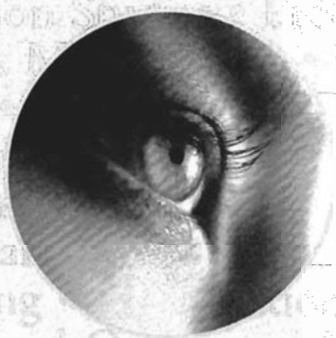
DEPARTMENT OF APPLIED ECONOMICS



RESEARCH REPORT

THE USE OF BUFFERS IN PROJECT MANAGEMENT: THE TRADE-OFF BETWEEN STABILITY AND MAKESPAN

STIJN VAN DE VONDER • ERIK DEMEULEMEESTER • WILLY HERROELEN • ROEL LEUS

OR 0404

The Use of Buffers in Project Management: The Trade-off between Stability and Makespan

Stijn Van de Vonder[1], Erik Demeulemeester[1], Willy Herroelen[1] and Roel Leus[1]

[1]Department of Applied Economics, K.U.Leuven, Belgium
e-mail: <first name>.<family name>@econ.kuleuven.ac.be

19th January 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 realised 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 in a clever way throughout the baseline project schedule in order to maximize schedule stability.

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 & Herroelen (2002) and Herroelen & Leus (2003b)).

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 fundamental working principles of *CC/BM* have been reviewed by Herroelen et al. (2002). *CC/BM* builds a baseline schedule using aggressive 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*), which should protect the project due date from variability in the critical chain activities. The *critical chain* is defined as the chain of precedence and resource dependent activities that determines the overall duration of a project. *Feeding buffers* (*FB*) are inserted whenever a non-critical chain activity joins the critical chain. Overall, the *CC/BM* idea is to protect the project due date against the disruptions that may occur during project execution. 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).

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 or whether it is preferable to insert time buffers scattered in a clever way throughout the project schedule.

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.

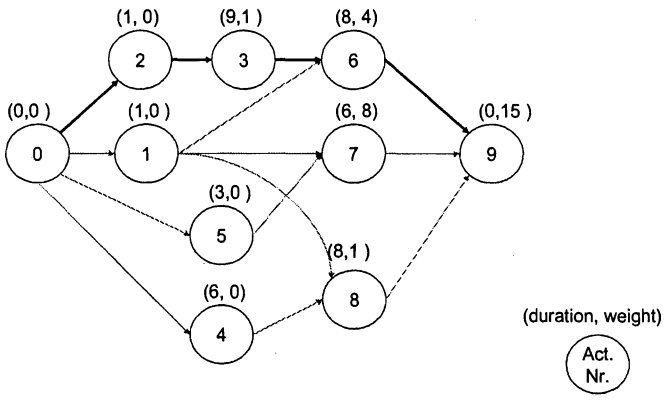


Figure 1: An example network

An example network with ten activities is given in Figure 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 schedule. The second number represents a weight that is attributed to the activity. These weights denote a relative cost of starting the corresponding activity one time unit earlier or later than originally scheduled. The weights will be input to the adapted float factor 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 process (*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 pushing back in time all non-gating tasks as much as possible. The buffered schedule for the example network in Figure 1, constructed using 50% feeding buffers and a 30% project buffer (choice of values is purely for illustration), is represented in Figure 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 chain <4-8> to be pushed back in time beyond the starting point of the critical chain. The simulation results reported in Section 3 have been obtained using a variant of this *CC/BM* buffer insertion mechanism that does not allow non-critical chain activities to start earlier than the starting time of the critical chain itself. The resulting schedule is shown in Figure 3. A justification for adhering to this variant of

