Technische Universität Berlin

FACHBEREICH 3
MATHEMATIK

APPROXIMATION ALGORITHMS FOR THE DISCRETE TIME-COST TRADEOFF PROBLEM

MARTIN SKUTELLA

No. 514/1996

Approximation Algorithms for the Discrete Time-Cost Tradeoff Problem *

Martin Skutella **

April 1996; revised November 1997

Abstract

Due to its obvious practical relevance, the TIME-COST TRADEOFF PROBLEM has attracted the attention of many researchers over the last forty years. While the LINEAR TIME-COST TRADEOFF PROBLEM can be solved in polynomial time, its discrete variant is known to be *NP*-hard. We present the first approximation algorithms with constant performance guarantee for the DISCRETE TIME-COST TRADEOFF PROBLEM. Specifically, given a fixed budget we consider the problem of finding a shortest schedule for a project. We give an approximation algorithm with performance ratio $3/2$ for the class of projects where all feasible durations of activities are either 0, 1, or 2. We extend our result by giving approximation algorithms with performance guarantee $O(\log \ell)$. Here, ℓ is the ratio of the maximum duration of any activity to the minimum non-zero duration of any activity. Finally, we discuss bicriteria approximations where both time and cost are within a constant factor of an optimum.

key words: time-cost tradeoff, approximation algorithm, scheduling, bicriteria optimization

1 Introduction

An instance P of the TIME-COST TRADEOFF PROBLEM is a *project* given by a finite set of *activities* $J^P = J$ together with a *partial order* $(J, \prec)$ on the set of activities. In order to carry out a project, all the activities have to be executed in accordance with the precedence constraints given by the partial order: if $j \prec k$ activity k may not be started before activity j is completed. The activities are indivisible tasks, hence their execution must not be interrupted. The *duration* of an activity $j \in J$, i. e., the difference between its completion and start time, depends on the amount of money that is paid for it. This correlation is described by a non-increasing non-negative *cost function* $c_j^P = c_j : \mathbb{R}_+ \to \mathbb{R}_+ \cup \{\infty\}$ for each activity $j \in J$, where $c_j(x_j)$ is the amount of money one has to pay to run j with duration x_j. We will drop the upper index P at any symbol whenever it is clear from the context.

Throughout this paper we will consider projects that are represented by an edge diagram. This is a directed acyclic graph where each activity $j \in J$ is represented by an edge of the graph, such that for any two activities $j, k \in J$ there is a directed path from j to k if and only if $j \prec k$. In general, *dummy edges* are needed to represent the precedence constraints in this way. Such a dummy edge corresponds to a *dummy activity* j with $c_j(x_j) = 0$ for all $x_j \geqslant 0$. We can without loss of generality merge all the sources (vertices with in-degree 0) of the edge diagram to a super-source s and all the sinks (vertices with out-degree 0) to a super-sink s'.

*An extended abstract of this paper appeared in Michael Saks et al., editors, Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'97). Society of Industrial and Applied Mathematics (1997), 501–508.

**FB Mathematik, MA 6-1, Technische Universität Berlin, Straße des 17. Juni 136, D-10623 Berlin, Germany, `skutella@math.tu-berlin.de`

In order to keep the size of the edge diagram small it is desirable to use as few dummy edges as possible. However, Krishnamoorthy and Deo (1979) proved that it is NP-hard to obtain a representation with a minimum number of dummy edges. On the other hand one can easily find an edge diagram where the number of dummy edges is polynomially bounded in the input size of the project. To be more precise, given a project by a set of activities J and precedence constraints $(J, \prec)$ on J, an edge diagram can be constructed by introducing for each pair of activities $j, k \in J$ with $j \prec k$ a dummy edge between the endpoint of edge j and the starting point of edge k.

A *realization* $x^P = x \in \mathbb{R}_+^J$ of project P is an assignment of durations x_j to activities $j \in J$. The *total cost* $c^P(x) = c(x)$ of realization x is given by

$$c(x) := \sum_{j \in J} c_j(x_j) \quad .$$

The *project duration* $t^P(x) = t(x)$ of the realization x is the makespan of the earliest start schedule which starts each activity at the earliest point in time obeying the precedence constraints and durations x_j. In other words, the project duration equals the length of a longest chain in the partial order which is itself the length of a longest directed s–s'–path in a corresponding edge diagram. Thereby, the length of an edge corresponding to an activity j is the chosen duration x_j.

Ideally, we would like to minimize both time and cost for a given project P. Unfortunately, there is a tradeoff between time and cost, i. e., short realizations are usually expensive and cheap realizations take a long time. Fixing either cost or time we get two related optimization problems with the objective to minimize the other parameter. The first problem is the

> BUDGET PROBLEM: For a given non-negative budget $B \geqslant 0$, find a shortest realization x satisfying $c(x) \leqslant B$.

Therefore, we are interested in the function $T_{opt}^P = T_{opt} : \mathbb{R}_+ \to \mathbb{R}_+ \cup \{\infty\}$ that gives the minimum time $T_{opt}(B)$ needed for realizing project P with budget B:

$$T_{opt}(B) := \min\big\{t(x) \mid x \in \mathbb{R}_+^J,\, c(x) \leqslant B\big\} \quad .$$

Since this minimum exists in all the cases that we will consider, the function is well-defined. The second problem is the

> DEADLINE PROBLEM: For a given project duration $T \geqslant 0$ (deadline) find a cheapest realization x satisfying $t(x) \leqslant T$.

Therefore, we are interested in the function $B_{opt}^P = B_{opt} : \mathbb{R}_+ \to \mathbb{R}_+ \cup \{\infty\}$ that gives the minimum budget $B_{opt}(T)$ needed to realize project P in time T:

$$B_{opt}(T) := \min\big\{c(x) \mid x \in \mathbb{R}_+^J,\, t(x) \leqslant T\big\} \quad .$$

Again, the minimum exists in all the cases that we will consider.

A realization x of the project P is an *optimal realization* if $c(x) = B_{opt}\big(t(x)\big)$ and $t(x) = T_{opt}\big(c(x)\big)$. That is, x is an optimal realization if and only if P can be realized neither cheaper nor shorter without increasing time or cost. An optimal realization x is called optimal for a deadline $T \geqslant 0$, if $c(x) = B_{opt}(T)$. It is called optimal for a budget $B \geqslant 0$, if $t(x) = T_{opt}(B)$.

The BUDGET PROBLEM and the DEADLINE PROBLEM are special parts of the TIME-COST TRADEOFF PROBLEM that was formulated almost forty years ago by Kelley and Walker (1959): find optimal realizations for all deadlines $T \geqslant 0$ (or equivalently for all budgets $B \geqslant 0$). They considered *linear projects* where all cost functions of activities are affine linear and decreasing functions over closed intervals. The LINEAR TIME-COST TRADEOFF PROBLEM has independently been solved by Fulkerson

(1961) and Kelley (1961). Later, Philips and Dessouky (1977) gave an improved version of the original algorithms.

In contrast to the LINEAR TIME-COST TRADEOFF PROBLEM, its discrete variant, the DISCRETE TIME-COST TRADEOFF PROBLEM, is known to be *NP*-hard (De et al., 1997). In a *discrete project* the duration of each activity can be chosen from a finite number of alternatives. In this paper we assume that for an activity j all possible durations are explicitly given such that the encoding length of j is linear in the number of possible durations. Since discrete alternatives are quite common in practice (Harvey and Patterson, 1979; Hindelang and Muth, 1979) and can also be used for approximating arbitrary time-cost relationships of jobs (Panagiotakopoulos, 1977; Robinson, 1975), the DISCRETE TIME-COST TRADEOFF PROBLEM has frequently been considered; for further references see (De et al., 1995).

Since one cannot find algorithms that compute optimal realizations for the DISCRETE TIME-COST TRADEOFF PROBLEM in polynomial time, unless $P = NP$, we are interested in algorithms that run in polynomial time and compute provably good realizations: an α*–approximation algorithm* is a polynomial-time algorithm that produces a feasible solution whose value is within a factor of α of the optimum; α is called *performance guarantee* or *performance ratio* of the algorithm. To the best of the author's knowledge no approximation algorithm with constant performance guarantee was known before for the DISCRETE TIME-COST TRADEOFF PROBLEM.

We present the following results: first of all we show that it suffices to consider projects with at most two alternatives for the duration of each activity, where the shorter of two possible durations is zero. This enables us to introduce a relaxation of discrete projects to linear projects. An optimal solution to this linear relaxation then serves as a surrogate for the true optimum in our estimations. Moreover, the structure of an optimal realization for the linear relaxation guides the construction of provably good realizations for the discrete project.

Using a simple rounding technique, we give approximations with performance guarantee ℓ for the BUDGET PROBLEM and the DEADLINE PROBLEM of projects where all possible durations of activities are in the range $\{0,\dots,\ell\}$. Using somewhat more sophisticated ideas these results can be improved. For the special class of projects where all feasible durations are either 0, 1, or 2, we present an approximation algorithm with performance guarantee $3/2$ for the BUDGET PROBLEM. We also show that there exists no approximation algorithm with a better performance guarantee for the considered class of instances, unless $P = NP$. Furthermore, for the more general class of discrete projects where all possible durations of activities lie in the set $\{0,\dots,\ell\}$ we present approximation algorithms for the BUDGET PROBLEM with performance guarantee depending logarithmically on ℓ. We also show that the analysis is tight. On the other hand we argue why we get much better results for wide classes of projects. Finally, we discuss bicriteria approximation algorithms that construct feasible realizations for arbitrary discrete projects such that both time and cost are within a constant factor of the optimal solution.

The paper is organized as follows: in the next section we state some important properties of *linear projects*; in particular, we describe the algorithm of Philips and Dessouky (1977). In Section 3 we consider *discrete projects*, present the reduction to the case of at most two alternatives for the duration of each activity, and describe the linear relaxation. This enables us to develop simple ℓ–approximation algorithms in Section 4. In Section 5 we present the improvement to performance guarantee $3/2$ for the BUDGET PROBLEM in case $\ell = 2$. For arbitrary ℓ, we give $O(\log \ell)$–approximations in Section 6. Finally, in Section 7 we discuss bicriteria approximation algorithms with constant performance ratio for the DISCRETE TIME-COST TRADEOFF PROBLEM.

2 The Linear Time-Cost Tradeoff Problem

In this section we consider projects P where the duration of each activity $j \in J$ can be chosen from a certain positive interval $[a_j^P, b_j^P] = [a_j, b_j]$ belonging to this activity. Moreover, the cost function c_j of j is affine linear and decreasing within that interval. Consequently, it is defined by the values $c_j(a_j), c_j(b_j) \in \mathbb{R}_+$ and can be written in the following form:

$$c_j(t) = \begin{cases} \infty & \text{if } t < a_j\,, \\ \frac{b_j - t}{b_j - a_j} c_j(a_j) + \frac{t - a_j}{b_j - a_j} c_j(b_j) & \text{if } a_j \leqslant t \leqslant b_j\,, \\ c_j(b_j) & \text{if } t \geqslant b_j\,. \end{cases}$$

Since c_j is constant for $t \geqslant b_j$, we will only consider realizations x satisfying $a \leqslant x \leqslant b$. An instance P of the described form is called a *linear project* and is an instance of the LINEAR TIME-COST TRADEOFF PROBLEM (LTCT PROBLEM).

