



The use of buffers in project management: The trade-off between stability and makespan

Stijn Van de Vonder*, Erik Demeulemeester, Willy Herroelen, Roel Leus

Department of Applied Economics, KU Leuven, Leuven, Belgium

Received 29 March 2004; accepted 20 August 2004

Available online 30 November 2004

Abstract

During execution projects may be subject to considerable uncertainty, which may lead to numerous schedule disruptions. Recent research efforts have focused on the generation of robust project baseline schedules that are protected against possible disruptions that may occur during schedule execution. The fundamental research issue we address in this paper is the potential trade-off between the quality robustness (measured in terms of project duration) and solution robustness (stability, measured in terms of the deviation between the planned and realized start times of the projected schedule). We provide an extensive analysis of the results of a simulation experiment set up to investigate whether it is beneficial to concentrate safety time in project and feeding buffers, or whether it is preferable to insert time buffers that are scattered throughout the baseline project schedule in order to maximize schedule stability.

© 2004 Elsevier B.V. All rights reserved.

Keywords: Project scheduling; Schedule stability; Quality robustness; Buffers

1. Problem description

The vast majority of the research efforts in project scheduling over the past several years have concentrated on the development of exact and suboptimal procedures for the generation of a *baseline schedule* (*pre-schedule*, *predictive schedule*) assuming a deterministic environment and complete information. During execution, however, a

project may be subject to considerable uncertainty, which may lead to numerous schedule disruptions. The recognition that uncertainty lies at the heart of project planning has induced a number of research efforts in the field of project scheduling under uncertainty (for an extensive review of the literature we refer to Demeulemeester and Herroelen (2002) and Herroelen and Leus (2004b)).

Critical chain scheduling/buffer management (CC/BM)—the direct application of the Theory of Constraints (TOC) to project management (Goldratt, 1997)—has received a lot of attention in the project management literature. The

*Corresponding author.

E-mail address: stijnvandevonder@econ.kuleuven.be
(S. Van de Vonder).

fundamental working principles of CC/BM have been reviewed by Herroelen et al. (2002). CC/BM builds a baseline schedule using aggressive¹ median or average activity duration estimates. The safety in the durations of activities that was cut away by selecting aggressive duration estimates is concentrated in the form of a *project buffer* (PB) that is positioned at the end of the so-called critical chain. The critical chain (CC) is defined as the longest chain² of precedence and resource dependent activities that determines the overall duration of a project. The project buffer should protect the project due date from variability in the critical chain activities. *Feeding buffers* (FB) are inserted whenever a non-critical chain³ activity joins the CC. Clearly, the CC/BM idea is to protect the project due date against the disruptions that may occur during project execution. As such it can be viewed as a procedure for generating *makespan (due date) protective schedules*. Due date protection, however, is only one side of the coin and relates to the sensitivity of the project makespan to activity disruptions, i.e. to the *quality robustness* of the baseline schedule. For executing a project, on the other hand, the CC/BM approach does not rely on the buffered schedule but on a so-called *projected schedule*. This schedule is precedence and resource feasible and is to be executed according to the roadrunner mentality, i.e. the so-called gating tasks (activities with no non-dummy predecessors) are started at their scheduled start time in the buffered schedule while the other activities are started as soon as possible. The projected schedule is recomputed when disruptions occur. Neither the buffered schedule nor the projected schedule are constructed with a view to stability (*solution robustness*, i.e. the insensitivity of planned activity start times to schedule disruptions).

¹Goldratt (1997) proposes to build schedules by using activity durations that cover the time to do the work without any safety, based on a 50% confidence level rather than on the 80–90% confidence levels that he claims to be in common use in project management practice. These durations are called aggressive.

²If there is more than one critical chain, an arbitrary choice is made.

³A non-CC can be every chain of one or more activities that does not determine the project lead time.

An ideal schedule should combine solution robustness (i.e. be stable) and quality robustness (i.e. be makespan protective). The fundamental research issue we address in this paper is the potential trade-off between quality robustness (measured in terms of project duration) and solution robustness (stability, measured in terms of the deviation between the planned and realized start times) of the projected schedule. By means of simulation we investigate whether it is beneficial to concentrate safety time in project and feeding buffers as done by the *original* CC/BM approach and a *modified* CC/BM approach developed in this paper, or whether it is preferable to insert time buffers scattered throughout the project schedule, as done by the *adapted float factor model* (ADFF) developed by Leus (2003).

The remainder of the paper is organized as follows. The next section describes the set-up of our computational experiment. We describe the scheduling mechanism used by the *original* and *modified* CC/BM approach and the ADFF heuristic. The *original* and *modified* CC/BM approaches are used as representatives of scheduling algorithms that aim at generating makespan protective schedules. ADFF is used as a representative of scheduling algorithms that aim at generating solution robust schedules. We also describe the metrics used for measuring solution and quality robustness. Section 3 describes the experimental results obtained by the three scheduling heuristics. The last section is reserved for our overall conclusions.

2. Set-up of the computational experiment

We assume that projects are represented in activity-on-the-node representation, where the precedence constraints are of the finish-start type with zero time-lag. An example network with ten activities is given in Fig. 1. Nodes 0 and 9 are the dummy start and end nodes, respectively. We make abstraction of resource usage and assume that activity durations are random variables with known distribution. The first number above each node represents the corresponding mean activity duration, to be used in generating a baseline

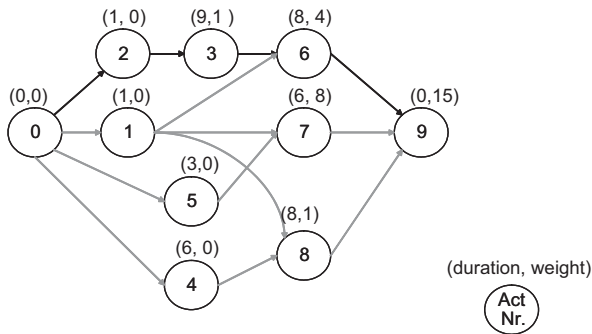


Fig. 1. An example network.

schedule. The second number represents a weight that is attributed to the activity. These weights denote the cost per unit of time that the corresponding activity is started earlier or later than originally planned. The weights will be input to the adapted float factor model described later in this section. The procedures for generating the projected schedules that are used in our experimental set-up will be clarified in the remainder of this section.

