Automation in Construction 39 (2014) 145–151

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

Optimized acceleration of repetitive construction projects

Ibrahim Bakry *, Osama Moselhi, Tarek Zayed

Department of Building, Civil and Environmental Engineering, Concordia University, Montreal, Canada

ARTICLE INFO

Article history:
Accepted 5 July 2013
Available online 27 July 2013

Keywords:
Repetitive projects
Acceleration
Linear scheduling

ABSTRACT

Contractors and/or owners frequently need to accelerate the delivery of construction projects. Contractors may have to accelerate in order to benefit from contractual bonus, avoid penalties, recover from delays and/or avoid undesirable weather and site conditions. Owners, on the other hand, may order acceleration to meat business and operational opportunities. This paper presents an algorithm for schedule updating, dynamic rescheduling and optimized acceleration of repetitive construction projects. Schedule updating captures the exact progress on site. Dynamic rescheduling aims at capitalizing on the repetitive nature of the project to fine-tune the remaining portion of the project. Optimized acceleration presents an optimized time–cost trade-off that is tailored for repetitive projects. Through a set of iterative steps, the optimized acceleration procedure divides each activity into segments and identifies the segments that would shorten project duration if accelerated. For those identified segments, the ones with the least cost slope are selected and queued for acceleration. Through the proposed segmentation of activities this algorithm provides optimum allocation of additional acceleration resources, thus is rendered capable of identifying least cost acceleration plans. The algorithm allows users to select among different acceleration strategies such as working overtime, working double shifts, working weekends, and employing more productive crews. The presented algorithm maintains work continuity and accounts for typical and non-typical activities. The algorithm is implemented in a spreadsheet application, which automates calculations, yet allows users to fine tune the algorithm to fit the project at hand. The developed algorithm is applied to a case study drawn from literature in order to illustrate its basic features and demonstrate its accuracy.

1. Introduction

Repetitive construction projects are identified as construction projects formed of recurring units, each unit consisting of the same group of sequential activities. This unique characteristic paves the way for making considerable savings on time and cost by maintaining the continuity of crews and different resources involved in repetitive projects. Maintaining work continuity in repetitive projects helps achieve those time and cost savings through maintaining a constant workforce by reducing firing and hiring of labor, retaining skilled labor, maximizing use of learning curve effect and minimizing equipment idle time [12]. However, maintaining work continuity forms an additional constraint when planning and managing repetitive projects. Consequently, using traditional scheduling and planning tools and techniques to manage repetitive projects has been widely criticized [24], which highlights the need for developing special tools and techniques specifically designed to suit repetitive projects.

Shortly after the launch of critical path method (CPM) as a scheduling technique for non-repetitive projects in the mid-1950s, researchers started investigating enhancements and additions to the CPM technique. Possibly the two most famous and most practical outcomes of researchers' efforts were resource leveling and schedule compression techniques [21]. This paper is concerned with schedule compression, which is sometimes referred to as project time reduction, least-cost expediting, project compression, least-cost scheduling, project cost optimization, optimized scheduling, scheduling with constraints, project acceleration and project crashing. Schedule compression is about finding the delicate balance between the increase in direct cost, due to assigning additional resources and the decrease in indirect cost, due to shorter project duration. Schedule compression techniques can be divided into two main groups, heuristic and optimization techniques. Heuristic techniques are simpler and require less computational effort but present good solutions that are not necessarily the optimum solutions. Examples of heuristic methods are genetic algorithms [17,27] and harmony search [8]. On the other hand, optimization techniques find the optimal solution but are more difficult to create, need considerable computation effort, and are not efficient when handling large scale projects [25]. Optimization techniques include integer programming [22], linear programming [18] and dynamic programming [4]. Recent but not so common additions to the common scope of the schedule compression problem include utilizing discounted cash flows and considering maximizing profit as a criterion instead of reducing total project cost [1,3,25]. The above mentioned techniques address different aspects of scheduling compression, but all of them only address traditional non-repetitive project schedules,

* Corresponding author.
E-mail address: bakryibrahim@yahoo.com (I. Bakry).

0926-5805/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.autcon.2013.07.003

thus rendered not suitable for repetitive projects as will be detailed later.

In comparison to traditional projects, repetitive projects usually have a smaller number of activities which makes exact optimization techniques a feasible optimization option. For repetitive projects, techniques are available for performing different optimization procedures at the initial scheduling stage but they perform schedule acceleration. Linear programming was used for single and multi-objective optimization [16,23] and dynamic programming was used as well [5]. These optimization techniques were designed to find crew formations - and optimum interruption vectors (El-Rayes1997) - that would yield a schedule with optimum duration and/or cost. Genetic algorithms (GA) were used as a heuristic technique [14], and in some efforts GA were used in conjunction with dynamic programming [7]. These techniques are capable of considering different alternatives and identifiying an optimum solution, all having varying yet acceptable degrees of practicality at the initial scheduling stage. However, when it comes to accelerating a project, each activity is divided into segments and each segment can be accelerated using different strategies, and this acceleration is performed using incremental assignment of acceleration resources, which results in an almost infinite number of possibilities. Therefore investigating the whole schedule for acceleration would lead to a lot of redundant calculations. That being said, the need is established to adopt an approach that is capable of identifying critical segments of activities and nominates these segments for acceleration, which is bound to save a lot of computational time and effort. Moreover, this problem calls for an algorithm that is capable of incremental assignment of additional resources until an optimum solution is reached, instead of investigating every single positive solution.

