Vlerick Leuven Gent Management School
the Autonomous Management School of
Ghent University and Katholieke Universiteit Leuven

Vlerick Leuven Gent Working Paper Series 2006/15

THE IMPACT OF VARIOUS ACTIVITY ASSUMPTIONS ON THE LEAD-TIME AND RESOURCE UTILIZATION OF RESOURCE-CONSTRAINED PROJECTS

DIETER DEBELS

MARIO VANHOUCKE

Mario.Vanhoucke@vlerick.be

D/2006/6482/15

THE IMPACT OF VARIOUS ACTIVITY ASSUMPTIONS ON THE LEAD-TIME AND RESOURCE UTILIZATION OF RESOURCE-CONSTRAINED PROJECTS

DIETER DEBELS

Ghent University

MARIO VANHOUCKE

Vlerick Leuven Gent Management School

Contact:

Mario Vanhoucke

Vlerick Leuven Gent Management School

Tel: +32 09 210 97 81

Fax: +32 09 210 97 00

Email: Mario.Vanhoucke@vlerick.be

ABSTRACT

The well-known resource-constrained project scheduling problem (RCPSP) schedules project activities within the precedence and renewable resource constraints while minimizing the total lead-time of the project. The basic problem description assumes non-pre-emptive activities with fixed durations, and has been extended to various other assumptions in literature.

In this paper, we investigate the effect of three activity assumptions on the total lead-time and the total resource utilization of a project. More precisely, we investigate the influence of variable activity durations under a fixed work content, the possibility of allowing activity pre-emption and the use of fast tracking to decrease a project's duration.

We give an overview of the procedures developed in literature and present some modifications to existing solution approaches to cope with our activity assumptions under study. We present computational results on a generated dataset and evaluate the impact of all assumptions on the quality of the schedule.

1. INTRODUCTION

The well-known *resource-constrained project scheduling problem* (RCPSP) is one of the most widely studied problems in project scheduling and can be stated as follows. In a project network $G(N,A)$ in an activity-on-the-node (AoN) format, the set of nodes N are used to represent the n activities (numbered from 1 to n, i.e. $|N| = n$) and a set of pairs of activities A represent the precedence relations between activities. Furthermore, project execution requires a set of resources R with a constant availability a_k for each resource type $k \in R$ throughout the project horizon. Each activity $i \in N$ is assumed to have a deterministic duration $d_i \in$ IN and requires $r_{ik} \in$ IN units of resource type k. The dummy start and end activities 1 and n have zero duration and zero resource usage. A schedule can be defined by an n-vector of start times $(s_1, ..., s_n)$, and implies an n-vector of finish times $(f_1, ..., f_n)$. A schedule is said to be *feasible* if it is non-pre-emptive and if both the precedence and renewable resource constraints are satisfied, and *optimal* if the project makespan f_n is minimized.

Figure 1 displays a project network example with 9 activities and one resource type with an availability a_1 of 6. This example will be used throughout the remainder of this paper. The duration d_i of each activity i has been displayed above the node, while the resource demand r_{i1} has been shown below the node. The optimal solution with a minimal project duration of 9 has been displayed at the right part of figure 1.

Insert Figure 1 About Here

In this paper, we relax the strict activity assumptions of the basic RCPSP and investigate the impact of these assumptions on the quality of the project schedule. More precisely, we investigate the effect of three activity assumptions, i.e. fixed or variable activity durations, activity pre-emption (splitting) and fast tracking (parallel execution of sub-parts of activities). The purpose of this research is twofold. First, we present some adaptations to current solutions approaches to cope with the activity assumptions under study. This allows the generation of optimal schedules for the various problem types that can be used for comparison purposes.

Second, we evaluate the impact of the various activity assumptions on the total project lead-time as well as on the efficiency of resource use. In doing so, we are able to provide some general guidelines to project schedulers for better choosing between the various activity options in their scheduling software.

The outline of the paper is as follows. In section 2 we discuss the three activity assumptions into detail. We show that many assumptions do not fundamentally change the problem description and can therefore be solved by any RCPSP solution procedure. In section 3, we propose some adaptations on a well-known branch-and-bound procedure for the basic RCPSP to cope with most of our new assumptions. Section 4 presents detailed computational results and investigates the impact of the activity assumptions on the quality of the schedule, both from a lead-time as from a resource point-of-view. Section 5 presents some overall conclusions and suggestions for future research.

2. PROJECT SCHEDULING UNDER THREE ACTIVITY ASSUMPTIONS

Many project scheduling software packages aim at the construction of resource feasible schedules in order to minimize the total lead-time of the project. Hence, an AoN project network with a list of activities with their corresponding precedence relations and resource requirements need to be given as an input. However, various activity assumptions need to be made by the user in order to construct a feasible schedule. We investigate three different activity assumptions, as summarized in figure 2. This figure displays the effect of the three assumptions on activity 2 of figure 1. These extensions are:

- Fixed duration or fixed work
- The presence of activity pre-emption
- The effect of fast tracking

Fixed duration or fixed work: The basic RCPSP assumes that each activity i consists of a deterministic work content W_{ik} for each resource-type k, and imposes a fixed duration d_i and fixed resource requirements r_{ik} on its execution. The extension to the *discrete time/resource trade-off problem* (DTRTP) still assumes a fixed work content but

allows variable activity durations. As an example, activity 2 of figure 1 still has a fixed work content W_{i1} of 9 for the single resource type 1, but can now be executed under different scenarios. Note that many commercial software packages pay a lot of attention to this activity assumption, and call for the well-considered use of this activity option before the construction of a schedule (see e.g. the many "Duration * Units = Work" examples of Uyttewaal (2005)).

Activity pre-emption: The basic RCPSP assumes that each activity, once started, will be executed until its finish. The extension to the *pre-emptive resource-constrained project scheduling problem* (PRCPSP) allows activities to be pre-empted at any integer time instant and restarted later on at no additional cost, and has been investigated in literature as an option to further reduce the total project lead-time. The literature for the *pre-emptive discrete time/resource trade-off problem* (PDTRTP) is, to the best of our knowledge, completely void. In most project scheduling software packages, the option of activity splitting can be made before the construction of a resource-feasible schedule. The option to split activities has an effect on the number of execution scenarios, as displayed in figure 2.