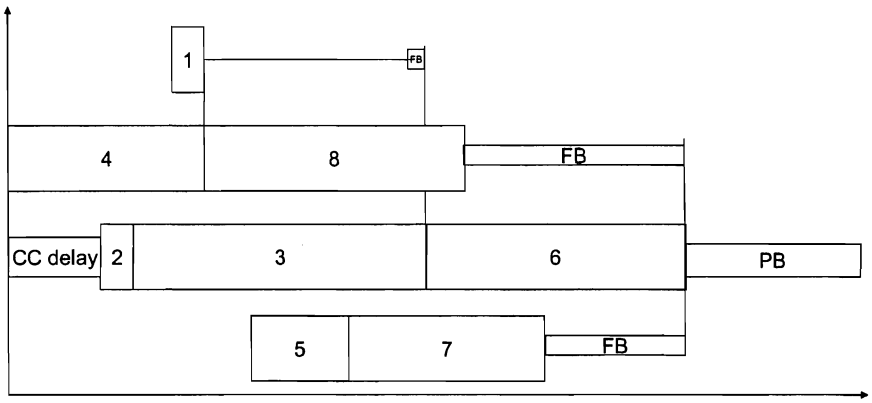


Figure 2: The buffered *CC/BM* schedule for the network of Figure 1

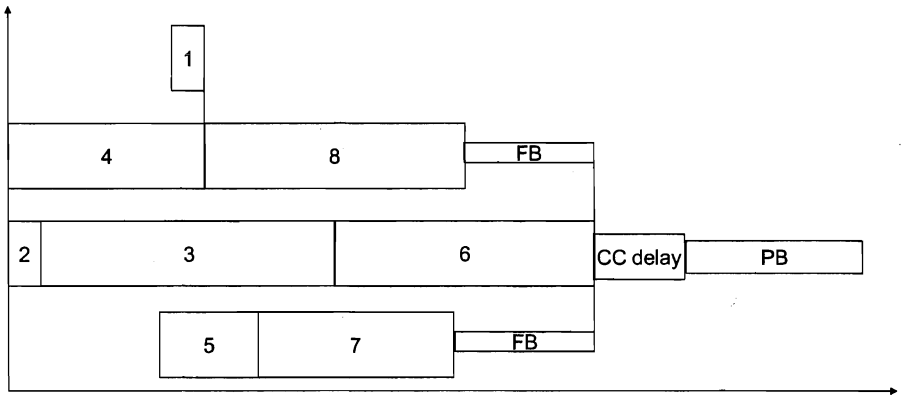


Figure 3: The *adapted CC/BM* schedule

buffer insertion is provided in section 3.3.2, by means of a comparison with the traditional *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). We will refer to this group of schedules as *makespan protecting schedules* (*MPS*), as opposed to the group of *solution robust schedules* which score in general better on *solution robustness.*

The *adapted float factor model* (*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 & Leus (2003a). 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, *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 transitive predecessors of activity i, 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. The 30% increase above the critical chain length is chosen to maintain comparability of the results with those obtained by *adapted CC/BM* in Figure 3, where a 30% project buffer was chosen for illustration purposes. The resulting schedule is shown in Figure 4.

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

Table 1: Calculation of starting times by using *ADFF*

During execution, an activity will never start before its scheduled starting time, calculated by *ADFF*. In Section 3.4, we will refer to this as *railway scheduling.*

In this paper we report on a simulation experiment set up to compare the solution robustness (stability) and quality robustness (realized makespan) of *makespan protecting schedules* (*adapted CC/BM* and traditional *CC/BM*) and *solution robust schedules* (*ADFF).* This comparison will be performed for varying project due date horizons. For *CC/BM* this boils down to increasing the project buffer size, while for *ADFF* increasing the available time for project

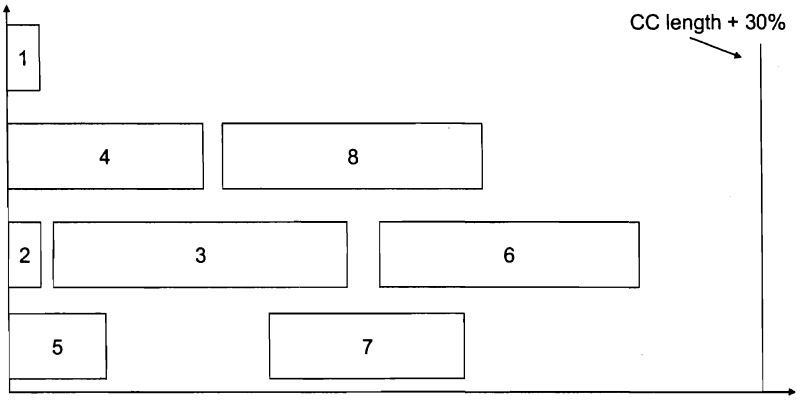


Figure 4: The schedule proposed by *ADFF* for the network of Figure 1

completion means increasing 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 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 number of precedence relations (Mastor 1970)). 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 set equal to 3 and a standard deviation set 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* (*adapted* and traditional *CC/BM*) results in a higher TPCP than a stable schedule (*ADFF*), 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. Moreover, we will report on the impact of important project metrics such as the number of activities, the weighting parameter and the order strength.

