Earliest-Completion Scheduling of Contract Algorithms with End Guarantees

Spyros Angelopoulos and **Shendan Jin**
Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, LIP6, F-75252 Paris, France
{spyros.angelopoulos, shendan.jin}@lip6.fr

Abstract

We consider the setting in which executions of contract algorithms are scheduled in a processor so as to produce an interruptible system. Such algorithms offer a trade off between the quality of output and the available computation time, provided that the latter is known in advance. Previous work on this setting has provided strict performance guarantees for several variants of this setting, assuming that an interruption can occur arbitrarily ahead in the future. In practice, however, one expects that the schedule will reach a point beyond which further progress will only be marginal, hence it can be deemed complete. In this work we show how to optimize the time at which the system reaches a desired performance objective, while maintaining interruptible guarantees throughout the entire execution. The resulting schedule is provably optimal, and it guarantees that upon completion each individual contract algorithm has attained a predefined end guarantee.

1 Introduction

One of the central objectives in the design of intelligent systems is the provision for anytime capabilities. In particular, several applications such as medical diagnostic systems and motion planning algorithms require that the system outputs a reasonably efficient solution even if interrupted during its execution. Hence the following problem arises: can we transform a given algorithm to its interruptible equivalent, without compromising the algorithm's performance?

This question has motivated research on interruptible systems based on *contract* algorithms. A contract algorithm receives, as input parameter, its available computation time (i.e., the intended query time). If queried before this deadline, a contract algorithm may output a meaningless result; in other words, the query deadline must be strictly observed (hence the term "contract"). Thus, contract algorithms inherently lack interruptible capabilities; however, they often use simpler data structures, and thus tend to be easier to implement and maintain than interruptible algorithms [Bernstein *et al.*, 2003].

A general technique for obtaining interruptible systems by means of contract algorithms was first given in [Russell and Zilberstein, 1991], and is based on iteratively doubling the available execution times (i.e., the contract lengths) of the contract algorithm. This can be described as a *schedule* of executions of the same algorithm in which the i-th execution is run for a time equal to 2^i. In this schedule, if an interruption occurs at time t, it is guaranteed that a contract of length at least $t/4$ has completed its execution. Thus, there is a multiplicative gap of 4 between the interruption t and the largest contract length completed by time t, which is called the *acceleration ratio* [Russell and Zilberstein, 1991] and is a worst-case measure of the efficiency of the schedule. Furthermore, [Zilberstein *et al.*, 2003] showed that no other schedule can perform better according to this measure.

A generalization of the above setting was studied in [Zilberstein *et al.*, 2003] in which n different instances of an optimization problem are given, and each instance is associated with its own contract algorithm. The objective is to design a schedule of interleaved executions of the n contract algorithms, and the acceleration ratio relates the interruption time t to the contract length of the problem instance that has made the least progress by time t (see Section 2 for a formal definition). Thus, the acceleration ratio can be interpreted as the multiplicative increase in processor speed that is required, in worst-case, to render the schedule as efficient as a contract algorithm with a known, given deadline.

Contract scheduling has been studied in a variety of settings. [Zilberstein *et al.*, 2003] gave an optimal schedule for multiple problem instances and a single processor. Optimal schedules were also shown for a single problem instance in several parallel processors [Bernstein *et al.*, 2002], a result that was later generalized to the multi-processor, multi-instance setting [Bernstein *et al.*, 2003; López-Ortiz *et al.*, 2014]. [Angelopoulos *et al.*, 2008] studied contract scheduling in the presence of soft interruptions, and [Angelopoulos and López-Ortiz, 2009] introduced performance measures alternative to the acceleration ratio.

Contract scheduling can be seen as an application of a broader problem in which we seek an intelligent allocation of resources among tasks under uncertainty. Thus, the optimization aspects of this problem can be of interest to other, seemingly unrelated applications. For example, the basic setting of minimizing the acceleration ratio for a single problem instance is equivalent to the *online bidding problem* which is used in [Chrobak and Mathieu, 2006] as a framework of efficient algorithms based on iterative doubling for several combinatorial optimization problems. It is also identical to the problem of *flood search on the line* [Baryshnikov *et al.*, 2004]; thus for all these problems, the same optimal multiplicative guarantee equal to 4 is obtained. Moreover, contract scheduling has interesting parallels to the problem of minimizing the *competitive ratio* of a search strategy in a star-like environment, as shown first in [Bernstein *et al.*, 2003], and later in [Angelopoulos, 2015].

