Optimal Scheduling of Contract Algorithms with Soft Deadlines

Spyros Angelopoulos
Max-Planck-Institut für Informatik
Campus E1 4
Saarbrücken 66123, Germany
sangelop@mpi-inf.mpg.de

Alejandro López-Ortiz
David R. Cheriton School
of Computer Science
University of Waterloo
Waterloo, Ontario, Canada, N2L 3G1
alopez-o@uwaterloo.ca

Angèle M. Hamel
Dept. of Physics and Computer Science
Wilfrid Laurier University
Waterloo, Ontario, Canada, N2L 3C5
ahamel@wlu.ca

Abstract

A contract algorithm is an algorithm which is given, as part of its input, a specified amount of allowable computation time. In contrast, interruptible algorithms may be interrupted throughout their execution, at which point they must report their current solution. Simulating interruptible algorithms by means of schedules of executions of contract algorithms in parallel processors is a well-studied problem with significant applications in AI.

In the classical case, the interruptions are hard deadlines in which a solution must be reported immediately at the time the interruption occurs. In this paper we study the more general setting of scheduling contract algorithms at the presence of soft deadlines. This is motivated by the observation of practitioners that soft deadlines are as common an occurrence as hard deadlines, if not more common. In our setting, at the time t of interruption the algorithm is given an additional window of time $w(t) \leq c \cdot t$ to continue the contract or, indeed, start a new contract (for some fixed constant c). We explore this variation using the acceleration ratio, which is the canonical measure of performance for these schedules, and derive schedules of optimal acceleration ratio for all functions w.

Introduction

Anytime algorithms, which were introduced and developed by Horvitz (Horvitz 1987) (Horvitz 1998) and Dean and Boddy (Dean and Boddy 1998), arise in situations where the available computation time is uncertain. Such algorithms will produce solutions of varying quality, depending on how much computation time they are allowed.

One can distinguish between two main types of anytime algorithms. *Interruptible algorithms* are algorithms whose allowable running time is not known a priori, and can be interrupted (queried) at any point during their execution. *Contract algorithms*, on the other hand, are algorithms whose allotted execution time is known beforehand as part of the algorithm's input. Such algorithms are thus less flexible than interruptible algorithms, however they tend to be simpler to implement.

Hence the following problem arises: We are given a set of n *problem instances* and a machine consisting of m identical *parallel processors*, and we seek an efficient strategy for scheduling multiple executions of n contract algorithms (one for each problem) on the processors. At any point of time an interruption may occur, during which a solution may be requested for any of the n problems. The performance of the algorithm at interruption time is related to the length (duration) of the longest *completed* execution of a contract algorithm for the requested problem: Specifically, we would like this length to be as large as possible, under the intuitive assumption that the longer the available computation time, the better the solution will be. This problem has been extensively studied in the literature, and we refer to it as the (classical) *hard deadline* problem.

However, in practice, it has been observed that often the time of interruption does not define an absolute hard deadline at which time a solution has to be produced. Instead, the specifications of the application may be such that a "grace period" may be desirable, especially if there are concerns about the quality of the solution. Consider the example of a medical diagnostic system: the doctor may query the system for a diagnosis, but if the diagnosis is not satisfactory, the doctor may allow the system some additional time, at the end of which she may reconsult. The amount of additional time should depend on the application: a hospital intensive care unit system should allow smaller windows of additional computation time than a system run by a consultation diagnostic center. To this goal, Zilberstein *et al.* (Zilberstein, Charpillet, and Chassaing 2003) consider the case in which there is a time dependent utility function which indicates the decrease in the value of the solution as time elapses. In our work, we consider the case in which the interruption marks a point in time from which a solution must be produced shortly. More precisely, at interruption time t, the interruptible algorithm is given a *window* $w(t)$ of additional time, and within which a solution must be reported for the queried problem. There is no penalty for using this time; on the other hand, the algorithm has to report a solution by time $t + w(t)$. This is similar to the work of Manolache et al. in which they consider the expected completion time in a setting where the deadlines are soft, all problems must be solved and there is a maximum global deadline which cannot be exceeded (Manolache, Eles, and Peng 2004).