3.1 Impact of the weighting parameter

Schedulers who implement *makespan protecting schedules,* traditionally assume that the cost of delaying the project completion outscales 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 critical chain (*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 *adapted CC/BM* and *ADFF* over 300 simulation runs. On average, *adapted CC/BM* only needs a 27% project buffer to reach a 95% TPCP. On the other hand, *ADFF* 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 *ADFF* will not be considered as a feasible alternative. By consequence, the project manager will have to opt for *adapted CC/BM* and cover the corresponding higher stability costs to be competitive.

Buffer Size	*Adapted CC/BM* cost*	*ADFF* cost*	*Adapted CC/BM* TPCP	*ADFF* TPCP
27%	110	9	95%	55%
50%	110	3	99%	76%
108%	110	≈ 0	100%	95%

**Cost refers to the sum of the weighted deviation of the realized activity start times from the planned activity start times*

Table 2: Comparison between *adapted CC/BM* and *ADFF* for *wp*=1

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 affect the projected schedule constructed by *adapted CC/BM*, a 27% project buffer will again be sufficient to obtain a 95% TPCP. On the other hand, *ADFF* 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. *Adapted CC/BM* still outperforms *ADFF* 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 *ADFF* 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.

Buffer Size	*Adapted CC/BM* cost*	*ADFF* cost*	*Adapted CC/BM* TPCP	*ADFF* TPCP
27%	98	14	95%	85%
40%	97	6	99%	95%
58%	97	4	99%	98%

**Cost refers to the sum of the weighted deviation of the realized activity start times from the planned activity start times*

Table 3: Comparison between *adapted CC/BM* and *ADFF* for *wp*=4

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 *ADFF* approach will now already result in the desired 95% TPCP for 29% of total CC buffering at much lower stability costs.

Buffer Size	*Adapted CC/BM* cost*	*ADFF* cost*	*Adapted CC/BM* TPCP	*ADFF* TPCP
27%	68	21	95	93
29%	67	19	96	95
50%	65	9	99	99

**Cost refers to the sum of the weighted deviation of the realized activity start times from the planned activity start times*

Table 4: Comparison between *adapted CC/BM* and *ADFF* for *wp*=15

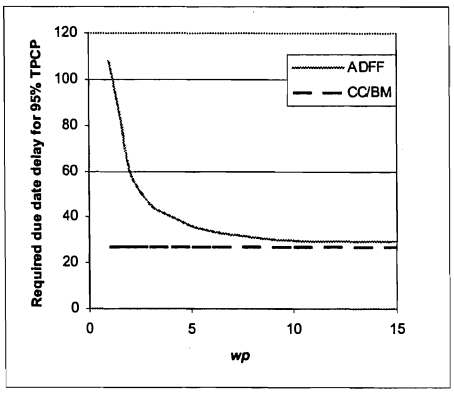


Figure 5: Required due date delay for 95% TPCP, expressed in % of the CC

As illustrated in this section, the weighting parameter has a substantial impact on the relative attractiveness of *adapted CC/BM* over *ADFF*. Figure 5 summarizes the quality robustness obtained by the two scheduling procedures for a wide range of weighting parameters, when *OS* equals 0.5 and n equals 20. It shows that *ADFF* indeed does need a later projected due date than *adapted CC/BM* to obtain the same 95% TPCP, but that this advantage of *adapted CC/BM* strongly decreases when *wp* increases.

Figure 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 *ADFF* over *adapted CC/BM* in terms of stability cost. It represents the ratio of stability cost of *adapted CC/BM* over stability cost of *ADFF*, for the case where the allowed project due date equals 150% of the critical chain length. The greater this ratio, the greater the stability advantage of *ADFF*. The second curve represents the makespan performance advantage of *adapted CC/BM*, expressed as the difference in TPCP for a 50% prolongation of the projected due date beyond the length of the critical chain.

