Available online at www.sciencedirect.com
SCIENCE DIRECT
ELSEVIER
International Journal of Project Management 24 (2006) 330–336
INTERNATIONAL JOURNAL OF PROJECT MANAGEMENT
www.elsevier.com/locate/ijproman

Non-delay scheduling as a managerial approach for managing projects

Ofer Zwikael [a,*], Yuval Cohen [b], Arik Sadeh [c,1]

[a] Victoria Management School, Victoria University of Wellington, PO Box 600, Wellington, New Zealand
[b] Department of Management and Economics, the Open University of Israel, Israel
[c] Management of Technology Department, Holon Academic Institute of Technology, Israel

Received 25 January 2005; received in revised form 8 September 2005; accepted 3 November 2005

Abstract

Many project managers face the dilemma of using delay versus non-delay scheduling while planning and managing a project. The first approach is called a non-delay schedule, also known as the "early start" or "as soon as possible" approach. The second approach, called a "delay schedule", based on "late start" scheduling, has recently been connected to some new models, among them, the Theory of Constraints. The paper compares these two approaches for managing projects. The non-delay concept was found to be more efficient in reducing project risk, reducing project duration, improving resource load, improving project tram development and improving the positioning of a project manager while negotiating for resources. However, some limitations in existing non-delay models were found and hence, a new non-delay scheduling algorithm was developed and is introduced in this paper. Comparing this model with other models from the literature, it was found that the suggested one gives better results while considering measures of computing time and project duration.
© 2005 Elsevier Ltd and IPMA. All rights reserved.

Keywords: Managing projects; Time; Resource constraints; Branch and bound; Non-delay scheduling; Theory of constraints

1. Introduction

Many project managers face the dilemma of whether to start executing certain non-critical activities, or to delay them, using their slacks. On the one hand, delaying non-critical activities saves the time-value of capital tied to their execution, frees the resources they would have used and improve project's cash flow. On the other hand, delaying activities increases their chance of becoming critical and delaying the project's due date. Roughly speaking, two scheduling approaches for this dilemma exist: the first approach is called *Non-Delay* scheduling, and is also known as the "early-start approach" or "ASAP approach" (as soon as possible approach). The second approach is the *Delay* concept, also known as the "late-start approach". The second approach has received a significant push from the Theory of Constraints and is used in capital-intensive projects for saving the time value of money.

The Non-Delay concept is commonly used among project managers. According to this approach, any activity that can start without deferring other activities should not be delayed. Delaying activities may increase the risk of not completing the project on time. Moreover, avoiding using available resources may cause project managers to be accused of wasting their resources and losing resources for other projects. Thus, it is the interest of a project manager to prefer the early-start approach, and as a result, most project management software packages use this approach as the default for project planning. Today, most project management software tools offer project managers a choice between these two types of scheduling.

Lately, some heuristics, models and project management software packages, based on the "Delay" concept have become very popular. The leading model is called Theory of Constraints (TOC), introduced by Goldratt [13], using the slacks to their maximal values and then protecting

* Corresponding author. Fax: +972 3 5026650.
E-mail addresses: tzviofer@netanya.ac.il (O. Zwikael), yuvalco@openu.ac.il (Y. Cohen), sadeh@hait.ac.il (A. Sadeh).
[1] A visiting faculty in the School of Management at Ben-Gurion University of the Negev, Israel.

0263-7863/$30.00 © 2005 Elsevier Ltd and IPMA. All rights reserved.
doi:10.1016/j.ijproman.2005.11.002

activities included in the Critical Chain by feeding and resource buffers. According to this approach, activities can be delayed, without delaying the duration of the entire project. Models include in the "Delay" concept sometimes require a "late start" approach, based on the natural urge of delaying any activity to its latest schedule, called the "student syndrome" [13]. The selection of an approach to be used in the project is a critical one, impacting all managerial aspects – the level of project risk, the chance to finish the project on time, cash flow, etc.

The objectives of this paper are to analyze the managerial meaning of each scheduling approach, analyze the existing models included in each approach and suggest an improved model. First, let us review the literature in this area.