Since the realization $x = b$ is obviously the shortest realization of a linear project P for the minimum budget $c(b)$ and since there can be no cheaper realization of P at all, it is an optimal realization for the deadline $t(b)$. Moreover, the duration $t(a)$ of the realization a is the shortest deadline that can be reached with finite cost, though a is not an optimal realization for this deadline in general.

The following result was discovered by Fulkerson (1961) and independently by Kelley (1961). It is not only crucial for solving the LINEAR TIME-COST TRADEOFF PROBLEM, but it also plays a fundamental role in the derivation of our approximation results for the discrete case. That is why we give a new proof for it here.

Lemma 2.1. *For a linear project P the functions B_{opt} and T_{opt} are piecewise affine linear, convex, non-increasing, and continuous, except for the intervals where the function values are infinite.*

Proof. Consider the subset of $\mathbb{R}_+^{|J|+2}$ consisting of all triples (x, T, B) where $x \in \mathbb{R}_+^J$, $a \leqslant x \leqslant b$, is a feasible realization, $T \geqslant t(x)$, and $B \geqslant c(x)$. Since function c is linear and t is the maximum of a finite number of linear functions (namely the maximum length of all directed s–s'–paths in the edge diagram), the considered subset is a polyhedron in $\mathbb{R}_+^{|J|+2}$. Therefore its projection onto the coordinates (T, B) is a two-dimensional polyhedron which we call *epi* and which is given by

$$\begin{aligned} epi &= \{(T,B) \mid T \geqslant t(x) \text{ and } B \geqslant c(x) \text{ for some } a \leqslant x \leqslant b\} \\ &= \{(T,B) \mid T \geqslant T_{opt}(B)\} \\ &= \{(T,B) \mid B \geqslant B_{opt}(T)\}\ . \end{aligned}$$

The last two equalities are direct consequences of the definitions of T_{opt} and B_{opt}. Thus the polyhedron *epi* is the epigraph of T_{opt} and B_{opt} and the functions have the desired properties. □

As a consequence of Lemma 2.1 it suffices to know all the breakpoints of the function B_{opt} in order to construct B_{opt} and T_{opt}. Moreover, given for each breakpoint a corresponding optimal realization, one can easily compute an optimal realization for an arbitrary deadline or budget as a convex combination of the optimal realizations corresponding to the two neighboring breakpoints.

The algorithm of Philips and Dessouky solves the LINEAR TIME-COST TRADEOFF PROBLEM by computing B_{opt} and optimal realizations for all the breakpoints. It starts with the optimal realization b and constructs a sequence of optimal realizations for decreasing deadlines and increasing budgets. In particular, this sequence contains optimal realizations for all the breakpoints of B_{opt} and T_{opt}; we refer to this algorithm as LTCT-SOLVER. Since it will be used as a subroutine in our approximation algorithms we give a more detailed explanation in what follows.

As already mentioned, algorithm LTCT-SOLVER computes optimal solutions for decreasing deadlines. Thus, given an optimal realization x, it has to find a way to shorten x without loosing optimality, i. e., without too much increase in cost. More precisely, the increase in cost must not exceed the absolute value of the left-hand derivative of B_{opt} at $t(x)$. Since it is obviously useless to shorten the durations x_j of those activities j whose corresponding edges do not lie in a longest s–s'–path at the moment, algorithm

LTCT-SOLVER only considers the subgraph that is induced by critical edges; we call an edge *critical* if it lies on a longest s–s'–path.

If the realization x can be shortened, i. e., if $t(x) > t(a)$, then there exists an (s,s')–cut S in the subgraph of critical edges such that all the activities j that correspond to forward edges in S can be shortened, i. e., $x_j > a_j$. Shortening all those durations uniformly by $\delta > 0$ leads to a decrease of the project duration by the same amount or a positive multiple of it as long as no other edge becomes critical. Of course, this can also only be done until one of the edges in the cut attains its minimum duration a_j. To save cost we can at the same time uniformly enlarge the durations of those activities j, with $x_j < b_j$, that correspond to backward edges in S by the same amount; this can be done until one of them has reached its upper bound b_j.

For each (s,s')–cut S we can in this way define the cost used per time to shorten all forward edges and enlarge backward edges of S together with a maximum possible δ_S. Changing durations along a *cheapest cut* S^* in the subgraph of critical edges by δ_{S^*} as described above preserves optimality and shortens the project duration by exactly δ_{S^*}. Moreover, such a cheapest cut can be found by solving a maximum flow problem on the subgraph of critical edges. We do not go into the details at this point, the interested reader is referred to the work of Philips and Dessouky (1977).

To summarize, algorithm LTCT-SOLVER starts with the optimal realization $x = b$ and then iteratively shortens this realization along a cheapest (s,s')–cut S^* in the subgraph of critical edges by δ_{S^*}. The algorithm stops as soon as the project duration $t(x)$ has reached the minimum possible duration $t(a)$. The running time of each iteration of the algorithm is dominated by the running time needed to find the minimum cut. This can be done in $O\big(nm\log(n^2/m)\big)$ time (Goldberg and Tarjan, 1988), where n denotes the number of vertices and m the number of edges of the current subgraph. Since there are no isolated vertices in the edge diagram and edges correspond to activities of the project, we get $n \leqslant m \leqslant |J|$. Hence the overall running time of algorithm LTCT-SOLVER is $O\big(\#\text{iterations} \cdot |J|^2 \log|J|\big)$.

We should mention here that the number of iterations could possibly be exponential in the input size, since there could exist exponentially many breakpoints for the function B_{opt} and the algorithm needs at least one iteration to get from one breakpoint to the next. However, it is not known if there really exist projects where B_{opt} has exponentially many breakpoints. Algorithm LTCT-SOLVER might not be an efficient tool for solving the DEADLINE PROBLEM or BUDGET PROBLEM of linear projects for a single deadline or budget. Of course, both problems can efficiently be solved, since they can be formulated as linear programs of polynomial size. It was observed by Fulkerson (1961) that the dual program can easily be transformed into a min-cost flow problem. Thus there even exists an efficient combinatorial algorithm.

The following lemma is crucial for finding good realizations of discrete projects, which will be considered in Section 4.

Lemma 2.2. *If a_j, b_j are integral for all $j \in J$, then algorithm* LTCT-SOLVER *computes in $O\big((t(b) - t(a))\,|J|^2 \log|J|\big)$ time for each integral deadline T an optimal, integral realization.*

Proof. The algorithm starts with the integral realization $x = b$. Since a and b are integral, the durations of activities along the selected cut can at least be changed by $\delta := 1$ in each iteration and the realization x stays integral. As mentioned above, the project duration is then also decreased by 1 in each iteration and the algorithm computes for each integral deadline an optimal, integral realization. Moreover, the number of iterations of the algorithm can be bounded by $t(b) - t(a)$ and we get the desired result for the running time. □

A similar integrality result was already achieved in (Fulkerson, 1961) and in (Kelley, 1961, Remark 4). There it is shown that under the conditions of Lemma 2.2 all the solutions computed by Algorithm LTCT-SOLVER are integral. For a more detailed discussion of the stated results for linear projects we refer to (Fulkerson, 1961), (Kelley, 1961), and (Philips and Dessouky, 1977). An order-theoretic view of the problem can be found in (Möhring and Radermacher, 1989).

3 The Discrete Time-Cost Tradeoff Problem

We consider projects P where the duration of each single activity j can attain at most two different non-negative values h_j and k_j, where h_j equals 0 or k_j. By ignoring fixed costs we assume $c_j(k_j) = 0$ such that $c(k) = 0$ and k is a feasible realization of P for any non-negative budget B. Moreover, if $0 = h_j < k_j$ we assume $0 < c_j(h_j) < \infty$. We can think of the cost function of activity j as a step function

$$c_j(t) = \begin{cases} \infty & \text{if } 0 \leqslant t < h_j\,, \\ c_j(h_j) & \text{if } h_j \leqslant t < k_j\,, \\ 0 & \text{if } k_j \leqslant t\,. \end{cases}$$

A project of the described form is a *discrete project* and an instance of the DISCRETE TIME-COST TRADEOFF PROBLEM (DTCT PROBLEM).

At first sight, allowing only two feasible durations for each activity might look like a substantial restriction. But we can in fact model any activity j with a finite number $q > 2$ of feasible non-negative durations $d_1 < \cdots < d_q$ as a set of q parallel activities with the properties described above. These parallel activities are represented by parallel edges in the edge diagram. Again, ignoring fixed cost we assume $c_j(d_q) = 0$ and think of c_j as a step function (recall that we assume c_j to be non-increasing).

We first introduce an activity j_1 with fixed length $h_{j_1} := k_{j_1} := d_1$ and $c_{j_1}(k_{j_1}) := 0$. This activity guarantees that we cannot get shorter than the minimum feasible duration d_1 of j. Then we model the cost structure of j by introducing for all the other feasible durations d_i, $1 < i \leqslant q$, an activity j_i. The idea of this construction is that activity j_i can only be shortened below duration d_i if the difference in cost to the next shorter feasible duration d_{i-1} of j is being paid. Therefore we define $h_{j_i} := 0$, $k_{j_i} := d_i$, $c_{j_i}(0) := c_j(d_{i-1}) - c_j(d_i)$, and $c_{j_i}(k_{j_i}) := 0$. It is an easy observation that the sum of the cost functions of the new activities $j_1, \dots, j_q$ exactly equals the cost function of j. Thus there is a canonical mapping of feasible durations x_j for j to tuples of feasible durations $x_{j_1}, \dots, x_{j_q}$ for $j_1, \dots, j_q$ such that $x_j = \max\{x_{j_1}, \dots, x_{j_q}\}$ and $c_j(x_j) = c_{j_1}(x_{j_1}) + \cdots + c_{j_q}(x_{j_q})$. Moreover, this mapping is bijective if we restrict ourselves without loss of generality to tuples of durations $x_{j_1}, \dots, x_{j_q}$ satisfying $x_{j_i} = k_{j_i}$ if $k_{j_i} \leqslant \max\{x_{j_1}, \dots, x_{j_q}\}$.

Since we assume that the encoding length of each activity j is linear in the number of possible durations, the input size of a project is only increased by a constant factor if activities with more than 2 feasible durations are replaced by a set of parallel activities. In the remainder we will only consider discrete projects with no more than two feasible durations for each activity.

For the same reason that $b^{\tilde{P}}$ is an optimal realization for a linear project $\tilde{P}$, we know that k^P is an optimal realization for the discrete project P since it is the cheapest possible realization. On the other hand, the duration $t(h)$ is the shortest deadline that can be reached with finite cost, though h is not an optimal realization in general. Therefore we only consider deadlines $T \geqslant t(h)$.