All previous work on contract scheduling based on the acceleration ratio has assumed that the schedule is unbounded since the interruption may occur arbitrarily far in the future. However, in practice, we expect that the execution of a schedule of contract algorithms will reach a point beyond which any progress over the problem instances will be only marginal. For instance [Boddy and Dean, 1994] show that functions of the form $Q(t) = 1 - e^{-\lambda t}$ can be used to model the expected performance of their anytime planner. This means that beyond a certain t_0, $Q(t)$ increases quite slowly (see also the discussion in [Zilberstein, 1996]). Similar sharp thresholds can be observed on the performance profiles of anytime algorithms that relate computation time and precision of output in a medical diagnostic system [Horvitz, 1988], as well as in anytime algorithms for planning and sensing [Zilberstein and Russell, 1993]. Therefore, one can argue that a schedule of contracts may be deemed *complete* once certain *end criteria* on the efficiency of completed contracts have been met, in terms of the computational time that has been allotted to the problem instances.

As another example, consider the class of Polynomial Time Approximation Scheme (PTAS) algorithms. The vast majority of such algorithms is based on Dynamic Programming, which means that they are not interruptible [López-Ortiz *et al.*, 2014]. These algorithms take as input a parameter $\epsilon > 0$ and output a solution within a factor of $(1 + \epsilon)$ of the optimal. Thus, for a given ϵ, one can find an upper bound of the required time (i.e., contract length) that is required to achieve the desired approximation. This implies that once the target approximation is determined, the designer can know exactly what is the required execution time so as to achieve it.

In our work we study contract scheduling in a setting that is motivated by the above applications. In particular, we seek a *finite* schedule with the following properties: i) it attains the optimal acceleration ratio, as in the standard model; and ii) it minimizes the time required to satisfy the end guarantees, among all schedules that obey property (i). In other words, if an interruption occurs during the execution of the schedule, the schedule has optimal performance; otherwise, if no interruption has occurred, it outputs a solution that meets the end guarantees the earliest possible, among all schedules optimal with respect to the acceleration ratio.

We adopt the following definition of end guarantees, which is in line with the worst-case nature of the acceleration ratio as well as the previous observation on the performance profiles of typical contract algorithms: we allow the schedule to complete once each problem instance has finished a contract of length at least a target value L. In particular, L can be defined in terms of the problem instance whose contract algorithm requires the largest deadline in order to achieve a satisfactory performance, among all instances.

1.1 Contribution

Our contribution is an optimal schedule for the problem described above, namely for earliest completion scheduling of contract algorithms with end guarantees. We propose a schedule that is theoretically optimal, and can be computed in time polynomial in the size of the end guarantee L (and thus in the size of the input, under the reasonable assumption that the number of problem instances n is constant, independent of L). In addition, we present computational results on its implementation which demonstrate that it achieves a considerable improvement over the known schedule that optimizes the acceleration ratio, but is oblivious of L.

The paper is structured as follows: We begin in Section 3 by showing that there exist optimal schedules for our problem that are *cyclic*, i.e., contracts are assigned to problems in round-robin fashion. This allows us to formulate our setting by means of a linear program (LP). Section 4 describes the main technical steps in the design and analysis of our schedule. More specifically, we prove that an optimal cyclic strategy saturates the constraints of the LP. In turn, this allows us to define an appropriate recurrence relation over the contract lengths, which yields the optimal schedule. Section 5 provides a computational evaluation of the schedule. Last, in Section 6 we show how to solve optimally a "dual" problem in which the interruptible system is given a *deadline*, and the objective is to maximize the worst-case performance among all problem instances while maintaining optimality according to the acceleration ratio. We illustrate an application of this setting in the context of *online bidding with budget*.

This work demonstrates that techniques based on linear programming, which have not been explored in previous work, can be very useful. Indeed, to our knowledge, all previous work is based on a variant of iterative doubling; in contrast, we use the structure of optimal LP solutions in order to obtain schedules that are described by more complex, yet still efficiently computable recurrence relations. A main technical obstacle one has to overcome is to derive the recurrence relations from the saturated LP constraints. We emphasize that we do not need to solve any linear program, and that it is used only as a guide in the design and analysis of the schedule. Due to space limitations some proofs are omitted or only sketched.

2 Preliminaries

We assume a single processor and n *problem instances* or simply *problems*, numbered $0, \ldots, n-1$. A schedule X of m contracts can be described as a sequence of the form $((x_i, p_i))_{i \in [1,m]}$, meaning that the i-th scheduled contract in X has length x_i and is assigned to problem $p_i \in \{0, \ldots, n-1\}$. For interruption time t, let $\ell_{p,t}(X)$ denote the length (duration) of the longest execution of a contract algorithm for problem p that has completed by time t in X. Then the *acceleration ratio of* X [Russell and Zilberstein, 1991] is defined as