2. Literature review

Scheduling a project with a limitation of resources is a common problem that most project managers have to cope with [32]. This problem is referred to in the literature as "Resource Constrained Project Scheduling Problem" (RCPSP) and is described in scheduling and operations research literature ([20,19,6]).

This problem has a set of one or more limited resource types and a set of activities to be scheduled. Each activity has a fixed duration and during its execution there is a fixed resource consumption level for each type of resource. In project scheduling, delays in the execution time of certain activities occur when resources required by these activities are not available in sufficient quantities [14]. The aim is to find a feasible schedule with the shortest duration. The RCPSP can be easily formulated, yet it is very hard to find an exact solution for it, especially for large-scale projects [6].

The RCPSP is known to be Non-Polynomial (NP) hard [6,18–20]. This means that none of the exact optimal techniques can solve the problems faced by large projects in a timely manner. Some of the known solutions for this problem will now be introduced.

Pritsker et al. [24] solved the RCPSP as an Integer Programming (IP) problem, using binary variables. Peterson and Huber [23] formulated the problem as a 0–1 integer linear program. Kaplan [16], Alvarez-Valdes and Tamarit [3], Mingozzi et al. [22], Kolisch [18] and Klein [17] improved the mathematical programming formulation of this problem. The Branch and Bound concept has also been used to solve RCPSP problems [1,7]. Demeulemeester and Herroelen [10] introduced an efficient branch and bound approach, which allowed for delays in schedule. This technique is most beneficial for addressing computer memory, search strategy and strong lower bound [8,9]. Sperecher et al. [26] defined "active" and "semi-active" schedules in the realm of project scheduling and their definition is used in our study.

According to Icmeli et al. [14], by the late 1970s, there were already more than 100 heuristic procedures and many exact solution techniques available for RCPSP. Analyzing these quoted techniques for solving the RCPSP, we may take into account two different approaches: *optimal* and *heuristic*.

The *optimal* approach includes three main groups of solutions: zero-one programming [23,22], dynamic programming [4] and implicit enumeration with branch and bound [10,27,29]. The NP-hard nature of the problem makes it difficult to reach an exact solution for realistic-sized projects. Hence, in practice, the use of *heuristics* is necessary. Tormos and Lova [30] classified heuristics for the RCPSP into four methodologies: (1) Priority rules based scheduling; (2) Truncated branch and bound; (3) Disjunctive arcs concepts and (4) Meta-heuristic techniques.

It is possible to categorize the solutions for this problem in order to ease the understanding of the two approaches [26]. Fig. 1 depicts the relationships among the different groups of schedules according to Non-Delay and Delay categorizations. Non-Delay schedules are subsets of Active schedules, themselves subsets of Semi-Active schedules, which are subsets of all feasible schedules. Sprecher et al. [26] showed that an optimal RCPSP schedule must be an Active schedule. A non-delay schedule typically contains near-optimal solutions. However, optimal schedules may or may not be a non-delay schedule.

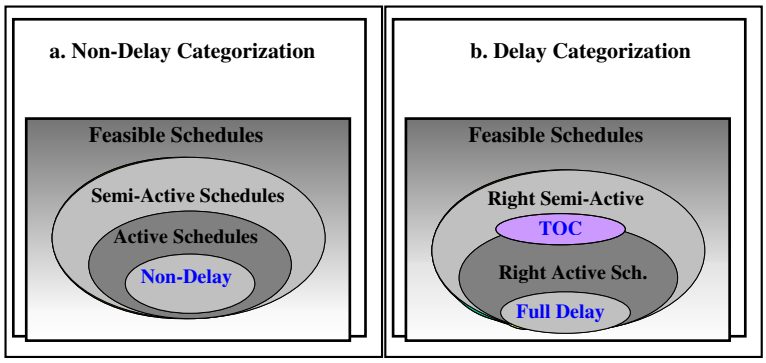

Fig. 1. Categorization of schedules using: (a) the Non-Delay concept and (b) the Delay concept.