In what follows we will study special classes of instances of the DISCRETE TIME-COST TRADEOFF PROBLEM: a discrete project P is an instance of the ℓ–DTCT PROBLEM for an integer $\ell \in \mathbb{N}$, if $k_j \in \{0, 1, \dots, \ell\}$ for each activity $j \in J$. If we consider arbitrary discrete projects with more than two alternative durations for activities, this is (by the above transformation) equivalent to the requirement that all feasible durations lie in the set $\{0, 1, \dots, \ell\}$. As mentioned above, it is shown in (De et al., 1997) that it is *NP*-hard to find optimal realizations for discrete projects. This is proved by reducing an *NP*-hard variant of 3SAT to instances of the 2–DTCT PROBLEM. Hence it is already *NP*-hard to find optimal realizations if we restrict ourselves to the 2–DTCT PROBLEM.

In order to design approximation algorithms for the BUDGET PROBLEM and the DEADLINE PROBLEM of discrete projects we introduce a *linear relaxation* which is used to get a lower bound on the value of an optimal solution. The linear relaxation $\tilde{P}$ of a discrete project P is a linear project that consists of the same set of activities, i. e., $J^{\tilde{P}} = J^P$. Its structure is defined by the same partial order $(J, \prec)$ on this set, hence the edge diagram corresponding to $\tilde{P}$ is the same as for P. The interval $[a_j^{\tilde{P}}, b_j^{\tilde{P}}]$ is given by $a_j^{\tilde{P}} := h_j^P$ and $b_j^{\tilde{P}} := k_j^P$ for each activity $j \in J$. The cost function $c_j^{\tilde{P}}$ is defined by $c_j^{\tilde{P}}(a_j) := c_j^P(h_j)$ and $c_j^{\tilde{P}}(b_j) := c_j^P(k_j)$. This definition of the linear relaxation is the main reason why we have transformed arbitrary discrete projects to those with at most two possible alternatives for the duration of each activity.

Lemma 3.1. *If $\tilde{P}$ is the linear relaxation of the discrete project P, then $T_{opt}^{\tilde{P}}(B) \leqslant T_{opt}^P(B)$ for all $B \geqslant 0$ and $B_{opt}^{\tilde{P}}(T) \leqslant B_{opt}^P(T)$ for all $T \geqslant 0$.*

The proof of the lemma follows immediately from the definition of the linear relaxation and is therefore left out. We shall often refer to the following two basic properties of realizations for arbitrary projects.

Lemma 3.2. *Let x, x' be realizations of the project P.*

a) If $\alpha \geqslant 0$ and $x_j \leqslant \alpha \cdot x'_j$ for each activity $j \in J$, then $t(x) \leqslant \alpha \cdot t(x')$.

b) If $\beta \geqslant 0$ and $x_j - x'_j \leqslant \beta$ for each activity $j \in J$, then $t(x) - t(x') \leqslant \beta \cdot |J|$.

Proof. Let $I \subseteq J$ be the subset of activities corresponding to a longest s–s'–path in the edge diagram with respect to x and I' the subset of J corresponding to a longest s–s'–path with respect to x', then

$$t(x) = \sum_{j \in I} x_j \leqslant \alpha \sum_{j \in I} x'_j \leqslant \alpha \sum_{j \in I'} x'_j = \alpha \cdot t(x')$$

in case a), and

$$t(x) - t(x') = \sum_{j \in I} x_j - \sum_{j \in I'} x'_j \leqslant \sum_{j \in I} x_j - \sum_{j \in I} x'_j \leqslant \sum_{j \in I} (x_j - x'_j) \leqslant \beta \cdot |J|$$

in case b). □

4 ℓ-Approximations for the ℓ-DTCT Problem

In this section, we consider the Budget Problem and the Deadline Problem for instances P of the ℓ-DTCT Problem, for arbitrary $\ell \in \mathbb{N}$. One idea to get good solutions to these problems is to compute an optimal realization $\tilde{x}$ of the linear relaxation $\tilde{P}$ of P and to round it appropriately to a feasible realization of P. The quality of this realization can then be tested by comparing its value, i. e., its duration or cost, to the value of the realization we started with.

Consider the following example: we are given a project P whose only activity j has feasible durations $h_j = 0$ and $k_j = 2$, where $c_j(2) = 0$ and $c_j(0) = q$, for some $q \in \mathbb{N}$; furthermore, we are allowed to spend the budget $B = q - 1$ and want to minimize the project duration. Since we cannot afford to choose duration 0 for activity j, the duration of the optimal realization is 2. However, the optimal solution to the linear relaxation $\tilde{P}$ has value $2/q$ and is thus a factor of q away from 2. Consequently, since q may be chosen arbitrarily large one cannot give any performance guarantee by comparing the value of a feasible realization of the discrete project to the optimal solution of its linear relaxation $\tilde{P}$. But we can overcome this drawback if we use as a lower bound the shortest integral realization of the linear relaxation for the given budget instead. This yields a duration of 1 in the our example which is only a factor of 2 away from the optimum for the discrete project P.

Therefore, we call a realization $\tilde{x}$ of $\tilde{P}$ *integral optimal* for a budget B (for a deadline T), if $\tilde{x}$ is the shortest (cheapest) integral realization of $\tilde{P}$ satisfying $c(\tilde{x}) \leqslant B$ (respectively $t(\tilde{x}) \leqslant T$). In contrast, if we talk about an *optimal, integral* realization for a budget (for a deadline) we mean one which is optimal for this budget (for this deadline) and integral. In the above example $\tilde{x}_j = 2/q$ is optimal and $\tilde{x}_j = 1$ is integral optimal for the budget $q-1$. However, $\tilde{x}_j = 1$ is not optimal, integral for the budget $q-1$ but it is optimal, integral for the budget $q/2$.

The following lemma states some important properties of integral optimal realizations.

Lemma 4.1. *Let P be an instance of the ℓ-*DTCT PROBLEM *and $\tilde{x}$ a realization for the linear relaxation $\tilde{P}$ of P.*

a) If $\tilde{x}$ is integral optimal for the deadline T with respect to $\tilde{P}$, then $c^{\tilde{P}}(\tilde{x}) \leqslant B^P_{opt}(T)$.

b) If $\tilde{x}$ is integral optimal for the budget B with respect to $\tilde{P}$, then $t(\tilde{x}) = \lceil T^{\tilde{P}}_{opt}(B) \rceil \leqslant T^P_{opt}(B)$.

c) Algorithm LTCT-SOLVER *can be used to compute integral optimal realizations of $\tilde{P}$ for all deadlines and budgets in time $O\big(\ell\,|J|^3 \log|J|\big)$.*

Proof. Since we are interested in integral realizations $\tilde{x}$ of $\tilde{P}$ only and all feasible realizations of P are integral, we can without loss of generality assume that the deadline T in part a) is integral too (because otherwise we can replace T by $\lfloor T \rfloor$). There exists an optimal, integral realization $\tilde{x}'$ for the integral deadline T by Lemma 2.2 and algorithm LTCT-SOLVER can be used to compute it. Moreover, $\tilde{x}'$ is by definition integral optimal for the deadline T and we get $c^{\tilde{P}}(\tilde{x}) = c^{\tilde{P}}(\tilde{x}') = B^{\tilde{P}}_{opt}(T) \leqslant B^P_{opt}(T)$ by Lemma 3.1.

To prove part b), consider an optimal, integral realization $\tilde{x}'$ for the deadline $\lceil T^{\tilde{P}}_{opt}(B) \rceil$. By definition of $T^{\tilde{P}}_{opt}(B)$ we know that $c(\tilde{x}') \leqslant B$ and since $\tilde{x}'$ is integral we get

$$t(\tilde{x}) \leqslant t(\tilde{x}') \leqslant \lceil T^{\tilde{P}}_{opt}(B) \rceil \leqslant T^P_{opt}(B) \;\;,$$

where the last inequality follows from the integrality of $T^P_{opt}(B)$ and Lemma 3.1. On the other hand, the integrality of $\tilde{x}$ yields $t(\tilde{x}) \geqslant \lceil T^{\tilde{P}}_{opt}(B) \rceil$. In particular $\tilde{x}'$ is integral optimal for the budget B and can be computed by algorithm LTCT-SOLVER.

As a consequence of part a) and b) we get integral optimal realizations by computing optimal realizations for integral deadlines. By Lemma 2.2 and Lemma 3.2 b) this can be done in time $O\big(\ell\,|J|^3 \log|J|\big)$. □

We get the following interesting corollary:

Corollary 4.2. *Let $r > 0$ and P be an instance of the* DTCT PROBLEM *satisfying $k_j \in \{0, r\}$ for all $j \in J$. Then algorithm* LTCT-SOLVER *can be used to compute optimal realizations of P for all possible deadlines and budgets in $O\big(|J|^3 \log|J|\big)$ time.*

Proof. Since all feasible durations are either 0 or r only multiples of r can occur as project durations and it suffices to consider those deadlines which are multiples of r. We may without loss of generality assume that $r = 1$ because otherwise one can rescale all feasible durations by multiplication with the positive scalar $\frac{1}{r}$. This leads to an instance of the 1-DTCT PROBLEM. The result now follows from Lemma 4.1 and the observation that for an instance P of the 1-DTCT PROBLEM all integral realizations of its linear relaxation $\tilde{P}$ are feasible for P. □

When we are looking for provably good solutions x to the DEADLINE PROBLEM of a discrete project P, we can first compute an integral optimal realization $\tilde{x}$ of the linear relaxation $\tilde{P}$ for the given deadline T. This gives a lower bound $c^{\tilde{P}}(\tilde{x})$ on the value of an optimal solution $B^P_{opt}(T)$ by Lemma 4.1 a). Unfortunately, $\tilde{x}$ is not a feasible solution for the discrete project P in general. But hopefully it is not too far away from an optimal feasible realization of P for the deadline T. Thus a straightforward approach is to round $\tilde{x}$ to a feasible realization x of P.

We only need to consider those activities j which have been assigned a duration $\tilde{x}_j$ that is not feasible for the discrete project, i. e., $0 < \tilde{x}_j < k_j$. Of course we would like to round $\tilde{x}_j$ to the less expensive duration k_j, but unfortunately this could possibly increase the project duration and thus violate the deadline T. To avoid this, we better round these durations to the more expensive alternative $x_j := 0$ such that $t(x) \leqslant t(\tilde{x}) \leqslant T$ by Lemma 3.2 a). If P is an instance of the ℓ–DTCT PROBLEM, we get by definition of the linear relaxation and by integrality of $\tilde{x}_j$

$$c_j^{\tilde{P}}(\tilde{x}_j) = \frac{k_j - \tilde{x}_j}{k_j} \cdot c_j^P(0) \geqslant \frac{1}{\ell} \cdot c_j^P(0) \ . \tag{4.1}$$

This yields $c_j^P(0) \leqslant \ell c_j^{\tilde{P}}(\tilde{x}_j)$ and hence $c^P(x) \leqslant \ell c^{\tilde{P}}(\tilde{x}) \leqslant \ell B_{opt}^P(T)$ by Lemma 4.1 a). Note that we had to start with an integral optimal realization $\tilde{x}$, because otherwise we could not give any bound on $k_j - \tilde{x}_j$ in (4.1).