This paper presents an algorithm for schedule updating, dynamic rescheduling and optimized acceleration algorithm for repetitive construction projects. Two types of updating are incorporated in the algorithm, a traditional updating procedure where site updates are utilized to bring the executed part of the project up to date, and a dynamic rescheduling procedure that allows for utilizing the exact durations of the executed portion of each activity to re-evaluate and reschedule the duration and the time buffers of the remaining portion of the schedule. After updating the project's schedule, the algorithm performs a heuristic time–cost tradeoff that is custom-made for repetitive projects in an effort to locate the optimum plan for accelerating the project.

2. Updating and dynamic rescheduling

More often than not, construction projects do not proceed exactly as planned. This makes the process of schedule monitoring and updating a basic component of any project management plan. The presented algorithm begins by schedule updating, followed by dynamic rescheduling and ends by optimized acceleration of repetitive projects. Updating aims at incorporating the actual data of the executed portion of the project into the schedule. This is performed through replacing planned start and end dates with the actual dates of completed activities. In addition, the user is given the opportunity to revise quantities of activities and productivity rates of crews for the uncompleted works of the project. This revision is deemed useful because after the project starts the user has a better knowledge of the project environment, which enables him to refine the estimates for the rest of the schedule. Based on the actual dates entered and the estimates revised, the schedule is recalculated and the project end date is changed accordingly.

The second task before starting the optimized acceleration procedure is dynamic rescheduling. Dynamic rescheduling aims at capitalizing on the repetitive nature of the project. It utilizes the experience and knowledge of project managers and their assessment of the project performance up to reporting date. The dynamic rescheduling aims at re-sizing the time buffers inserted between successive activities. Time buffers are usually built according to the expected uncertainty level affecting the activities involved. Activities in a highly uncertain environment are more likely to suffer from interruptions and/or delays and hence are followed by bigger buffers to provide adequate protection to work continuity, and vice versa. It is common practice to rely on experience when estimating the size of buffers. This subjective approach highlights the need for monitoring and fine-tuning time buffers once the project starts and delays occur. The proposed technique works under the assumption that buffers are calculated for each unit and then aggregated to form a single buffer for each activity. The algorithm at hand utilizes relative weights to add delays that happened and those estimated for the remaining activities. The relative weights are calculated based on the number of units completed to the total number of units, and the number of units to be completed to the total number of units, respectively. The equation below shows how the buffer size is fine-tuned using weighted average.

$$B_n = \left(\frac{U_c}{U_t} \times D_a\right) + \left(\frac{U_r}{U_t} \times B_o\right) \quad (1)$$

where

- B_n is the new buffer per unit
- U_c is the number of units completed
- D_a is the average delay per unit for the completed units
- U_r is the number of units remaining
- B_o is the originally estimated buffer per unit
- U_t is the total number of units

As such, for an activity of 10 day duration scheduled to be repeated throughout 40 units with an estimated delay (buffer) of 1 day per unit, the total activity duration will be 400 days followed by a 40 days buffer. If the schedule is updated after 10 units have been completed and it was noticed that the average delay per unit is 1.2 days instead of 1 day. The new buffer per unit can be expressed as:

$$B_n = \frac{(10 \times 1.2) + (30 \times 1)}{40} = 1.05 \text{ days of delay per unit}$$

This example shows how the estimated buffer for each unit is refined based on the performance experienced on site up to reporting date. When the 1.05 days of anticipated delays for each unit is aggregated, a buffer of 31.5 days will be needed to cover the remaining 30 units. The described updating and dynamic rescheduling flowchart are illustrated in Fig. 1.

3. Identifying activities to accelerate

Selecting the right activities to accelerate in a repetitive project is a key step toward successful project acceleration. Accelerating the wrong activity will lead to spending more money without any effect on a projects duration, or spending more money than needed. In traditional projects such a decision is made easier by the existence of a critical path. In CPM every project schedule includes a critical path or paths, which is a group of sequential activities with a total duration longer than other paths, hence, determining the project's total duration. Crashing any activity on this path would shorten the project's duration. This remains valid until that critical path is no longer the longest path in the network [21]. Things are different in repetitive projects, as many alternatives exist in literature for identifying which activities control a repetitive project's duration. Two well-known methods to identify the critical activities controlling a repetitive projects' total duration are "Controlling Activity Path" for schedules built using Linear Scheduling Model (LSM) [9,10], and "Controlling Sequence" for schedules built using Repetitive Scheduling Method (RSM) [11]. Many comparisons have been made between these two methods highlighting their advantages and disadvantages. Although both successfully identify critical activities, both techniques only account for sequential activities with