Delay schedules are fully analogous to non-delay schedules. The only differences are that the project finish time replaces the project start time as the point of reference, and activities' right shifts replace activities' left shifts. Here, too, the optimum is contained within the Right Active solutions group, which contains the Full-delay schedules.

The main difference between the two approaches is that the Delay scheduling approach allows shifting the start dates of project activities to the right (delaying). Lichtenberg [21] claims that the optimal solution may be found in a midway use of the slack. Based on more than 300 projects of all kind, risk and uncertainty were analyzed to introduce the Merge Event Bias delays [28].

The most popular Delay concept is the TOC [12], which postulates a late-start policy when computed time-buffers allow it. The TOC model is based on the principle that every system has a constraint, and system performance can only be improved by enhancing the performance of the constraining resource. Critical Chain [13], an extension of TOC designed specifically for project management [25], aims at developing a sound schedule, using buffer management, in order to avoid project overruns. The Critical Chain methodology provides project managers with a heuristic framework and guidelines on how to plan, schedule, and control their projects, and it is up to the user of the method to complete the details. Critical Chain scheduling includes (1) reducing activity durations by eliminating safety margins, (2) identifying the Critical Chain, (3) creating a project buffer and (4) creating feeding buffers [5]. The Critical Chain results in start times which still provide time-buffers of security for the critical path, but are unmistakably Delay schedules based on late start of the project activities [5,25].

Based on models described in the literature survey, let us demonstrate the two approaches, delay versus non delay scheduling, in order to better understand the sequences of each on project planning. Fig. 2 represents a network diagram of a sample project, taken from [6]. Durations of each activity (in days) are presented on top of the activity's code and resource consumptions (per unit of time) are presented under the activity's code. For example, activity "D" requires two employees to work together for 3 days, or 6 man-days. Analyzing data, effort required for accomplishing this project equals 32 man-days. The maximal available resource level of this example is five employees. This means that theoretically the project cannot be finished in less than 7 days, since $32/5 = 6.4$.

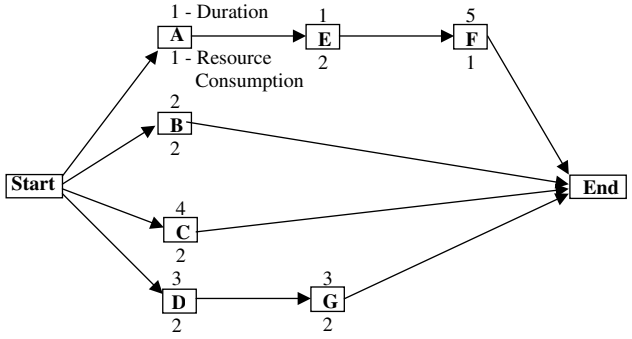

Fig. 2. Project network of the sample project.

Using the above example, the two scheduling approaches will be illustrated. Fig. 3(a) is derived using "Concerto" software, which uses the TOC methodology. The "Concerto" run was executed with the software's default definitions, without reducing project task durations. The final duration of the sample project according to TOC is even longer than the eight days that appears in the Gantt chart, since it includes an additional project buffer.

Yet, TOC is not the only solution that uses the Delay concept. Most project managers who prefer the "as late as possible" approach will choose to use one of many available project management packages (i.e. [31]. Hence, another "Delay" solution is presented in Fig. 3(b), this time using the function "as late as possible" in MS-Project. The duration of the project remained eight days, but the starting dates of activities "A", "B" and "E" were changed, compared to the previous schedule.

A Non-Delay schedule is introduced in Fig. 3(c). This schedule is the outcome of the resource leveling function of MS-project and PS-Next software. According to this schedule, the project starts with executing three activities together: A, B and F. The minimal project duration of the example project, under the constraints of resource level, is 7 days, as presented in Fig. 3(d).