We have proved performance guarantee ℓ for the approximation algorithm that first computes an integral optimal solution to the linear relaxation and then rounds this solution to a feasible realization of P. We cannot get a better bound in this way since our analysis of the cost for the rounded solution is tight: consider a discrete project P where both duration and cost are dominated by only one activities j with $h_j = 0$ and $k_j = \ell$ together with the deadline $T := \ell - 1$. In this case equality holds in (4.1) and thus the bound is tight. In particular, our lower bound can be away from the value of an optimal solution by a factor of ℓ.

When we are looking for good solutions to the BUDGET PROBLEM, we can apply a similar idea. In a first step we compute an integral optimal solution $\tilde{x}$ of the linear relaxation for the given budget B. By Lemma 4.1 b) we get a lower bound $t(\tilde{x})$ on the duration $T_{opt}^P(B)$ of an optimal solution. In the rounding step we should now set the durations of activities j with $0 < \tilde{x}_j < k_j$ to the less expensive duration $x_j = k_j$, because rounding to 0 increases cost and we would possibly overspend the budget B. If P is an instance of the ℓ–DTCT PROBLEM we therefore get

$$x_j \leqslant k_j \tilde{x}_j \leqslant \ell \tilde{x}_j \ ,$$

since $\tilde{x}_j \geqslant 1$ by integrality of $\tilde{x}$. This yields $t(x) \leqslant \ell t(\tilde{x}) \leqslant \ell T_{opt}^P(B)$ by Lemma 3.2 a) and thus we have also developed an approximation algorithm with performance guarantee ℓ for the BUDGET PROBLEM.

Theorem 4.3. *For instances of the* ℓ-DTCT PROBLEM*, rounding the durations of an integral optimal realization to the linear relaxation uniformly in one direction as described above gives approximation algorithms with performance guarantee* ℓ *for the* BUDGET PROBLEM *and the* DEADLINE PROBLEM.

For the BUDGET PROBLEM we can even get better results. Because, unlike the situation for the DEADLINE PROBLEM, we can now repair the violation of the budget caused by rounding some durations to the shorter but more expensive alternative 0, if we save money by rounding durations of other activities to the less expensive alternative k_j. We will use this idea in the next section to get a better approximation result for the 2–DTCT PROBLEM.

5 The Approximability of the 2–DTCT PROBLEM

We consider the BUDGET PROBLEM for instances of the 2–DTCT PROBLEM. Hence we are given a discrete project P with $k_j \in \{0, 1, 2\}$ for each activity $j \in J$ and a budget $B \geqslant 0$. By construction of the linear relaxation $\tilde{P}$ the cost functions c_j^P and $c_j^{\tilde{P}}$ coincide for all feasible durations x_j in P. Therefore, to

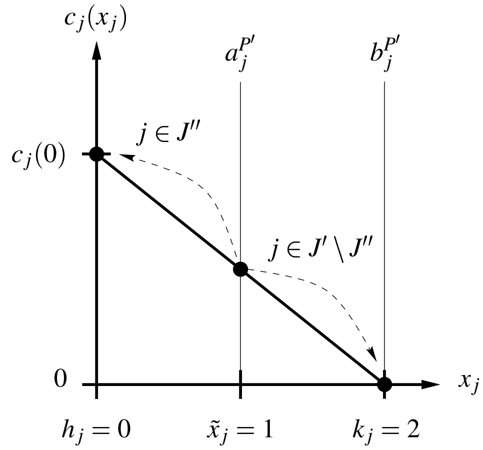


Figure 1: The cost function of an activity $j \in J'$.

simplify notation, we only use the symbols $c_j := c_j^{\tilde{P}}$ and $c := c^{\tilde{P}}$ throughout this section. For the same reason we denote both functions t^P and $t^{\tilde{P}}$ simply by t.

To compute a provably good, feasible realization of P we first use algorithm LTCT-SOLVER in order to find an integral optimal realization $\tilde{x}$ of the linear relaxation $\tilde{P}$ for the given budget B. Then an algorithm called REINVEST rounds this realization $\tilde{x}$ in a somewhat more intricate way according to the idea given at the end of the last section in order to get a good, feasible realization of P. First of all, it fixes the durations of all the activities j that are already feasible to $x_j := \tilde{x}_j \in \{h_j, k_j\}$. Now it remains to consider those activities j where $h_j = 0$, $k_j = 2$, and $\tilde{x}_j = 1$. We denote the subset of J consisting of all these activities by J'.

The feasible realization that we get by just setting $x_j := 2$ for all $j \in J'$ is a 2–approximation by Theorem 4.3. Notice that this realization does not use an amount of $B' := B - c(x) = B - c(\tilde{x}) + \sum_{j \in J'} c_j(1)$ of the budget B. The main idea of the algorithm is to reinvest this saved amount in order to round some of the durations back from 2 to 1 and then to the feasible duration 0. In other words, algorithm REINVEST rounds not all but only some of the jobs in J' from 1 to 2 such that at least $\frac{1}{2}B'$ of the whole budget is left. This amount suffices to round all the other jobs in J' from 1 to 0; the reason is that the amount saved by rounding a job from 1 to 2 exactly equals the cost that is needed to round it from 1 to 0, see Figure 1. A more precise argument is given in the proof of Theorem 5.1. We denote the subset of J' consisting of those activities whose duration is rounded to 0 by J''.

It remains to decide which activities should be rounded from 1 to 2 preserving an amount of $\frac{1}{2}B'$ with minimum increase in the project duration. This problem can be solved optimally: we construct another linear project P' where the durations of activities $j \in J'$ can be chosen out of the interval $[1,2]$, see Figure 1, and the durations and costs of all the other activities are fixed, i. e., $a_j^{P'} = 1$, $b_j^{P'} = 2$ for $j \in J'$ and $a_j^{P'} = b_j^{P'} = \tilde{x}_j$ for $j \in J \setminus J'$. Note that the realization $b^{P'}$ is the solution found by the 2–approximation algorithm of the last section. Therefore it is at most twice as long as an optimal realization for the budget B. Moreover, by definition of B' we get $c(b^{P'}) = B - B'$. We compute an integral optimal realization x' of P' for the budget $B - \frac{1}{2}B'$, i. e., we force the new realization x' to preserve an amount of $\frac{1}{2}B'$ with minimum increase in the project duration. The convexity of the function $T_{opt}^{P'}$ yields that the duration of x' is (up to integrality) at most the average of $t(\tilde{x})$ and $t(b^{P'})$ since its budget is the average of the cost of $b^{P'}$ and the budget B. Therefore it is within a factor $\frac{3}{2}$ of the optimum (again, up to integrality), see Figure 2.

Finally, as mentioned above, we turn x' into a feasible realization for P by setting the durations of the remaining activities $j \in J'' \subseteq J'$ from 1 to 0 using the remaining budget $\frac{1}{2}B'$. A formal description of

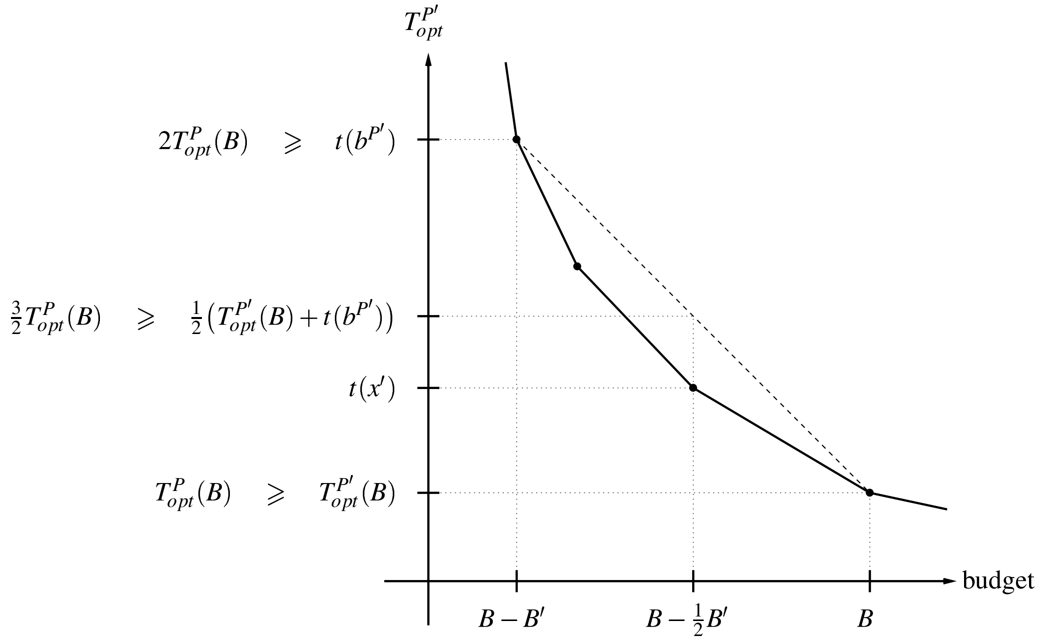


Figure 2: The convexity of $T_{opt}^{P'}$ yields $t(x') \leqslant \left\lceil \frac{3}{2}T_{opt}^{P}(B)\right\rceil$.

algorithm REINVEST is given in Figure 3.

Theorem 5.1. *Given an instance P of the* 2–DTCT PROBLEM *and a budget $B \geqslant 0$, algorithm* REINVEST *computes a feasible realization x such that $c(x) \leqslant B$ and $t(x) \leqslant \left\lceil \frac{3}{2}T_{opt}^{P}(B)\right\rceil$. The running time of the algorithm is $O\big(|J|^3 \log |J|\big)$.*

Proof. By construction of the linear project P' its cost functions $c_j^{P'}$ and $c^{P'}$ coincide with $c_j^{\tilde{P}}$ respectively $c^{\tilde{P}}$ for all feasible durations, see Figure 1. Therefore we also use the notation c_j, c, and t for the project P'.

First of all we show that the realization x of P does not violate the given budget B. Since $x_j = x'_j$ for all $j \in J \setminus J''$, we get

$$\begin{aligned} c(x) - c(x') &= \sum_{j \in J''} \big(c_j(0) - c_j(1)\big) && \text{by step 4,} \\ &= \sum_{j \in J''} \big(c_j(1) - c_j(2)\big) && \text{by linearity of } c_j\text{, see Figure 1,} \\ &= c(x') - c(b^{P'}) && \text{by definition of } b^{P'}. \end{aligned}$$

Since $c(x') \leqslant B - \frac{1}{2}B'$ and $c(b^{P'}) = B - B'$, we get

$$c(x) = 2c(x') - c(b^{P'}) \leqslant B \ .$$

Now we want to show that $t(x) \leqslant \left\lceil \frac{3}{2}T_{opt}^{P}(B)\right\rceil$. We know from the last section that $t(b^{P'}) \leqslant 2t(\tilde{x})$. Considering $\tilde{x}$ as a realization of P' yields $T_{opt}^{P'}(B) \leqslant t(\tilde{x})$ because $c(\tilde{x}) \leqslant B$. Since $t(\tilde{x}) \leqslant T_{opt}^{P}(B)$ by Lemma 4.1 b), we get

$$T_{opt}^{P'}(B) \leqslant T_{opt}^{P}(B) \qquad \text{and} \qquad t\big(b^{P'}\big) \leqslant 2T_{opt}^{P}(B) \ . \tag{5.1}$$

The remaining part of the proof is described in Figure 2. Since the budget for the integral optimal realization x' of P' is the average of the budget B and the cost $B - B'$ of the realization $b^{P'}$, the convexity

Input: instance P of the 2–DTCT PROBLEM, budget $B \geqslant 0$;
Output: feasible realization x of P.