The CC/BM schedule is constructed following the principles described by Goldratt (1997). To reduce work in process⁴ (WIP), we initially calculate a late start baseline schedule, which we expand with feeding buffers and a project buffer. Afterwards, this buffered baseline schedule is converted into a projected schedule by scheduling all non-gating tasks as early as possible. The buffered schedule for the example network in Fig. 1, constructed using 50% feeding buffers and a 30% project buffer (these values are chosen purely for illustrative purposes), is represented in Fig. 2. Note that the critical chain (0–2–3–6–9) does not start at time zero, because the introduction of the feeding buffer following activity 8 causes the starting time of the non-CC (4–8) to be scheduled before the start of the first activity of the CC.

⁴Work in process is defined as work inside a system that is started but not yet complete (Newbold, 1998). We follow Leus (2003) and estimate total WIP as the sum over all activities of the average floats between the end of the activity and the starting times of its successors.

The simulation results reported in Section 3 have been obtained using a variant of this *original* CC/BM buffer insertion mechanism. This variant does not allow non-CC activities to start earlier than the starting time of the critical chain itself. All non-CC that, as the result of the insertion of a feeding buffer, would start before the starting time of the critical chain will thus be scheduled to start at the same time as the CC by crashing its feeding buffer size. The feeding buffer size is not allowed to exceed the float of the gating task heading the non-CC. For our example project, this means that the non-CC (4–8) will start at the same time as activity 2 and that the feeding buffer at the end of the chain (4–8) will be smaller than the case in the *original* CC/BM schedule of Fig. 2. The resulting schedule is shown in Fig. 3. We will refer to this variant approach as the *modified* CC/BM model. A justification for adhering to this variant of buffer insertion is provided in Section 3.3.2, by means of a comparison with the *original* CC/BM buffer insertion mechanism.

CC/BM may be seen as a heuristic that builds a schedule that mostly obtains good results on timely completion of the project (quality robustness). As mentioned earlier, we will refer to this group of schedules as *makespan protecting schedules*, as opposed to the group of *solution robust schedules* which score in general better on *solution robustness*.

ADFF has been shown (Leus, 2003) to produce good results in the group of solution robust schedules when the number of disruptions is rather high (which is the case in our experiment). ADFF generates a stable projected schedule according to the procedure described in Leus (2003) and Herroelen and Leus (2004a). The procedure is an adaptation of the float factor model that was originally introduced by Tavares et al. (1998) to generate a schedule S in which the start time of activity i is obtained as $s_i(S) := s_i(ESS) + \alpha_i(s_i(LSS) - s_i(ESS))$, where $\alpha_i \in [0, 1]$ is the so-called float factor, $s_i(ESS)$ denotes the earliest possible start time of activity i and $s_i(LSS)$ represents the latest allowable start time of activity i . Both start times are derived from critical path calculations for a given project deadline. Instead of using a single float factor α for all the activities,

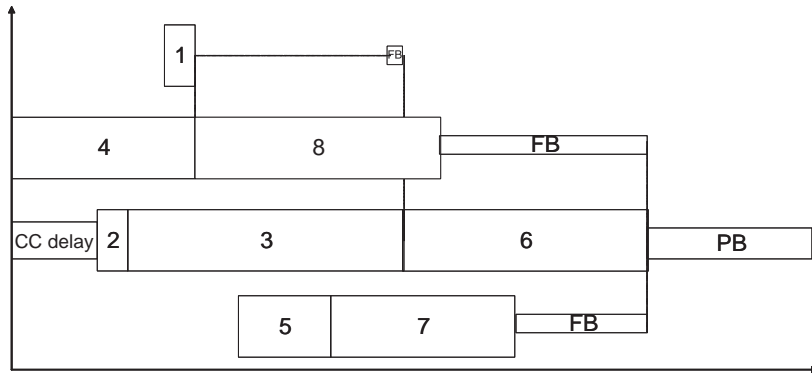


Fig. 2. The buffered *original* CC/BM schedule for the network of Fig. 1.

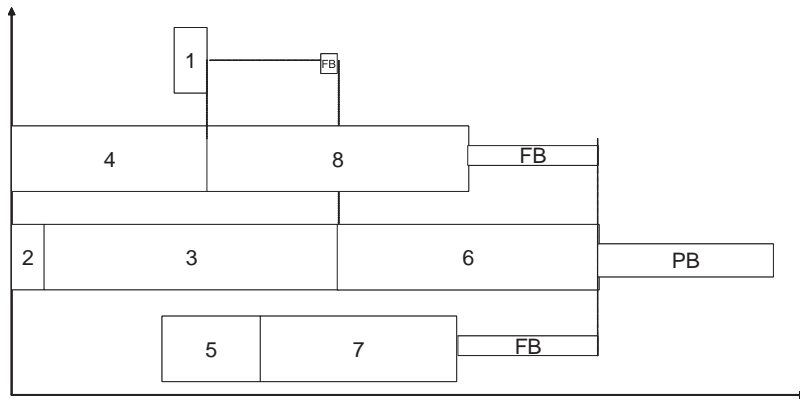


Fig. 3. The *modified* CC/BM schedule.

ADFF adopts an activity-dependent float factor that is calculated as $\alpha_i = \beta_i / (\beta_i + \delta_i)$, where β_i is the sum of the weight of activity i and the weights of all its transitive predecessors, while δ_i is the sum of the weights of all transitive successors of activity i . In doing so, ADFF inserts longer time buffers in front of activities that would incur a high cost if started later than originally planned. Table 1 calculates the ADFF starting times when the imposed due date equals 130% of the critical chain length (expressed as the sum of the durations of all activities on the critical chain). The 30% increase above the critical chain length is chosen to maintain comparability of the results with those obtained by *modified* CC/BM in Fig. 3, where a 30% project buffer was chosen for illustrative purposes. The resulting schedule is shown in Fig. 4.