After introducing the scheduling differences between these two concepts, let us examine the managerial aspects of the solutions. Managerial aspects may include project risk, resource load, negotiation for resources and project tram development. Comparing one Delay scheduling with a Non-Delay scheduling, we found the following managerial differences in the plans:

1. *Project risk* – a Non-Delay concept leaves spare time for schedule overruns of specific activities and hence reduces project risk. A Delay concept does not always leave room for mistakes, although the TOC approach treats this problem using buffers. Non-delay scheduling may also help us get closer to reaching minimal duration of a project, due to reducing the level of risk.
2. *Resource load* – Analyzing resource load we found that in the sample project, most resources in the Non-Delay concept will be used during the first six days of the project. The Delay concept does not require the work of all five team members until the fourth day of the project.
3. *Negotiation for resources* – Using delay approach may cause problems to a project manager who wants to ask for additional resources. The fact that he does not use all of his resources will make his arguments for additional resources looks odd. Moreover, choosing a delay technique and leaving some of his resources unutilized for a period of time may signal management that adding resources to that project was a mistake.

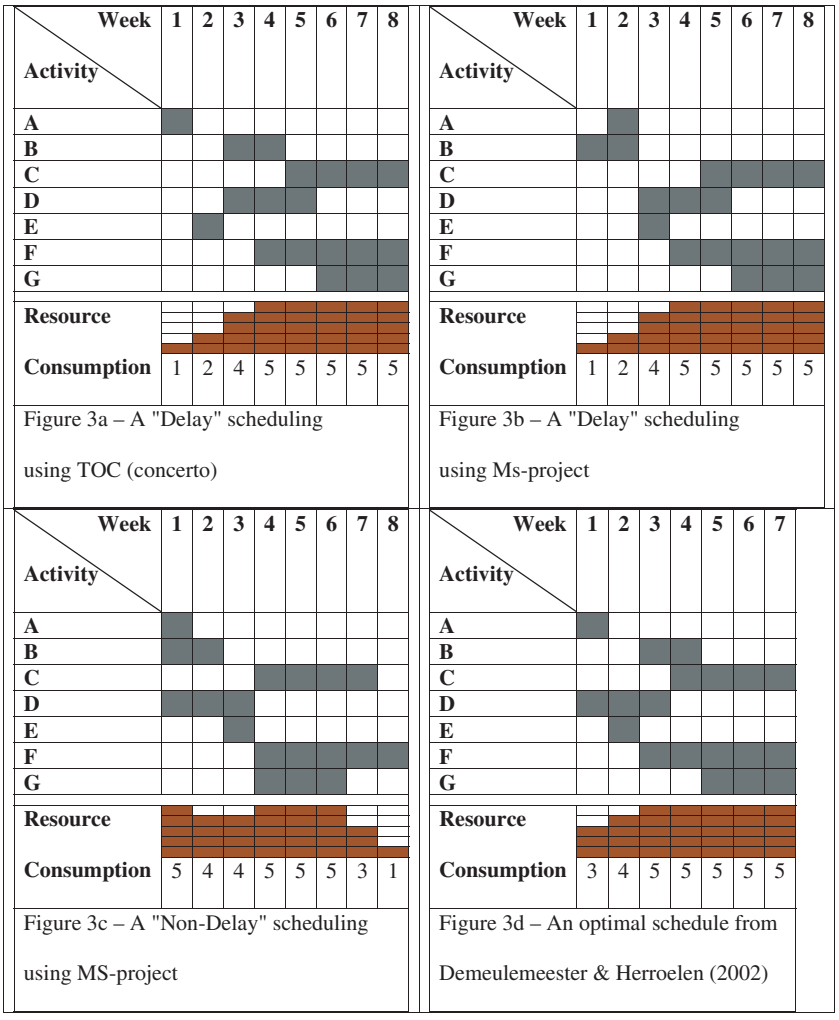

Fig. 3. Different project schedules for the example problem.

4. *Project team development* – Delay approach will cause a situation in which not all resources work together until a very late stage of the project. This may disturb the project manager in developing his project team as a united team.

Analyzing these managerial aspects we found many advantages to using the non-delay approaches for project scheduling. Due to large computing time required for exact solutions, the suggested direction of improving is developing an efficient heuristic. The next section introduced a new model for project scheduling, based on the non-delay approach.