We focus on window functions which are upper-bounded by linear functions of time: namely, $w(t) \leq c \cdot t$ for some constant c. This is motivated by the observation that it would

be unrealistic to allow larger windows, since otherwise the algorithm could hardly qualify as "interruptible". We also assume that $w(t)$ is a continuous, increasing function of time, and that the function is known to the algorithm in advance. Note that if $w(t)$ is not known in advance, but rather revealed at the time of interruption, in the worst case the problem degenerates to the problem of hard deadlines, if the window is too small.

For the classical problem, the standard measure of the efficiency of an interruptible algorithm is the *acceleration ratio*. Informally, the acceleration ratio measures the increase in processor speed required for the simulation to produce a solution of the same quality as the optimal solution. Here, by optimal solution we refer to an algorithm which knows beforehand both the exact interruption time and the queried problem; such an optimal, ideal algorithm can dedicate a single processor and run a single contract of length equal to the interruption time for the problem in question.

Simulating interruptible algorithms using contract algorithms has been studied under several variants. Examples include the case of one problem and one processor, in work by Russell and Zilberstein (Russell and Zilberstein 1991) and Zilberstein *et al.* (Zilberstein, Charpillet, and Chassaing 1999), many problems and one processor by Zilberstein *et al.* (Zilberstein, Charpillet, and Chassaing 1999), and one problem and many processors by Bernstein *et al.* (Bernstein et al. 2002). For all of these problems, schedules of optimal acceleration ratio were derived. For the most general case, namely solving many problems using many processors, Bernstein *et al.*(Bernstein, Finkelstein, and Zilberstein 2003) showed an optimal simulation under the restrictive, but natural assumption that the schedule has a cyclic format. In subsequent work, López-Ortiz *et al.* (Lopez-Ortiz, Angelopoulos, and Hamel 2006) removed the assumption of cyclicality, and showed that the schedule of (Bernstein, Finkelstein, and Zilberstein 2003) is optimal among all possible schedules. The acceleration ratio of the optimal schedule is a function of n and m, and is equal to $(1+\frac{n}{m})(1+\frac{m}{n})^{\frac{n}{m}}$.

The main contribution of this paper is a schedule of contract algorithms which guarantees optimal acceleration ratio under no restrictive assumptions (c.f. Corollary 2 and Theorem 4). To simplify exposition and analysis, in what follows we concentrate on the the case of linear windows, namely $w(t) = \epsilon \cdot t$ for some constant ϵ. It turns out that this is the most challenging case: a very similar analysis can be applied to a much wider class of window functions, namely for all increasing functions w such that $w(t) \leq c \cdot t$, for some constant c: in this more general case, ϵ is defined as $\lim_{t\to\infty}(w(t)/t)$ (which always exists), and the same bounds apply w.r.t. ϵ as in the case of a linear function.

Due to space limitation, certain technical proofs are either omitted or only sketched in this paper.

Preliminaries

We follow the notation of Bernstein *et al.* (Bernstein, Finkelstein, and Zilberstein 2003) and Lopez-Ortiz *et al.* (Lopez-Ortiz, Angelopoulos, and Hamel 2006). Let P denote a set of n problems. A *contract* c is a pair (p, L), where $p \in P$ is the problem c is working on and L is the length of time (duration) the contract is designated to work on p. Let C denote a potentially infinite set of contracts. Then a *schedule* X of contracts is a feasible assignment of contracts to processors. More formally, $X = \{(c_i, m_i, x_i) : c_i \in C\}$, where m_i is the processor to which c_i is scheduled and x_i is the length of contract c_i. For a contract c_i we denote by T_{c_i}, G_{c_i} and L_{c_i} its *start time*, its *completion or finish time* and its *length or duration*, respectively. Since idle time in the schedule leads to suboptimal solutions, it will always be the case that $L_{c_i} = G_{c_i} - T_{c_i}$. Define by $\ell_{p,t}$ the length of the longest contract for problem p that has been completed by time t in X, and by $c_{p,t}$ the contract of length $\ell_{p,t}$ (we can assume without loss of generality that no two contracts in the schedule are of the same length). We use the notation t^+ (resp. t^-) to denote time infinitesimally bigger (resp. smaller) than t.

Given a time t of interruption and a problem p that an answer is required for, we define an *interruption* as the pair $i = \langle t, p\rangle$. As it is common in the field we assume that interruptions occur only after one contract for each problem has been completed.