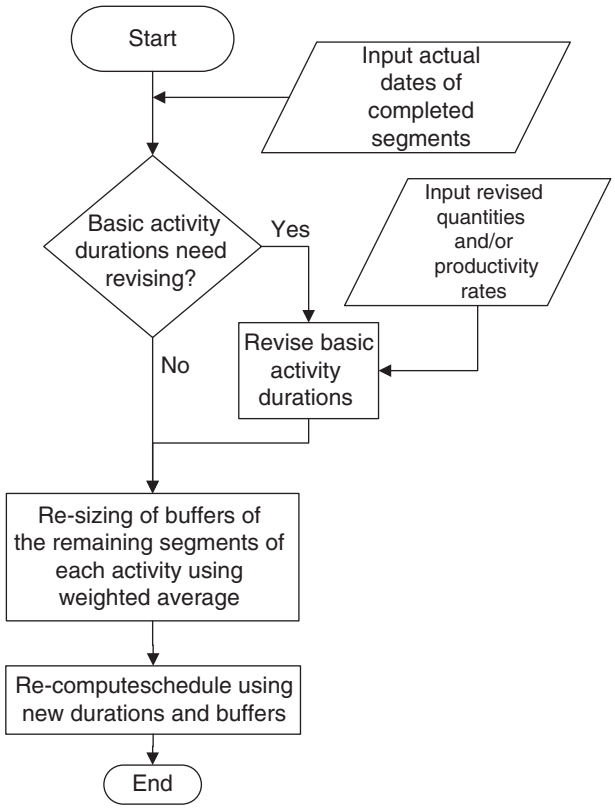

Fig. 1. Updating and dynamic rescheduling flowchart.

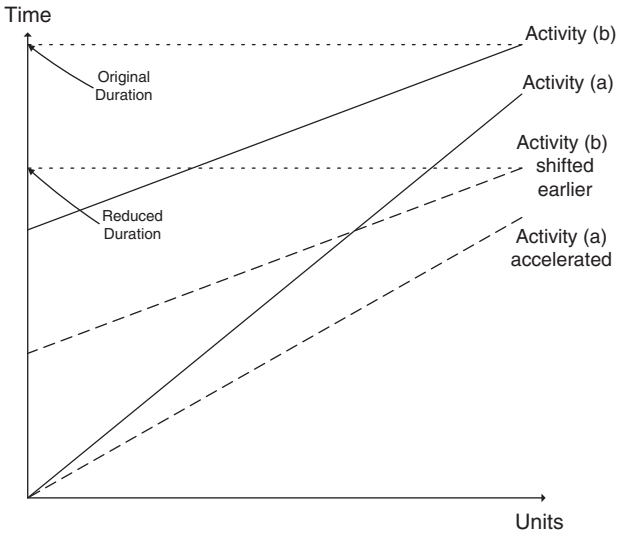

Fig. 2. Effect of acceleration on repetitive activities.

constant production rates (typical activities), therefore limiting their practical use [20].

Although in the context of this algorithm Linear Scheduling Model (LSM) is used to calculate and represent schedules, a different technique was used to identify which activities to accelerate than targeting the controlling activities defined earlier. The technique adopted in this research is a modified version of the technique presented by [13]. In their technique, Hassanein and Moselhi aimed at identifying activities that are least aligned with their successors. As the case is in all repetitive projects scheduling techniques, when a successor activity has a higher rate than its predecessor, its starting time is determined by backward calculations based on the predecessor activity ending time. Consequently, reducing the duration of a least aligned predecessor will advance its ending time, thus enabling an earlier start for its successor. Fig. 2 shows an example of a repetitive project's schedule, and shows how accelerating the least aligned activity leads to shortening project duration.

Previous repetitive projects acceleration techniques chose a critical activity to accelerate throughout all units. For example, they would choose to accelerate activity "earthwork" for every kilometre throughout a highway construction projects. This might be the correct choice in some cases where units are all typical. Typical activities are activities comprising units having the same quantity for each unit and are performed by crews having same productivity rates. This leads to a repetitive schedule formed of activities represented by straight lines with a single slope for each line. Once an activity is identified to be less aligned with its successor or predecessor (i.e. critical), it will continue to be less aligned through all units. However, this is a very special case. The general case, which this algorithm carefully addresses, is that projects consist of non-typical activities. Non-typical activities have different quantities for each unit, and utilize crews and equipment operating at different productivity rates. Repetitive project's schedules consisting of non-typical activities are represented by broken lines with varying slopes for each unit. An example of which can be seen in Fig. 3. This general case makes it unlikely that the least aligned activity will be the same throughout all units. The presented algorithm identifies the least aligned activity separately for each unit. So instead of identifying an activity to be accelerated throughout all units, the algorithm might select different segments in different activities to accelerate utilizing different strategies. By doing so, two main benefits are realized. Firstly, needed duration shortening can be achieved through assigning less additional accelerating resources, as these excess resources were previously assigned to non-critical parts of the identified activity, thus not reducing total project duration. Secondly, it helps avoid productivity loss due to assigning too many overtime hours, as literature shows that maintaining a 1 hour per day of overtime for 4 weeks, results in a 16% less efficient process than keeping regular working hours for 4 weeks [2].