3. A non-delay scheduling model

The proposed algorithm's logic is based on the Branch and Bound (BB) principal. However, instead of choosing from all potential solutions, the technique enumerates only the efficient feasible non-delay schedules (a small promising subset). In contrast to most BB techniques, the solution is developed in stages. Each stage of the proposed branch and bound is based on an addition of an activity j to a previous stage's partial schedule.

The algorithm starts with empty schedule at stage zero and a set of immediate feasible candidate activities (activities that have no precedence). The first stage generates a set of possible first activity candidates. The second stage generates all distinct, feasible non-delay schedules that can combine two activities, and so on. Overall for a project with J activities, the algorithm has J stages. It generates only the efficient and feasible Non-Delay schedules and eventually chooses the one that minimizes the project's duration. Two new concepts have been a key to the proposed method: (1) *a closed interval*, and (2) *an open set*.

A closed interval is a set of activities scheduled for a given time interval so that no other non-scheduled activity can be processed simultaneously during the same time interval (whether it is because of precedence constraints or resource availability). Fig. 4 depicts an example of a closed interval made of activities A and B (and the only one succeeding candidate – activity C). The closed interval lasts until the end of activity A (and is marked by a double arrow).

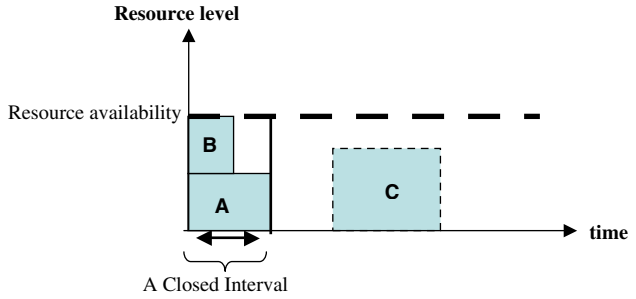

Fig. 4. An example illustration of a Closed Interval.

The algorithm forms consecutive *closed intervals* of maximal resource utilization, ensuring that no simple improvements could be made by moving of a single activity. This is illustrated in the example shown in the Fig. 5.

Open set – A set of activities, ongoing and scheduled directly after the closed intervals so that at least one *more* activity could be processed simultaneously directly after the closed intervals. Open-set is composed of an added activity and/or previously scheduled activities (in the open set) (Fig. 6).

The algorithm categorizes activities into four groups:

1. Activities in scheduled *Closed Intervals*.
2. Activities scheduled in an *Open-set* (more activities could be scheduled in a parallel manner).
3. Immediate, feasible candidate activities (activities that could be added to a certain schedule).
4. Other activities (to be scheduled during future stages).

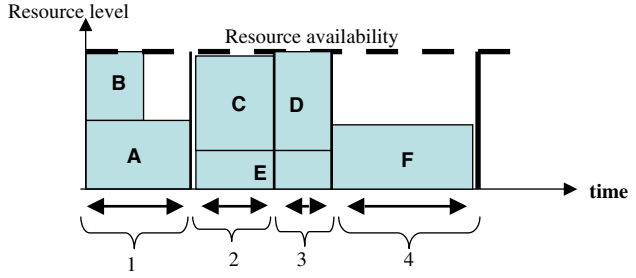

Fig. 5. An example illustrating a schedule of four consecutive closed intervals.

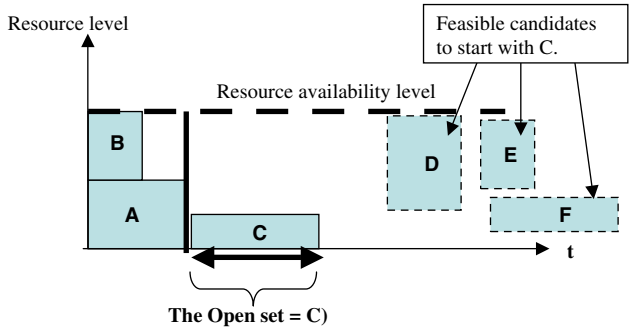

Fig. 6. An example illustrating the open set.