Bernstein *et al.* (Bernstein, Finkelstein, and Zilberstein 2003) define the class of *cyclic* schedules as schedules which have the following natural properties: Let P_{c_i} be the problem instance worked on by contract c_i.

1. Property 1 (Problem Round Robin) If c_i is the ith contract, the problem instance P_{c_i} is such that $P_{c_i} = i \bmod n$.
2. Property 2 (Length Increasing) For all c_i, c_j if $P_{c_i} = P_{c_j}$ and $i < j$, then $L_i < L_j$.
3. Property 3 (Processor Round Robin) $m_i = i \bmod m$ for all i.

An *exponential* schedule is a cyclic schedule in which the lengths of contracts in the round-robin order increase exponentially. More formally, the i-th contract in the order has length b^i for some fixed number b.

Definition 1 *Given a set P of n problem instances and a set M of m processors of identical speed, the acceleration ratio of a schedule X for P, denoted by $\alpha(X)$ (or simply α when X is implied from context) is defined as the maximum ratio (over all problems $p \in P$ and interruption times t) of the ratio of the interruption time over the longest contract length completed by the interruption. Formally: $\alpha(X) = \max_{p,t} \frac{t}{\ell_{p,t}}$. Then the acceleration ratio for P and a set M of processors of identical speed is defined as $\alpha^* = \inf_X \alpha(X)$. A schedule X is* optimal *if $\alpha(X) = \alpha^*$.*

Lemma 1 (Bernstein *et al.* 2003) *For all cyclic schedules X, $\alpha(X) = \sup_k \frac{G_{k+n}}{L_k}$, where k denotes a contract index in the cyclic schedule, and G_{k+n} denotes the completion time of contract c_{k+n}.*

Since exponential schedules are also cyclic, the above holds also for exponential schedules.

Acceleration ratio for the soft-deadline problem

In this section we propose expressions for the acceleration ratio for the soft deadline problem and we prove a first lower-bound on the quality of any schedule of contract algorithms with respect to these measures.

First of all, it is easy to observe that the acceleration ratio α as defined for the classical case (see Definition 1) is not appropriate for the version we study. This is due to the fact that α does not take into account the additional window of time, nor the progress the interruptible algorithm could make within the additional window (for instance $\ell_{p,(1+\epsilon)t}$ could be much bigger than $\ell_{p,t}$). We thus need to adapt the measure to the requirements of the problem.

The algorithm may perform one of several actions in the additional window of time. Suppose that at interruption time t the algorithm is queried for a solution to problem p. If the algorithm is working on a contract for p, then the algorithm may choose to continue working on the contract (e.g., if it can finish the contract by time $t(1+\epsilon)$). A further possibility is for the algorithm to start a new contract for problem p at time t: this is a situation in which the algorithm ignores the schedule altogether and dedicates its resources to a new contract which may run for a total duration of ϵt.

Note that the requirements of the problem allow the algorithm to work for as much time as it deems necessary within the interval $[t,(1+\epsilon)t]$. This means the algorithm may stop, and return a solution, at any time in $[t,(1+\epsilon)t]$. With this requirement in mind, we propose the following definition for the acceleration ratio:

$$\beta(X) = \max_{t,p} \min \left\{ 1+\frac{1}{\epsilon}, r(p,t) \right\}, \qquad (1)$$

where $r(p,t)$ is equal to $t/\ell_{p,t}$, if no contract for p terminates within $(t,(1+\epsilon)t]$, and equal to $G_{C_{p,(1+\epsilon)t}}/\ell_{p,(1+\epsilon)t}$, otherwise.

In words, if at time t an interruption occurs concerning problem p, then the interruptible algorithm has two options (and will choose the best). The first option is to start a new contract for p at time t and of total duration $t(1+\epsilon)$, in which case its acceleration ratio should be defined as $(t(1+\epsilon))/(\epsilon t) = 1+1/\epsilon$. The second option is to consider whether some contract for p is bound to terminate within the additional window of time: if this is the case, the algorithm will wait until the longest contract for p finishes, at which point it returns the solution, otherwise it will not take any further action and terminate at time t. The function $r(p,t)$ is meant to reflect the ratio of the total elapsed time over the progress made by the algorithm on problem p (largest contract finished), in a manner similar to the definition of the acceleration ratio for hard interruptions. Note that we assume, without loss of generality, that the schedule does not start a contract that is smaller than the one that is already completed, for any given problem.