$$\rho(X) = \sup_{t,p\in[0,\dots,n-1]} \frac{t}{\ell_{p,t}(X)}. \qquad (1)$$

We denote by $T_j(X)$ the completion time of the j-th contract in X and by $T(X)$ the completion time of X (i.e., of its last contract). It is not hard to see that the worst-case interruptions occur infinitesimally prior to the completion of a contract, hence the following useful formula for a schedule of m contracts:

$$\rho(X) = \sup_{j\in[n+1,\dots m], p\in[0,\dots,n-1]} \frac{T_j(X)}{\ell_{p,T_j^-(X)}(X)}, \qquad (2)$$

where $T_j^-(X)$ denotes a time right before $T_j(X)$. We also make the standard assumption that no interruption occurs unless each problem instance has completed a contract in the schedule, and assume, without loss of generality, that X does not schedule a contract of length l for problem p if it has already finished a contract for p of length at least l by that time.

A schedule is called *cyclic* if its i-th contract is assigned to problem $i \bmod n$, and *monotone* if $x_{i+1} \geq x_i$, for all i. The optimal acceleration ratio, denoted by ρ_n^*, can be attained by a cyclic (and monotone) strategy such that $x_i = b^i$, with $b = \frac{n+1}{n}$ [Zilberstein *et al.*, 2003], from which it follows that $\rho_n^* = n\left(\frac{n+1}{n}\right)^{n+1}$. We will denote by S_n^* the set of all schedules of optimal acceleration ratio ρ_n^*.

Given a schedule X, and an end guarantee $L \in \mathbb{R}^+$, we say that X is *feasible for L* if for each problem p there is a contract for p in X that has length at least L. A schedule X that is feasible for L is called *earliest for L* if for any other schedule X' feasible for L, $T(X) \leq T(X')$. Using this notation, we can state our problem as follows: Find schedule X^* (feasible for L) such that $X^* \in S_n^*$, and for every $X' \in S_n^*$, $T(X) \leq T(X')$. We call such a schedule *optimal*.

Example 1. *Consider the simple case $n = 1$, and $L = 30$. An example of a schedule feasible for L is a schedule X with contract lengths $1, 2, 4, 8, 16, 30$, which has completion time $T(X) = 61$. Note also that X is in S_1^*, since it has optimal acceleration ratio equal to 4.*

3 Cyclic Schedules and the LP Formulation

In this section we show that for a given end guarantee L, there is a cyclic schedule that is optimal for L. This will allow us to focus exclusively on this class of schedules, which we will later analyze by means of an LP. To this end, we first define a property that will be instrumental in the proof.

Definition 2. *A schedule $X = ((x_i, p_i))_{i=1}^m$ is called* normalized *if for each $i \in [1, m]$, $\ell_{p_i,T_{i-1}(X)}(X) \leq \ell_{q,T_{i-1}(X)}(X)$, for all $q \neq p_i$.*

Informally, a normalized schedule X assigns, at each time, a contract to the problem that has been worked the least among all problems up to that time. The following lemma shows that for every L there exists an optimal normalized schedule. Its proof expands the property that is already known, namely that there exists a normalized schedule that has optimal acceleration ratio [López-Ortiz *et al.*, 2014].

Lemma 3. *For every schedule X feasible for L, there exists a normalized schedule X' feasible for L such that $\rho(X') \leq \rho(X)$ and $T(X') \leq T(X)$.*

The following property follows easily from the definition of a normalized schedule.

Lemma 4. *Any optimal normalized schedule X is such that for every problem p, X schedules at most one contract for p of length at least L.*

Proof. By way of contradiction, suppose that there is a problem p for which X schedules two contracts of length at least L. Let C_1, C_2 denote these contracts, in the order they appear in the schedule. Since X is normalized, at the moment C_2 is about to start, every other problem in X has completed a contract of length at least the length of C_1, and thus at least L. Thus, at that moment, X has met the end guarantee. One could then obtain a schedule X' that is identical to X up to, but not including C_2. X' is also feasible for L; moreover, $\rho(X') \leq \rho(X)$. However, $T(X') < T(X)$, a contradiction. □

The next theorem is central in that it allows us to obtain an LP formulation from the problem. The theorem implies that optimal schedules can be found in the space of cyclic and monotone schedules.

Theorem 5. *Given the end guarantee L, there is a cyclic and monotone schedule that is optimal.*

Proof. Let $X = ((x_i, p_i))_{i=1}^m$ denote an optimal schedule given L. From Lemma 3 and 4, we can assume that X is normalized, and that for every problem p, it schedules at most one contract of length at least L. For each problem $p \in [0, n-1]$, define by C_p the contract of longest length for p in X, and let $C = \cup_p\{C_p\}$. We denote by $\overline{C} = X \setminus C$ the set of all remaining contracts in X. Note that $|C| = n$, and $|\overline{C}| = m-n$. Let the permutation $\pi : [1, m-n] \to [1, m-n]$ describe the contract lengths of $\overline{C}$ in increasing order, namely $x_{\pi(i)} \leq x_{\pi(i+1)}$ for all $i \in [1, m-n]$.