Fast tracking: Fast tracking is a scheduling technique used to reduce the total project lead-time during project execution. When projects are fast-tracked, it usually indicates the compression of a project schedule by doing certain activities in parallel that would normally be done in a sequence. Hence, it violates the precedence relations between activities which implies activity execution at incomplete information. In our paper, we investigate the impact of within-activity fast tracking, which allows the execution of pre-emptive sub-parts of an activity in parallel. The fast tracking option removes precedence relations between sub-parts of pre-empted activities and increases the number of execution scenarios. The within-activity fast tracking option is inspired on the idea that activities are executed by groups of resources (with a fixed availability), but the total work can often be done by multiple groups (in parallel). The *pre-emptive resource-constrained project scheduling problem with fast tracking* (PRCPSP-FT) assumes pre-emptive activities with fixed durations, which results in d_i parallel sub-activities with each a resource requirement r_{ik}. The *pre-emptive discrete time/resource trade-off problem with fast tracking* (PDTRTP-FT) assumes variable activity durations

(under a fixed work content) and allows the pre-emptive and parallel execution of each sub-activity with a duration and resource requirement equal to 1, as shown in the bottom part of figure 2. To the best of our knowledge, the literature of resource-constrained project scheduling with a fast tracking option between pre-emptive sub-parts of activities is completely void.

Insert Figure 2 About Here

In the next subsection, we show that the PRCPSP, the PRCPSP-FT and the PDTRTP-FT can be solved by any solution approach for the basic resource-constrained project scheduling problem. In section 2.2, we elaborate on the DTRTP and the PDTRTP.

2.1 The sub-activity network for the PRCPSP, PRCPSP-FT and PDTRTP-FT

In this section, we show that the resource-constrained project scheduling problem can be easily extended to cope with 3 of our activity assumptions, i.e. PRCPSP, PRCPSP-FT and PDTRTP-FT, and hence, these problem instances can be solved by any solution algorithm for the RCPSP.

Kaplan (1988, 1991) was the first to study the PRCPSP, but she did not present a correct exact solution procedure (Demeulemeester and Herroelen, 1996). Ballestin et al. (2006) have developed a meta-heuristic procedure to solve the PRCPSP. Demeulemeester and Herroelen (1996) have translated the RCPSP to the PRCPSP by means of a subactivity project network $G(N',A')$ and developed a branch-and-bound procedure to optimally solve the problem. In a sub-activity network, each activity i is splitted into d_i sub-activities i_s ($s = 1, \ldots, d_i$) with a sub-activity duration $d_{i_s} = 1$ and a corresponding resource requirement $r_{i_s k} = r_{ik}$. The PRCPSP allows activity pre-emption and assumes that the remaining part of the activity is scheduled later in the schedule. Hence, a precedence constraint between each pair (i_s, i_{s+1}) is added in the sub-activity network. The complete PRPCSP sub-activity network has been displayed in figure 3(a) and splits the 7 non-dummy activities into 16 sub-activities. The optimal schedule is displayed in

the right part of figure 3(a) and leads to an overall project lead-time reduction from 9 to 8 thanks to the pre-emption of activities 4 and 5.

Insert Figure 3 About Here

The option to fast track pre-empted sub-parts of activities boils down to the option to schedule sub-activities of the same activity in parallel, and hence, implies the removal of all precedence relations between sub-activities of the same activity. Consequently, the sub-activity network for the PRCPSP-FT assumes that each activity i is splitted into d_i sub-activities i_s (s = 1, …, d_i) with a sub-activity duration d_{i_s} = 1, resource requirements $r_{i_s k} = r_{ik}$, and no precedence relations between sub-activities of the same activity. The fast track option for the PDTRTP-FT assumes a sub-activity network where each activity i is splitted into W_{ik} sub-activities i_s (s = 1, …, W_{ik}) with a sub-activity duration d_{i_s} = 1, resource requirements $r_{i_s k}$ = 1, and no precedence relations between sub-activities of the same activity.

Figures 3(b) and 3(c) represent the sub-activity networks and corresponding optimal schedules for the PRCPSP-FT and the PDTRTP-FT, respectively. The PRCPSP-FT schedule shows a decreased lead-time from 8 to 7 time units, thanks to the parallel execution of pre-emptive sub-part for activities 2, 4, 5, 6 and 7. The sub-activity network for the PDTRTP-FT contains 36 sub-activities, with all durations and resource requirements equal to 1. The optimal resource feasible schedule has a minimal project lead time of 7 time units with a more efficient resource consumption over time.

Since the PRCPSP, PRCPSP-FT and DTRTP-FT can be represented by a sub-activity network, these problem types can be solved by any algorithm for the RCPSP. Many exact and (meta-)heuristic RCPSP procedures have been presented in literature, and overviews can be found in Icmeli et al. (1993), Özdamar and Ulusoy (1995), Herroelen et al. (1998), Brucker et al. (1999), Hartmann and Kolisch (2000), Kolisch and Padman (2001) and Kolisch and Hartmann (2004). In our current paper, we rely on the efficient branch-and-bound procedure of Demeulemeester and Herroelen (1992) to solve various problem instances. Their depth-first approach builds up partial schedules starting

at time 0 and continuing systematically throughout the search process by iteratively adding (sub-)activities until a complete feasible schedule is obtained. A partial schedule at level p of the search tree will be further build by determining the next decision moment dm at which unscheduled activities might start. All unscheduled activities which are a candidate to start at time dm are calculated and collected in the set E of eligible activities. The previously scheduled but at dm unfinished activities belong to the set S of activities in progress. If scheduling all activities from $E \cup S$ at dm would cause a resource conflict, the procedure starts to branch to the next level $p + 1$ and delays subsets (delaying alternatives) of $E \cup S$ to resolve resource conflicts. It has been shown that it is sufficient to limit the search to the *minimal delaying alternatives*, which contain no other delaying alternatives as a subset. Then, a minimal delaying alternative needs to be selected, which involves that only the unselected activities of $E \cup S$ will be scheduled at dm while all previously scheduled activities of S and the activities of E that belong to the alternative are postponed. This process is repeated until a feasible schedule is found, followed by a backtracking mechanism and the algorithm continues as a usual branch-and-bound procedure. The branch-and-bound procedure has been made very efficient thanks to a number of dominance rules (probably the best known is the cutset dominance rule) and efficient lower bound calculations.

Demeulemeester and Herroelen (1996) have adapted their original RCPSP branch-and-bound procedure to cope with pre-emptive activities. To that purpose, they rely on the sub-activity network (see figure 3(a)) with all activity durations equal to one. Furthermore, they removed some inefficient or redundant lower bound calculations and dominance rules and simplified their branching strategy. Indeed, since all sub-activities that are scheduled at a decision moment dm automatically end one time unit later, the next decision moment automatically equals $dm + 1$, resulting in an empty set S of activities in progress. The authors observe a clear trade-off between computational effort to solve the PRCPSP and the resulting schedule quality improvements compared to the RCPSP, and show that activity pre-emption has only a small positive effect on a project's lead-time. However, Ballestin et al. (2006) recently showed that high-quality heuristic solutions can be obtained more easily for the PRCPSP than for the RCPSP.

In the current paper, we rely on the original branch-and-bound procedure of Demeulemeester and Herroelen (1992) to solve the RCPSP, and adapt this procedure to make it more efficient for solving the RCPSP-FT and the PDTRTP-FT (see section 3).