In accordance with the notation for the hard deadline problem, we will denote by β^* the optimum acceleration ratio for the soft deadline problem, namely $\beta^* = \min_X \beta(X)$. We will also be using β instead of $\beta(X)$ when X is implied from context.

It should be mentioned that there is a second natural way of defining the acceleration ratio for soft contracts, namely one which involves restricting the algorithm to wait until the end of the window to return the result. This gives rise to the following measure.

$$\gamma := \max_{t,p} \min \left\{ \frac{t(1+\epsilon)}{\ell_{p,t(1+\epsilon)}}, \frac{(1+\epsilon)t}{\epsilon t} \right\}.$$

It is straightforward to show that $\gamma = \min\left\{\alpha, 1+\frac{1}{\epsilon}\right\}$ (essentially, it suffices to substitute $t(1+\epsilon)$ with a new variable t') and thus an optimal algorithm for this measure is the (known) optimal schedule with respect to measure α (with the possibility of starting a new contract at the time of the interruption).

Note, however, that the definition of γ reflects that the algorithm will be using the entire additional window, even though it may have completed the longest contract for the queried problem long before the window elapses. Instead, one should expect that the algorithm anticipates this situation and promptly returns after the contract in question terminates, thus saving time. This situation is captured only by β, and hence from this point on we focus on β as the performance measure of the interruptible algorithm. For instance, going back to the example of the diagnostic system, a system that notifies the expert right after the best diagnosis is derived will have the same performance w.r.t. γ as a system that waits until the whole window expires. This is not desirable of course, since the former system should be far superior than the latter, and β is a measure which can make this distinction.

Lemma 2 *For every X, $\beta(X) \geq \min\{\frac{\alpha(X)}{1+\epsilon}, 1+\frac{1}{\epsilon}\}$ hence $\beta^* \geq \min\{\frac{\alpha^*}{1+\epsilon}, 1+\frac{1}{\epsilon}\}$.*

The exponential schedule

In this section we propose an exponential schedule of length base b, for some appropriate value of b. In the subsequent section we will prove that this schedule is optimal. Recall that we denote by L_k and G_k the length and finish times of the k-th contract in round-robin order, respectively. By definition, $L_k = b^k$. It is known (see (Bernstein, Finkelstein, and Zilberstein 2003)) that

$$G_k = \frac{b^{k+m} - b^{(k+m) \bmod m}}{b^{m-1}}.$$

Given interruption $\langle t_i, p_i \rangle$, denote by $\overline{c_i}$ the first contract for p_i in the schedule which is completed after time t. Let also $\overline{G_i}$ denote the finish time of contract $\overline{c_i}$. We consider two cases concerning the nature of interruption:

- $t_i(1+\epsilon) < \overline{G_i}$, namely no contract for problem p_i can finish within the additional time window.
- $t_i(1+\epsilon) \geq \overline{G_i}$, namely the schedule is such that the interruptible algorithm can complete a contract for p_i within the additional window of time.

We can thus partition I into disjoint sets A or B, depending on which of the above cases applies. More formally,

$A = \{\langle t_i, p_i\rangle \mid t_i(1+\epsilon) < \overline{G_i}\}$ and $B = \{\langle t_i, p_i\rangle \mid t_i(1+\epsilon) \geq \overline{G_i}\}$.

Given an interruption $i = \langle t_i, p_i\rangle \in A$ define $\beta_i^A = \frac{t_i}{\ell_{p_i,t_i}}$, whereas if $i \in B$ define $\beta_i^B = \frac{G_{C_{p_i,(1+\epsilon)t_i}}}{\ell_{p_i,(1+\epsilon)t}}$. In addition, define $\beta_A = \max_{i\in A}\beta_i$ and $\beta_B = \max_{i\in B}\beta_i$. The following lemma dictates that β_A, β_B (and of course ϵ) suffice to bound the acceleration ratio of the schedule.

Lemma 3 $\beta \leq \min\{\max\{\beta_A, \beta_B\}, 1 + 1/\epsilon\}$.