Each stage of the algorithm starts with the results of the previous stage. These results include the partial schedule, the closed intervals, the open set, and a list of immediate feasible candidate activities (for addition) for each schedule. Having this data for a particular schedule enables building a new branch for each activity in the feasible candidate set. Closing schedule intervals within a stage may sometimes require additional minor iterations.

The development of a schedule begins by moving an activity from the immediate feasible candidate activity set of a schedule to the Open-set of a schedule. This requires an update of resource consumption of the Open-set activities.

At this point, a decision is required regarding whether or not to close an interval. Closing an interval occurs only when no more feasible candidates could be started immediately after the last closed interval. If a feasible successor has been found, no closing is possible and the stage is finished for that particular schedule.

However, if no feasible successor could be found, an interval closing procedure starts. The first completion of an activity in the Open-set marks the end of the new closed interval (to be added to the closed intervals of the schedule). The duration of this closed interval is subtracted from the activities in the open set.

The feasible candidate set (that had been empty) is updated for deciding whether to close an additional interval (if the candidate set is empty) or not. Closing intervals may take several minor iterations until a feasible candidate (or more) is found. Having an immediate feasible candidate to be added to the Open-set immediately completes the stage for the schedule under consideration.

The calculations for each stage are computed for all schedules. At the end of the stage, inferior schedules are found by comparison and are disposed off, to limit the branching.

The proposed model has been computerized and is capable of calculating large-scale projects. Several projects were scheduled using this tool and compared to other known solutions, as is described in the next section.

4. Efficiency of the proposed technique

The objective of this section is to demonstrate the efficiency of the proposed solution. This will be demonstrated in three parts: (1) to ascertain that the proposed method finds the best non-delay schedule, (2) to compare the quality of the proposed solution to the optimal solution and to other heuristics and (3) to point out other advantages and limitations of the proposed method.

4.1. The best non-delay schedule

A *Closed Interval* is an interval during which no other activity can be added due to resource and precedence constraints. Having a continuous sequence of such intervals ensures a non-delay schedule. The proposed algorithm is

constructed accordingly and is based on the principle of sequentially filling and closing intervals.

The schedules of each stage are generated by considering each feasible successor to each of the partial previous schedules. These successors' start time is at the end of the closed intervals. Successors are added until no more activities can be added (due to both precedence and resources constraints) and another closed interval is formed. Thus, by enumerating all the feasible options for adding activities to the generated schedules, we ensure that all the relevant non-delay solutions are considered. The bounding operation culls only inefficient schedules for which a better schedule (with the same set of activities) can be found.

Since we consider all feasible and efficient non-delay schedules, all we need to do is choose the best one. However, if we also wanted to consider "Delay" options, we would have left some intervals "unclosed". This, of course, would have had a devastating effect on the complexity of the method.

4.2. Comparison to some results from the literature

The proposed method was also applied to several examples reported in the literature. We compared the project duration achieved by each method versus the method presented here and with respect to the minimal project duration that could have been achieved. The optimal solution is taken from literature cited in the right column. Other methods include "early start", representing Non-Delay scheduling and "late start", representing Delay scheduling, which were solved by resource leveling function in MS-project software.

Indeed, Table 1 shows that the proposed method improves the software solutions in four cases, and is equal to it in five more cases. Comparing the suggested method to the optimal solution, we found that the optimal solution was reached in 6 out of 9 cases, and a near optimal solution was found in the other cases. This comparison is somewhat biased since it includes deliberate examples where the optimal solution includes a delay. On the other hand, among the non-delay procedures, the proposed method will always achieve the best results.