2.2 The solution approach for the DTRTP

The DTRTP assumes variable activity durations and resource requirements with a fixed work content W_{i1} for a single resource type (note that only 1 resource type is considered, and hence, no resource/resource trade-offs between multiple resources are included). Each activity can be executed according to a set of feasible execution modes M_i. Every mode m represents a combination of duration $d_{i(m)}$ and resource requirements $r_{i1(m)}$, for which $d_{i(m)} * r_{i1(m)} \geq W_{i1}$. De Reyck et al. (1998) have shown that it is sufficient to consider only efficient modes for which all other feasible modes are either higher in duration or higher in resource requirements. As an example, set M_2 of figure 4 contains 5 efficient modes $m = (d_{i(m)}, r_{i1(m)})$, i.e. $M_2 = \{(1,9), (2,5), (3,3), (5,2), (9,1)\}$. Note that modes (2,5) and (5,2) exceed the minimal work content of 9 by 1 unit, and mode (1,9) is infeasible towards to renewable resource constraints. The optimal schedule has a decreased lead-time from 9 to 7 time units when shifting from fixed durations (RCPSP) to fixed work content (DTRTP), thanks to the selection of a different mode for activities 4, 6 and 7.

Insert Figure 4 About Here

Demeulemeester et al. (2000) have presented a branch-and-bound procedure to solve the DTRTP that relies on *activity-mode combinations* branching strategy as an extension of the minimal delaying alternatives branching strategy. Activity-mode combinations are subsets of the candidate activities of set ($E \cup S$), executed in a specific mode. The authors have shown that only feasible and maximal combinations need to be considered. An activity-mode combination is *feasible* if the activities can be executed in parallel in the specified mode without causing a resource conflict, and *maximal* if no other activity can be added in one of its modes without causing a resource conflict. The authors mention the importance of efficient resource-based lower-bounds since the resource utilization for a DTRTP schedule is often much higher than for an RCPSP schedule. The literature for the PDTRTP is, to the best of our knowledge, completely

void. In the current paper, we do not consider the PDTRTP since the problem type can not be transformed to a sub-activity network as is the case for the PRCPSP, PRCPSP-FT and the PDTRTP-FT. Hence, we restrict the research of activity pre-emption to the PRCPSP.

In the remainder of this paper, we consider various approaches for the PRCPSP, PRCPSP-FT and the PDTRTP-FT, since they can represented by a sub-activity network and solved by any procedure for the basic RCPSP (as indicated by dashed lines in figure 2). Hence, we do not present new solution procedures for the DTCTP, but only rely to an existing DTCTP procedure to compare its results with our newly obtained solutions.

3 A BRANCH-AND-BOUND PROCEDURE

In this section, we present two adaptations to the branch-and-bound procedure of Demeulemeester and Herroelen (1992) in order to solve the PRCPSP-FT and the PDTRTP-FT more efficiently. Section 3.1 presents an adapted minimal delaying alternatives approach to solve the PRCPSP-FT. In section 3.2, we present adapted lower bound and upper bound calculations for the PDTRTP-FT

3.1. The minimal delaying alternatives for the PRCPSP-FT and the PDTRTP-FT

Demeulemeester and Herroelen (1992) have shown that only minimal delaying alternatives need to be investigated during their branch-and-bound procedure. A minimal delaying alternative is a subset of activities to delay in order to resolve a resource conflict, that contains no other delaying alternative as a subset. Since the PRCPSP-FT removes precedence relations between sub-activities, all sub-activities i_s of an activity i become eligible to be scheduled at the same decision moment, and hence, the number of minimal delaying alternatives at each level of the search tree grows exponentially. However, an extension of the minimal delaying alternatives principle limits the search of delaying alternatives to subsets of the eligible activities of set E, and dominates many nodes in the branch-and-bound tree, as follows:

Theorem: In order to define the set of minimal delaying alternatives for the PRCPSP-FT and the PDTRTP-FT, it is sufficient to define the number of sub-activities e_i for each activity i that should be chosen from the eligible set E

The theorem implies that it is not required to define which sub-activities should be selected from the eligible set for entrance in each minimal delaying alternative. All sub-activities have a duration of 1 and the set S of activities in progress is always empty at the decision moment dm. Hence, if a minimal delaying alternative selects e_i sub-activities of activity i from the eligible set E, then every other combination of e_i sub-actives of i in E will lead to an equivalent schedule. In our specific implementation, we always select the e_i highest numbered sub-activities of activity i to enter the minimal delaying alternative, such that the remaining lower numbered sub-activities are scheduled at dm.

Insert Table 1 About Here

Table 1 illustrates the theorem at the initial decision moment 0 for the example PRCPSP-FT problem of figure 3(b). The set E of eligible sub-activities contains all sub-activities of activities 2, 3 and 4, i.e. $E = \{2_1, 2_2, 2_3, 3_1, 4_1, 4_2, 4_3\}$. Scheduling all sub-activities in parallel results in a total resource demand of 14 units, which exceeds the availability of 6. In order to solve this resource conflict, the branch-and-bound procedure of Demeulemeester and Herroelen (1992) generates 16 minimal delaying alternatives. The theorem selects only one delaying alternative for each combination e_2, e_3 and e_4, and hence, only alternatives 1, 3, 6 and 16 need to be considered in the tree.

3.2. The lower and upper bound calculations for the PDTRTP-FT

The PDTRTP-FT assumes sub-activities with all durations and resource requirements equal to 1 and no precedence relations between within-activity sub-activities. Hence, the problem type is a strong relaxation of the RCPSP for which many alternative optimal schedules exist. In this section, we present straightforward yet

efficient lower and upper bounds that dramatically improve the efficiency of the adapted branch-and-bound algorithm of section 3.2.

Lower bound LB_p: the algorithm calculates at each node of the branch-and-bound tree the *minimal remaining duration* L_{i_s} of each sub-activity i_s of the eligible set E (i.e. ready to be scheduled) at decision moment dm. Hence, the lower bound LB_p at level p of the tree equals $dm + \max_{i_s \in E}(L_{i_s})$ and is based on the backward calculations (from the dummy end node to the dummy start node) of L_{i_s}, as follows:

$$L_{i_s} = \max\left(\left\lceil \frac{|S_{i_s}|}{a_1} \right\rceil, \max_{j_s \in S_{i_s}}(L_{j_s})\right) + 1 + \left\lfloor \frac{u_{i_s}}{a_1} \right\rfloor$$

where S_{i_s} is used to denote the set of all (immediate and transitive) non-dummy successor sub-activities of sub-activity i_s and $|S_{i_s}|$ is used to represent the number of sub-activities in this set. Moreover, u_{i_s} is used to represent the number of sub-activities of activity i with a higher subscript than the sub-script s of sub-activity i_s (these are sub-activities i_{s+1}, i_{s+2},.., $i_{W_{ik}}$). (note that all higher numbered sub-activities can not be scheduled earlier than sub-activity i_s due to our specific delaying alternatives approach of theorem 1). This lower bound calculates the minimal remaining length of each activity as the maximum of the resource-based remaining schedule length $\left\lceil \frac{|S_{i_s}|}{a_1} \right\rceil$ and the minimal remaining length of its successors $\max_{j_s \in S_{i_s}}(L_{j_s})$, increased by a factor $1 + \left\lfloor \frac{u_{i_s}}{a_1} \right\rfloor$ to represent the minimal required extra time needed to schedule i_s and its u_{i_s} higher subscripted sub-activities of the same activity i.