We define a cyclic schedule X' that is derived from X as follows. X' is produced by assigning the following m contract lengths to problem instances, in a cyclic manner:

$$X' = (x_{\pi(1)}, \dots, x_{\pi(m-n)}, L \dots, L).$$

In words, X' is by construction cyclic, in that we assign length $x_{\pi(i)}$ to problem $\pi(i) \bmod n$. The same rule is applied to to the last n contracts of length L. Note that X' is feasible for L, since it completes a contract of length L for each problem. Moreover, $T(X') \leq T(X)$, since each contract in X' can be mapped bijectively to a contract of no great length in X. By definition, $x_{\pi(i)} < L$, for all $i \in [1, m-n]$, since X schedules at most one contract of length at least L per problem. Hence, X' is also monotone.

It remains then to show that $\rho(X') \leq \rho(X)$. We partition the interruptions for X' into two classes: Those occurring right before the length $x_{\pi(j)}$ contracts, and those occurring right before the length L contracts. Let $T_{m-n} = \sum_{i=1}^{m-n} x_{\pi(i)}$. Then we can express the acceleration ratio of

X' as the maximum among these types of interruptions by applying (2), which gives

$$\rho(X') = \max\{\max_{\substack{j\in[n+1,\\ m-n]}}\{\frac{\sum_{i=1}^{j} x_{\pi(i)}}{x_{\pi(j-n)}}\},\ \max_{j\in[1,n]}\{\frac{T_{m-n}+jL}{x_{\pi(m+j-2n)}}\}\}. \tag{3}$$

Next, we will lower-bound $\rho(X)$. For $j \in [1, m-n]$ let p_j denote the problem to which $x_{\pi(j)}$ is assigned in X. Since $x_{\pi(j)} \in \overline{C}$, by definition there is another contract in X for problem p_j that is executed *after* $x_{\pi(j)}$ in X; let $\tilde{c}_j$ denote the first such contract. Since X is monotone, it follows that right at the moment $\tilde{c}_j$ is to be scheduled in X, all contracts $x_{\pi(i)}$, with $i \le j$ have already been executed, as well as n contracts that are at least as long as $x_{\pi(j)}$. We will consider two cases:

Case 1: $j \in [1, m-2n]$. In this case $\tilde{c}_j$ completes at time at least $\sum_{i=1}^{j+n} x_{\pi(i)}$ in X, and since the largest contract completed for p_j right before $\tilde{c}_j$ terminates is $x_{\pi(j)}$ it follows that

$$\rho(X) \ge \max_{j\in[1,m-2n]} \frac{\sum_{i=1}^{j+n} x_{\pi(i)}}{x_{\pi(j)}} = \max_{j\in[n+1,m-n]} \frac{\sum_{i=1}^{j} x_{\pi(i)}}{x_{\pi(j-n)}}. \tag{4}$$

Case 2: $j \in [m-2n+1, m-n]$. In this case, $\tilde{c}_j$ completes at time at least $T_{m-n} + (j-(m-2n))L$ in X, and since the largest contract completed for p_j right before $\tilde{c}_j$ terminates is $x_{\pi(j)}$, we have that

$$\begin{aligned}\rho(X) \quad &\ge \max_{j\in[m-2n+1,m-n]} \frac{T_{m-n}+(j-(m-2n))L}{x_{\pi(j)}} \\ &= \max_{j\in[1,n]} \frac{T_{m-n}+jL}{x_{\pi(m+j-2n)}}.\end{aligned} \tag{5}$$

Combining (3), (4), and (5), shows that $\rho(X') \le \rho(X)$, which completes the proof. □

Theorem 5 allows us to formulate our problem using an LP. More precisely, it suffices to show that there exists a cyclic schedule X^* of m^* contracts of the form $X^* = (x_1^*, \dots, x_{m^*}^*)$ whose contracts lengths are the optimal solution to the following LP, which we denote by P_m.

$$\begin{aligned} \min \quad & \sum_{i=1}^{m} x_i \\ \text{subject to} \quad & x_i \ge L, \quad i \in [m-n+1, m] && (F_i) \\ & \sum_{j=1}^{i} x_j \le \rho_n^* \cdot x_{i-n}, \quad i \in [n+1, m] && (C_i) \\ & x_i \le x_{i+1}, \quad i \in [1, m-1] && (M_i) \\ & x_1 \le \tau. && (I) \end{aligned}$$

Here, constraints (F_i) model the feasibility of X for L; constraints (C_i) imply that X has optimal acceleration ratio, using (2); and constraints (M_i) model the monotonicity of X. Last, we need an initialization constraint for x_1, namely (I), which states that x_1 has length at most a fixed number τ, which is a constant that does not depend on other parameters. This is a reasonable assumption, but is also required in order to exclude some unacceptable solutions. Otherwise, a cyclic schedule that starts with n contracts of length L would be feasible and optimal for the LP, but this is by no means an interruptible schedule.