4. Acceleration strategies

There are common acceleration strategies, which project managers often use when accelerating repetitive projects [13,20]. These strategies include: (1) working overtime; (2) working double shifts; (3) working weekends or (4) employing more productive crews. Clearly each of the

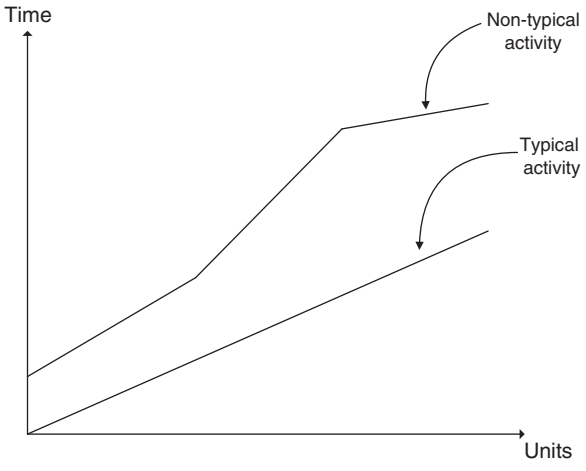

Fig. 3. Typical and non-typical activities.

acceleration strategies mentioned above is accompanied with additional costs. Examples of these additional costs are increased direct costs as in labor wages and equipment running costs, and loss of productivity due to congestion in case of increased crews size [2]. On the other hand, as projects' total duration decrease, indirect costs also decrease. Two additional strategies are also considered, namely [6,26]: (5) relaxing activities and (6) introducing intentional work breaks. Those two strategies can have the effect of decreasing projects total duration only if applied to converging activities [6,26]. As displayed in Fig. 4, converging activities are activities having a higher rate than their predecessor and a lower rate than their successor. By relaxing a converging activity's rate or introducing an intentional break it can start earlier, and its successor can start earlier. Relaxing an activity might cost less money as it leads to assigning fewer resources, however it might cost more. For example relaxing an activity could mean increased renting period for equipment and increased supervision man hours. Similarly introducing intentional breaks comes at an increased cost, especially in equipment extensive projects like highway projects as rented or procured equipment would be left idle on site. The associated costs leave strategies (5) and (6) less likely to be chosen by a project manager to accelerate a project; however, they are included as options in the proposed algorithm.

5. Optimized acceleration procedure

The optimized acceleration procedure automates the identification of which activities of a repetitive project to accelerate and which accelerating strategies to use. The starting point is after successfully updating the repetitive project schedule and revising time buffers. Identifying less aligned activities is done using a technique similar to minimum moment algorithm used for resource leveling [15]. The algorithm calculates the areas trapped between lines representing successive activities, and then calculates the moment these areas cause around an imaginary centerline. Ω is a value revealing activity alignment; it is calculated by subtracting the moment of area of an activity from the moment of area of the predecessor activity. Less aligned activities result in bigger areas with bigger eccentricities, hence resulting in bigger moments, and bigger values of Ω, and vice versa. One of the features introduced in this research is that the alignment calculations are carried out for each unit separately instead of the whole activity. Although this requires more calculations, yet it allows more efficient allocation of additional acceleration resources.

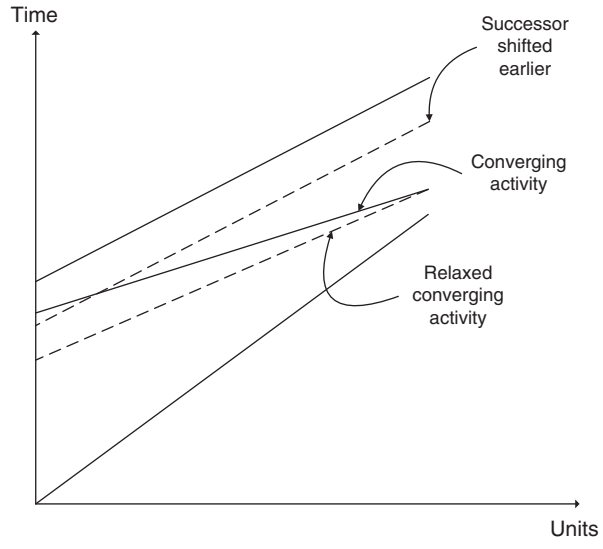

Fig. 4. Acceleration by relaxing converging activities.