1) compute an integral optimal realization $\tilde{x}$ of the linear relaxation $\tilde{P}$ of P for the budget B;

2) construct a new linear project P':

- let $(P', \prec) := (P, \prec) = (\tilde{P}, \prec)$;
- if $\tilde{x}_j \in \{h_j, k_j\}$ for $j \in J$ then set $a_j^{P'} := b_j^{P'} := \tilde{x}_j$ and $c_j^{P'}(a_j^{P'}) := c_j(\tilde{x}_j)$;
- if $\{h_j, k_j\} = \{0, 2\}$ and $\tilde{x}_j = 1$ for $j \in J$ (i.e., $j \in J'$), then set $a_j^{P'} := 1$, $b_j^{P'} := 2$, $c_j^{P'}(1) := c_j(1)$, and $c_j^{P'}(2) := c_j(2)$;

3) compute an integral optimal realization x' of P' for the budget $B - \frac{1}{2}B'$;

4) set $x_j := \begin{cases} x'_j & \text{if } x'_j \in \{h_j, k_j\}, \\ 0 & \text{if } x'_j = 1 \text{ and } j \in J', \end{cases}$ for all $j \in J$ and return x.

Figure 3: Algorithm REINVEST.

of the function $T_{opt}^{P'}$ yields that $t(x')$ is up to integrality at most the average of $T_{opt}^{P'}(B)$ and $t(b^{P'})$ which can be bounded by (5.1). Putting these results together, we get

$$\begin{aligned} t(x') &= \left\lceil T_{opt}^{P'}\left(\tfrac{1}{2}\left(B + c(b^{P'})\right)\right)\right\rceil && \text{by Lemma 4.1 b)},\\ &\leqslant \left\lceil \tfrac{1}{2}\left(T_{opt}^{P'}(B) + T_{opt}^{P'}\left(c(b^{P'})\right)\right)\right\rceil && \text{by Lemma 2.1},\\ &= \left\lceil \tfrac{1}{2}\left(T_{opt}^{P'}(B) + t(b^{P'})\right)\right\rceil && \text{by optimality of } b^{P'},\\ &\leqslant \left\lceil \tfrac{3}{2}T_{opt}^{P}(B)\right\rceil && \text{by (5.1).} \end{aligned}$$

Since $x_j \leqslant x'_j$ for all $j \in J$, the result follows by Lemma 3.2 a).

Steps 2 and 4 of algorithm REINVEST can be done in linear time. Its running time is therefore dominated by the two calls of algorithm LTCT-SOLVER in steps 1 and 3. Since $t(b^{\tilde{P}}) - t(a^{\tilde{P}}) \leqslant 2|J|$ and $t(b^{P'}) - t(a^{P'}) \leqslant |J|$ by Lemma 3.2 b), the overall running time is $O\left(|J|^3 \log |J|\right)$ by Lemma 2.2. ☐

In the next section we make use of the following slightly stronger result.

Corollary 5.2. *The duration $t(x)$ of the realization x computed by algorithm* REINVEST *can be bounded from above by* $\left\lceil \frac{3}{2} \lceil T_{opt}^{\tilde{P}}(B) \rceil \right\rceil$.

Proof. Lemma 4.1 b) yields $t(\tilde{x}) = \lceil T_{opt}^{\tilde{P}}(B) \rceil$. Thus we can replace $T_{opt}^{P}(B)$ by $\lceil T_{opt}^{\tilde{P}}(B) \rceil$ in (5.1). The result now follows using the same arguments as in the proof of Theorem 5.1. ☐

Finally, we want to show that there can be no better approximation algorithm for the BUDGET PROBLEM of the 2–DTCT PROBLEM, unless $P = NP$.

Theorem 5.3. *There is no polynomial-time algorithm computing a realization x for arbitrary instances P of the* 2–DTCT PROBLEM *and for arbitrary budgets $B \geqslant 0$, such that $c(x) \leqslant B$ and $t(x) < \frac{3}{2}T_{opt}^{P}(B)$, unless $P = NP$.*

Proof. De et al. (1997) show that the following decision problem is NP-complete. Given an instance of the 2–DTCT PROBLEM and a fixed budget, does there exist a realization x obeying the budget, such that $t(x) \leqslant 2$? If there was a polynomial-time approximation algorithm for the BUDGET PROBLEM with performance guarantee $\frac{3}{2} - \varepsilon$ for some $\varepsilon > 0$, it would find optimal realizations for all the instances P with $T^P_{opt}(B) \leqslant 2$ and could therefore solve the NP-complete decision problem. □

The statement of Theorem 5.3 may be of little relevance in some sense since we have only shown it to be tight for instances with optimal value 2. Thus, like for the edge-coloring problem, it could be the case that the BUDGET PROBLEM can be approximated within an additive constant of 1 for instances of the 2–DTCT PROBLEM. Unfortunately, there does not seem to be the possibility to carry over the non-approximability result directly to instances of the 2–DTCT PROBLEM with arbitrarily large optimal value. On the one hand, the problem lacks a straightforward scaling property. On the other hand, a simple serial concatenation of several copies of a project whose duration is NP-hard to approximate within a factor of $\frac{3}{2} - \varepsilon$ does not necessarily lead to a longer project with the same non-approximability property; the reason is that we cannot force a fixed distribution of the given budget among those copies.

However, the statement of Theorem 5.3 is certainly of relevance in the context of arbitrary discrete projects. There can always be a sub-project of a given discrete project which dominates both time and cost and is up to rescaling an instance of the 2–DTCT PROBLEM. Therefore, for arbitrary discrete projects the BUDGET PROBLEM cannot be approximated within a constant $\frac{3}{2} - \varepsilon$ for $\varepsilon > 0$ even if the optimal duration of such a project is large.

6 $O(\log \ell)$–Approximations for the BUDGET PROBLEM

In this section we consider instances P of the ℓ-DTCT PROBLEM and present approximation algorithms with performance guarantee $O(\log \ell)$ for the BUDGET PROBLEM. The algorithms even work for the more general class of discrete projects where the ratio of the maximum feasible duration of any activity to the minimum allowed non-zero duration of any activity is bounded by ℓ. Using the representation of discrete projects described in Section 3 and rescaling all the durations by the inverse of the minimum allowed non-zero duration of any activity as in the proof of Corollary 4.2, those instances can be described by discrete projects with at most two possible durations $h_j \leqslant k_j$ where $h_j \in \{0, k_j\}$ and $k_j \in \{0\} \cup [1, \ell]$ for each activity $j \in J$. To simplify notation we set $\lambda := \lfloor \log_2 \ell \rfloor$ in this section.

The main idea of the $O(\lambda)$–approximation algorithm, which is called PARTITION and is formally described in Figure 4, is to divide the project P into $\lambda + 1$ sub-projects. We first cover the interval $[1, \ell]$ with $\lambda + 1$ intervals $[2^i, 2^{i+1})$, $0 \leqslant i \leqslant \lambda$, of geometrically increasing size. Then the activities of P are partitioned according to their maximum duration k_j: for $0 \leqslant i \leqslant \lambda$ let $J_i := \{j \in J \mid 2^i \leqslant k_j < 2^{i+1}\}$. Each subset J_i of J induces a sub-project P_i of P which is given by $(J_i, \prec|_{J_i})$, where $\prec|_{J_i}$ denotes the restriction of the partial order $\prec$ of J to the subset J_i. Notice that we did not take activities j with $k_j = 0$ into account since they are dummy activities and thus part of the partial order.

All the sub-projects P_i of P have the nice property that the maximum durations k_j of activities $j \in J_i$ have up to a factor of 2 the same value 2^i. In the approximation algorithm we want to compute optimal realizations for all the sub-projects of P. Thus, in view of Corollary 4.2, we round those durations for activities of sub-projects P_i uniformly to 2^i, i. e., we set for $j \in J_i$

$$\begin{aligned} k_j^{P_i} &:= 2^i \ , & c_j^{P_i}(k_j^{P_i}) &:= c_j^P(k_j^P) = 0 \ , \\ h_j^{P_i} &:= \begin{cases} 0 & \text{if } h_j^P = 0, \\ 2^i & \text{otherwise,} \end{cases} & c_j^{P_i}(h_j^{P_i}) &:= c_j^P(h_j^P) \ . \end{aligned} \tag{6.1}$$

The main idea in algorithm PARTITION is to combine optimal realizations of the sub-projects to a provably good realization of project P. For every tuple $x^{P_0}, \ldots, x^{P_\lambda}$ of feasible realizations for the sub-

Input: discrete project P, budget $B \geqslant 0$;
Output: feasible realization x^P of P;
1) construct the sub-projects $P_0, \dots, P_\lambda$ as defined in (6.1);
2) for $0 \leqslant i \leqslant \lambda$ compute $B_{opt}^{P_i}$ and corresponding optimal realizations of P_i;
3) compute the minimum deadline T satisfying $\sum_{i=0}^{\lambda} B_{opt}^{P_i}(T) \leqslant B$;
4) combine optimal realizations of $P_0, \dots, P_\lambda$ for the deadline T to a realization x^P of P as defined in (6.2) and return x^P.

Figure 4: Algorithm PARTITION.

projects $P_0, \dots, P_\lambda$ we can construct a corresponding feasible realization x^P of P and vice versa: given realizations $x^{P_0}, \dots, x^{P_\lambda}$ we set for $0 \leqslant i \leqslant \lambda$ and $j \in J_i$

$$x_j^P := \begin{cases} h_j^P & \text{if } x_j^{P_i} = h_j^{P_i}, \\ k_j^P & \text{otherwise.} \end{cases} \tag{6.2}$$

This defines a bijection between realizations for P and tuples of realizations for $P_0, \dots, P_\lambda$.

In order to decide which optimal realizations of sub-projects should be combined to a realization of P, algorithm PARTITION computes the minimum deadline T such that all sub-projects can be finished at time T and the sum of the corresponding costs does not exceed the budget B, i. e., $\sum_{i=0}^{\lambda} B_{opt}^{P_i}(T) \leqslant B$. Note that such a deadline T always exists and $\max_i t^{P_i}(k^{P_i})$ is an upper bound on it. In the proof of Theorem 6.1 we will show that T is a lower bound on the optimal value $T_{opt}^P(B)$. Algorithm PARTITION computes optimal realizations of all the sub-projects for this deadline T and combines them to a realization of P as described in (6.2).