During execution an activity will never start before its scheduled starting time, calculated by ADFF. In Section 3.4 we will refer to this scheduling principle as *railway scheduling*. This *railway scheduling* principle opposes the road-runner mentality that is typically applied during the execution of a CC/BM schedule.

Our simulation experiment aims at comparing the solution robustness (stability) and quality robustness (realized makespan) obtained by *original* and *modified* CC/BM (as representatives of schedulers aiming at makespan protection) and ADFF (as representative of schedulers aiming at solution robustness). Our analysis will mainly focus on the comparison of *modified* CC/BM and ADFF. It will be shown that the conclusions made are also valid for the *original* CC/BM approach.

Table 1
Calculation of starting times by using ADFP

Activity i	duration	$s_i(ESS)$	$s_i(LSS)$	float	weight	$\alpha[i]$	$s_i(S)$
0	0	0	5.4	5.4	0	0	0
1	1	0	14.4	14.4	0	0	0
2	1	0	5.4	5.4	0	0	0
3	9	1	6.4	5.4	1	0.05	1.27
4	6	0	9.4	9.4	0	0	0
5	3	0	14.4	14.4	0	0	0
6	8	10	15.4	5.4	4	0.25	11.35
7	6	3	17.4	14.4	8	0.35	8.01
8	8	6	15.4	9.4	1	0.06	6.59
9	0	18	23.4	5.4	15	1	23.4

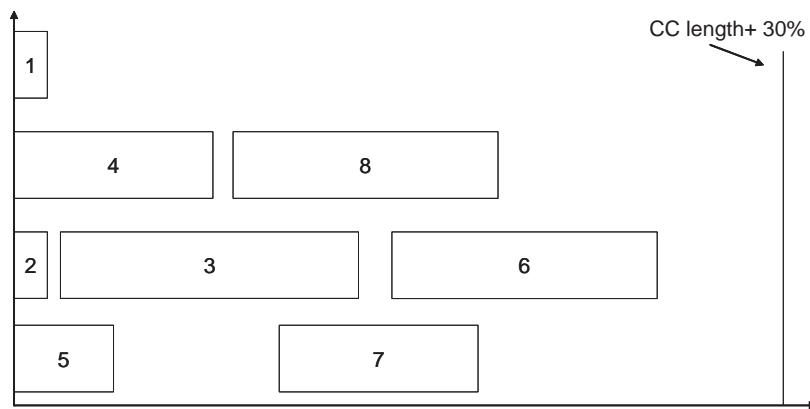


Fig. 4. The schedule proposed by ADFP for the network of Fig. 1.

The reader may wonder why our comparative study relies on simulation and not on mathematical analysis or approximations thereof. For the restrictive case where exactly one anticipated disturbance will occur in the network due to an increase in the duration of a single activity, Herroelen and Leus (2004a) have developed a mathematical programming model that allows for the construction of a buffered pre-schedule that minimizes the expected deviation in activity start times in the realized schedule from those in the pre-schedule. Computations become messy when the restrictive assumptions are removed. Leus and Herroelen (2004b) have shown that the problem becomes NP-hard if resource constraints are

present and have developed a branch-and-bound procedure for optimally solving a single-machine scheduling problem with stability objective, when a single job is anticipated to be disrupted (Leus and Herroelen, 2004a). The reason why we have opted in this paper to rely on the ADFP heuristic for generating solution-robust schedules is that the same authors have shown that this heuristic performs well. Moreover, ongoing research demonstrates that the heuristic can be easily extended for the resource-constrained case (Van de Vonder et al., 2004).

In order to evaluate the quality and solution robustness of the schedules generated in our computational experiment, it is necessary to

calculate the distribution function of the activity start times. Hagstrom (1988) has shown that the characterization of this distribution function is #P-complete. For the general activity network structures studied in our experiment, we could have used the approximation schemes developed in the literature as the datum against which the heuristics are compared (for a review we refer to Adlakha and Kulkarni (1989), Demeulemeester and Herroelen (2002) and Elmaghraby (1977)). We could also have tried to compute stochastic bounds on the makespan distribution using any of the procedures described in the open literature (the bounds of Kleindorfer (1971) and Dodin (1985) are considered good approximations for the makespan distribution and can be found within a small computational burden (Ludwig et al., 2001)).

The reasons why we have chosen to rely on the simulation methodology are twofold. First, simulation allows for an effective evaluation of the quality and solution robustness of the generated schedules with an efficiency that is sufficient for our purposes. Second, the simulation methodology can be readily extended to the resource-constrained case for which our research is ongoing (Van de Vonder et al., 2004).

The comparison of the three scheduling heuristics will be performed for varying project due date horizons. For both CC/BM versions this boils down to increasing the project buffer size, while for ADFP increasing the available time for project completion means increasing the amount of project buffering that can be spread along the activities on the critical chain.

The baseline scheduling methods have been applied to network instances generated by the *RanGen* project scheduling instances generator developed by Demeulemeester et al. (2003). The networks differ in the number of activities (n) and in order strength OS (defined as the number of precedence relations, including the transitive ones, divided by the theoretical maximum number⁵ of precedence relations (Mastor, 1970)).

⁵The maximum number of precedence relations in a network with n activities equals $n(n-1)/2$.

For every network 300 project executions have been simulated using a right-skewed beta-distribution for the activity durations (mean duration value equal to the deterministic duration used in the baseline schedule, minimum value equal to half the baseline duration and maximum value equal to 2.25 times the baseline duration). For every run new activity weights are drawn from a normal distribution with mean equal to 3 and a standard deviation equal to 2 (the weights are adapted in such a way that they cannot be negative). The weighting parameter (wp) is defined as the ratio between the weight of the dummy end activity and the average of the distribution of the weights of the other activities.

Quality robustness (makespan performance) is measured by the probability that a project ends within the projected deadline (further referred to as *timely project completion probability* or TPCP), while stability cost is computed as the weighted sum of the absolute deviations between the actually realized activity starting times and the starting times indicated in the initial projected schedule as anticipated before project execution.