4.3. Advantages and limitations of the proposed technique

The advantages of the proposed method are:

1. The number of algorithm stages is always the number of activities in the project, and the information summary of each stage is sufficient for generating the next stage. These characteristics have attenuating effects on the solution complexity.
2. The algorithm can deal with any number of limiting resources using the same number of stages.
3. The more constraints there are, the less the number of feasible schedules, and therefore the less complex the algorithms. The complexity decreases with:
   (a) the number of limiting resources,
   (b) the lower the resource availability levels,
   (c) the number of precedence constraints (immediate predecessors) and,
   (d) the number of activities that require most of the resource level.
4. Considering only the project duration, the solution is the best non-delay solution, which is a class of schedules known to contain near optimal solutions.
5. The algorithm can be made to generate multiple solutions by either generating a set of optimal solutions to choose from. Alternatively some bounding can be removed to generate more near optimal solutions. This could be important if a certain schedule is not feasible due to circumstances that cannot be incorporated into the model.

The limitation of the method lies in its being a heuristic and its inability to ensure the optimal solution (which may be a delay schedule). If the precedence constraints and the resource constraints are very loose, the complexity could grow in a non-polynomial manner. In such cases, more restrictive bounding could help reduce the complexity, but at the expense of giving up on finding the best non-delay solution and compromising on a "good" non-delay solution. Therefore, more restrictive bounding should be adopted only in extreme cases. Moreover, similarly to other RCPSP scheduling models, the proposed technique does

Table 1
Comparison of the proposed method to other methods

Project #	Resource limitation	Solution approach				Example source
		Minimal (optimal)	The proposed model	MS-Project "early start" resource leveling	MS-Project "late start" resource leveling	
1	8	15	15	17	15	[6] p. 207
2	5	7	8	9	8	[6] p. 265
3	3	22	22	22	22	[11] p. 224
4	2	5	6	6	6	[26]
5	3	12	12	12	12	Numerical example
6	9	35	35	35	35	Numerical example
7	2	34	34	37	40	[15]
8	5	17	17	17	17	[2]
9	20	20	21	23	20	[7]

not consider factors such as the size of the project delay loss, the cost of sudden establishing of resources, the start-up cost, the cost of idle resources and the uncertainty in the evaluation of activity duration.

5. Conclusion

In this paper, we presented the advantages of using non-delay scheduling and the limitations of existing non-delay models. The Non-Delay concept is efficient in reducing project risk and reducing project duration. Moreover, it often happens that a project manager who negotiated for resources in his project cannot afford to use a Delay concept in scheduling. In such cases, non-delay scheduling would yield the best solution.

A new branch and bound algorithm was introduced. This model efficiently enumerates all the efficient non-delay schedules and gives a near optimal (or in some cases, optimal) solution to the RCPSP. The model is fully computerized and provides a solution within seconds. The surprising simplicity of the model enables easy programming, insignificant computing time, and memory requirements.

The algorithm was compared with other heuristics and with the optimal solutions of a set of RCPSP problems from the literature and demonstrated its ability to find near optimal (and optimal) solutions. The difference between a global optimal solution and the solution provided by the suggested procedure is negligible. While finding the global optimal solution a tedious process and requires a complicated computing procedure, the suggested procedure converges to its optimal solution very fast. The factors affecting the complexity of the algorithm were discussed as well as the advantages and limitations of the algorithm.