Here is the intuition behind our approach: An efficient schedule must guarantee that both β_A and β_B attain small values, as suggested by Lemma 3. We will argue that β_A is bounded by $\alpha/(1+\epsilon)$, and thus minimized for b chosen as in the classical case (hard deadlines). On the other hand, one can think of β_B as the acceleration ratio (again for hard deadlines) assuming interruptions only *right after a contract has finished* (c.f. Lemma 4): the resulting expression is a decreasing function of b. Hence the two measures are in a trade-off relation, but still we can choose an appropriate value of b that minimizes the maximum of β_A and β_B (c.f. Corollary 2).

We aim then to provide expressions for β_A and β_B. Consider first an interruption $i \in A$. By definition of $\overline{c_i}$, $\ell_{p_i,t_i} = \ell_{p_i,\overline{G_i}}$. Hence

$$\beta_A = \max_{i\in A}\beta_i = \max_{\langle t_i,p_i\rangle\in A}\frac{t_i}{\ell_{p_i,t_i}} = \frac{1}{1+\epsilon}\max_{i\in A}\frac{\overline{G_i}}{\ell_{p_i,\overline{G_i}}}, \qquad (2)$$

where the last equality follows from the fact that the value of t_i which maximizes the ratio is $t_i = \overline{G_i}/(1+\epsilon) - \delta$, for arbitrarily small δ.

Note also that that the above equality implies

$$\beta_A \leq \frac{1}{1+\epsilon}\max_i\frac{G_i}{\ell_{p_i,G_i}} = \frac{\alpha}{1+\epsilon}. \qquad (3)$$

The following lemma and corollary provide an expression for β_B.

Lemma 4 $\beta_B = \max_{k\geq 1}\frac{G_{n+k}}{L_{n+k}}$.

Corollary 1 $\beta_B = \frac{\alpha}{b^n} = \frac{b^m}{b^m-1}$.

We are now ready to determine the appropriate base of the exponential schedule. Our strategy is to perform a case–by–case analysis: for each case we find the corresponding best value of b. The value chosen by the algorithm is is then the one that guarantees the best acceleration ratio, among all cases. Observe that combining Lemma 3, Corollary 1 and (3) we have that

$$\beta \leq \min\left\{\max\left\{\frac{b^{n+m}}{(1+\epsilon)(b^m-1)}, \frac{b^m}{b^m-1}\right\}, 1+\frac{1}{\epsilon}\right\} \qquad (4)$$

More specifically, we consider the following cases:

Case 1: $b^n > 1+\epsilon$. Then (4) gives $\beta \leq \min\left\{\frac{\alpha}{1+\epsilon}, 1+\frac{1}{\epsilon}\right\}$. Since the interruptible algorithm can always guarantee a ratio of $1 + 1/\epsilon$ (by starting a new contract at interruption time), we seek the value of b which minimizes α, subject to $b^n > 1+\epsilon$. To this end, we use the fact that $\alpha(b)$ is minimized at $b^* = (1+\frac{m}{n})^{\frac{1}{m}}$ (Bernstein, Finkelstein, and Zilberstein 2003). In addition, since b^* is the unique local minimum of $\alpha(b)$, $\alpha(b)$ is a decreasing function of b for $b < b^*$, and an increasing function of b for $b > b^*$. Consider the subcases:

Subcase 1a) $b^* > (1+\epsilon)^{\frac{1}{n}}$, then we choose $b = b^*$ in which case $\beta \leq \min\{\alpha^*/(1+\epsilon), 1+1/\epsilon\}$, and the algorithm is optimal from Lemma 2.

Subcase 1b) $b^* \leq (1+\epsilon)^{\frac{1}{n}}$, then we need to choose the smallest value of b given the constraints, namely we will choose b infinitesimally larger than $(1+\epsilon)^{\frac{1}{n}}$.

Case 2: $b^n \leq 1+\epsilon$. In this case, (4) becomes $\beta \leq \min\left\{\frac{b^m}{b^m-1}, 1+\frac{1}{\epsilon}\right\}$. It follows that we need to minimize $\frac{b^m}{b^m-1}$ subject to $b < (1+\epsilon)^{\frac{1}{n}}$. The optimal choice of b is then $b = (1+\epsilon)^{\frac{1}{n}}$.

Corollary 2 *The strategy defined above chooses a value b such that the right-hand side of (4) is minimized.*

Optimality of the exponential schedule

Let X be a given schedule of contracts. We will follow the convention of denoting by lower case d and upper case D the lengths of a pair of consecutively completed contracts of a given problem p, respectively. More precisely, if (p_i, d_i) denotes a contract of length d_i for problem p_i, then the earliest contract in X for p_i which is completed after (p_i, d_i) finishes is denoted by (p_i, D_i).