4 Obtaining an Optimal Schedule

We will show how to obtain an optimal schedule given L. We will denote by $T^*(L)$ the completion time of an optimal schedule. Let $\mathcal{C}_m^*$ denote the class of all cyclic schedules with m contracts that have optimal acceleration ratio ρ_n^*. For simplicity we denote $T_i(X)$ by T_i when X is clear from context, with $T_0 = 0$.

We first give a road map of our approach. We begin by showing a lower bound on the lengths x_i of any schedule of optimal acceleration ratio (Lemma 7). This lower bound is expressed inductively in terms of T_i and two sequences a, b, which are defined appropriately in order to satisfy the inductive arguments. Next, we need to find the best value of m. To this end, we first show that if P_m is feasible, then the lower bounds on the x_i's hold with equality. This allows us to express the objective in terms of the parameters n, L and the sequences a, b (Lemma 8). Finally, to find the best value of m, we argue that it suffices to identify the smallest m for which P_m is feasible (Lemma 9).

We define the sequence a and b recursively as follows:

$$a_i = \begin{cases} 1, & i \in [0, n-1] \\ \frac{\sum_{j=0}^{n-1} a_{i-n+j} \prod_{k=0}^{j-1}(b_{i-n+k}+1)}{\rho_n^* - \prod_{k=0}^{n-1}(b_{i-n+k}+1)}, & i \ge n \end{cases} \tag{6}$$

and

$$b_i = \begin{cases} 0, & i \in [0, n-1] \\ \frac{\prod_{k=0}^{n-1}(b_{i-n+k}+1)}{\rho_n^* - \prod_{k=0}^{n-1}(b_{i-n+k}+1)}, & i \ge n. \end{cases} \tag{7}$$

Lemma 6. *For every $i \ge 0$, it holds that $a_i > 0$ and $b_i \in [0, \frac{1}{n}]$. In addition, $(b_i)_{i\ge 0}$ is monotone increasing with $\lim_{i\to+\infty} b_i = \frac{1}{n}$.*

The following lemma lower bounds the contract lengths of schedules in $\mathcal{C}_m^*$.

Lemma 7. *For any positive integers m, n with $m > n$ and for every schedule $X = (x_1, \dots, x_m)$, with $X \in \mathcal{C}_m^*$, it holds that $x_i \ge a_{m-i} \cdot x_{m-n+1} + b_{m-i} \cdot T_{i-1}$ for $i \in [1, m]$. In addition, $x_i = a_{m-i} \cdot x_{m-n+1} + b_{m-i} \cdot T_{i-1}$ if constraints (C_j) for $j \in [i, m]$ and (M_i) for $i \in [m-n+1, m-1]$ are tight.*

Proof. The proof is by induction on i, for $i \in [1, m]$. The base cases can be readily verified. For the inductive step, suppose that for $i \le m-n$ it holds that $x_j \ge a_{m-j}x_{m-n+1} + b_{m-j}T_{j-1}$ with $j \in [i+1, m]$. We will show that $x_i \ge a_{m-i}x_{m-n+1} + b_{m-i}T_{i-1}$.

$$\begin{aligned} \rho_n^* x_i \ge T_{i+n} &= x_{i+n} + T_{i+n-1} \\ &\ge a_{m-i-n} \cdot x_{m-n+1} + (b_{m-i-n}+1)T_{i+n-1} \\ &= a_{m-i-n} \cdot x_{m-n+1} \\ &+ (b_{m-i-n}+1)(x_{i+n-1} + T_{i+n-2}). \end{aligned}$$

Inductively, it follows that

$$\rho_n^* x_i \geq \left(\sum_{j=0}^{n-1} a_{m-n-i+j} \prod_{k=0}^{j-1} (b_{m-n-i+k} + 1) \right) x_{m-n+1} + \prod_{k=0}^{n-1} (b_{m-n-i+k} + 1) T_i,$$

which is equivalent to $x_i \geq a_{m-i} \cdot x_{m-n+1} + b_{m-i} T_{i-1}$. □

Lemma 7 allows us to find the optimal objective value of P_m, for a given m, assuming P_m has a feasible solution. This is shown in the next lemma.