3. Experimental results

It is quite normal that a *makespan protecting schedule* (original and modified CC/BM) results in a higher TPCP than a stable schedule (ADFP), while a solution robust schedule is expected to have a lower stability cost. The interesting issue addressed in this section is the magnitude of the possible loss in makespan performance when a stable schedule is used. We are fully aware that every project has unique characteristics and thus we do not pretend to give a full experimental design, which would be practically impossible to accomplish. Rather, we limit our scope to the description of the main effects of the different parameters to wit the weighting parameter (wp), the order strength (OS) and the network size measured by the number of activities (n).

3.1. Impact of the weighting parameter

Schedulers who implement *makespan protecting schedules*, traditionally assume that the cost of delaying the project completion outcales the cost of deviating from the planned (non-dummy) activity starting times. In this section we gradually increase the relative importance attributed to timely project completion. The number of activities and the order strength are fixed at 20 and 0.5, respectively. Buffer sizes are expressed as a percentage of the CC length.

Let us first consider the case where the weight of the dummy end activity equals the average of the distribution of all other activity weights ($wp = 1$). Table 2 shows the average results obtained by *modified CC/BM* and ADFP over 300 simulation runs. On average, *modified CC/BM* only needs a 27% project buffer to reach a 95% TPCP. On the other hand, ADFP needs 108% of due date delay to achieve the same result. Obviously, an increase in project duration of 108% of the critical chain length will most likely not be acceptable for a project manager. The schedule built by ADFP will not be considered as a feasible alternative. By consequence, the project manager will have to opt for *modified CC/BM* and cover the corresponding higher stability costs to be competitive.

In order to obtain additional insight into the extent to which the importance (weight) attributed to the last activity affects the performance/stability trade-off under study, we have obtained experimental results for increasing values of the weighting parameter. The results for $wp = 4$ are given in Table 3. Because the weighting parameter does not

Table 3

Comparison between *modified CC/BM* and ADFP for $wp = 4$

Buffer size (%)	Modified CC/BM cost ^a	ADFP cost ^a	Modified CC/BM TPCP(%)	ADFP TPCP(%)
27	98	14	95	85
40	97	6	99	95
58	97	4	99	98

^aCost refers to the sum of the weighted deviations of the realized activity start times from the planned activity start times.

affect the projected schedule constructed by *modified CC/BM*, a 27% project buffer will again be sufficient to obtain a 95% TPCP. On the other hand, ADFP does not need 108% due date extension in this case. On average, the addition of 40% of the critical chain length already results in the required 95% TPCP. Contrary to the case with $wp = 1$, this may be a valid alternative for project management. *Modified CC/BM* still outperforms ADFP on quality robustness, but the stability cost increase required to achieve a better TPCP is rather substantial. In other words, a project manager who opts for ADFP will agree with either a lower TPCP or a later promised project deadline, but will save on stability cost by doing so. Neither method dominates the other. The choice of a scheduling policy must face the makespan performance/stability trade-off.

Finally, consider the case where the weight of the dummy end activity equals 15 times the average of the distribution of the other activity weights ($wp = 15$). Note that this assumption is by no means unrealistic and that *CC/BM* theory ascribes high value to makespan performance, and thus would opt for a larger weighting parameter. Table 4 shows that the ADFP approach will now already result in the desired 95% TPCP for 29% of total CC buffering at much lower stability costs.

As illustrated in this section, the weighting parameter has a substantial impact on the relative attractiveness of *modified CC/BM* over ADFP. Fig. 5 summarizes the quality robustness obtained by the two scheduling procedures for a wide range

Table 2

Comparison between *modified CC/BM* and ADFP for $wp = 1$

Buffer size (%)	Modified CC/BM cost ^a	ADFP cost ^a	Modified CC/BM TPCP (%)	ADFP TPCP (%)
27	110	9	95	55
50	110	3	99	76
108	110	≈ 0	100	95

^aCost refers to the sum of the weighted deviations of the realized activity start times from the planned activity start times.

Table 4
Comparison between *modified* CC/BM and ADFP for $w_p = 15$

Buffer size (%)	Modified CC/BM cost ^a	ADFF cost ^a	Modified CC/BM TPCP(%)	ADFF TPCP(%)
27	68	21	95	93
29	67	19	96	95
50	65	9	99	99

^aCost refers to the sum of the weighted deviations of the realized activity start times from the planned activity start times.

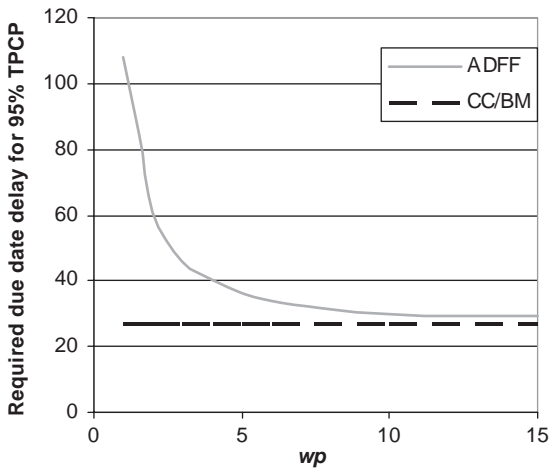


Fig. 5. Required due date delay for 95% TPCP, expressed in % of the CC.

of weighting parameters, when OS equals 0.5 and n equals 20. It shows that ADFP indeed does need a later projected due date than *modified* CC/BM to obtain the same 95% TPCP, but that this advantage of *modified* CC/BM strongly decreases when w_p increases.