Theorem 6.1. *Algorithm* PARTITION *returns for every discrete project P and budget $B \geqslant 0$ a realization x of P satisfying $c(x) \leqslant B$ and $t(x) \leqslant 2(\lambda+1)T_{opt}^P(B)$, where $\lambda = \lfloor \log_2 \ell \rfloor$ and ℓ is the ratio of the maximum feasible duration of any activity to the minimum allowed non-zero duration of any activity. The algorithm can be implemented to run in strongly polynomial time.*

In the proof of Theorem 6.1 we use the fact that the deadline T in step 3 is a lower bound on $T_{opt}^P(B)$. If we combine the optimal realizations of sub-projects in step 4 we nearly have to double the durations of activities in the worst case because of the rounding of $k_j^{P_i}$ in (6.1). This contributes a factor of two to the performance guarantee of algorithm PARTITION. Moreover, in the worst case the durations of the realizations for the $\lambda+1$ sub-projects can add up to the duration of the final realization x^P. This yields another factor of $\lambda+1$.

In order to give the formal proof of Theorem 6.1 we need the following lemma:

Lemma 6.2. *If x^P and $x^{P_0}, \dots, x^{P_\lambda}$ are realizations of P and $P_0, \dots, P_\lambda$ as in* (6.2) *the following relations hold:*

a) $c^P(x^P) = \sum_{i=0}^{\lambda} c^{P_i}(x^{P_i})$,

b) $t^P(x^P) \leqslant 2 \sum_{i=0}^{\lambda} t^{P_i}(x^{P_i})$,

c) $t^{P_i}(x^{P_i}) \leqslant t^P(x^P)$ *for* $0 \leqslant i \leqslant \lambda$.

Proof. Part a) of the lemma follows from the definition of sub-projects in (6.1). In order to prove part b) let $I \subseteq J$ be the elements of a longest chain in the partial order of the set J with respect to x^P. Since for any activity $j \in J_i$ the duration x^P_j is at most twice as long as the duration $x^{P_i}_j$ by (6.1) and $I \cap J_i$ is a chain in the partial order of the set J^{P_i}, we get

$$t^P(x^P) = \sum_{j\in I} x^P_j = \sum_{i=0}^{\lambda} \sum_{j\in I\cap J_i} x^P_j \leqslant \sum_{i=0}^{\lambda} \sum_{j\in I\cap J_i} 2x^{P_i}_j \leqslant 2\sum_{i=0}^{\lambda} t^{P_i}(x^{P_i}) \ .$$

The last part of the lemma is a direct consequence of the fact that $x^{P_i}_j \leqslant x^P_j$ for all $j \in J_i$, $0 \leqslant i \leqslant \lambda$. $\square$

Proof of Theorem 6.1. Since the realization x^P is composed of realizations x^{P_i} with $t(x^{P_i}) \leqslant T$ for $0 \leqslant i \leqslant \lambda$, Lemma 6.2 b) yields $t(x^P) \leqslant 2(\lambda+1)T$. Moreover, by Lemma 6.2 a) and the choice of T in step 3 of the algorithm we get $c(x^P) \leqslant B$. Hence it remains to show that $T \leqslant T^P_{opt}(B)$. Let $\hat{x}^P$ be an optimal realization of P for the budget B. By Lemma 6.2 a) the corresponding realizations $\hat{x}^{P_0},\dots,\hat{x}^{P_\lambda}$ of $P_0,\dots,P_\lambda$ satisfy

$$\sum_{i=0}^{\lambda} c^{P_i}(\hat{x}^{P_i}) = c^P(\hat{x}^P) \leqslant B$$

and $t^{P_i}(\hat{x}^{P_i}) \leqslant t^P(\hat{x}^P) = T^P_{opt}(B)$ for $0 \leqslant i \leqslant \lambda$ by Lemma 6.2 c). Hence we get $T \leqslant T^P_{opt}(B)$.

Finally we want to show that algorithm PARTITION can be implemented to run in strongly polynomial time. In the algorithm we only have to consider those sub-projects P_i of P with $J_i \neq \emptyset$. Thus their number can be bounded by $|J|$. In particular we can implement all the loops and summations over the set of sub-projects to run in strongly polynomial time.

Steps 1 and 4 can obviously be done in linear time. By Corollary 4.2 we can use algorithm LTCT-SOLVER to perform step 2 in strongly polynomial time. It remains to show that the minimum deadline T in step 3 can be computed in strongly polynomial time. We replace step 3 by the following subroutine:

i) set $q := 0$;

ii) while $\sum_{i=0}^{\lambda} B^{P_i}_{opt}\big(t^{P_q}(k^{P_q})\big) > B$ do $q := q+1$;

iii) determine the smallest $T \in 2^q\mathbb{N}_0$ with $t^{P_q}(h^{P_q}) \leqslant T \leqslant t^{P_q}(k^{P_q})$ satisfying $\sum_{i=0}^{\lambda} B^{P_i}_{opt}(T) \leqslant B$;

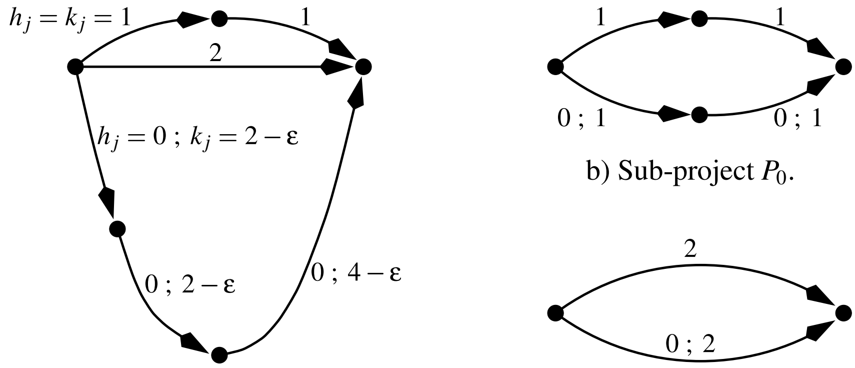
Note that there always exists a $0 \leqslant q \leqslant \lambda$ which terminates the while-loop in step ii: consider q with $t^{P_q}(k^{P_q}) = \max_i t^{P_i}(k^{P_i})$, then $\sum_{i=0}^{\lambda} B^{P_i}_{opt}\big(t^{P_q}(k^{P_q})\big) = 0 \leqslant B$. We have to make sure that the value T computed by the subroutine equals the minimum deadline $\hat{T}$ with $\sum_{i=0}^{\lambda} B^{P_i}_{opt}(\hat{T}) \leqslant B$. Thus we have to show $t^{P_q}(h^{P_q}) \leqslant \hat{T} \leqslant t^{P_q}(k^{P_q})$ and $\hat{T} \in 2^q\mathbb{N}_0$. By construction of q in step ii we know that $\hat{T} > t^{P_i}(k^{P_i})$ for $0 \leqslant i < q$ and $\hat{T} \leqslant t^{P_q}(k^{P_q})$. Moreover, since $t^{P_q}(h^{P_q})$ is the smallest deadline that can be reached with finite cost for the sub-project P_q, we get $\hat{T} \geqslant t^{P_q}(h^{P_q})$.

By contradiction we assume $\hat{T} \notin 2^q\mathbb{N}_0$. Since for $q \leqslant i \leqslant \lambda$ all feasible durations of activities in P_i are multiples of 2^q, the same holds for the breakpoints of the step function $B^{P_i}_{opt}$. This yields

$$B^{P_i}_{opt}\big(\hat{T}\big) = B^{P_i}_{opt}\big(\hat{T}-1\big) \tag{6.3}$$

for $q \leqslant i \leqslant \lambda$. Moreover, since $\hat{T} > t^{P_i}(k^{P_i})$ for $0 \leqslant i < q$, equation (6.3) also holds for these values of i. Hence $\sum_{i=0}^{\lambda} B^{P_i}_{opt}(\hat{T}-1) = \sum_{i=0}^{\lambda} B^{P_i}_{opt}(\hat{T}) \leqslant B$ in contradiction to the minimality of $\hat{T}$.

Since we only have to consider values for q with $J_q \neq \emptyset$, step ii can be implemented to run in strongly polynomial time. In step iii we can simply enumerate all possible values of T since $t(k^{P_q}) - t(h^{P_q}) \leqslant 2^q|J_q|$ by Lemma 3.2 b). $\square$



a) Project P.

b) Sub-project P_0.

c) Sub-project P_1.

Figure 5: A bad instance for Algorithm PARTITION with $\lambda = 1$.

The proven performance guarantee of algorithm PARTITION is tight: let $\lambda \in \mathbb{N}$ and $\ell := 2^{\lambda+1} - \varepsilon$ for some small $\varepsilon > 0$ and consider the project P consisting of $\lambda + 2$ parallel chains $0, \dots, \lambda + 1$ of serial activities. For $0 \leqslant i \leqslant \lambda$, the i^{th} chain contains $2^{\lambda-i}$ identical activities j_i with $h_{j_i} := k_{j_i} := 2^i$ and its length is thus fixed to 2^λ. The last chain is the serial concatenation of all the other chains, with the small but crucial difference that we set $h_{j_i} := 0$ and $k_{j_i} := 2^{i+1} - \varepsilon$ for all the activities of the i^{th} chain now. In Figure 5 a) we give an edge diagram of P for the case $\lambda = 1$. We choose the budget B such that we can afford to shorten the last chain to length 0, i. e., $B \geqslant c^P(h^P)$. Thus we get $T^P_{opt}(B) = t(h^P) = 2^\lambda$, see Figure 5 a).

But algorithm PARTITION performs very poorly on this class of instances. For $0 \leqslant i \leqslant \lambda$ sub-project P_i consists of two parallel chains: the i^{th} chain of P with fixed length 2^λ and a copy of it from the $(\lambda+1)^{\text{th}}$ chain of P with the only difference that $h_{j_i} = 0$ for all the activities, see Figure 5 b) and c). Therefore $t^{P_i}(h^{P_i}) = t^{P_i}(k^{P_i}) = 2^\lambda$ and the only optimal realization of P_i is k^{P_i}. Hence algorithm PARTITION returns the trivial realization k^P whose length is given by the length of the last chain, i. e., $t(k^P) = \sum_{i=0}^{\lambda} 2^{\lambda-i}(2^{i+1} - \varepsilon) = 2(\lambda + 1 - \varepsilon)2^\lambda + \varepsilon = 2(\lambda + 1 - \varepsilon)T^P_{opt}(B) + \varepsilon$.