Lemma 8. *Given $m \geq n$, assuming that P_m has a feasible solution, then the objective value of P_m is minimized if constraints (F_i) for $i \in [m-n+1, m]$, (M_i) for $i \in [m-n+1, m-1]$ and (C_i) for $i \in [n+1, m]$ are tight. Moreover, the minimum objective value is*

$$\left(n + \sum_{j=n}^{m-1} a_j \prod_{k=n}^{j-1} (b_k + 1) \right) L,$$

Proof. Given $m \geq n$, for any feasible solution $X = (x_1, \ldots, x_m)$ of P_m, by Lemma 7, we have

$$\begin{aligned} T_m &= \sum_{i=m-n+1}^{m} x_i + T_{m-n} \\ &\geq nL + T_{m-n} = nL + (x_{m-n} + T_{m-n-1}) \\ &\geq (n + a_n)L + (b_n + 1) T_{m-n-1} \\ &= (n + a_n)L + (b_n + 1)(x_{m-n-1} + T_{m-n-2}). \end{aligned}$$

It follows inductively that

$$T_m \geq \left(n + \sum_{j=n}^{m-1} a_j \prod_{k=n}^{j-1} (b_k + 1) \right) L.$$

We note that this lower bound is attained if constraints (F_i) for $i \in [m-n+1, m]$, (M_i) for $i \in [m-n+1, m-1]$ and (C_i) for $i \in [n+1, m]$ are tight. □

From Lemma 7 and 8, it follows that for a given m, under the assumption that P_m has a feasible solution, the optimal solution is derived by means of the recurrence relation

$$x_i = a_{m-i} x_{m-n+1} + b_{m-i} T_{i-1}, \text{ with } x_1 = a_{m-1} L.$$

The initial condition on x_1 comes from the constraint (C_{n+1}) in the LP P_m. Note that if $a_{m-1} L > \tau$, then P_m is not feasible, from Lemma 7.

Moreover, from the statement of x_i, we have that $T_i = \rho_n^* \cdot x_{i-n}$ for all $i \in [n+1, m]$, which implies that $x_i = \rho_n^* (x_{i-n} - x_{i-n-1})$, for all $i \in [n+2, m]$. Moreover, from Lemma 7, $x_i > 0$. Therefore, $x_{i-n} > x_{i-n-1}$ for $i \in [n+2, m]$. This in turn means that the solution defined above satisfies constraints (M_i) of P_m (since $x_i = L$ for $i \in [m-n+1, m]$).

Lemma 9. *The optimal objective value of P_m is monotone increasing in m. Thus, $T^*(L)$ is attained by the optimal objective value of P_M, where M is the smallest integer m such that P_m has a feasible solution.*

Proof. For $m \geq n$, by Lemma 8, the optimal objective value of P_m is $\alpha(m) \cdot L$, with $\alpha(m) = n + \sum_{j=n}^{m-1} a_j \prod_{k=n}^{j-1} (b_k + 1)$. By Lemma 6, it follows that $\alpha(m)$ is monotone increasing in m. This concludes the proof. □

It remains to find the smallest integer m such that P_m has feasible solutions; denote such m by m^*. To this end, we give an upper bound to m^*, as follows: we observe that the exponential cyclic strategy in which the i-th contract has length b^{i-1}, where $b = \frac{n+1}{n}$ has optimal acceleration ratio ρ_n^* (see the discussion in Section 2). Thus, there are at most $\lceil \frac{\log L}{\log b} + n \rceil = O(n \log L)$ candidate values for m^*, and the overall complexity of the algorithm is $O(n^2 \log^2 L)$. Algorithm 1 summarizes the steps needed to obtain the optimal schedule.

Algorithm 1: Earliest-completion scheduling of contract algorithms with end guarantee L

```
Input: $n \geq 1, L > 0$ and $\tau > 0$
$U \leftarrow \lceil \frac{\log L}{\log b} + n \rceil$
Compute $(a_i)_{i \in [1,U]}, (b_i)_{i \in [1,U]}$ using the recurrence
  relations (6) and (7)
for $m = n+1 \ldots U$ do
    if $a_{m-1} L \leq \tau$ then
        Output $x_i$, using the recurrence relation
          $x_i = a_{m-i} L + b_{m-i} T_{i-1}(X)$, and stop.
    end
end
```

5 Computational Evaluation

In this section we present computational results on the implementation of our schedule, whose completion time recall we denote by $T^*(L)$. In particular, we compare $T^*(L)$ to the completion time of the exponential cyclic schedule with base $b = \frac{n+1}{n}$, whose completion time we denote by $T_{\texttt{exp}}(L)$. Recall that the latter is the known strategy with optimal acceleration ratio, which, however is oblivious of L. We choose τ to be equal to 1, and L to be integral in the range $[1, 10^6]$.

Figure 1 illustrates the completion times of the two schedules for $n = 5$. We observe that $T^*(L)$ is almost linear, unlike $T_{\texttt{exp}}(L)$ which is a step function. Similar almost-linear shapes were observed for $T^*(L)$ for the values of n for which we evaluated our schedule, with slopes increasing in n.