Fig. 6 quantifies the trade-off between makespan performance and stability, also for fixed order strength ($OS = 0.5$) and network size ($n = 20$). The upper curve (labelled stability) shows the advantage of ADFP over *modified* CC/BM in terms of stability cost. It represents the ratio of stability cost of *modified* CC/BM over stability cost of ADFP, for the case where the allowed project due date equals 150% of the critical chain length. The greater this ratio, the

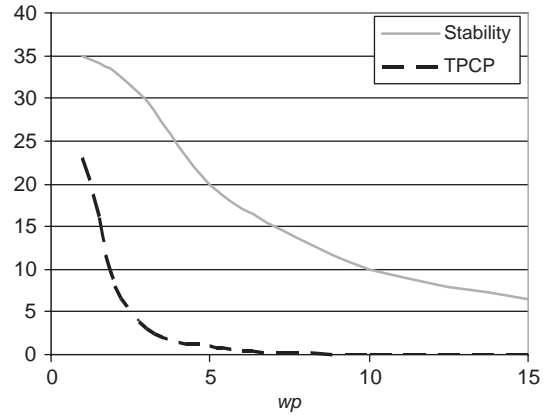


Fig. 6. Comparing ADFP and *modified* CC/BM for total buffering equal to 50% of CC length.

greater the stability advantage of ADFP. The second curve represents the makespan performance advantage of *modified* CC/BM, expressed as the difference in TPCP for a 50% prolongation of the projected due date beyond the length of the CC.

It is obvious that both the makespan performance and the stability cost advantage decrease when the weighting parameter increases. However, the makespan performance advantage decreases much more rapidly. This means that for higher w_p values, the stability advantage of ADFP remains, while the makespan performance advantage of *modified* CC/BM tends to disappear. In this case, ADFP will also provide a large buffer for the heavily weighted last activity, which acts as a project buffer and protects for project completion overruns.

The above leads to a rather paradoxical conclusion. While the critical chain methodology aims at makespan protection, *modified* CC/BM is less attractive than the stable scheduling method ADFP when the last activity is deemed relatively more important, i.e. when makespan performance really matters. In this case, a negligibly small allowance in makespan can lead to an enormous gain in stability by opting for a *solution robust scheduling method*. Choosing for a *makespan protecting schedule* and ignoring stability will become difficult to defend.

3.2. Impact of the order strength and the number of activities

In the previous section both the order strength and the network size were kept fixed. In this section we investigate the impact of both the order strength (set to 0.3, 0.5 and 0.7) and the network size (set to 10, 20 and 30 activities) on the makespan performance/stability trade-off.

3.2.1. Impact of the order strength

The percentage difference in allowed due date to obtain a 95% TPCP between ADFP and *modified* CC/BM is shown in Fig. 7 for different values of w_p and OS and for 20-activity networks. As already stated above, this extra buffer size needed by ADFP strongly decreases for increasing w_p . The examined extra due date delay in percentage of the CC length seems to be dependent on the order strength. For example, Fig. 7 shows that a project where the project completion has a weight that equals three times the average of the distribution of the other activity weights ($w_p = 3$) needs approximately 10% of extra buffer size if the order strength equals 0.3, while order strengths of 0.5 and 0.7 require, respectively, 19% and 25% of extra due date delay.

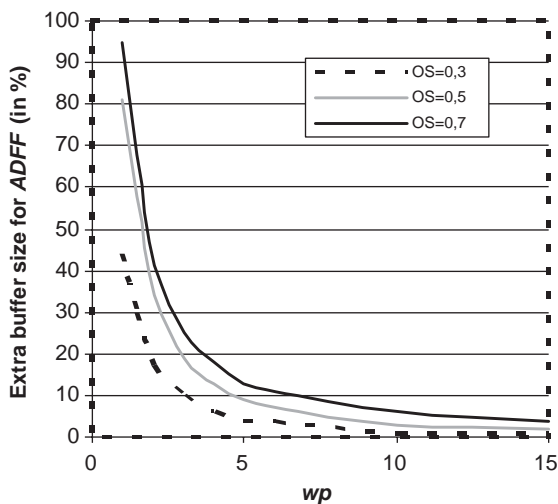


Fig. 7. Impact of the OS on the required buffer size for a 95% TPCP.

It is important to note that the extra-allowed due date delay, expressed in time units, is even more dependent on the order strength. Indeed, buffer sizes are expressed as a percentage of the CC length and this length highly depends on the order strength. This is no surprise because a higher OS induces more precedence relations and thus increases the CC length by allowing less activities to be scheduled in parallel. Table 5 shows that 20-activity networks with an order strength $OS = 0.3$ have a CC length of 25 time units on average, while networks with $OS = 0.7$ have an average CC length of 43 time units. 10% required buffer size of a 25-unit CC length for $OS = 0.3$ results in a much smaller delay than 25% of a 43-unit CC length for $OS = 0.7$.

The order strength also affects the project buffer size required for *modified* CC/BM to obtain a 95% TPCP. A higher order strength means that a smaller percentage of (the larger) CC length should be added to achieve this goal. We observed in Section 3.1 that for 20-activity projects with $OS = 0.5$ *modified* CC/BM needed an average project buffer of 27% of the CC length. When the order strength goes up to 0.7, only 22% of the CC length is needed to achieve a 95% TPCP. It shows that the proposed 50% project buffer sizing rule always provides a safe buffer size, but is overprotective for project networks with high order strength values.

3.2.2. Impact of the network size

Fig. 8 shows the impact of the weighting parameter w_p and the number of activities n on the higher due date values (expressed in % of CC length) required by ADFP to achieve 95% TPCP. As in the previous section, we observe that the required buffer size is dependent on both w_p and n . While, for example, 11% extra-allowed due date

Table 5
Average CC length (in time units) as a function of the order strength

OS	CC length
0.3	25
0.5	32
0.7	43

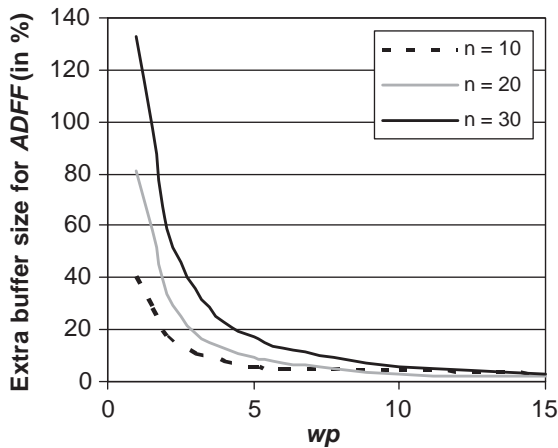


Fig. 8. Impact of w_p and n on the buffer size required to achieve a 95% TPCP.