Lower bound LB_0: At the initial node of the branch-and-bound tree, the algorithm replaces the decision moment dm by the earliest possible start time EST_{i_s} of each sub-activity i_s, and hence, the lower bound LB_p can be replaced by $LB_0 = \max_{\forall i_s \in N}(EST_{i_s} + L_{i_s})$, with

$$EST_{i_s} = \max\left(\left\lceil \frac{|P_{i_s}|}{a_1} \right\rceil, \max_{j_s \in P_{i_s}} \left(EST_{j_s}\right) + 1\right) + \left\lfloor \frac{l_{i_s}}{a_1} \right\rfloor$$

where P_{i_s} is used to denote the set of all (immediate and transitive) non-dummy predecessor sub-activities of sub-activity i_s and $|P_{i_s}|$ is used to represent the number of sub-activities in this set. Moreover, l_{i_s} is used to represent the number of sub-activities of activity i with a lower subscript than the subscript s of activity i_s (these are sub-activities i_1, i_2,.., i_{s-1}).

Upper bound UB_0: At the start of the search process, a priority-rule based upper bound by generating a resource feasible schedule with the serial schedule generation scheme will be constructed. The algorithm relies on the L_{i_s} ranking to construct the priority rule, with the maximum $|S_{i_s}|$ value and the index of the sub-activity as tie-breaking rules.

Figure 5 displays the sub-activity network of figure 1 for the PDTRTP-FT, with the values of EST_{i_s} and L_{i_s}. The lower bound LB_0 equals 7 (see nodes 7_7, 7_8, 7_9, 8_1, 8_2 and 9_1) and the upper bound UB_0 has a total duration of 7 and corresponds to the schedule of figure 3(c). Hence, the resource-feasible schedule constructed for the upper bound is optimal, and no branching is needed.

Insert Figure 5 About Here

4 COMPUTATIONAL RESULTS

In this section, we test the impact of the three activity assumptions on the problem complexity and the schedule quality based on 1,920 randomly generated project networks generated by RanGen (Demeulemeester et al. 2003). The number of non-dummy activities ($n - 2$) has been set at 10, 12, 14, 16, 18 and 20 with an order strength OS and a resource-constrainedness RC fixed at 0.2, 0.4, 0.6 and 0.8. All project instances require a single resource type with an availability of 10 units. The activity durations have been chosen randomly between 1 and 5. Using 20 instances for each problem setting, we obtain a problem set of 6 * 4 * 4 * 20 = 1920 network instances. In section 4.1, we measure the problem complexity by means of different branch-and-bound procedures. Section 4.2 investigates the impact of all assumptions on the total project lead-time and the utilization of resources.

4.1 Impact of the activity assumptions on the problem complexity

In this section, we report computational results for the resource-constrained problems discussed in section 2. Thanks to the transformation to sub-activity networks, the PRCPSP, PRCPSP-FT and the DTRTP-FT can be solved by the efficient RCPSP procedure of Demeulemeester and Herroelen (1992). Moreover, the contribution of the adaptations of section 3 will be tested for the PRCPSP-FT and PDTRTP-FT.

The results have been displayed in table 2. The abbreviation 'DH92' refers to the branch-and-bound procedure of Demeulemeester and Herroelen (1992) while the abbreviation 'DH96' refers to the branch-and-bound procedure of Demeulemeester and Herroelen (1996) for the PRCPSP. The adapted branch-and-bound approach as discussed in sections 3 will be abbreviated with 'DV06'. The rows labeled "sub-activities" displays the average number of sub-activities in the sub-activity network (this number equals to our number of activities for the RCPSP row).

The row "Avg. OS" displays the average value for the order strength OS, defined as the number of immediate and transitive precedence relations between the $(n - 2)$ non-dummy (sub-)activities in relation to a maximal number $((n - 2).(n - 3))/2$ of precedence relations between (sub-)activities. The average OS value equals 0.5 for the original problem instances, which is an average of our RanGen input settings 0.2, 0.4, 0.6 and 0.8. The row "Avg. CPU" displays the average CPU-time (in seconds) needed to solve a problem instance and the row "% Opt" reports the number of problem instances that could be optimally solved within a time limit of 100 seconds.

The results can be summarized as follows. First, the table clearly reveals the increase in problem complexity as we move further from the basic assumptions of the RCPSP. All RCPSP relaxations lead to an increase in the number of sub-activities (e.g. an increase from 20 to approximately 61.85 sub-activities for the PRCPSP and PRCPSP-FT to 300.25 sub-activities for the PDTRTP-FT). However, note that the PRCPSP is still easier to solve than the PRCPSP-FT, although both rows show an equal number of sub-activities. Hence, the order strength, that measures the presence of precedence relations in the sub-activity network, has decreased from 0.5 to approximately 0.46 for the PRCPSP-FT and increased to approximately 0.53 for the PRCPSP. This lower (higher) amount of precedence relations is responsible for the difference in problem complexity between the PRCPSP and the PRCPSP-FT, which is completely in line with the negative effect of OS on problem complexity in literature (Herroelen and De Reyck, 1999).

Second, the table shows that dedicated and problem-specific algorithms always perform better than the RCPSP branch-and-bound procedure applied on the sub-activity networks. Though the DH92 procedure shows relatively good results for the PRCPSP, the dedicated DH96 clearly outperforms it. The RCPSP-FT and the PDTRTP-FT could not be efficiently solved by the DH92 procedure. The DV06 adaptations clearly improve the results, both from a CPU-time as from a percentage solved to optimality point-of-view. The DV06 procedure is able to optimally solve all PRCPSP-FT problem instances with up to 16 non-dummy activities, and can optimally solve 309 20-activity problem instances within the pre-specified limit of 100 seconds. The average CPU-times decrease drastically compared to the DH92 procedure from 23.82 to less than 1 second for the 20-activity instances. The PDTRTP-FT instances could not be solved by the DH92

procedure, but show excellent results for the DV06 procedure. Almost all problem instances could be optimally solved at very low CPU requirements. Thanks to the removal of all within-activity precedence relations (fast tracking) as well as the presence of variable durations (fixed work), optimal schedules often utilize all available resources almost completely. Hence, the initial lower bound LB_0 is often equal to the initial UB_0, such that no branching is needed.