Definition 2 *For a contract c of length D_c and start time T_c define $\alpha_X(c) = (T_c + D_c^-)/d_c$, $\lambda_X(c) = (T_c + D_c)/D_c$ and $\lambda(X) = \sup_{c\in X}\lambda_X(c)$.*

Lemma 5 *For every schedule X,*

$$\beta(X) \geq \min\left\{\max\left\{\frac{\alpha(X)}{1+\epsilon}, \lambda(X)\right\}, 1+\frac{1}{\epsilon}\right\}. \qquad (5)$$

The intuition behind our approach is as follows: $\alpha(X)/(1+\epsilon)$ and $\lambda(X)$ are related to β_A and β_B, as defined in the analysis of the exponential schedule, and are in the same trade-off relation. We need to quantify this tradeoff by first providing appropriate lower bounds for α and λ (c.f. Theorem 1 and Theorem 2), and then arguing that their contribution is minimized when X is an appropriate exponential schedule (c.f. the details in the proof of Theorem 4).

In (Lopez-Ortiz, Angelopoulos, and Hamel 2006) it is shown that for every schedule X there exists another schedule X' such that $\alpha(X') \leq \alpha(X)$; moreover X' satisfies a set of properties (which at an intuitive level introduce some structure into the schedule). Such a schedule is called *normalized*, and to stress that the normalization pertains to the measure α we will call it *α-normalized*. The specific properties are not important for the purpose of our analysis; however, it is important to observe that a normalized schedule X'

is derived from X by swapping the problem tags of appropriate pairs of contracts in X. Such swappings do not affect the λ value of the schedule, or more formally, it is also the case that $\lambda(X') = \lambda(X)$. This implies that for every optimal schedule w.r.t. β, there exists an optimal α-normalized schedule of the same λ-value.

Theorem 1 *(Lopez-Ortiz et al. 2006) For any α-normalized schedule $X = ((c_0, m_0, x_0), (c_1, m_1, x_1), \ldots)$ for n problems with m processors*

$$\alpha(X) \geq \sup_{k \geq 0} \left\{ \sum_{i=0}^{k+n} x_i^s \Big/ \sum_{i=k-m+1}^{k} x_i^s \right\}, \qquad (6)$$

where $X^s = (x_0^s, x_1^s, \ldots)$ is the sequence of the sorted x values of X and $x_i^s := 0$ if $i < 0$.

We now introduce an additional normalization criterion, this time with respect to measure λ.

Lemma 6 *For any schedule X one of the following hold: i) For any two contracts (p_i, D_i), (p_j, D_j) scheduled to start at times T_i and T_j, respectively, if $T_i + D_i > T_j + D_j$ then $D_i > D_j$, or ii) we can define a schedule X' such that $\lambda(X') \leq \lambda(X)$ and, additionally, X' observes property (i).*

We call a schedule that obeys property (i) of Lemma 6 *λ-normalized*. The following is a basic theorem concerning λ-normalized schedules.

Theorem 2 *For any λ-normalized schedule $X = ((c_0, m_0, x_0), (c_1, m_1, x_1), \ldots)$ for n problems with m processors*

$$\lambda(X) \geq \sup_{k \geq 0} \left\{ \sum_{i=0}^{k+m} x_i^s \Big/ \sum_{i=k+1}^{k+m} x_i^s \right\} \qquad (7)$$

where $X^s = (x_0^s, x_1^s, \ldots)$ is the sequence of the sorted x values of X and $x_i^s := 0$ if $i < 0$.

Proof sketch. Suppose that at time T_0 processor M_0 is about to start a new contract C_0 for problem P_0; let D_0 denote its length. Denote by C_j, for all $0 < j \leq m-1$, the latest contract which is scheduled in M_j and which is completed by time $T_0 + D_0$ (and by D_j its corresponding length).

Consider now the sequence of contract lengths for processor M_j completed up to $T_0 + D_0$ inclusively. These time spans are elements in the sequence X^s; let R_j be the set of indices in X^s of these scheduled time spans for processor M_j. The λ-value for contract C_j is then given by $\lambda_X(C_j) = \frac{\sum_{i \in R_j} x_i^s}{D_j}$.

