](#)
37. Kanit, R.; Ozkan, O.; Gunduz, M. Effects of project size and resource constraints on project duration through priority rule-based heuristics. *Artif. Intell. Rev.* **2009**, *32*, 115. [\[CrossRef\]](#)
38. Blazewicz, J.; Lenstra, J.K.; Kan, A.H.G.R. Scheduling subject to resource constraints: Classification and complexity. *Discret. Appl. Math.* **1983**, *5*, 11–24. [\[CrossRef\]](#)
39. Pritsker, A.A.B.; Waiters, L.J.; Wolfe, P.M. Multiproject scheduling with limited resources: A zero-one programming approach. *Manag. Sci.* **1969**, *16*, 93–108. [\[CrossRef\]](#)
40. Bedworth, D.D.; Bailey, J.E. *Integrated Production Control Systems: Management, Analysis, Design*; John Wiley & Sons, Inc: New York, NY, USA, 1999; ISBN 0471821799.
41. Browning, T.R.; Yassine, A.A. Resource-constrained multi-project scheduling: Priority rule performance revisited. *Int. J. Prod. Econ.* **2010**, *126*, 212–228. [\[CrossRef\]](#)
42. Özkan, Ö.; Gülçiçek, Ü. A neural network for resource constrained project scheduling programming. *J. Civ. Eng. Manag.* **2015**, *21*, 193–200. [\[CrossRef\]](#)
43. Kolisch, R. *Project Scheduling under Resource Constraints: Efficient Heuristics for Several Problem Classes*; Physica-Verlag HD: Heidelberg, Germany, 1995; ISBN 3642502962.
44. Ulusoy, G.; Özdamar, L. Heuristic performance and network/resource characteristics in resource-constrained project scheduling. *J. Oper. Res. Soc.* **1989**, 1145–1152. [\[CrossRef\]](#)
45. Bedworth, D.D. *Industrial Systems: Planning, Analysis, Control*; Ronald Press: New York, NY, USA, 1973.
46. Elsayed, E.A.; Nasr, N.Z. Heuristics for resource-constrained scheduling. *Int. J. Prod. Res.* **1986**, *24*, 299–310. [\[CrossRef\]](#)
47. Cohen, I.; Mandelbaum, A.; Shtub, A. Multi-Project Scheduling and Control: A Process-Based Comparative Study of the Critical Chain Methodology and Some Alternatives. *Proj. Manag. J.* **2004**, *35*, 39–50. [\[CrossRef\]](#)
48. Messelis, T.; De Causmaecker, P. An automatic algorithm selection approach for the multi-mode resource-constrained project scheduling problem. *Eur. J. Oper. Res.* **2014**, *233*, 511–528. [\[CrossRef\]](#)
49. Vázquez, E.P.; Calvo, M.P.; Ordóñez, P.M. Learning process on priority rules to solve the RCMPSP. *J. Intell. Manuf.* **2015**, *26*, 123–138. [\[CrossRef\]](#)
50. Wang, Y.; He, Z.; Kerkhove, L.-P.; Vanhoucke, M. On the performance of priority rules for the stochastic resource constrained multi-project scheduling problem. *Comput. Ind. Eng.* **2017**, *114*, 223–234. [\[CrossRef\]](#)
51. Franco-Duran, D.M.; de la Garza, J.M. Review of Resource-Constrained Scheduling Algorithms. *J. Constr. Eng. Manag.* **2019**, *145*, 3119006. [\[CrossRef\]](#)
52. Franco-Duran, D.M.; de la Garza, J.M. Performance of Resource-Constrained Scheduling Heuristics. *J. Constr. Eng. Manag.* **2020**, *146*, 4020026. [\[CrossRef\]](#)
53. Chakraborty, R.K.; Rahman, H.F.; Ryan, M.J. Efficient priority rules for project scheduling under dynamic environments: A heuristic approach. *Comput. Ind. Eng.* **2020**, *140*, 106287. [\[CrossRef\]](#)
54. Schwindt, C. *ProGen/Max: A New Problem Generator for Different Resource-Constrained Project Scheduling Problems with Minimal and Maximal Time Lags*; Institut für Wirtschaftstheorie und Operations Research, Universität Karlsruhe: Karlsruhe, Germany, 1995.
55. Kim, J.-L.; Ellis, R.D. Comparing Schedule Generation Schemes in Resource-Constrained Project Scheduling Using Elitist Genetic Algorithm. *J. Constr. Eng. Manag.* **2010**, *136*, 160–169. [\[CrossRef\]](#)
56. Pascoe, T.L. Allocation of resources CPM. *Rev. Fr. Rech. Oper.* **1966**, *10*, 31.