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 *wp* values, the stability advantage of *ADFF* remains, while the makespan performance advantage of *adapted CC/BM* tends to disappear. In this case, *ADFF* 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

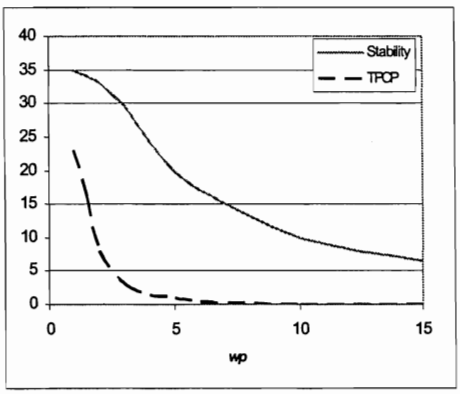


Figure 6: Comparing *ADFF* and *CC/BM* for total buffering equal to 50% of CC length

methodology aims at makespan protection, *adapted CC/BM* is less attractive than the stable scheduling method *ADFF* 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 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 *ADFF* and *adapted CC/BM* is shown in Figure 7 for different values of *wp* and *OS* and for 20-activity networks. As already stated above, this extra buffer size needed by *ADFF* strongly decreases for increasing *wp*. The examined extra due date delay in percentage of the critical chain length, seems to be dependent of order strength. For example, Figure 7 shows that a project where the project completion has a weight that equals three times the average of the distribution

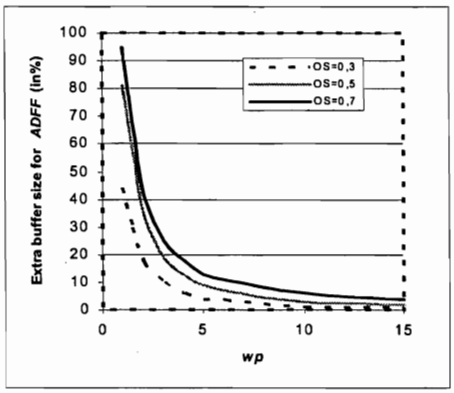


Figure 7: Impact of the *OS* on the required buffer size for a 95% TPCP

of the other activity weights ($wp = 3$) 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.

It is important to note that the extra allowed due date delay, expressed in time units, is even more dependent on order strength. Indeed, buffer sizes are expressed as a percentage of the critical chain length and this length highly depends on order strength. Table 5 shows that 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 critical chain 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 43-unit CC length for $OS = 0.7$.

OS	CC length
0.3	25
0.5	32
0.7	43

Table 5: Average CC length as a function of order strength

The order strength also affects the project buffer size required for *adapted CC/BM* to obtain a 95% TPCP. A higher order strength means that a smaller percentage of CC length should be added to achieve this goal. We observed in section 3.1 that for 20-activity projects with *OS* 0.5, *adapted 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.

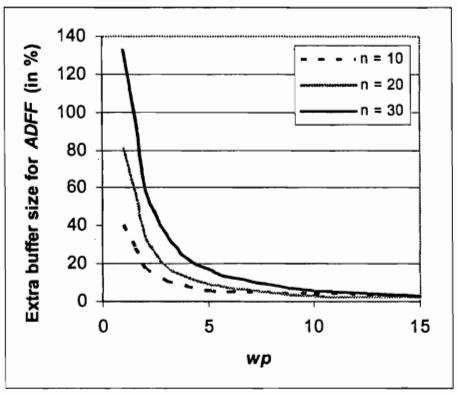


Figure 8: Impact of wp and n on the buffer size required to achieve a 95% TPCP

3.2.2 Impact of the network size

Figure 8 shows the impact of wp and the number of activities n, on the higher due date values (expressed in % of CC length) required by *ADFF* to achieve 95% TPCP. As in the previous section, we observe that the required buffer size is dependent on both the wp and the number of activities. While, for example, 11% extra allowed due date delay would be sufficient when order strength = 0.5, $wp = 3$ 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 *adapted CC/BM* over *ADFF* in terms of TPCP for a given weighting parameter is more pronounced for larger networks. The paradox described above (and illustrated in Figure 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 *adapted CC/BM* and *ADFF* as introduced above (in Figure 6), remains high for any combination of *OS*, wp and n. Consequently, if stability is the issue, we would always opt for *ADFF*. If the reduced quality robustness of *ADFF* 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 & Leus 2001)) to develop more advanced buffer sizing procedures.

3.3 Feeding Buffers

Adapted 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[1], 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 pushes back gating tasks from their latest allowable start time by inserting a feeding buffer sized at 50% of the feeding chain length. However, in *adapted 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. In section 3.3.2, we will justify the use of the *adapted CC/BM* and compare its performance with the traditional *CC/BM* approach. Before doing so, we rely on our adapted model 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 critical chain 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 as 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 Figure 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[2] (WIP) for networks with $n = 20$,

[1] Tasks with only dummy predecessors

[2] Work 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.