Note that $D_j = x_{l_j}^s$ for some $l_j \geq 0$. Hence

$$\lambda(X) \geq \max_{0 \leq j \leq m-1} \left\{ \frac{\sum_{i \in R_j} x_i^s}{D_j} \right\} \geq \frac{\sum_{j=0}^{m-1} \sum_{i \in R_j} x_i^s}{\sum_{j=0}^{m-1} x_{l_j}^s}. \qquad (8)$$

Let N denote the set of contracts completed by $T_0 + D_0$, *excluding* contracts $C_0, \ldots C_{m-1}$, and C' denote the contract of largest length in N, then the length of C' is equal to $x_{k_0}^s$, for some k_0. The proof is completed by showing that $\sum_{j=0}^{m-1} \sum_{i \in I_j} x_i^s \geq \sum_{i=0}^{k_0+m} x_i^s$, and $\sum_{j=0}^{m-1} D_j \leq \sum_{i=k_0+1}^{k_0+m} x_i^s$. □

In order to prove a lower bound on the right hand side of inequality (5) we make use of the results by Gal (Gal 1980) and Schuierer (Schuierer 2001) which, for completeness, we state here without proof and in a simplified form. Define $G_a = (1, a, a^2, \ldots)$ to be the geometric sequence in a and $X^{+i} = (x_i, x_{i+1}, \ldots)$ the suffix of sequence X starting at x_i.

Theorem 3 ((Schuierer 2001)) *Let $X = (x_0, x_1, \ldots)$ be a sequence of positive numbers, r an integer, and $a = \overline{\lim}_{n \to \infty}(x_n)^{1/n}$, for $a \in \mathbb{R} \cup \{+\infty\}$. If F_k, $k \geq 0$, is a sequence of functionals which satisfy*

1. *$F_k(X)$ only depends on $x_0, x_1, \ldots, x_{k+r}$,*
2. *$F_k(X)$ is continuous, for all $x_i > 0$, with $0 \leq i \leq k + r$,*
3. *$F_k(\alpha X) = F_k(X)$, for all $\alpha > 0$,*
4. *$F_k(X + Y) \leq \max(F_k(X), F_k(Y))$, and*
5. *$F_{k+i}(X) \geq F_k(X^{+i})$, for all $i \geq 1$,*

then

$$\sup_{0 \leq k < \infty} F_k(X) \geq \sup_{0 \leq k < \infty} F_k(G_a).$$

Theorem 4 *The exponential schedule which observes Corollary 2 is optimal.*

Proof. Let $X = ((c_0, m_0, x_0), (c_1, m_1, x_1), \ldots)$ denote an optimal α-normalized schedule (w.r.t. the acceleration ratio β), and X^s the sequence of the sorted X values (as argued earlier, such an optimal schedule exists). In (Lopez-Ortiz, Angelopoulos, and Hamel 2006), Theorem 1 can be used to show

$$\alpha(X) \geq \frac{a^{n+m}}{a^m - 1}, \qquad (9)$$

where $a = \overline{\lim}_{n \to \infty}(x_n)^{1/n}$.

Let $Y = ((c_0, m_0, y_0), (c_1, m_1, y_1), \ldots)$ denote the λ-normalized schedule derived from X by applying (if necessary) the process described in Lemma 6. Then, from the details of the proof of Lemma 6, the sequence Y^s of the sorted Y values is such that $y_i^s \leq x_i^s$. Thus, if $\tilde{a} = \overline{\lim}_{n \to \infty}(y_n)^{1/n}$, we have that $\tilde{a} \leq a$.

Define $F_k := \sum_{i=0}^{k+m} y_i^s \Big/ \sum_{i=k+1}^{k+m} y_i^s$. It is easy to see that F_k satisfies all the conditions of Theorem 3. Hence

$$\sup_{0 \leq k < \infty} F_k(Y^s) \geq \sup_{0 \leq k \leq \infty} \left\{ \sum_{i=0}^{k+m} \tilde{a}^i \Big/ \sum_{i=k+1}^{k+m} \tilde{a}^i \right\}. \qquad (10)$$