A deeper look at the algorithm at hand reveals that studying each unit separately has a weakness. This approach would identify the criticality of an activity based only on the productivity of the assigned crew, regardless of the number of crews working on the same activity in other units. For example if 3 crews assigned to an activity each producing 1 unit per day, their total productivity is 3 units per day. Comparing each activity's rate locally (at each unit separately) and neglecting the global perspective would identify this activity to be more critical than an activity assigned to a single crew producing 2 units per day, although clearly the later activity progresses at a slower rate. To address this issue the equations for calculating areas and their moment around the imaginary center line had to be modified to include also the number of crews, which enables correctly conveying the rate of an activity according to the productivity and number of crews assigned. Fig. 5 together with the following equations below demonstrate how identifying the least aligned activity in a repetitive project is formulated. The activity with the largest value for Ω is the least aligned activity.

$$\text{Area}_{(i)} = \text{L.Side}_{(i)} + \text{R.Side}_{(i)}/2 \tag{2}$$

$$\text{L.Side}_{(i)} = S_{(i)} - S_{(i-1)} \tag{3}$$

$$\text{R.Side}_{(i)} = \left[F_{(i)} - F_{(i-1)}\right] - \left[D_{(i)}(n-1)/n\right] + \left[D_{(i-1)}(n'-1)/n'\right] \quad \text{if } \text{L.Side}_{(i)} > \text{R.Side}_{(i)} \tag{4}$$

$$C_{(i)} = \left(\text{L.Side}_{(i)} + 2 \times \text{R.Side}_{(i)}\right) / \left[3\left(\text{L.Side}_{(i)} + \text{R.Side}\right)_{(i)}\right] \tag{5}$$

$$e_{(i)} = C_{(i)} - 0.5 \tag{6}$$

$$\text{If } \text{L.Side}_{(i)} > \text{R.Side}_{(i)} \quad C_{(i)} = \left(\text{R.Side}_{(i)} + 2 \times \text{L.Side}_{(i)}\right) / \left[3\left(\text{R.Side}_{(i)} + \text{L.Side}_{(i)}\right)\right] \tag{7}$$

$$e_{(i)} = 0.5 - C_{(i)} \tag{8}$$

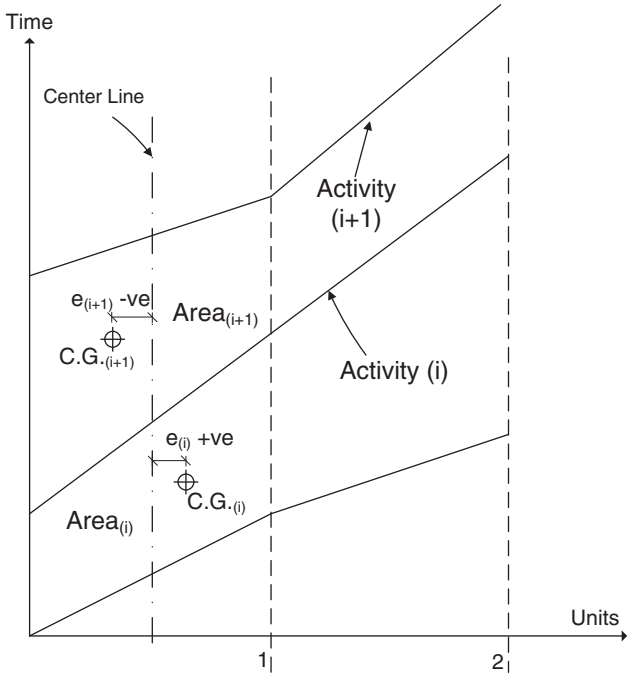

Fig. 5. Identifying least aligned activities.

$$\Omega_{(i)} = \text{Area}_{(i)} \times e_{(i)} - \text{Area}_{(i+1)} \times e_{(i+1)} \quad (9)$$

where

- $\text{Area}_{(i)}$ is the area between activity (i) and $(i-1)$
- $S_{(i)}$ is the start time of activity (i)
- $S_{(i-1)}$ is the start time of activity $(i-1)$
- $F_{(i)}$ is the end time of activity (i)
- $F_{(i-1)}$ is the end time of activity $(i-1)$
- $D_{(i)}$ is the duration of activity (i)
- $D_{(i-1)}$ is the duration of activity $(i-1)$
- n is the number of crews assigned to activity (i)
- n' is the number of crews assigned to activity $(i-1)$
- $C_{(i)}$ is the distance between the area's edge to the area's center of gravity
- $e_{(i)}$ is the eccentricity of the center of gravity to the center line of area (i)
- $\Omega_{(i)}$ is the value reflecting the degree of misalignment of activity (i)

For different reasons, planners often introduce time or space buffers between successive activities, the presence of a buffer of any size does not affect the identification of the least aligned activity, as buffers shift activities without changing their alignment. As shown in Fig. 5, Ω is calculated for each activity in each repetitive unit separately, to identify the least aligned activity for each unit. Activities with negative values of Ω are converging activities. These activities are considered for acceleration through relaxing their rate or applying intentional work stoppages. After identifying which activity to accelerate, now the acceleration strategy has to be carefully selected. In this research, selection is based on the least associated costs. Here it is up to the project manager to provide the cost of each acceleration strategy in the form of cost per hour or cost of intentional work stoppage. Then this cost is translated into a cost slope for each strategy, where each strategy's cost is in the form of cost per day of duration reduction. In the developed algorithm, priorities for acceleration are established based on the least cost slope. This enables finding least cost acceleration plans. In case there is more than one segment with the same cost slope, priorities are generated based on Ω value for each competing activity (the higher the Ω value is the higher the priority).