Since the performance ratio of algorithm PARTITION mainly depends on the number of sub-projects that have to be considered, it can be significantly better if some of the sets J_i are empty. If for example $k_j \in \{0, 1, \ell\}$ for each activity $j \in J$, we only have to consider the projects P_0 and P_λ. Moreover, we do not need the rounding of durations in this situation and get performance guarantee 2. More generally we can state the following corollary:

Corollary 6.3. *For a given project P and budget B algorithm* PARTITION *returns a realization x of P with $c(x) \leqslant B$ and $t(x) \leqslant 2qT^P_{opt}(B)$, where q is the number of non-empty sets J_i.*

Algorithm PARTITION can even be slightly improved for general instances by combining the idea of partitioning P into sub-projects with the rounding technique of algorithm REINVEST. Remember that the factor $\lambda + 1$ in the performance guarantee of algorithm PARTITION equals the number of sub-projects that have to be combined in step 4. The main idea for the improved variant of the algorithm (see Figure 6) is to partition project P into half as many sub-projects as before in order to save a factor 2 in the performance guarantee. The new sub-projects are (up to rescaling) instances of the 2–DTCT PROBLEM and can therefore only be approximated within a factor $\frac{3}{2}$ of an optimal solution. These two effects together yield an improvement in the performance guarantee of Algorithm PARTITION by a factor of $\frac{3}{4}$.

For $0 \leqslant i \leqslant \left\lfloor \frac{\lambda}{2} \right\rfloor$ we combine sub-projects P_{2i} and P_{2i+1} to a new sub-project $\bar{P}_i$ (if λ is even, $P_{\lambda+1}$ is defined to be a trivial project with an empty set of activities $J_{\lambda+1} := \emptyset$). The new sub-project $\bar{P}_i$ consists

Input: discrete project P, budget $B \geqslant 0$;
Output: feasible realization x^P of P;

1') for $0 \leqslant i \leqslant \lfloor \frac{\lambda}{2} \rfloor$ construct the sub-project $\bar{P}_i$ together with P^i and $\tilde{P}^i$;

2') for $0 \leqslant i \leqslant \lfloor \frac{\lambda}{2} \rfloor$ compute $B^{\tilde{P}^i}_{opt}$ and corresponding integral optimal realizations of $\tilde{P}^i$;

3') compute the minimum deadline T with $\sum_{i=0}^{\lfloor \lambda/2 \rfloor} B^{\tilde{P}^i}_{opt}\big(\lfloor 2^{-2i}T \rfloor\big) \leqslant B$;

4') compute for each $0 \leqslant i \leqslant \lfloor \frac{\lambda}{2} \rfloor$ a realization x^{P^i} of P^i satisfying $c(x^{P^i}) \leqslant B^{\tilde{P}^i}_{opt}\big(\lfloor 2^{-2i}T \rfloor\big)$ and $t(x^{P^i}) \leqslant \lceil \frac{3}{2} \lfloor 2^{-2i}T \rfloor \rceil$;
for $0 \leqslant i \leqslant \lfloor \frac{\lambda}{2} \rfloor$ rescale x^{P^i} by a factor of 2^{2i} to get a realization $x^{\bar{P}_i}$ of $\bar{P}_i$ and combine these realizations to a realization x^P of P.

Figure 6: Improved variant of algorithm PARTITION.

of the set of activities $J_{2i} \cup J_{2i+1}$ together with the induced partial order. The durations of activities are again rounded as described in (6.1).

Thus, up to rescaling by a factor of 2^{-2i}, the sub-project $\bar{P}_i$ is an instance of the 2–DTCT PROBLEM. We denote the corresponding instance of the 2–DTCT PROBLEM by P^i and its linear relaxation by $\tilde{P}^i$. Since it is *NP*-hard to compute $B^{\bar{P}_i}_{opt}(T)$ and corresponding optimal realizations of $\bar{P}_i$, we use $B^{\tilde{P}^i}_{opt}\big(\lfloor 2^{-2i}T \rfloor\big)$ and integral optimal realizations of $\tilde{P}^i$ instead. By construction and Lemma 3.1 we get

$$B^{\tilde{P}^i}_{opt}\big(\lfloor 2^{-2i}T \rfloor\big) \leqslant B^{P^i}_{opt}\big(\lfloor 2^{-2i}T \rfloor\big) = B^{\bar{P}_i}_{opt}(T) \ .$$

Thus, the same argument as in the proof of Theorem 6.1 yields that the deadline T computed in step 3' is a lower bound on $T^P_{opt}(B)$. Moreover, using the rounding technique of algorithm REINVEST, we can construct realizations x^{P^i} of the projects P^i, $0 \leqslant i \leqslant \lfloor \frac{\lambda}{2} \rfloor$, with $c(x^{P^i}) \leqslant B^{\tilde{P}^i}_{opt}\big(\lfloor 2^{-2i}T \rfloor\big)$ and $t(x^{P^i}) \leqslant \lceil \frac{3}{2} \lfloor 2^{-2i}T \rfloor \rceil$ by Corollary 5.2. These realizations can finally be rescaled to realizations of the sub-projects $\bar{P}_i$ and combined to a realization x^P of P as described in (6.2).

Theorem 6.4. *The improved variant of algorithm* PARTITION *given in Figure 6 achieves performance guarantee* $\frac{3}{2}\lambda + 3$ *and can be implemented to run in strongly polynomial time.*

Since the proof of Theorem 6.4 is very similar to the proof of Theorem 6.1 we only highlight the main differences that lead to the improved performance guarantee. In particular, we do not give the analysis of the running time.

Sketch of proof. As mentioned above, the reason for the improved performance guarantee is that we only have to combine half as many realizations as before and can therefore save a factor 2. We get an additional factor $\frac{3}{2}$ since we cannot compute optimal realizations for the sub-projects but use algorithm REINVEST with performance guarantee essentially $\frac{3}{2}$ instead. Using the same arguments as in the proof of Theorem 6.1 we get $c(x^P) \leqslant B$ and

$$t^{\bar{P}_i}(x^{\bar{P}_i}) \leqslant \lceil \tfrac{3}{2} \lfloor 2^{-2i}T \rfloor \rceil 2^{2i} \leqslant \begin{cases} 0 & \text{if } T < 2^{2i}, \\ \frac{3}{2}T + 2^{2i-1} & \text{if } T \geqslant 2^{2i}, \end{cases}$$

for $0 \leqslant i \leqslant \lfloor \frac{\lambda}{2} \rfloor$. If λ is even we can get a better bound for the case $i = \lfloor \frac{\lambda}{2} \rfloor$ since in this case $J_{\lambda+1} = \emptyset$ and $\bar{P}_{\lfloor \lambda/2 \rfloor}$ is up to rescaling an instance of the 1-DTCT PROBLEM as in Corollary 4.2. Thus we can find an optimal realization $x^{\bar{P}_{\lfloor\lambda/2\rfloor}}$ of $\bar{P}_{\lfloor\lambda/2\rfloor}$ for the deadline T such that $t^{\bar{P}_{\lfloor\lambda/2\rfloor}}(x^{\bar{P}_{\lfloor\lambda/2\rfloor}}) \leqslant T$. Moreover, since all feasible durations of $\bar{P}_{\lfloor\lambda/2\rfloor}$ are multiples of 2^λ we get $t^{\bar{P}_{\lfloor\lambda/2\rfloor}}(x^{\bar{P}_{\lfloor\lambda/2\rfloor}}) = 0$ if $T < 2^\lambda$. This yields

$$\sum_{i=0}^{\lfloor\lambda/2\rfloor} t^{\bar{P}_i}(x^{\bar{P}_i}) \leqslant \begin{cases} \frac{3}{2}\left(\frac{\lambda-1}{2}+1\right)T + \sum_{i:\,T\geqslant 2^{2i}} 2^{2i-1} & \text{if } \lambda \text{ is odd,} \\ \left(\frac{3}{2}\frac{\lambda}{2}+1\right)T + \sum_{i=0}^{\lambda/2-1} 2^{2i-1} & \text{if } \lambda \text{ is even and } T \geqslant 2^\lambda, \\ \frac{3}{2}\frac{\lambda}{2}T + \sum_{i:\,T\geqslant 2^{2i}} 2^{2i-1} & \text{if } \lambda \text{ is even and } T < 2^\lambda. \end{cases}$$

All the right hand sides can be bounded from above by $\frac{1}{2}\left(\frac{3}{2}\lambda+3\right)T$. Thus, an appropriate adaption of Lemma 6.2 b) yields

$$t(x^P) \leqslant 2\sum_{i=0}^{\lfloor\lambda/2\rfloor} t^{\bar{P}_i}(x^{\bar{P}_i}) \leqslant \left(\tfrac{3}{2}\lambda+3\right)T \quad .$$

It remains to be shown that T is a lower bound on $T^P_{opt}(B)$ and that T can be computed in strongly polynomial time. This can be done using the same ideas that have already been described in the proof of Theorem 6.1. □

7 Bicriteria Results

From a practical point of view the results in Sections 4 and 6 are certainly of minor interest. On the one hand, the proven performance guarantee ℓ for the DEADLINE PROBLEM and $O(\log \ell)$ for the BUDGET PROBLEM are somewhat weak. On the other hand, in many situations it may not be realistic to assume a hard given deadline or budget. Thus, it could be a better idea to treat both time and cost as parameters having equal rights. This means that we allow some restricted tolerance in both directions. More precisely, we consider the following problem: we are given a discrete project P together with a deadline T (or budget B) which implicitly defines an optimal time-cost pair (T,B) where $B = B^P_{opt}(T)$ (respectively $T = T^P_{opt}(B)$). We are looking for a realization x of P such that $t(x) \leqslant \kappa T$ and $c(x) \leqslant \lambda B$ for given parameters $\kappa, \lambda \geqslant 1$ which define the allowed tolerance for the project duration and cost. Algorithms that compute such solutions in polynomial time are called *bicriteria approximation algorithms* or *pseudo approximation algorithms*.

The main idea for getting bicriteria approximation results for the DISCRETE TIME-COST TRADEOFF PROBLEM is again to start with an optimal solution to the linear relaxation and to round this solution to a feasible realization of the discrete project. Thus, in the following we will always assume that $\tilde{x}$ is an optimal realization of the linear relaxation for the given deadline or budget. As described in Section 2, $\tilde{x}$ can be computed in polynomial time.

In contrast to our considerations in Section 4, we may now round the duration of an activity in both directions depending on which feasible duration is closer in some sense. In our rounding procedure called BICRITERIA-ROUNDING(μ) we partition for each activity j the interval $[0,k_j]$ into two parts $[0,\mu k_j)$ and $[\mu k_j, k_j]$ depending on a parameter $0 < \mu < 1$. If the duration $\tilde{x}_j$ of activity j is within the interval $[0,\mu k_j)$, we round it to $x_j := 0$, otherwise to $x_j := k_j$. In the first case, the cost of the solution is increased by a factor less than $1/(1-\mu)$, in the second case, the duration of j is increased by $1/\mu$ in the worst case. Thus we get a realization x of the discrete project P with $c(x) < c(\tilde{x})/(1-\mu)$ and $t(x) \leqslant t(\tilde{x})/\mu$ by Lemma 3.2 a). Thus, as a consequence of Lemma 3.1 we get the following theorem.