Insert Table 2 About Here

The increase in problem complexity by relaxing the basic assumptions of the RCPSP is, from an algorithmic point-of-view, straightforward. However, the increase in problem complexity can also be considered from a project manager's point-of-view, who is using a commercial software scheduling package to find a resource-feasible schedule. The many options (activity splitting, fast tracking, fixed duration versus fixed work) all have a result on the quality of the schedule, and the more complex the problem description, the more degrees of freedom the software has. Hence, an optimal schedule for the PRCPSP-FT or PDTRTP-FT, for example, might lead to a very tight schedule in which many original activity input durations and resource units have been changed due to activity interruptions (pre-emption), precedence relation violations (fast tracking) and multiple execution modes (fixed work instead of fixed duration). Due to the adaptations of the original DH92 procedures and the computational experience of this section, we are able to measure the impact of all activity assumptions on the quality of the schedule, which is the subject of the next sub-section. In doing so, the project manager can get insight into the impact of various activity assumptions and better balance on the trade-off between relaxed activity assumptions and too many degrees of scheduling freedom.

4.2 The impact of the activity assumptions on project lead-time and resource utilization

In this section, we report results for the various activity assumptions and their impact on the quality of the schedule. To that purpose, we rely on the dedicated algorithms for the problem types under study:

The DH92 procedure for the RCPSP, the DH96 procedure for the PRCPSP, the DV06 procedures of section 3 for the PRCPSP-FT and the PDTRTP-FT and the Demeulemeester et al. (2000) procedure (DDH00) for the DTRTP.

The impact of the activity assumptions on the schedule quality has been measured both from a project lead-time and a resource utilization point-of-view. The row "Avg. LT" of table 3 displays the average decrease of total project duration compared to the minimal makespan found by solving the RCPSP problem. The row "Avg. Res" displays the *average resource utilization ratio* (ARUR), defined as the *resource utilization ratio* (Valls et al., 2002) for all resource types averaged over the complete scheduling horizon, i.e. $ARUR = \frac{1}{f_n \cdot |R|} \sum_{k=1}^{|R|} \sum_{i=1}^{n} \frac{W_{ik}}{a_k}$. As an example, the average resource utilization ratio equals (6 + 4 + 4 + 4 + 4 + 4 + 3 + 3 + 2) / (9 * 6) = 62.9% (only 1 resource type) for the RCPSP schedule of figure 1 and 70.8% (80.9%) for the optimal schedule for the PRCPSP (PRCPSP-FT and PDTRTP-FT) of figure 2.

Insert Table 3 About Here

The results can be summarized as follows. First, allowing pre-emption in the RCPSP has almost no effect on both the lead-time and the resource utilization. Hence, the 'task splitting' option of project scheduling software, which results in pre-emptive and often less clear schedules, is no good alternative to improve the schedule quality. Second, the shift from fixed duration activities to fixed work activities (DTRTP), however, has a major effect on both the lead-time (an improvement with approximately 20%) and resource utilization (from approximately 75% to 92% or more). Hence, the 'fixed work' option should be carefully considered as a default option, since – although resulting in an increasing problem complexity – it has a major beneficial effect on the schedule quality. Third, 'within-activity fast tracking' turns out to have a beneficial effect on the fixed duration activities (PRCPSP-FT), leading to approximately 15% lead-time improvement and 88% resource utilization, but the extra benefits when using fixed work activities (PDTRTP-FT) are relatively small compared to the very efficient schedules found by the

DTRTP. Hence, allowing fixed work activities already results in a very efficient schedule, making the within-activity fast tracking a redundant alternative to improve schedule quality.

5 CONCLUSIONS

In this paper, we provided a computational experiment of a project network dataset in order to measure the impact of three different activity assumptions on the overall quality of the schedule. The three activity assumptions, fixed duration or fixed work, activity splitting and fast tracking, are closely related to project scheduling software options and need to be made by the project manager. The schedule quality has been measured both from a lead-time as from a resource utilization point-of-view. All activity assumptions can be considered as relaxations from the activity assumptions for the well-known resource-constrained project scheduling problem (RCPSP)

The results show that all relaxations lead to an increase of the problem complexity, and hence, problem specific procedures are needed to solve problem instances to optimality. Activity pre-emption has only a small positive effect on the schedule quality. The extension to fixed work and/or fast tracking has a major effect on both the project lead-time and resource utilization. The additional effect of fast tracking on the DTRTP seems to be negligible compared to the increase in problem complexity. We believe that the provided insights are valuable for project managers when using commercial project scheduling software packages to help them choosing the activity options and carefully balancing on the trade-off between complexity and schedule quality impact.

Our future intensions are twofold. First, we aim at the construction of efficient meta-heuristic solution procedures to solve the PRCPSP-FT and the PDTRTP-FT where setups are incorporated between pre-emptive sub-activities. These solution procedures should be able to cope with large-sized and realistic problem settings. Second, we want to extend this approach to a flexible activity assumptions problem setting, where each of the activity assumption can differ among activities in the same project. In doing so, we allow an option for each activity, and tighten the gap between commercial software packages and various operations research based solution procedures from literature.

REFERENCES

Ballestin, F., Valls, V., Quintanilla, S., 2006, Pre-emption in resource-constrained project scheduling, working paper, Universidad Publica de Navarra, Spain.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., Pesch, E., 1999. Resource-constrained project scheduling: notation, classification, models and methods, European Journal of Operational Research, 112, 3-41.

Demeulemeester, E., Herroelen, W., 1992, A branch-and-bound procedure for the multiple resource-constrained project scheduling problem, Management Science, 38 (12), 1803-1818.

Demeulemeester, E., Herroelen, W., 1996, An efficient optimal solution procedure for the pre-emptive resource-constrained project scheduling problem, European Journal of Operational Research, 90, 334-348.

Demeulemeester, E., De Reyck, B., Herroelen, W., 2000, The discrete time/resource trade-off problem in project networks: a branch-and bound approach, IIE Transactions, 32 (11), 1059-1069.

Demeulemeester, E., Vanhoucke, M., Herroelen, W., 2003. A random network generator for activity-on-the-node networks, Journal of Scheduling, 6, 13-34.

De Reyck, B., Demeulemeester, E., Herroelen, W., 1998, Local search methods for the discrete time/resource trade-off problem in project networks, Naval Research Logistics, 45 (6), 553-578.

Hartmann, S., Kolisch, R., 2000. Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem, European Journal of Operational Research, 127, 394-407.

Herroelen, W., De Reyck, B., Demeulemeester, E., 1998. Resource-constrained project scheduling: a survey of recent developments, Computers and Operations Research, 25 (4), 279-302.