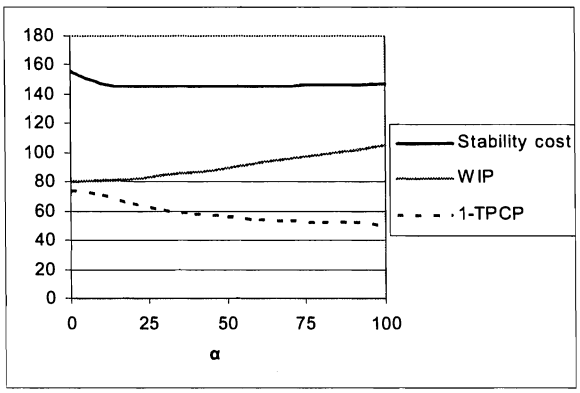


Figure 9: Evaluating makespan protective schedules for fluctuating α

$OS = 0.5$, $wp = 5$ 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 pushing back some tasks 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 Figure 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. To obtain a solution with a reasonable WIP and no explicit makespan problems, Figure 9 proposes intermediate solutions (for instance $\alpha = 30\%$).

The stability cost pattern shown in Figure 9 needs clarification. Figure 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 are pushed back further in time (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. Figure 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. Traditionally (Goldratt 1997), 50% of the critical chain length is proposed as

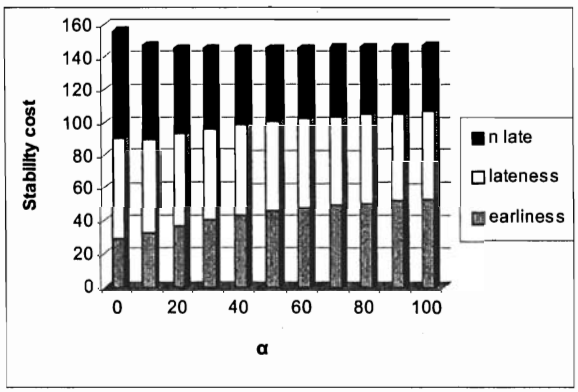


Figure 10: Stability cost split into three components

the feeding buffer size. In our set-up, we experimented with other buffer sizes, expressed as a percentage of the CC length. Results (Figure 11) show a similar trade-off between WIP and makespan protection as could be observed in Figure 9. A 50% feeding buffer size seems to be a nice compromise. The stability cost, however, is slightly lower than the best case in Figure 9, where the feeding buffer size was expressed as a percentage of the total float of the gating tasks. Pushing back activities by including a feeding buffer, increases the total earliness cost and decreases the total lateness cost in the same way as described in Figure 10. However, when we compare the total costs in Figures 9 and 11, it is obvious that the feeding buffer mechanism needs a slightly lower total stability cost. Like we introduced $\alpha = 30\%$ as a good trade-off value between WIP and quality robustness in Figure 9, Figure 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-critical chains, compared to the cases with small α-values where only $\alpha\%$ of the total float is used as a feeding buffer in Figure 9. Indeed, near critical chains 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, Herroelen & Leus (2001) have shown

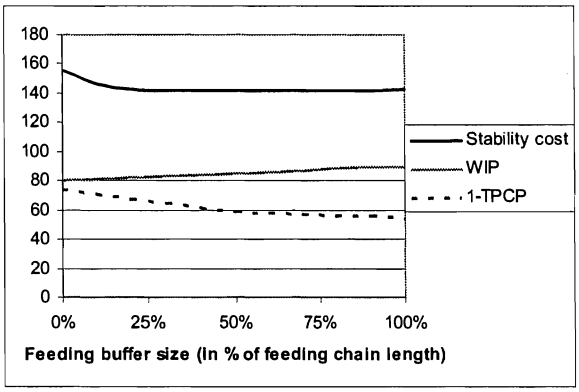


Figure 11: Evaluating makespan protective schedules for fluctuating feeding buffer size

that the 50% buffer sizing rule is overprotective for large projects.