References

[1] Agin N. Optimum seeking with branch and bound. Manage Sci 1966;13(4):176–85.
[2] Ahsan MK, Tsao D. A heuristic search algorithm for solving resource-constrained project scheduling problems. Asia-Pacific J Oper Res 2003;20(2):143–60.
[3] Alvarez-Valdes R, Tamarit JM. Heuristic algorithms for a resource-constrained project scheduling: A review and an empirical analysis. In: Slowinski R, Weglarz J, editors. Advances in project scheduling. Amsterdam: Elsevier; 1989. p. 113–34.
[4] Carruthers JA, Battersby A. Advances in critical path methods. Operation Res Quart 1966;17:359–80.
[5] Cohen I, Mandelbaum A, Shtub A. Multi-project scheduling and control: a process-based comparative study of the critical chain methodology and some alternatives. Project Manage J 2004;35(2):39–50.
[6] Demeulemeester ED, Herroelen WS. Project scheduling – a research handbook. Kluwer International; 2002.
[7] Demeulemeester E, De-Reyck B, Herroelen W. The discrete time/resource trade-off problem in project networks: a branch and bound approach. IIE Trans 2000;32:1059–69.
[8] Demeulemeester EL, Herroelen WS. New benchmark results for the resource-constrained project scheduling problem. Manage Sci 1997;43(11):1485–92.
[9] Demeulemeester E. Optimal algorithms for various classes of multiple resource-constrained project scheduling problems, unpublished Ph.D. Dissertation, Katholieke Universiteit Leuven, Belgium, 1992.
[10] Demeulemeester E, Herroelen W. A branch-and-bound procedure for the multiple resource-constrained project scheduling problem. Manage Sci 1992;38:1803–18.
[11] Globerson S, Shtub A. Project management – planning, executing and control, Dyunon, Tel-Aviv, Israel (in Hebrew). 2004.
[12] Goldratt EM. The goal. Great Barrington, MA: The North River Press; 1984.
[13] Goldratt EM. Critical Chain. Great Barrington, MA: The North River Press; 1997.
[14] Icmeli O, Erenguc S, Selcuk Z, Christopher J. Project scheduling problems: A survey. Int J Operation Prod Manage 1993;13(11):80–91.
[15] Ingalls RG, Morrice DJ. PERT scheduling with resource-constraints using qualitative simulation graphs. Project Manage J 2004;35(3):5–14.
[16] Kaplan LA. Resource-constrained project scheduling with preemption of jobs, Unpublished PhD Thesis, University of Michigan. 1988.
[17] Klein R. Scheduling of resource-constrained projects. Boston: Kluwer Academic Publishers; 2000.
[18] Kolisch R. Efficient priority rules for the resource-constrained project scheduling problem. J Oper Manage 1996;14(3):179–92.
[19] Kolisch R, Hartman S. Heuristic algorithms for solving the resource-constrained project scheduling problem: classification and computational analysis. In: Weglarz J, editor. Handbook of recent advances in project scheduling. Kluwer Academic Publishers; 1999. p. 147–78 [Chapter 7].
[20] Leung JY-T. Handbook of scheduling. CRC press/Chapman & Hall; 2004.
[21] Lichtenberg S. Management by projects achieving success in a changing world. Int J Project Manage 1999;17(4):266. Kidlington: 3 pgs.
[22] Mingozzi A, Maniezzo V, Ricciardelli S, Bianco L. An exact algorithm for project scheduling with resource-constraints based on a new mathematical formulation. Manage Sci 1998;44(5):714–29.
[23] Patterson JH, Huber WD. A horizon-varying, zero-one approach to project scheduling. Manage Sci 1974;20(6):990–8.
[24] Pritsker AAB, Watters LJ, Wolfe PM. Multi project scheduling with limited resources: a zero-one programming approach. Manage Sci 1969;16:93–108.
[25] Raz T, Barnes R, Dvir D. A critical look at critical chain project management. Project Manage J 2003;34(4):24–32.
[26] Sprecher A, Kolish R, Drexl A. Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem. Eur J Oper Res 1995;80:94–102.
[27] Stinson JP, Davis EW, Khumawala BM. Multiple resource-constrained scheduling using branch-and-bound. AIIE Trans 1978;10(3):252–9.
[28] Sunde L, Lichtenberg S. Net-present-value cost/time tradeoff. Int J Project Manage 1995;13(1):45, 5p.
[29] Talbot B, Patterson J. An efficient integer programming algorithm with network cuts for solving resource-constrained scheduling problems. Manage Sci 1978;24:1163–74.
[30] Tormos P, Lova A. A competitive heuristic solution technique for resource-constrained project scheduling. Ann Oper Res 2001;102(1):65.
[31] Zwikael O, Globerson S. Evaluating the quality of project planning: a model and field results. Int J Prod Res 2004;42(8):1545–56.
[32] Zwikael O, Shimizu K, Globerson S. Cultural differences in project management processes: a field study. Int J Project Manage 2005;23(6):454–62.