University of Szeged

Department of Computer Algorithms and Artificial Intelligence

Online lookahead on parameter learning algorithms for data acknowledgment and scheduling problems

Summary of the Ph.D. thesis

by

Tamás Németh

Thesis advisor

Csanád Imreh, Ph. D.

University of Szeged

Ph.D. School in Computer Science

Szeged

2013

-

1 Introduction

We are dealing with online optimisation problems where we only know of a partial input and have to make decisions based on the information we already have, without knowledge of further data.

We cannot expect from an algorithm like this to provide us with the optimal solution like one that is supplied by algorithms that possess the whole information. The algorithms that have complete knowledge of the input are called offline algorithms.

There are two essential methods of investigating the effectiveness of online algorithms. One of them is the analysis of the average case. In this case, we need to assume some kind of probability distribution of the possible inputs, while investigating the expected value of the objective function on the distribution.

Another approach to this problem is the worst case analysis, which is called competitive analysis. In this case, we compare the objective function value of the solution provided by the online algorithm with the optimal offline objective function value.

In what follows, we denote the value of the objective function of an arbitrary ALG online algorithm on the I input by $ALG(I)$. The value of the optimal offline objective function of the I input is $OPT(I)$. Using this system of notations, we may define the competitiveness on minimisation problem as follows: the ALG algorithm is C-competitive if $ALG(I) \leq C \cdot OPT(I)$ is true for all input I. The competitive ratio of an algorithm is the smallest C number, on which the algorithm is C-competitive.

2 The acknowledgement problem

Here we summarise our results on the competitive ratio on the algorithms for the lookahead version of the data acknowledgement problem, presented in [7].

In the communication of a computer network the information is sent by packets. If the communication channel is not completely safe then the arrival of the packets must be acknowledged. The TCP implementations are also using acknowledgement of the packets (see [12]). In the data acknowledgement problem we try to determine the time of sending acknowledgements. One acknowledgement can acknowledge many packets but waiting for long time can cause the resending of the packets and that results the congestion of the network. On the other hand sending immediately an acknowledgement about the arrival of each packet would cause again the congestion of the network.

In the mathematical model of the problem input is the list of the arrival times $a_1, \dots, a_n$ of the packets. We also denote the packets by their arrival time. The decision maker has to determine when to send acknowledgements, these times are denoted by $t_1, \dots, t_k$. We consider the objective function $\gamma k + (1-\gamma)\sum_{j=1}^{k} L_j$, where k is the number of the sent acknowledgements and L_j is the extra latency belonging to the acknowledgement t_j and $\gamma \in (0,1)$ is a constant. We consider two different cases. We obtain the objective function $f_{\max}$ if $L_j = \max_{t_{j-1} < a_i \le t_j}(t_j - a_i)$, the maximal delay collected by j-th acknowledgement. We obtain the objective function f_{sum} if $L_j = \sum_{t_{j-1} < a_i \le t_j}(t_j - a_i)$, the sum of the delays collected by the j-th acknowledgement.

We consider a semi-online model with time lookahead c, where at time t the decision maker knows the arrival times of the packets already arrived and also knows the arrival times of the packets arriving in the time interval $(t, t+c]$. When $c < \gamma/(1-\gamma)$ we define an extended version of the alarming algorithm developed in [6]. This *time lookahead alarming algorithm* (TLA in short) works as follows.

In the case of function f_{max} at the arrival time a_j an alarm is set for time $a_j + \gamma/(1-\gamma) - c$. If the packet a_{j+1} arrives before the alarm or we can see a_{j+1} at time $a_j + \gamma/(1-\gamma) - c$ in the lookahead interval $(a_j + \gamma/(1-\gamma) - c, a_j + \gamma/(1-\gamma)]$ then we postpone the alarm to the time $a_{j+1} + \gamma/(1-\gamma) - c$. In the opposite case (no packet arrives in the time interval $(a_j, a_j + \gamma/(1-\gamma)]$) an acknowledgement is sent at time $a_j + \gamma/(1-\gamma) - c$ which acknowledges all of the unacknowledged packets.