Fig. 6 shows the algorithm flowchart. The formulated algorithm is implemented in a spread sheet application that automates all calculations. The application accepts projects initial schedule, along with activities quantities and crews rates. It allows updating the schedule with exact dates, and re-sizing of inserted buffers. The application automatically runs the needed calculations for Ω values for each segment, and identifies the activity nominated for acceleration in each segment. As the user responds to the identified nominations and assigns acceleration resources, the application continuously re-identifies the next activity to be accelerated in each segment in an iterative manner until the required duration reduction is achieved.

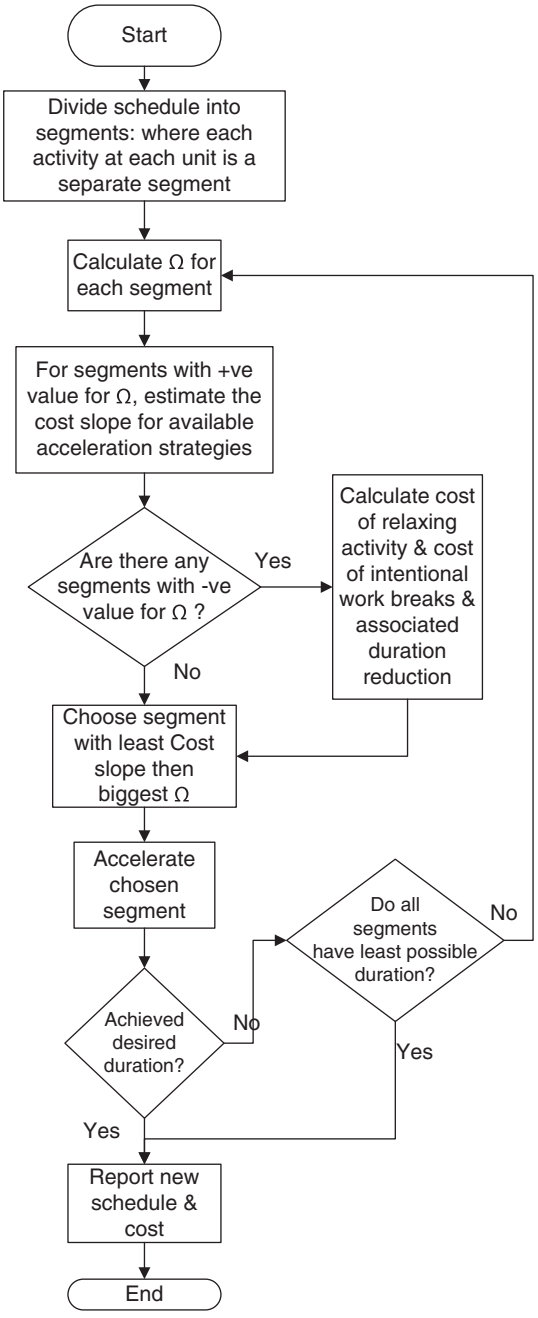

Fig. 6. Optimized acceleration flowchart.

6. Case study

The presented algorithm was applied to a case study drawn from literature to demonstrate its basic features. The case study presented in literature [5] is a 15 km three-lane highway project, consisting of 5 repetitive activities. These activities, in their order of precedence, are: (1) cut and chip trees; (2) grub and remove stumps; (3) excavation; (4) base; and (5) paving, and all precedence relations are finish to start, with no lag time. The project is divided into 15 segments of equal lengths, each is 1 km. This project includes typical activities and non-typical activities. Typical activities are (4) base and (5) paving, as they have same quantities for each segment and same crew productivity for different crews. While activities (1) cut and chip trees, (2) grub and remove stumps and (3) excavation are non-typical activities, as their quantities change from one unit to another and also their different crews have different productivities. The included activities are all sequential except for activity (3) excavation; this activity is non-sequential as its starts by units 4 to 1, then units 5 to 15. The original project data can be found in [5], the initial schedule had a normal duration of 84 days.

When Hassanein (2005) applied his repetitive projects acceleration technique, the goal was to accelerate the project to complete it in 77 days. And he only considered working overtime hours as an acceleration strategy, and the cost of overtime hours was considered the same for different crews. Be assuming so, he neutralized all variables accept the segment to accelerate. Accordingly the case study was specifically built to test the capability of identifying which activities to accelerate. Also the limit for additional overtime hours per day was set to 4 hours per day for activities Cut and chip trees, Grub stumps and Base; while

Table 1
Added overtime hours.