3.3.2 Traditional *CC/BM*

In this section, we compare the traditional *CC/BM* approach (see Figure 2) with the *adapted CC/BM* approach, as presented in Figure 12. The critical chain delay that appears in Figure 2 (see also Herroelen et al. (2002)) is a minimum prolongation of the critical chain length in traditional *CC/BM*. For 20-activity projects with $OS = 0.5$, we note that on average approximately 4 units of *CC delay* are added. This means that traditional *CC/BM* would not be able to meet an imposed project completion time of less than average critical chain 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 adapted *CC/BM* model as a minimal required project buffer.

For the case with order strength 0.5 and 20 activities, we note that the traditional *CC/BM* 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 adapted model clearly outperforms traditional *CC/BM* 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 *adapted 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 *adapted*

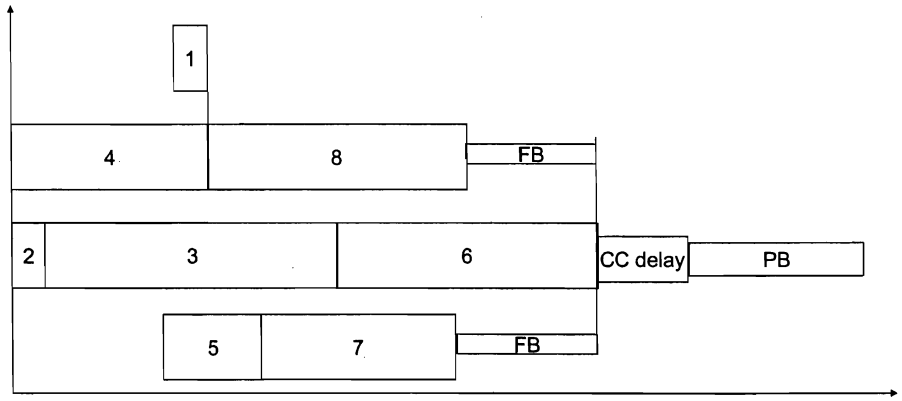


Figure 12: The *adapted CC/BM* schedule with CC delay

version is more appropriate to use due to its much higher quality robustness than traditional *CC/BM.*

3.4 Railway scheduling

The differences in quality and solution robustness between the *ADFF* and *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 standard *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*). *ADFF,* on the other hand, will never 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. Figure 13 represents the average *railway/ADFF* stability cost ratio for networks with 20 activities and order strength equal to 0.5. The curve with *adapted CC/BM* label is equal to the curve in Figure 6 and represents the enormous gap in stability cost between makespan protecting schedules and stability based schedules.

The curve with the *adapted CC/BM railway* label combines the adapted critical chain methodology with the railway scheduling mode, as used in ADFF. Apparently, railway scheduling (combined with *adapted 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

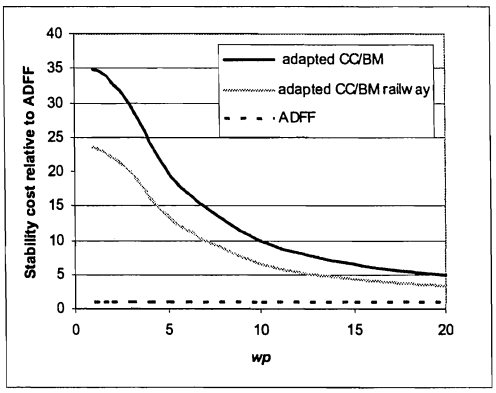


Figure 13: Stability cost of *adapted CC/BM* combined with railway execution.

railway scheduling in *adapted 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 main conclusion of this paper 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. *ADFF*, 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 *critical chain 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.

Acknowledgment

This research has been supported by project OT/03/14 of the Research Fund K.U.Leuven.

References

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, pp 17–38.

Goldratt, E. (1997). *Critical Chain.* The North River Press Publishing Corporation Great Barrington.

Herroelen, W. & Leus, R. (2001). On the merits and pitfalls of critical chain scheduling. *Journal of Operations Management*, 19, pp 559–577.

Herroelen, W. & Leus, R. (2003a). The construction of stable baseline schedules. *European Journal of Operational Research, to appear.*

Herroelen, W. & Leus, R. (2003b). 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, pp 48–60.

Leus, R. (2003). *The generation of stable project plans.* PhD thesis. Department of applied economics, Katholieke Universiteit Leuven.

Mastor, A. (1970). An experimental and comparative evaluation of production line balancing techniques. *Management Science*, 16, pp 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, pp 451–469.