Herroelen, W., De Reyck, B., 1999. Phase transitions in project scheduling, Journal of Operational Research Society, 50, 148-156.

Icmeli, O., Erenguc, S.S., Zappe, C.J., 1993. Project scheduling problems: a survey, International Journal of Operations and Productions Management, 13 (11), 80-91.

Kaplan, L., 1988, Resource-constrained project scheduling with pre-emption of jobs, Unpublished Phd Dissertation, University of Michigan.

Kaplan, L., 1991, Resource-constrained project scheduling with setup times", Unpublished paper, Department of Management, University of Tenessee, Knoxville.

Kolisch, R., Hartmann, S., 2004. Experimental investigation of Heuristics for resource-constrained project scheduling: an update, working paper, Technical University of Munich, Munich.

Kolisch, R., Padman, R., 2001. An integrated survey of deterministic project scheduling, Omega, 49 (3), 249-272.

Kolisch, R., Sprecher, A., 1996. PSPLIB – A project scheduling library, European Journal of Operational Research, 96, 205-216.

Özdamar, L., Ulusoy, G., 1995. A survey on the resource-constrained project scheduling problem, IIE Transactions, 27, 574-586.

Uyttewaal, E., 2005, Dynamic scheduling with Microsoft Office Project 2003 – the book by and for professionals, J. Ross Publishing, Florida.

Valls, V., Ballestin, F. and Quintanilla, S., 2002. A hybrid genetic algorithm for the resource-constrained project scheduling problem with the peak crossover operator, Eighth International Workshop on Project Management and Scheduling, 368-371.

FIGURE 1

Example activity network and optimal schedule of the RCPSP

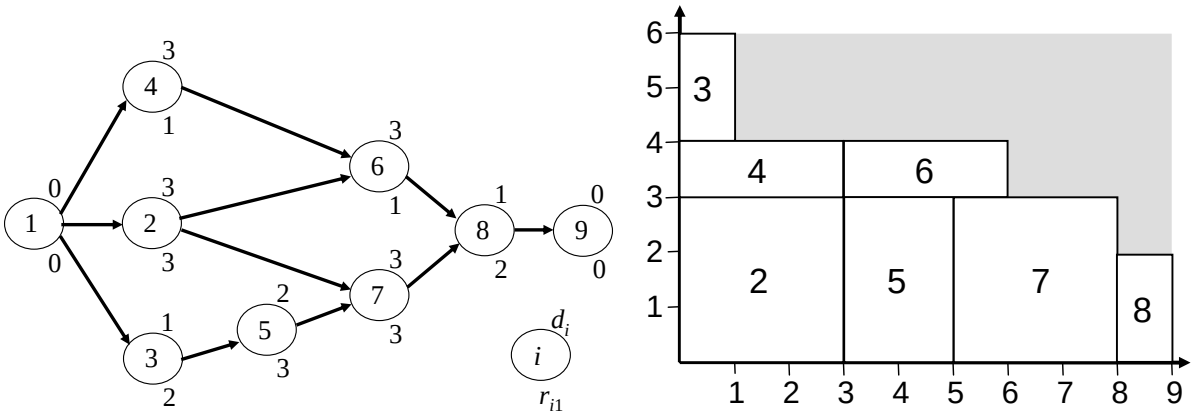


FIGURE 2

Resource-constrained project scheduling under various activity assumptions

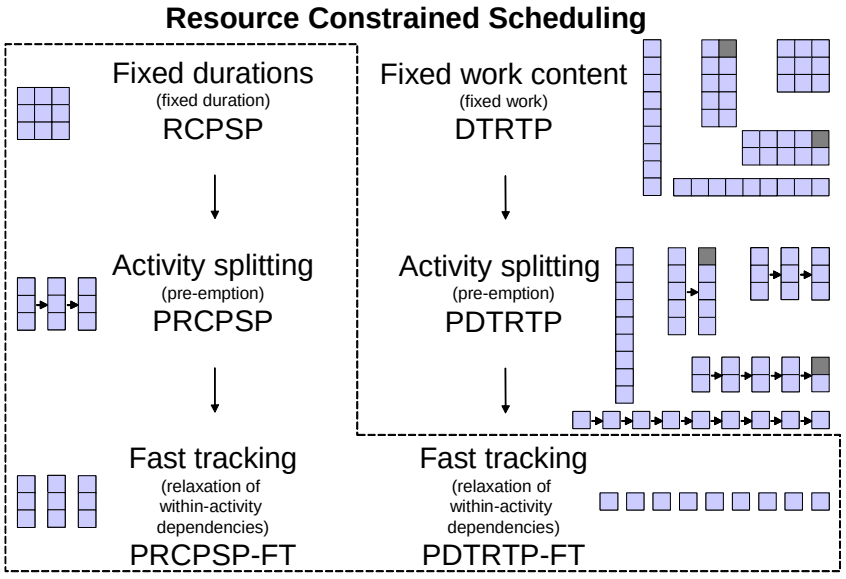
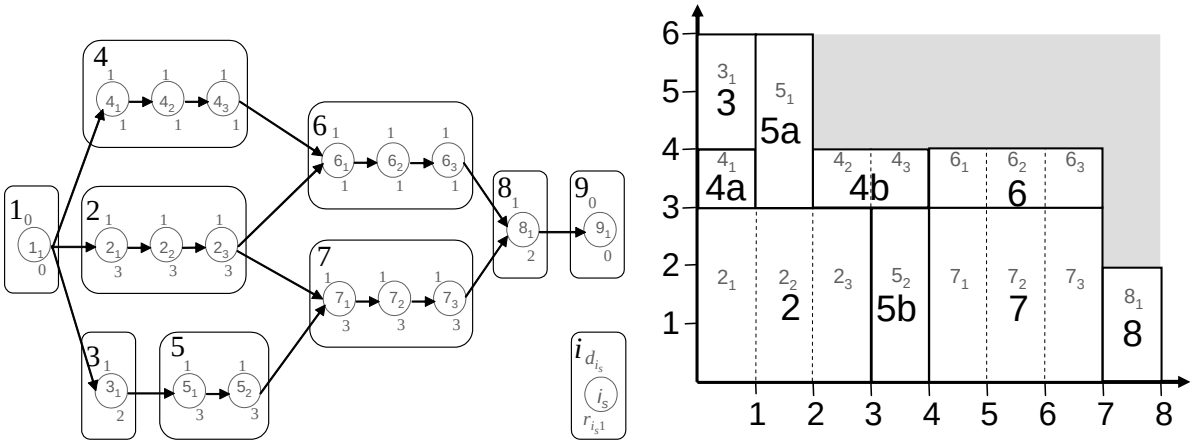