Unit	Cut and chip trees		Grub and remove stumps		Excavation		Base		Paving	
	Over-time (hr)	New duration (days)	Over-time (hr)	New duration (days)	Over-time (hr)	New duration (days)	Over-time (hr)	New duration (days)	Over-time (hr)	New duration (days)
1	3	4	–	3	3	6	–	10	–	8
2	3	4	–	3	2	4	3	8	–	8
3	2	6	–	4	3	6	–	10	–	8
4	–	4	–	3	3	5	–	10	–	8
5	–	6	–	5	–	6	–	10	–	8
6	–	8	–	6	–	7	–	10	–	8
7	3	10	3	7	–	6	–	10	–	8
8	1	8	3	5	–	5	–	10	–	8
9	–	7	–	4	2	6	–	10	–	8
10	–	7	–	4	1	5	–	10	–	8
11	–	6	–	5	3	6	–	10	–	8
12	–	4	–	3	3	4	3	8	–	8
13	-	4	–	2	3	4	3	8	–	8
14	–	4	–	3	3	6	–	10	–	8
15	–	4	–	3	3	4	3	8	–	8

it was set to 3 hours per day for activities Earthmoving and Paving. His acceleration technique achieved the required end date by accelerating activities “grub and remove stumps” and “excavation” throughout all units, by adding three and two hours of overtime per day respectively.

The modified algorithm proposed in this research was then applied to the selected case study. The original schedule was input to the developed spreadsheet application. The spreadsheet application automatically identified the critical activities to be considered for acceleration for each unit. Additional overtime hours were added to the identified activities. Automatically the schedule was updated and the new critical activity segments were identified and more additional overtime hours were added in an iterative manner, taking into consideration the limit of available overtime hours for each crew. The final 77 days duration schedule targeted by Hassanein (2005) was achieved by the combination of overtime hours displayed in Table 1.

Fig. 7 shows the original schedule (in solid lines) and the accelerated schedule (in dashed lines), while Table 2 shows the original duration and results of both acceleration algorithms. It can be seen that the modified acceleration algorithm presented in this research achieved the same duration reduction by assigning 59 instead of 75 overtime hours, thus significantly reducing acceleration costs. This reduction in additional resources utilized reflects the algorithms' capability of better targeting of additional acceleration resources.

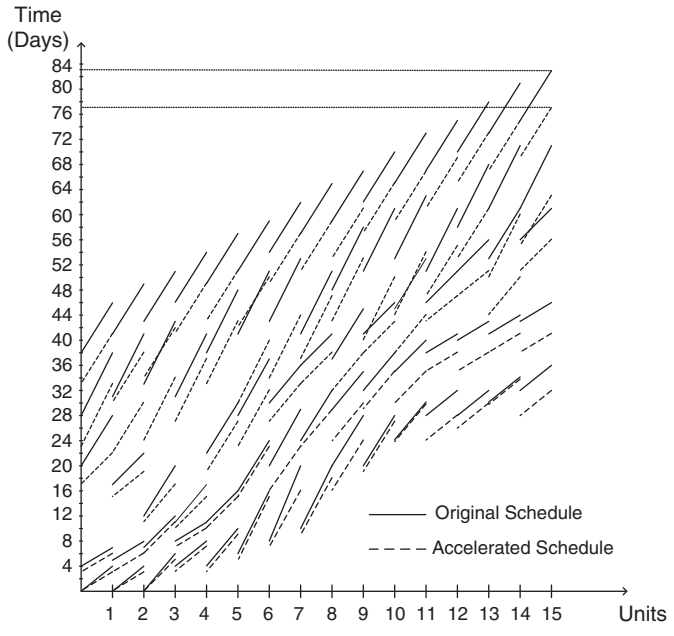

Fig. 7. Original and accelerated schedules.

7. Concluding remarks

This paper presented an algorithm for schedule updating, dynamic rescheduling and optimized acceleration of repetitive construction projects. The algorithm allows for capturing project progress on site. It capitalizes on the repetitive nature of projects by capturing their respective cumulative onsite performance and utilizing it to dynamically reschedule the remaining work on each project. The algorithm offers a targeted allocation of additional acceleration resources that are selected primarily based on least cost slope and secondly on activity criticality. It can accommodate typical and non-typical activities, and sequential and non-sequential activities. The developed spreadsheet application enables simple and straightforward use of the algorithm, and allows users to fine tune the algorithm to fit project conditions at hand. The algorithm allows for selection of acceleration strategies from six available alternatives while maintaining resource continuity. The incremental assignment of additional acceleration resources allows for considering more alternatives with the least possible amount of calculation time and effort, which is the main advancement over existing optimization techniques. The analyzed project case demonstrated the use of the developed algorithm and its ability to accelerate repetitive projects while using less additional resources. A number of limitations accompany the presented algorithm. These limitations include not accounting for any decrease in crew productivity that could result from site congestion or overtime hours. As well the incremental assignment of acceleration resources adopted by this algorithm does not maintain the “natural rhythm” of the activity that was described by [19]. Another limitation is that although the algorithm utilizes LSM to perform basic scheduling operations, the algorithm does not accommodate “Bar” and “Block” activities, which are sometimes utilized to represent special cases of repetitive and non-repetitive activities in a repetitive construction project [9]. Recommendation for future

Table 2
Results comparison.

Schedule	Total duration	Total added overtime hours
Original Schedule by [5]	83	–
Acceleration by Hassanein 2005	77	75
New algorithm	77	59

expansions of the presented algorithm include researching methodological buffer building and insertion techniques that scientifically account for imbedded uncertainty, incorporating the effects of learning curve and loss of productivity due to changes in resource assignment, and researching new visual schedule representations.