In the case of function f_{sum} at the arrival time a_j an alarm is set for time $a_j + e_j$ where $e_j = (\gamma/(1-\gamma) - \sum_{a_i \in \sigma_j}(a_j - a_i))/|\sigma_j|$. If the packet a_{j+1} arrives before the time $\max\{a_j, a_j + e_j - c\}$ or we can see a_{j+1} at this time in the lookahead interval, then we move to a_{j+1} and reset the alarm. In the opposite case (no packet arrives in the time interval $(a_j, a_j + e_j]$) an acknowledgement is sent at time $\max\{a_j, a_j + e_j - c\}$ which acknowledges all of the unacknowledged packets.

We summarise our results about this algorithms in Thesis 1 below.

Thesis 1

Theorem 1 *TLA is* $\max\{1, 2 - \frac{1-\gamma}{\gamma}c\}$*-competitive in the* f_{max} *model.*

Theorem 2 *No semi-online algorithm in the* $f_{\max}$ *model with lookahead* c *may have smaller competitive ratio than* $\max\{1, 2 - \frac{1-\gamma}{\gamma}c\}$.

Theorem 3 *The competitive ratio of TLA is* 2 *for arbitrary* c *in the* f_{sum} *model.*

In the case when $c > \gamma/(1-\gamma)$ we can achieve smaller competitive ratio than 2 by the following algorithm. We present the Lookahead Interval Planning Algorithm (LIP in short). The algorithm partitions the packets into blocks and for each block determines the acknowledgments based on an offline optimal solution. The block always starts at the first unacknowledged packet. First the algorithm examines whether there exist packets a_i, a_{i+1} in the c length lookahead interval with the property $a_{i+1} - a_i > \gamma/(1-\gamma)$. If there exists such pair, then the block is ended at the first such a_i, otherwise the block has length c. Then LIP calculates the optimal solution of the offline acknowledgement problem for the packets in the block, it can use one of the algorithms which solves the offline problem (such algorithm is presented in [6]) and sends the acknowledgements according to this solution and considers the next block.

We collect our results about the f_{sum} model in Thesis 2.

Thesis 2

Theorem 4 *LIP is* $1 + \frac{\gamma}{(1-\gamma)c}$*-competitive in the* f_{sum} *model.*

Theorem 5 *No online algorithm with lookahead* $c > \gamma/(1-\gamma)$ *in the* f_{sum} *model may have smaller competitive ratio than* $1 + \Omega(1/c^2)$.

Theorem 6 *No online algorithm with lookahead* $c \leq \gamma/(1-\gamma)$ *in the* f_{sum} *model can have smaller competitive ratio than* $2\gamma/(c(1-\gamma)+\gamma)$.

3 Parameter learning algorithms for the online data acknowledgment problem

In this part we present a new online algorithm and a new lookahead semi-online algorithm for the data acknowledgement problem with the

f_{sum} objective which has better performance in average case than the algorithms known from the literature. We published the results about these algorithms in [8, 9]. The algorithm works in stages and for each stage it uses an alarming or a TLA algorithm to acknowledge the packets. First, we define the subclass of the alarming algorithms which can be used. For a positive parameter p, we can define an algorithm $Alarm_p$ as follows. $Alarm_p$ uses the value $e_j = (p\gamma/(1-\gamma) - \sum_{a_i \in \sigma_j}(a_j - a_i))/|\sigma_j|$ for each j. This e_j implies that if no new packet arrives then sending an acknowledgment at time $a_j + e_j$ has a latency cost of $\sum_{a_i \in \sigma_j}(a_j - a_i)) + |\sigma_j| \cdot e_j = p \cdot \gamma/(1-\gamma)$. We should mention that $Alarm_1$ is the 2-competitive algorithm defined in [6]. We can extend TLA into TLA_p in the same way in the lookahead model

The new parameter learning algorithms depend on two parameters, namely r and q. LearnAlarm solves the online problem and uses $Alarm_p$ as a subroutine, LearnTLA solves the lookahead problem and uses the lookahead algorithm TLA_p as subroutine. The algorithm itself can be described as follows.

Algorithm Learnalarm(r,q)/LearnTLA(r,q)

- *Stage 1* Use the $Alarm_1/TLA_1$ algorithm to acknowledge the first r packets. Go on to Stage 2.
- *Stage j ($j > 1$)*
  - Let I_j represent the input containing the last $r \cdot q$ packets. If fewer packets have arrived, then I_j contains all of them.
  - Use the SimpleOpt algorithm to determine the value p which minimizes the cost among the algorithms $Alarm_p/TLA_p$ on input I_j. Denote this value by p^*.
  - Use the $Alarm_{p^*}/TLA_{p^*}$ algorithm to acknowledge the next r packets and go on to Stage $j+1$.