Figure 2 illustrates the ratio $T_{\texttt{exp}}(L)/T^*(L)$ for $n \in \{1, 2, 20\}$. We observe that the fluctuations of the ratio, as function of L, tend to decrease in n.

Figure 2 motivates the evaluation of the ratio $T_{\texttt{exp}}(L)/T^*(L)$, for large L. This is shown in Figure 3. The experiments suggest that a constant multiplicative gain is

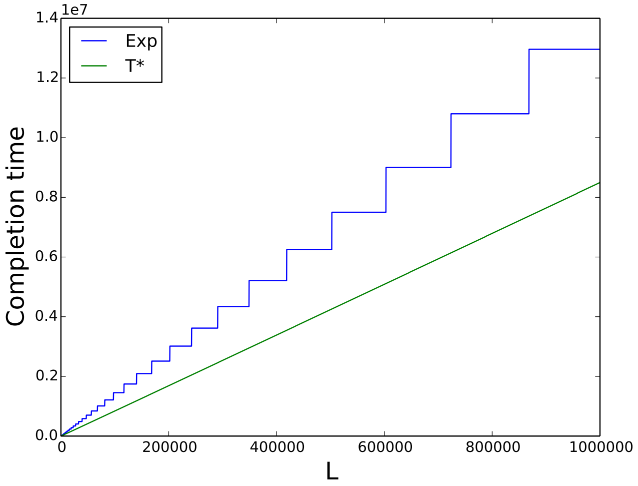

Figure 1: Completion times $T^*(L)$ and $T_{\texttt{exp}}(L)$ as function of L for $n = 5$.

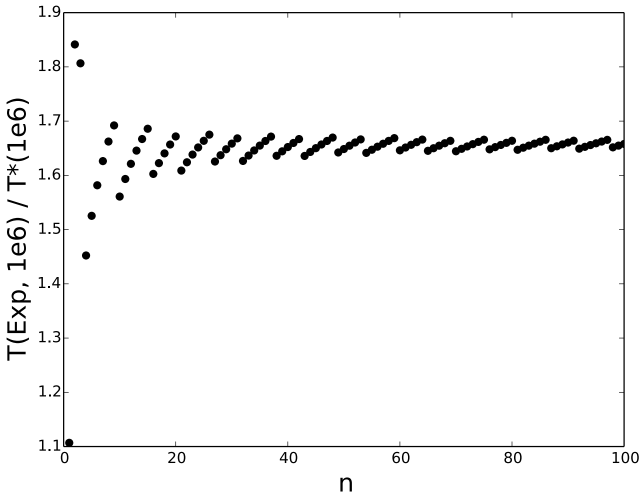

Figure 3: Ratio $T_{\texttt{exp}}(L)/T^*(L)$, for $L = 10^6$.

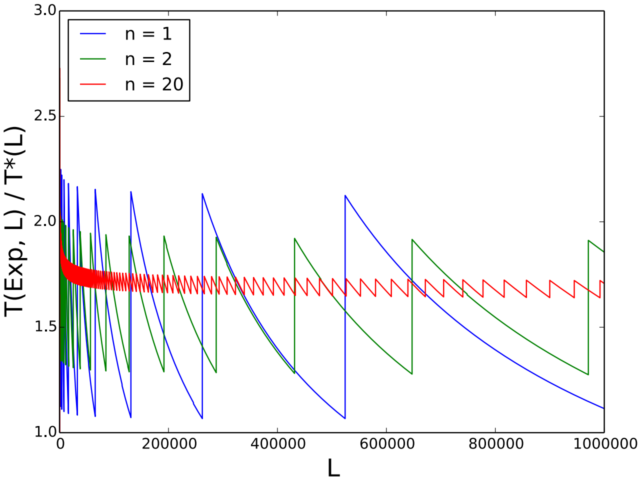

Figure 2: Ratio $T_{\texttt{exp}}(L)/T^*(L)$ as function of L.

achieved by the optimal schedule, and for $L = 10^6$ the ratio tends to approximately 1.65, for large n.

We conclude this section with an observation on the run time of the implementation. In our computational evaluation, we observed that the values (a_i) described by (6) appear to be monotone decreasing in i, although this is hard to prove analytically. If this indeed holds, then one can show the following fact concerning the LP: If there exists m such that P_m has a feasible solution, then so does P_{m+1}. This implies a heuristic in which instead of $O(n \log L)$ candidate values for m, one needs to check only $O(\log n + \log\log L)$ values, using binary search, which improves the complexity to $O(n \log(\log n + \log\log L))$.