Theorem 7.1. *For a given deadline T (or budget B) and a fixed parameter $0 < \mu < 1$ BICRITERIA-ROUNDING(μ) computes a realization x such that $c(x) < B/(1-\mu)$ and $t(x) \leqslant T/\mu$, where $B = B^P_{opt}(T)$ (respectively $T = T^P_{opt}(B)$).*

If we choose $\mu = 1/2$ for example, we get a realization which is at most twice as expensive and twice as long as an optimal realization for the given deadline or budget. Our analysis is tight: consider a project P consisting of two parallel activities 1 and 2, where $h_1 = h_2 = 0$, $k_1 = 1$, $k_2 = 1+\varepsilon$, $c_1(0) = \varepsilon$, and $c_2(0) = 1$ for some $\varepsilon > 0$. An optimal realization $\tilde{x}$ of the linear relaxation $\tilde{P}$ of P for the deadline μ is obviously given by $\tilde{x}_1 = \tilde{x}_2 = \mu$ such that $t(\tilde{x}) = \mu$ and $c(\tilde{x}) = (1-\mu)\varepsilon + 1 - \mu/(1+\varepsilon)$. BICRITERIA-ROUNDING($\mu$) leads to the realization x of P with $x_1 = 1$, $x_2 = 0$, $t(x) = 1$, and $c(x) = 1$. This yields $t(x)/t(\tilde{x}) = 1/\mu$ and $c(x)/c(\tilde{x}) \to 1/(1-\mu)$ when ε goes to 0.

We get another kind of bicriteria result if we allow our algorithm to use randomness. The motivation to consider randomized algorithms in this context is that for a given project P and a fixed realization $\tilde{x}$ not all possible choices of μ can lead to an increase in cost by a factor $1/(1-\mu)$ (notice that we have constructed different worst case examples for different values of μ in the last paragraph). By choosing μ randomly, we can avoid the worst case behavior of BICRITERIA-ROUNDING(μ) that can occur in the deterministic case and improve the *expected* performance of our algorithm:

Theorem 7.2. *If the parameter μ is drawn at random with uniform distribution from the interval $(\gamma, 1)$ for some $0 < \gamma < 1$,* BICRITERIA-ROUNDING(μ) *computes for a given deadline T (or budget B) a realization x such that the expected duration $E\big(t(x)\big)$ is bounded by $T \cdot \ln(1/\gamma)/(1-\gamma)$ and the expected cost $E\big(c(x)\big)$ is bounded by $B/(1-\gamma)$, where $B = B^P_{opt}(T)$ (respectively $T = T^P_{opt}(B)$).*

Proof. Since $E\big(c(x)\big) = \sum_{j\in J} E\big(c_j(x_j)\big)$ it suffices to show that $E\big(c_j(x_j)\big) \leqslant c_j(\tilde{x}_j)/(1-\gamma)$ for all $j \in J$. By construction of the algorithm, we have

$$E\big(c_j(x_j)\big) = \Pr(\mu > \tilde{x}_j/k_j) \cdot c_j(0) \leqslant \frac{1-\tilde{x}_j/k_j}{1-\gamma} \cdot c_j(0) = \frac{c_j(\tilde{x}_j)}{1-\gamma} \quad .$$

For each fixed choice of μ we get $t(x) \leqslant t(\tilde{x})/\mu$ by Lemma 3.2 a) and therefore

$$E\big(t(x)\big) \leqslant t(\tilde{x}) \int_\gamma^1 \frac{1/\mu}{1-\gamma}\, d\mu = \frac{t(\tilde{x})}{1-\gamma} \ln(1/\gamma) \quad .$$

□

Instead of the $(2,2)$–approximation by deterministically choosing $\mu = 1/2$ we can now randomly compute a realization where both duration and cost are expected to be within a factor of $e/(e-1) \approx 1.58$ of an optimal solution; this is done by setting $\gamma = 1/e$ and drawing μ randomly from $(\gamma, 1)$ as in Theorem 7.2. We should however mention that such a randomized bicriteria approximation result is in some sense weaker than a deterministic one. We cannot guarantee that there exists a feasible realization which simultaneously achieves both bounds. Consider the example with two parallel jobs given above. In this situation we either have to increase cost or time by a factor of 2.

Acknowledgments: I would like to thank Andreas S. Schulz for helpful comments, ideas, discussions, and many useful remarks on the content and presentation of this paper. Also many thanks to Rolf H. Möhring for helpful comments and ideas and the encouragement to work on this topic. I am also grateful to the anonymous referees for many useful comments that helped to improve the presentation of the paper.

References

De, P., Dunne, E. J., Ghosh, J. B., and Wells, C. E. (1995). The discrete time-cost tradeoff problem revisited. *Eur. J. Operational Research*, 81:225–238.

De, P., Dunne, E. J., Ghosh, J. B., and Wells, C. E. (1997). Complexity of the discrete time-cost tradeoff problem for project networks. *Operations Research*, 45:302–306.

Fulkerson, D. R. (1961). A network flow computation for project cost curves. *Management Science*, 7:167–178.

Goldberg, A. and Tarjan, R. E. (1988). A new approach to the maximum flow problem. *Journal of the ACM*, 35:921–940.

Harvey, R. T. and Patterson, J. H. (1979). An implicit enumeration algorithm for the time/cost tradeoff problem in project network analysis. *Foundations of Control Engineering*, 4:107–117.

Hindelang, T. J. and Muth, J. F. (1979). A dynamic programming algorithm for Decision CPM networks. *Operations Research*, 27:225–241.

Kelley, J. E. (1961). Critical path planning and scheduling: Mathematical basis. *Operations Research*, 9:296–320.

Kelley, J. E. and Walker, M. R. (1959). Critical path planning and scheduling: An introduction. Mauchly Associates, Inc., Ambler, Pa.

Krishnamoorthy, M. and Deo, N. (1979). Complexity of the minimum-dummy-activities problem in a PERT network. *Networks*, 9:189–194.

Möhring, R. H. and Radermacher, F. J. (1989). The order-theoretic approach to scheduling: The deterministic case. In Slowinski, R. and Weglarz, J., editors, *Advances in Project Scheduling*, pages 29–66. Elseviers Science Publ., Amsterdam.

Panagiotakopoulos, D. (1977). A CPM time-cost computational algorithm for arbitrary activity cost functions. *INFOR*, 15:183–195.

Philips, S. and Dessouky, M. I. (1977). Solving the project time/cost tradeoff problem using the minimal cut concept. *Management Science*, 24:393–400.

Robinson, D. R. (1975). A dynamic programming solution to cost-time tradeoff for CPM. *Management Science*, 22:158–166.

Reports from the group

“Algorithmic Discrete Mathematics”

of the Department of Mathematics, TU Berlin

566/1997 *Jens Gustedt:* Minimum Spanning Trees for Minor-Closed Graph Classes in Parallel

565/1997 *Andreas S. Schulz, David B. Shmoys, and David P. Williamson:* Approximation Algorithms

564/1997 *Uta Wille:* On Extending Closure Systems to Matroids

559/1997 *Matthias Müller–Hannemann and Karsten Weihe:* Improved Approximations for Minimum Cardinality Quadrangulations of Finite Element Meshes

554/1997 *Rolf H. Möhring and Matthias Müller–Hannemann:* Complexity and Modeling Aspects of Mesh Refinement into Quadrilaterals

551/1997 *Hans Bodlaender, Jens Gustedt and Jan Arne Telle:* Linear-Time Register Allocation for a Fixed Number of Registers and no Stack Variables

550/1997 *Karell Bertet, Jens Gustedt and Michel Morvan:* Weak-Order Extensions of an Order

549/1997 *Andreas S. Schulz and Martin Skutella:* Random-Based Scheduling: New Approximations and LP Lower Bounds

542/1996 *Stephan Hartmann:* On the NP–Completeness of Channel and Switchbox Routing Problems

536/1996 *Cynthia A. Phillips, Andreas S. Schulz, David B. Shmoys, Cliff Stein, and Joel Wein:* Improved Bounds on Relaxations of a Parallel Machine Scheduling Problem

535/1996 *Rainer Schrader, Andreas S. Schulz, and Georg Wambach:* Base Polytopes of Series-Parallel Posets: Linear Description and Optimization

533/1996 *Andreas S. Schulz and Martin Skutella:* Scheduling–LPs Bear Probabilities: Randomized Approximations for Min–Sum Criteria

530/1996 *Ulrich H. Kortenkamp, Jürgen Richter-Gebert, Aravamuthan Sarangarajan, and Günter M. Ziegler:* Extremal Properties of 0/1-Polytopes

524/1996 *Elias Dahlhaus, Jens Gustedt and Ross McConnell:* Efficient and Practical Modular Decomposition

523/1996 *Jens Gustedt and Christophe Fiorio:* Memory Management for Union-Find Algorithms

520/1996 *Rolf H. Möhring, Matthias Müller-Hannemann, and Karsten Weihe:* Mesh Refinement via Bidirected Flows: Modeling, Complexity, and Computational Results

519/1996 *Matthias Müller-Hannemann and Karsten Weihe:* Minimum Strictly Convex Quadrangulations of Convex Polygons

517/1996 *Rolf H. Möhring, Markus W. Schäffter, and Andreas S. Schulz:* Scheduling Jobs with Communication Delays: Using Infeasible Solutions for Approximation

516/1996 *Leslie A. Hall, Andreas S. Schulz, David B. Shmoys, and Joel Wein:* Scheduling to Minimize Average Completion Time: Off-line and On-line Approximation Algorithms

515/1996 *Christophe Fiorio and Jens Gustedt:* Volume Segmentation of 3-dimensional Images

514/1996 *Martin Skutella:* Approximation Algorithms for the Discrete Time-Cost Tradeoff Problem

509/1996 *Soumen Chakrabarti, Cynthia A. Phillips, Andreas S. Schulz, David B. Shmoys, Cliff Stein, and Joel Wein:* Improved Scheduling Algorithms for Minsum Criteria

508/1996 *Rudolf Müller and Andreas S. Schulz:* Transitive Packing

506/1996 *Rolf H. Möhring and Markus W. Schäffter:* A Simple Approximation Algorithm for Scheduling Forests with Unit Processing Times and Zero-One Communication Delays

505/1996 *Rolf H. Möhring and Dorothea Wagner:* Combinatorial Topics in VLSI Design: An Annotated Bibliography

504/1996 *Uta Wille:* The Role of Synthetic Geometry in Representational Measurement Theory

502/1996 *Nina Amenta and Günter M.Ziegler:* Deformed Products and Maximal Shadows of Polytopes

500/1996 *Stephan Hartmann and Markus W. Schäffter and Andreas S. Schulz:* Switchbox Routing in VLSI Design: Closing the Complexity Gap

498/1996 *Ewa Malesinska, Alessandro Panconesi:* On the Hardness of Allocating Frequencies for Hybrid Networks

496/1996 *Jörg Rambau:* Triangulations of Cyclic Polytopes and higher Bruhat Orders

Reports may be requested from: S. Marcus
Fachbereich Mathematik, MA 6–1
TU Berlin
Straße des 17. Juni 136
D-10623 Berlin – Germany

e-mail: Marcus@math.TU-Berlin.DE

Reports are available via anonymous ftp from: ftp.math.tu-berlin.de
cd pub/Preprints/combi
file Report–<number>–<year>.ps.Z