delay would be sufficient when the $OS = 0.5$, $w_p = 3$ and $n = 10$, a network with 30 activities would require, ceteris paribus, approximately 36% of extra delay. Bearing in mind that the critical chain length is also dependent on the number of activities, we may conclude that the number of activities strongly affects the due date required to achieve a certain TPCP. By consequence, the advantage of *modified CC/BM* over ADFP in terms of TPCP for a given weighting parameter is more pronounced for larger networks. The paradox described above (and illustrated in Fig. 6) holds for all network sizes, but is less pronounced for large networks. The attentive reader could wonder why we make this conclusion without looking at stability. The reason for omitting stability from the analysis is due to the fact that results show that the stability ratio between *modified CC/BM* and ADFP as introduced above (in Fig. 6), remains high for any combination of OS , w_p and n . Consequently, if stability is the issue, we would always opt for ADFP. If the reduced quality robustness of ADFP is deemed acceptable by project management, a project manager may choose for a stable scheduling policy and take advantage of the lower stability costs. If not, the high stability cost of a makespan protecting schedule has to be accounted for in order to be competitive. Thus, quality robustness

is a much more important factor than solution robustness, if a choice has to be made between the two project scheduling heuristics.

Regarding the buffer size that is required to achieve a 95% TPCP, the same conclusion can be drawn as in Section 3.2.1. The required percentage of CC length decreases when the network size goes up. The 50% buffer sizing rule, initially proposed by Goldratt (1997), overprotects large networks. This insight incited researchers (Newbold, 1998; Herroelen and Leus, 2001) to develop more advanced buffer-sizing procedures.

3.3. Feeding buffers

Modified CC/BM protects the makespan by including feeding buffers in non-critical feeding chains. It allows all activities to start as soon as possible, except for the gating tasks, which start at an intermediate time between their earliest possible and latest allowable start time. This intermediate time is calculated by the feeding buffer mechanism, which forces the gating tasks to start earlier than their latest allowable start time by inserting a feeding buffer sized at 50% of the feeding chain length. However, in *modified CC/BM* the feeding buffer size is not allowed to exceed the total float of the gating task. Consequently, when 50% of the feeding chain size exceeds the total float of that activity, the starting time of the gating task would equal its earliest start time, which is zero by definition.

We alert the reader to the fact that we defer the justification for the use of *modified CC/BM* and the comparison of its performance with the *original CC/BM* approach to Section 3.3.2. Before doing so, we rely on *modified CC/BM* in the next section to investigate the optimal feeding buffer size.

3.3.1. Feeding buffer size

In this section we will take a look at the feeding buffer mechanism and investigate the required size of these buffers. Before discussing the optimal feeding buffer size in terms of a percentage of the CC length, we will first examine the influence of gating task starting times, expressed as a percentage of total activity float.

We set the starting time, $s_j(S)$, of a gating task j at its latest allowable starting time, $s_j(LSS)$, reduced by a percentage (α) of its total float, i.e. $s_j(S) := s_j(LSS) - \alpha(s_j(LSS) - s_j(ESS))$. In Fig. 9 we let α fluctuate between 0 (all gating tasks start as late as allowed) and 100% (all gating tasks start as early as possible) and compare intermediate results in terms of stability, quality robustness and work in process (WIP) for networks with $n = 20$, $OS = 0.5$, $wp = 5$ and no project buffer (the due date is equal to the CC length).

We observe that WIP increases when α increases. This is a logical result because scheduling some tasks earlier has an obvious negative effect on WIP. On the other hand, the probability that a project can be completed on time, goes up when more buffering is inserted. However, it is very important to note that Fig. 9 only gives this TPCP for the zero-sized project buffer case. For larger project buffers this difference in makespan performance will decrease and eventually disappear.

The planner may experiment with appropriate α -values in searching for schedules with acceptable WIP and makespan performance. For the illustrative parameter settings of Fig. 9, for example, $\alpha = 30\%$ seems to yield satisfactory results.

The stability cost pattern shown in Fig. 9 needs clarification. Fig. 10 partitions the stability costs into its constituent components. The first stability cost component accounts for all activities that end earlier than planned. The number of such activities and by consequence also this part of the cost will increase when the starting times of the gating tasks

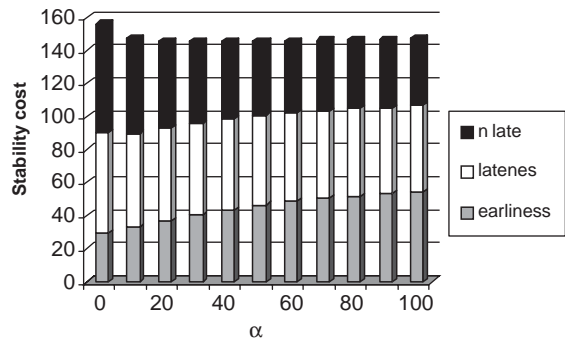


Fig. 10. Stability cost split into three components.

are scheduled earlier (i.e. when α increases). The second cost component corresponds with the cost originating from activities that end later than scheduled. This cost will decrease for increasing α . The last stability cost component refers to the cost of the delay of the dummy end activity. It again decreases with an increasing α . For a project buffer of 50% of the CC length, however, the third stability cost component would completely disappear, resulting in an increasing total stability cost function. Fig. 10 also allows for an easy interpretation of the stability cost change when, for example, the weight of being late would be twice the weight of being early.