Note that if $\tilde{a} \leq 1$, then the ratio in (10) tends to infinity as $k \to \infty$, then (5) and Theorem 2 imply $\beta(X) \geq 1 + 1/\epsilon$, which is trivially matched by any schedule. Hence, we can assume that $\tilde{a} > 1$ and obtain

$$\begin{aligned} \sup_{0 \leq k < \infty} F_k(Y^s) &\geq \sup_{0 \leq k \leq \infty} \left\{ \sum_{i=0}^{k+m} \tilde{a}^i \Big/ \sum_{i=k+1}^{k+m} \tilde{a}^i \right\} \\ &= \sup_{0 \leq k < \infty} \left\{ \frac{\tilde{a}^m - \tilde{a}^{-k+1}}{\tilde{a}^m - 1} \right\} \end{aligned}$$

$$\stackrel{(\tilde{a}\geq 1)}{=} \quad \frac{\tilde{a}^m}{\tilde{a}^m-1} \geq \frac{a^m}{a^m-1}, \tag{11}$$

where the last inequality follows from the fact that the function $f(x) = x^m/(x^m-1)$ is decreasing in x, and $\tilde{a} \leq a$. Combining (9), (11), with Lemma 5, Theorem 2 and the definition of $\lambda(X)$ it follows that

$$\beta(X) \geq \min\left\{\max\left\{\frac{a^{n+m}}{(1+\epsilon)(a^m-1)}, \frac{a^m}{a^m-1}\right\}, \frac{1+\epsilon}{\epsilon}\right\}.$$

The theorem follows then from Corollary 2. □

Conclusions and future work

In this paper we formulated and addressed the problem of designing interruptible algorithms in a setting in which the interruption is not a hard deadline, but rather an additional time window is available to the algorithm to complete its execution. We presented an exponential schedule of optimal acceleration ratio in this setting.

Several other formulations of a soft deadline are possible (e.g., by defining a penalty proportional to the amount of time between the actual interruption and the time that a solution is returned). It would be interesting to address such alternative formulations and compare them in the context of a real-time application. Another topic of future work includes average-case analysis of schedules, namely the case in which the interruption and/or the window are drawn from a known distribution. Last, the problems of scheduling contract algorithms and parallel ray-searching are surprisingly interrelated, as shown in (Bernstein, Finkelstein, and Zilberstein 2003). Do similar connections arise when dealing with soft, as opposed to hard, deadlines?

References

Bernstein, D.; Perkins, T. J.; Zilberstein, S.; and Finkelstein, L. 2002. Scheduling contract algorithms on multiple processors. In *Proceedings of the 18th National Conference on Artificial Intelligence*, 702–706.

Bernstein, D. S.; Finkelstein, L.; and Zilberstein, S. 2003. Contract algorithms and robots on rays: Unifying two scheduling problems. In *Proceedings of the 18th International Joint Conference in Artificial Intelligence*, 1211–1217.

Dean, T., and Boddy, M. S. 1998. An analysis of time-dependent planning. In *Proceedings of the 15th National Conference on Artificial Intelligence*, 49–54.

Gal, S. 1980. *Search Games*. Academic Press.

Horvitz, E. 1987. Reasoning about beliefs and actions under computational resource constraints. In *Proceedings of the 3rd Annual Conference on Uncertainty in Artificial Intelligence*, 301–324.

Horvitz, E. 1998. Reasoning under varying and uncertain resource constraints. In *Proceedings of the 15th National Conference on Artificial Intelligence*, 111–116.

Lopez-Ortiz, A.; Angelopoulos, S.; and Hamel, A. 2006. Optimal scheduling of contract algorithms for anytime problems. In *Proceedings of the 21st National Conference on Artificial Intelligence*.

Manolache, S.; Eles, P.; and Peng, Z. 2004. Optimization of soft real-time systems with deadline miss ratio constraints. In *Proc. 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'04)*, 562.

Russell, S. J., and Zilberstein, S. 1991. Composing real-time systems. In *Proceedings of the 12th International Joint Conference in Artificial Intelligence*, 212–217.

Schuierer, S. 2001. Lower bounds in online geometric searching. *Computational Geometry: Theory and Applications* 18(1):37–53.

Zilberstein, S.; Charpillet, F.; and Chassaing, P. 1999. Real-time problem-solving with contract algorithms. In *Proceedings of the 16th International Joint Conference in Artificial Intelligence*, 1008–1015.

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