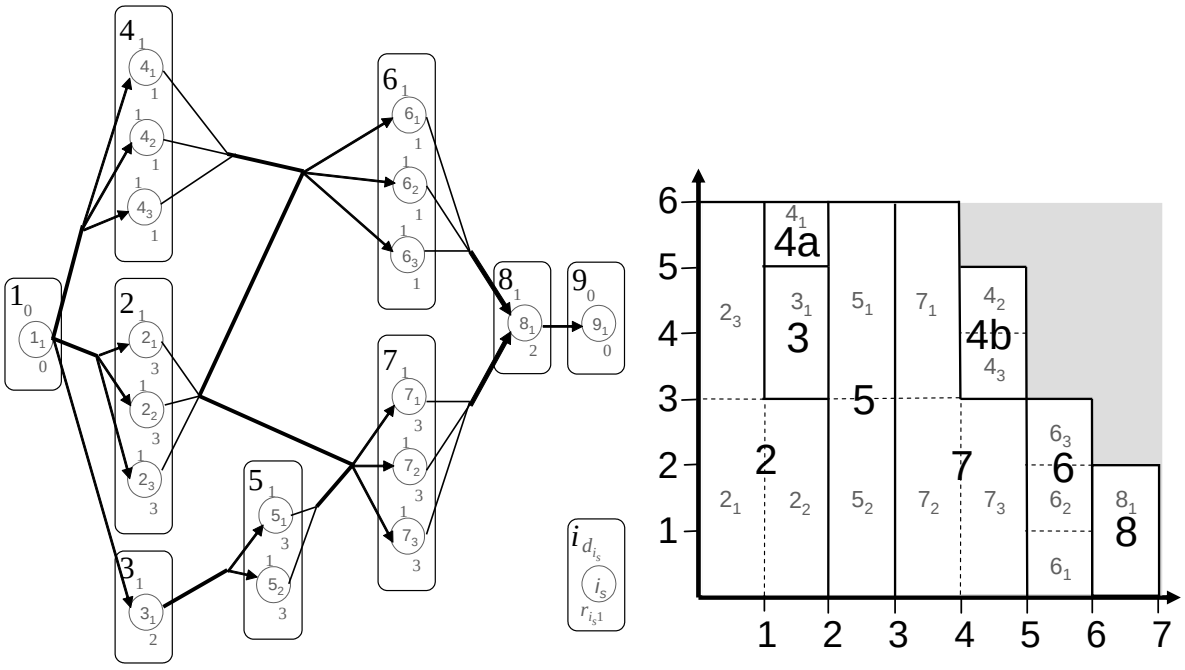
FIGURE 3

The sub-activity network and corresponding optimal schedule for the PRCPSP, PRCPSP-FT and the PDTRTP-FT

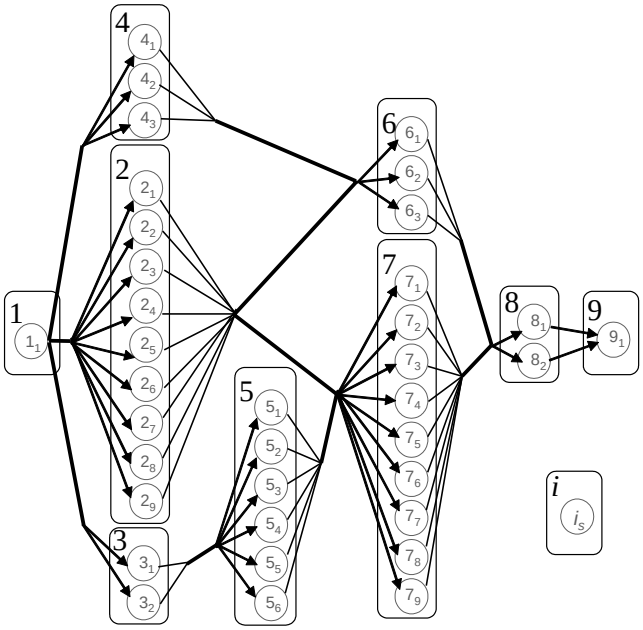
(a) The PRCPSP



(b) PRCPSP-FT



(c) PDTRTP-FT



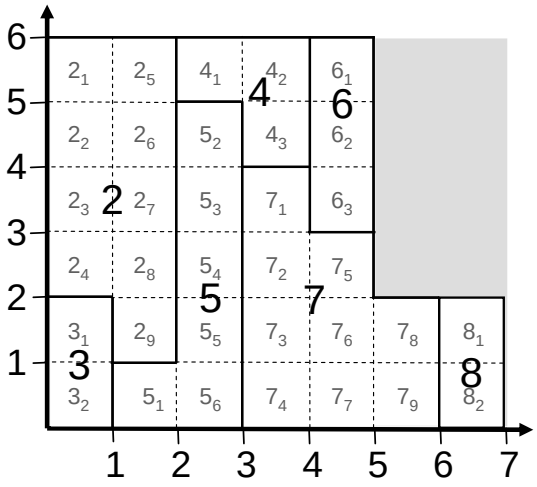


FIGURE 4

The activity network and corresponding optimal schedule for the DTRTP

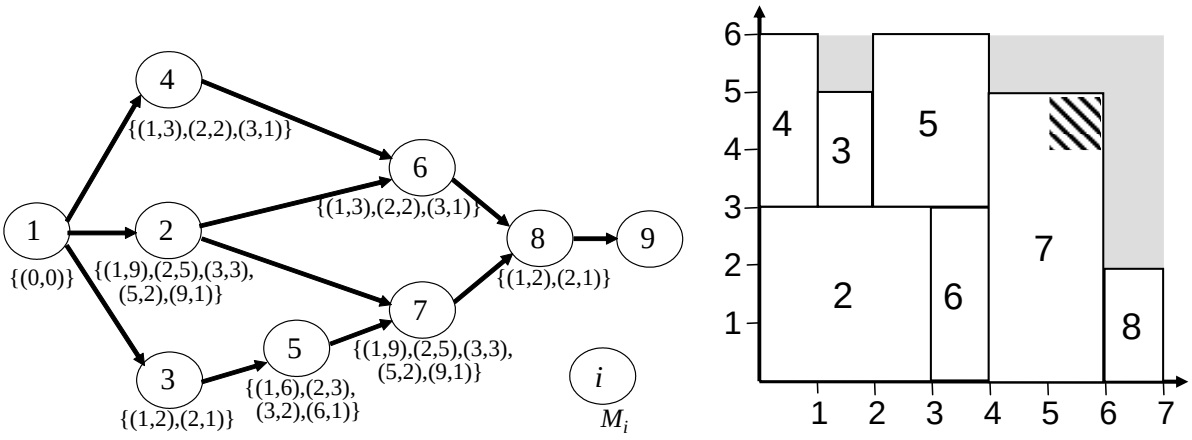


TABLE 1

The minimal delaying alternatives at the initial level of the example problem

Activity	$i = 2$			$i = 3$	$i = 4$						
Sub-activity	1	2	3	1	1	2	3	e_2	e_3	e_4	selected
Alternative											
1			×	×	×	×	×	1	1	3	Yes
2		×		×	×	×	×	1	1	3	No
3		×	×			×	×	2	0	2	Yes
4		×	×		×		×	2	0	2	No
5		×	×		×	×		2	0	2	No
6		×	×	×				2	1	0	Yes
7	×			×	×	×	×	1	1	3	No
8	×		×			×	×	2	0	2	No
9	×		×		×		×	2	0	2	No
10	×		×		×	×		2	0	2	No
11	×		×	×				2	1	0	No
12	×	×				×	×	2	0	2	No
13	×	×			×		×	2	0	2	No
14	×	×			×	×		2	0	2	No
15	×	×		×				2	1	0	No
16	×	×	×					3	0	0	Yes