6 Extensions and Further Applications

Consider the following problem, which can be seen as the "dual" of earliest-completion scheduling with end guarantees. Suppose we are given a *deadline* D, and we would like to obtain a schedule X of contract algorithms for n problems with the following properties: i) $T(X) \leq D$, namely the schedule respects the deadline; ii) $\rho(X) = \rho^*_n$; and iii) X maximizes the parameter $\min_p \ell(p, D)$, among all problems p. In words, we seek an interruptible system which has optimal interruptible behavior up to the deadline, and which maximizes the progress that has been made by the deadline among all problem instances. This formulation can be useful in the context of medical diagnostic systems, in which the deadline models the absolute time by which a diagnosis needs to be procured. To solve this dual problem, one can apply Algorithm 1 in combination with binary search over the space of end guarantees, namely with $O(\log D)$ applications of our algorithm.

For the special case $n = 1$, this problem has an equivalent statement, which we call the *online bidding problem with budget* D. In the online bidding problem [Chrobak and Mathieu, 2006], there is some hidden value u, and a player submits a sequence (x_i) of *bids* until a bid is greater than or equal to u. The player pays the sum of the bids, and the efficiency of the sequence is measured in terms of the *competitive ratio*, which is the worst-case ratio of the cost paid by the player over the value u. It is known that a simple doubling bidding strategy is competitively optimal, but the setting does not take into account the budget of the bidder. In contrast, our solution provides a competitively optimal bidding strategy with overall cost that does not exceed the bidder's budget D.

7 Conclusion

We studied contract scheduling in a model in which the interruptible system is deemed complete once certain performance guarantee has been reached on all problem instances. In future work we would like to consider more general models of end guarantees. For instance, each problem instance p may be associated with its own end guarantee L_p. We also believe that our approach can be applicable to other budgeted optimization problems. More precisely, we would like to study the setting in which the searcher has a budget in terms of the total area it can cover.

References

[Angelopoulos and López-Ortiz, 2009] S. Angelopoulos and A. López-Ortiz. Interruptible algorithms for multi-problem solving. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 380–386, 2009.

[Angelopoulos *et al.*, 2008] S. Angelopoulos, A. López-Ortiz, and A. Hamel. Optimal scheduling of contract algorithms with soft deadlines. In *Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI)*, pages 868–873, 2008.

[Angelopoulos, 2015] S. Angelopoulos. Further connections between contract-scheduling and ray-searching problems. In *Proceedings of the 24th International Joint Conference in Artificial Intelligence (IJCAI)*, pages 1516–1522, 2015.

[Baryshnikov *et al.*, 2004] YM M. Baryshnikov, E. G. Coffman Jr., P. R. Jelenkovic, Petar Momcilovic, and D. Rubenstein. Flood search under the california split rule. *Oper. Res. Lett.*, 32(3):199–206, 2004.

[Bernstein *et al.*, 2002] D.S. Bernstein, T. J. Perkins, S. Zilberstein, and L. Finkelstein. Scheduling contract algorithms on multiple processors. In *Proceedings of the 18th AAAI Conference on Artificial Intelligence (AAAI)*, pages 702–706, 2002.

[Bernstein *et al.*, 2003] D.S. Bernstein, L. Finkelstein, and S. Zilberstein. Contract algorithms and robots on rays: unifying two scheduling problems. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1211–1217, 2003.

[Boddy and Dean, 1994] M. S. Boddy and T. L. Dean. Deliberation scheduling for problem solving in time-constrained environments. *Artif. Intell.*, 67(2):245–285, 1994.

[Chrobak and Mathieu, 2006] M. Chrobak and C. Mathieu. Competitiveness via doubling. *SIGACT News*, pages 115–126, 2006.

[Horvitz, 1988] E. Horvitz. Reasoning about beliefs and actions under computational resource constraints. *Int. J. Approx. Reasoning*, 2(3):337–338, 1988.

[López-Ortiz *et al.*, 2014] A. López-Ortiz, S. Angelopoulos, and A.M. Hamel. Optimal scheduling of contract algorithms for anytime problems. *Journal of Artificial Intelligence Research*, (51):533–554, 2014.

[Russell and Zilberstein, 1991] S. J. Russell and S. Zilberstein. Composing real-time systems. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 212–217, 1991.

[Zilberstein and Russell, 1993] S. Zilberstein and S. J. Russell. Anytime sensing, planning and action: A practical model for robot control. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1402–1407, 1993.

[Zilberstein *et al.*, 2003] S. Zilberstein, F. Charpillet, and P. Chassaing. Optimal sequencing of contract algorithms. *Ann. Math. Artif. Intell.*, 39(1-2):1–18, 2003.

[Zilberstein, 1996] S. Zilberstein. Using anytime algorithms in intelligent systems. *AI Magazine*, 17(3):73–83, 1996.