Next, we investigate the actual impact of the feeding buffer size. In the original CC/BM literature (Goldratt, 1997) 50% of the critical chain length is proposed as the feeding buffer size. In our set-up, we have experimented with other buffer sizes, expressed as a percentage of the CC length. Results (Fig. 11) show a similar trade-off between WIP and makespan protection as could be observed in Fig. 9. A 50% feeding buffer size seems to be a nice compromise. The stability cost, however, is slightly lower than the best case in Fig. 9, where the feeding buffer size was expressed as a percentage of the total float of the gating tasks. Scheduling activities earlier by including a feeding buffer increases the total earliness cost and decreases the total lateness cost in the same way as described in Fig. 10. However, when we compare the total costs in Figs. 9 and 11, it is obvious that the feeding buffer mechanism

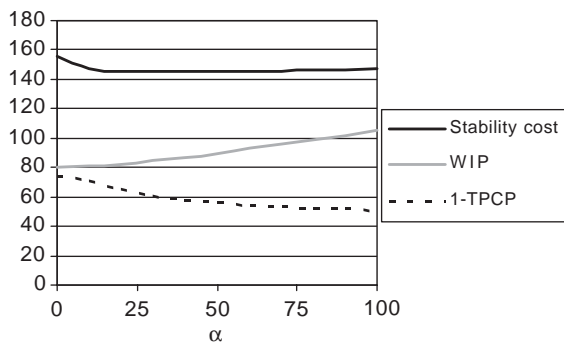


Fig. 9. Evaluating makespan protective schedules for fluctuating α .

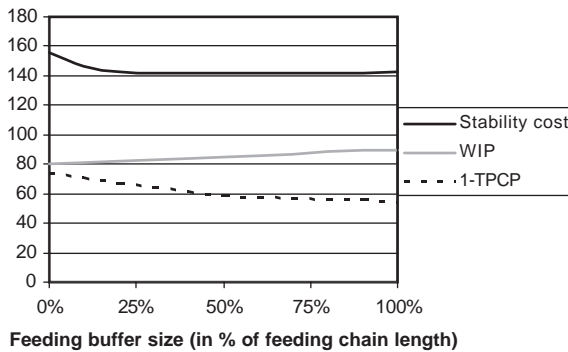


Fig. 11. Evaluating makespan protective schedules for fluctuating feeding buffer size.

needs a slightly lower total stability cost. Similar to the introduction of $\alpha = 30\%$ as a good trade-off value between WIP and quality robustness in Fig. 9, Fig. 11 shows that the traditional 50% rule delivers good results. Moreover, both rules of thumb score almost equally well on WIP and makespan protection. Nevertheless, the stability cost is lower for the feeding buffer approach. A more detailed examination shows that, especially, the lower total lateness cost makes the feeding buffer approach appealing. This lower cost is due to a larger feeding buffer for near-CC, compared to the cases with small α -values where only $\alpha\%$ of the total float is used as a feeding buffer in Fig. 9. Indeed, near-CC will start at their earliest possible start time in the feeding buffer approaches, resulting in a reduction of the number of activities that end late.

We can conclude that expressing the feeding buffer size as a percentage of the incoming chain length yields better results than expressing the feeding buffer size as a percentage of the total float of the gating tasks. It should be observed, however, that our analysis has been made in the absence of resource requirements. If resources come into play, as Herroelen and Leus (2001) have shown, the 50% buffer sizing rule is overprotective for large projects.

3.3.2. Original CC/BM

In this section, we compare the *original* CC/BM approach (see Fig. 2) with the *modified* CC/BM

approach, as presented in Fig. 12. The CC delay that appears in Fig. 2 (see also Herroelen et al. (2002)) is a minimum prolongation of the critical chain length in the *original* CC/BM approach. For 20-activity projects with $OS = 0.5$, we note that on average approximately 4 units of CC delay are added. This means that the *original* CC/BM approach would not be able to meet an imposed project completion time of less than the average CC length plus four when activity durations would have been deterministic. To have a more honest comparison of both methods, i.e. with equal projected project completion times, we have to add this CC delay to the *modified* CC/BM model as a minimal required project buffer.

For the case with order strength 0.5 and 20 activities, we found that the *original* CC/BM approach has a 57% TPCP when the project buffer is 0 and needs a project buffer of 23% of CC length to obtain 95% TPCP. Our modified model clearly outperforms the *original* CC/BM approach on quality robustness. Without project buffer, TPCP increases up to 73%, while the needed buffer size for 95% TPCP is only 17% of CC length. For stability cost, there is no large difference between both methods, except for very small project buffer sizes. Then, the *modified* CC/BM model performs substantially better because of the lower stability cost contribution of the last dummy activity. However, because CC/BM serves in our paper as an example of a *makespan protecting schedule*, we feel that the *modified* version is more appropriate to use due to its much higher quality robustness than the *original* CC/BM approach.

3.4. Railway scheduling

The differences in quality and solution robustness between the ADFP and the CC/BM approach may be due to two reasons. First of all, the projected schedules are different (with or without intermediate buffers), but secondly, also the execution policy varies. The CC/BM literature denies the importance of intermediate milestones and therefore suggests to start a non-gating task as soon as all its predecessors are finished (*roadrunner mentality*). ADFP, on the other hand, will never

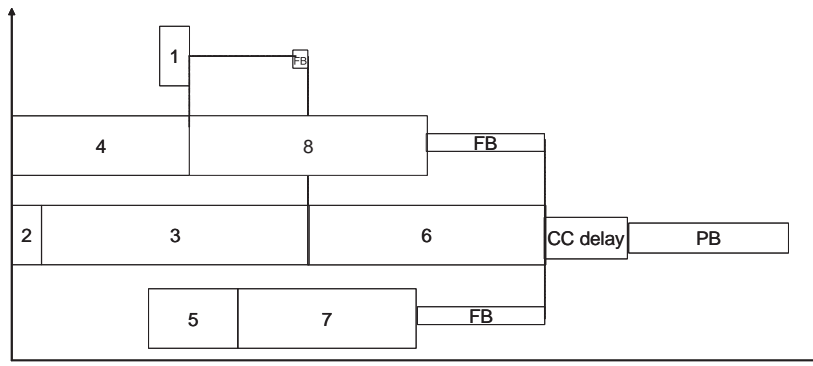


Fig. 12. The *modified CC/BM* schedule with CC delay.

allow an activity to start earlier than planned (we refer to this execution policy as *railway scheduling*, because of its comparability with the scheduling of trains in a railway station).

In this section we investigate the impact of the use of feeding buffers combined with the railway scheduling policy. Fig. 13 represents the average *railway*/ADFF stability cost ratio for networks with 20 activities and order strength equal to 0.5. The curve with *modified CC/BM* label is equal to the curve in Fig. 6 and represents the enormous gap in stability cost between makespan protecting schedules and stability-based schedules.