FIGURE 5

The sub-activity network for the PDTRTP-FT with EST_{i_s} and L_{i_s} values

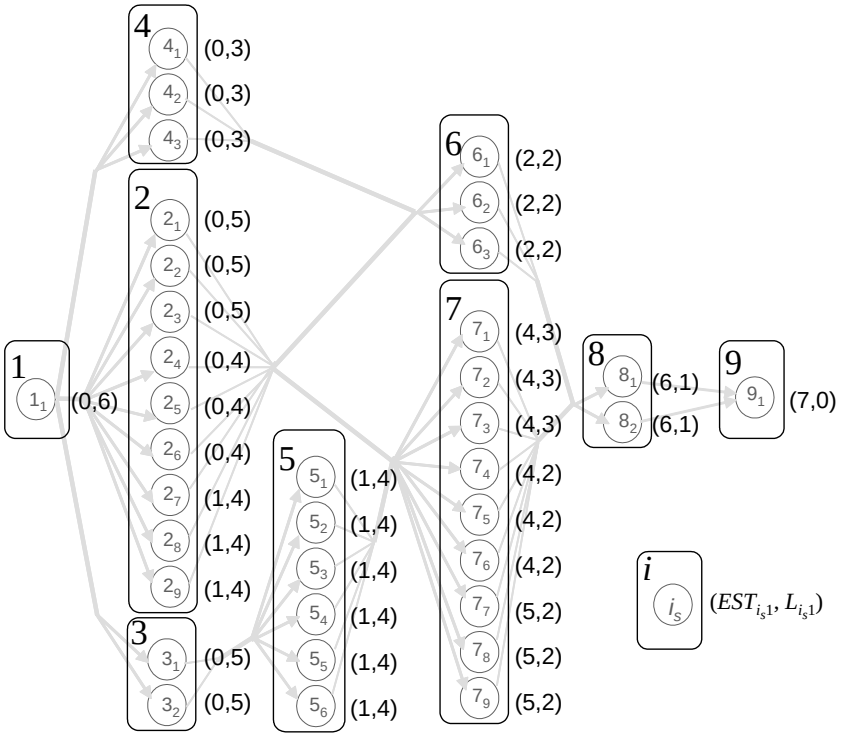


TABLE 2

The RCPSP and the impact of the pre-emption (PRCPSP) and fast tracking (PRCPSP-FT and PDTRTP-FT) on problem complexity

Number of activities		10	12	14	16	18	20	Total
RCPSP								
Sub-activities		10	12	14	16	18	20	
Avg. OS		0.5	0.5	0.5	0.5	0.5	0.5	0.5
DH92	Avg. CPU	0.00	0.00	0.00	0.00	0.00	0.01	0.00
	% Opt	100%	100%	100%	100%	100%	100%	100%
PRCPSP								
Sub-activities		31.83	37.82	44.21	50.21	55.75	61.85	
Avg. OS		0.55	0.53	0.53	0.53	0.53	0.52	0.53
DH92	Avg. CPU	0.00	0.00	0.04	0.09	0.85	3.67	0.77
	% Opt	100%	100%	100%	100%	99%	97%	99%
DH96	Avg. CPU	0.00	0.00	0.03	0.06	0.57	3.20	0.64
	% Opt	100%	100%	100%	100%	100%	97%	99%
PRCPSP-FT								
Sub-activities		31.83	37.82	44.21	50.21	55.75	61.85	
Avg. OS		0.46	0.46	0.46	0.47	0.48	0.48	0.47
DH92	Avg. CPU	0.43	2.64	7.59	11.67	16.95	23.82	10.52
	% Opt	100%	98%	93%	90%	86%	80%	91%
DV06	Avg. CPU	0.00	0.00	0.05	0.22	1.22	3.96	0.91
	% Opt	100%	100%	100%	100%	99%	97%	99%
PDTRTP-FT								
Sub-activities		151.56	181.13	212.33	243.00	269.80	300.25	
Avg. OS		0.41	0.41	0.43	0.44	0.44	0.45	0.43
DH92	Avg. CPU	97.04	98.46	99.88	100.00	100.00	100.00	99.23
	% Opt	3%	2%	0%	0%	0%	0%	1%
	Avg. CPU	0.04	0.36	0.40	0.35	1.30	0.50	0.49
DV06	% Opt	100%	100%	100%	100%	99%	100%	100%
	$LB_0 = UB_0$	91%	92%	94%	92%	92%	93%	92%

Note: DH92 : procedure used for RCPSP, PRCPSP, PRCPSP-FT and PDTRTP-FT
DH96 : procedure used for the PRCPSP
DV06 : procedure for PRCPSP-FT and PDTRTP-FT

TABLE 3

The schedule quality for the RCPSP, PRCPSP, PRCPSP-FT, DTRTP and the PDTRTP-FT

Number of activities		10	12	14	16	18	20	Total
DH92	**RCPSP**							
	Avg. LT	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Avg. Res	69.66%	72.60%	75.50%	76.55%	78.00%	79.25%	75.26%
DH96	**PRCPSP**							
	Avg. LT	0.47%	0.47%	0.59%	0.51%	0.49%	0.49%	0.50%
	Avg. Res	70.05%	72.30%	76.01%	76.98%	78.44%	79.67%	75.58%
DV06	**PRCPSP-FT**							
	Avg. LT	18.91%	17.01%	14.76%	14.00%	12.75%	11.85%	14.88%
	Avg. Res	85.63%	87.44%	88.55%	89.08%	89.36%	89.60%	88.28%
DDH00	**DTRTP**							
	Avg. LT	25.25%	23.13%	20.81%	20.20%	19.10%	18.06%	21.09%
	Avg. Res	92.18%	93.71%	94.76%	95.61%	95.97%	96.31%	94.76%
DV06	**PDTRTP-FT**							
	Avg. LT	26.19%	23.85%	21.37%	21.08%	19.65%	18.72%	21.81%
	Avg. Res	93.43%	94.66%	95.99%	96.49%	96.63%	97.18%	95.73%