References

[1] M.A. Ammar, Optimization of project time–cost trade-off problem with discounted cash flows, Journal of Construction Engineering and Management, ASCE 137 (1) (2011) 65–71.
[2] D. Arditi, O. Tokdemir, K. Suh, Challenges in line-of-balance scheduling, Journal of Construction Engineering and Management, ASCE 128(6) (2002) 545–556.
[3] A.M. Elazouni, F.G. Metwally, Finance-based scheduling: tool to maximize project profit using improved genetic algorithms, Journal of Construction Engineering and Management, ASCE 131 (4) (2005) 400–412.
[4] S.E. Elmaghraby, Resource allocation via dynamic programming in activity networks, European Journal of Operational Research 64 (1993) 199–215.
[5] K. Elrayes, Optimized Scheduling for Repetitive Construction Projects. (PhD thesis) Department of Building, civil and Environmental Engineering, Concordia University, Montreal, Quebec, 1997.
[6] K. El-Rayes, O. Moselhi, Resource-driven scheduling of repetitive activities, Journal of Construction Engineering and Management, ASCE 16 (1998) 433–446.
[7] A.S. Ezeldin, A. Soliman, Hybrid time–cost optimization of non-serial repetitive construction projects, Journal of Construction Engineering and Management, ASCE 135 (1) (2009) 42–55.
[8] Z.W. Geem, Multi-objective optimization of time cost trade-off using Harmony Search, Journal of Construction Engineering and Management, ASCE 136 (6) (2010) 711–716, (ASCE).
[9] D.J. Harmelink, Linear Scheduling Model: The Development of a Linear Scheduling Model With Micro Computer Applications for Highway Construction Project Control. (PhD thesis) Iowa State University, Ames, Iowa, 1995.
[10] D.J. Harmelink, J. Rowings, Linear scheduling model: development of controlling activity path, Journal of Construction Engineering and Management, ASCE 124(4) (1998) 263–268.
[11] R. Harris, P. Ioannou, Scheduling projects with repeating activities, Journal of Construction Engineering and Management, ASCE 124(4) (1998) 269–278.
[12] A. Hassanein, Planning and Scheduling Highway Construction Using GIS and Dynamic Programming. (PhD thesis) Concordia University, Montreal, Quebec, 2002.
[13] A. Hassanein, O. Moselhi, Accelerating linear projects, Journal of Construction Engineering and Management, ASCE 23(4) (2005) 377–385.
[14] T. Hegazy, N. Wassef, Cost optimization in projects with repetitive non-serial activities, Journal of Construction Engineering and Management, ASCE 127 (3) (2001) 183–191.
[15] M. Hiyassat, Applying modified minimum moment method to multiple resource leveling, Journal of Construction Engineering and Management, ASCE 127(3) (2001) 192–198.
[16] P.G. Ipsilandis, Multiobjective linear programming model for scheduling linear repetitive projects, Journal of Construction Engineering and Management, ASCE 133 (6) (2007) 417–424.
[17] S.S. Leu, C.H. Yang, GA-based multicriteria optimal model for construction scheduling, Journal of Construction Engineering and Management, ASCE 125 (6) (1999) 420–427.
[18] L. Liu, S.A. Burns, C.W. Feng, Construction time–cost trade-off analysis using LP/IP hybrid method, Journal of Construction Engineering and Management, ASCE 121 (4) (1995) 446–454.
[19] P. Lumsden, The Line-of-Balance Method, Pergamon Press Limited, Great Britain, 1968.
[20] K. Mattila, A. Park, Comparison of linear scheduling model and repetitive scheduling model, Journal of Construction Engineering and Management, ASCE 129(1) (2003) 56–64.
[21] O. Moselhi, Schedule compression using the direct stiffness method, Canadian Journal of Civil Engineering 20 (1) (1993) 65–72.
[22] A. Pagnoni, Project Engineering: Computer Oriented Planning and Operational Decision Making, Springer, Berlin, 1990.
[23] S. Perera, Resource sharing in linear construction, Journal of Construction Engineering and Management, ASCE 109 (1) (1983) 102–111.
[24] A.D. Russell, W.C.M. Wong, New generation of planning structures, Journal of Construction Engineering and Management, ASCE 119 (2) (1993) 196–214.
[25] A. Senouci, K. El-Rays, Time–profit trade-off analysis for construction projects, Journal of Construction Engineering and Management, ASCE 135 (8) (2009) 718–725.
[26] C. Srisuwanrat, The Sequence Step Algorithm: A Simulation-Based Scheduling Algorithm for Repetitive Projects With Probabilistic Activity Duration. (PhD thesis) Michigan University, Michigan, USA, 2009.
[27] D. Zheng, T. Ng, M. Kumaraswamy, Applying a genetic algorithm-based multiobjective approach for time–cost optimization, Journal of Construction Engineering and Management, ASCE 130 (2) (2004) 168–176.