The curve with the *modified CC/BM railway* label combines the *modified CC* methodology with the railway scheduling mode, as used in ADFE. Apparently, railway scheduling (combined with *modified CC/BM*) improves stability for all weighting parameter values. However, the largest reduction in stability cost of stable schedules is gained through intermediate buffering. The TPCP only deteriorates for very small project buffers. Incorporating railway scheduling in *modified CC/BM*, as opposed to relying on the roadrunner mentality that is typically assumed in *CC/BM*, seems to offer an interesting alternative.

4. Conclusions

The basic objective of this paper was to research the potential trade-off between solution robustness (stability) and quality robustness (makespan performance) using three heuristics as the vehicle of

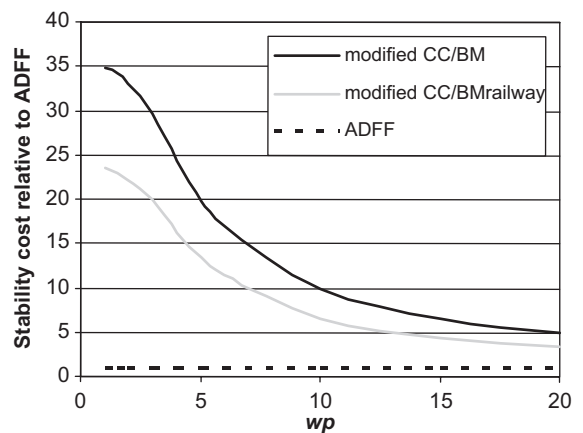


Fig. 13. Stability cost of *modified CC/BM* combined with railway execution.

analysis: the *original CC/BM* and *modified CC/BM* approaches, being heuristics that aim at makespan performance, and ADFE, being a heuristic that aims at stability. The main conclusion of the comparative study is that the expected difference in makespan performance between *makespan protecting schedules* and *solution robust schedules* tends to disappear for some projects. Where this is the case, a *solution robust schedule* will most likely be preferred because of the considerably lower stability cost. ADFE, for example, seems to be particularly interesting for projects for which a heavy weight is given to timely completion, i.e. for which quality robustness really matters.

Paradoxically, the pioneers of *CC management* focus on due date performance, while their

approach seems to be hard to defend when the timely realization of the projects is deemed important. The fact that stability costs of the real activities are relatively small does not justify them to be ignored. Indeed, *solution robust scheduling techniques* ascribe accordingly little attention to intermediate activities, which will also result in a projected schedule with a large project buffer, but opposing to *makespan protecting schedules*, stability is not ignored and is substantially better.

CC/BM-based schedules will in general suffer less from the solution/quality robustness trade-off when the project has a larger number of activities or higher order strength. An important lesson to be learned from this paper is that project managers should be aware of the trade-off between stability and makespan performance. The buffering strategy should be chosen with an eye on the characteristics of the project to be scheduled.

Acknowledgements

This research has been supported by project OT/03/14 of the Research Fund K.U. Leuven. We are also very grateful to the anonymous referees for their valuable suggestions.

References

- Adlakha, V., Kulkarni, V., 1989. A classified bibliography of research on stochastic PERT networks. *INFOR* 27, 272–296.
- Demeulemeester, E., Herroelen, W., 2002. Project scheduling—A Research Handbook. Vol. 49 of International Series in Operations Research & Management Science. Kluwer Academic Publishers, Boston.
- Demeulemeester, E., Vanhoucke, M., Herroelen, W., 2003. RanGen: A random network generator for activity-on-the-node networks. *Journal of Scheduling* 6, 17–38.
- Dodin, B., 1985. Bounding the project completion time distribution in PERT networks. *Operations Research* 33, 862–881.
- Elmaghraby, S., 1977. Activity Networks—Project Planning and Control by Network Models. Wiley, New York.
- Goldratt, E., 1997. Critical Chain. The North River Press Publishing Corporation, Great Barrington, MA.
- Hagstrom, J., 1988. Computational complexity of PERT problems. *Computers and Operations Research* 18, 139–147.
- Herroelen, W., Leus, R., 2001. On the merits and pitfalls of critical chain scheduling. *Journal of Operations Management* 19, 559–577.
- Herroelen, W., Leus, R., 2004a. The construction of stable baseline schedules. *European Journal of Operational Research* 156, 550–565.
- Herroelen, W., Leus, R., 2004b. Project scheduling under uncertainty—Survey and research potentials, *European Journal of Operational Research*, to appear.
- Herroelen, W., Leus, R., Demeulemeester, E., 2002. Critical chain project scheduling: Do not oversimplify. *Project Management Journal* 33, 48–60.
- Kleindorfer, G., 1971. Bounding distributions for a stochastic acyclic network. *Operations Research* 19, 1586–1601.
- Leus, R., 2003. The generation of stable project plans. Ph.D. Thesis, Department of Applied Economics, Katholieke Universiteit Leuven, Belgium.
- Leus, R., Herroelen, W., 2004a. A branch-and-bound algorithm for stable scheduling in single-machine production systems. Research Report 0428, Department of Applied Economics, Katholieke Universiteit Leuven, Belgium.
- Leus, R., Herroelen, W., 2004b. The complexity of machine scheduling for stability with a single disrupted job. *Operations Research Letters*, to appear.
- Ludwig, A., Möhring, R., Stork, F., 2001. A computational study on bounding the makespan distribution in stochastic project networks. *Annals of Operations Research* 102 (1/4), 49–64.
- Mastor, A., 1970. An experimental and comparative evaluation of production line balancing techniques. *Management Science* 16, 728–746.
- Newbold, R., 1998. Project management in the fast lane—Applying the Theory of Constraints. APICS Series on Constraints Management. The St. Lucie Press.
- Tavares, L., Ferreira, J., Coelho, J., 1998. On the optimal management of project risk. *European Journal of Operational Research* 107, 451–469.
- Van de Vonder, S., Demeulemeester, E., Herroelen, W., Leus, R., 2004. The trade-off between stability and makespan in resource-constrained project scheduling. Research Report 0423, Department of Applied Economics, Katholieke Universiteit Leuven, Belgium.