We proved that the optimal value of p is such that some of the acknowledgments are sent exactly at the arrival time of a packet. Based on this observation we developed the SimpleOpt algorithm which checks only those values which satisfy this property.

The SimpleOpt Algorithm

- *Initialization:* Let $p^* = 1$ and $Z = \infty$

- *Search:* For each $1 \leq i < j \leq n$ do
  - Let $s := \sum_{k=i}^{j}(a_j - a_i)$, and let $p = (1-\gamma)s/\gamma$.
  - For p, execute $Alarm_p/TLA_p$ on the input, denote the number of acknowledgments by k,
  - If $k\gamma(1+p) < Z$, then $p^* := p$ and $Z := k\gamma(1+p)$
- Return p^*

We evaluated the efficiency of the new parameter learning algorithm via an experimental analysis. Based on the results we could draw the following set of conclusions:

- Both algorithm Learnalarm(r,q) and LearnTLA(r,q) gave significantly better results on the tests than best competitive algorithms from the literature.
- Our second observation is that the tests clearly show that in contrast to the competitive analyis the lookahead property and also the parameter learning can bring about a significant improvement in the average efficiency of the Alarm algorithm.
- Thirdly we see that the new algorithm is sensitive to the parameter settings. A careful parameter setting yields an additional decrease in the ALG/OPT ratio. The best setting of the parameters depends on the input, but we can conclude some general rule. It seems that choosing a medium learning sequence gives better results than a smaller one or a larger one.

We summarise these results in Thesis 3.

Thesis 3

We developed new online algorithms, one for the solution of the online data acknowledgement and one for the online data acknowledgement with lookahead problems which are based on parameter learning. We proved by an empirical analysis that the new algorithms give significantly better results in the average case than the best known algorithms for these problems.

4 Parameter learning online algorithm for multiprocessor scheduling with rejection

Here we present a new algorithm for the scheduling with rejection problem. These results are published in [10]. The problem of scheduling with rejection is defined in [2]. In this model, it is possible to reject the jobs. The jobs are characterised by a processing time p_j and a penalty w_j. The goal is to minimise the makespan of the schedule for the accepted jobs plus the sum of the penalties of the rejected jobs. One basic idea in scheduling with rejection is to compare the penalty and the load (processing time divided by the number of machines) of the job, and reject the job in the case when the penalty is smaller. This algorithm is called Greedy. In [2] a 2.618-competitive algorithm called RTP is defined. It is a refined version of Greedy, it also rejects some large jobs with $w_j > p_j/m$, these jobs are collected in set R. We can define this algorithm as follows.

Algorithm RTP(α)

```
• 1. *Initialization.* Let $R := \emptyset$.
• 2. When job $j$ arrives
    – (i) If $w_j \le \frac{p_j}{m}$, then reject.
    – (ii) Let $r = \sum_{i \in R} w_i + w_j$. If $r \le \alpha \cdot p_j$, then reject job $j$, and
      set $R = R \cup \{j\}$.
    – (iii) Otherwise, accept $j$ and schedule it by LIST
```

Proposition 1 *RTP is* $(3 + \sqrt(5))/2$ *competitive, with parameter* $\alpha = (\sqrt(5) - 1)/2$

Proposition 2 *There exists no online algorithm that is β-competitive for some constant $\beta < (3 + \sqrt{5})/2$ and for all m.*

These propositions show that RTP is optimal in the sense that it has the smallest possible competitive ratio. But it often happens that some algorithms with worse competitive ratio has better performance in the average case [1].

Parameter learning algorithm

We present a new algorithm PAROLE (Parameter Online Learning) which tries to learn the best parameter during its execution. The algorithm works in phases, after each phase it chooses a new parameter based on the known part of the input. First we define a frame algorithm which uses the selection of the new parameter as a subrutin, then we define the subrutin which finds the new parameter. We defined the phases by the number of arriving jobs, the phase is finished after 250 jobs.

Algorithm PAROLE (PHASE i)

- At the beginning of phase i, use algorithm CHOOSE to find a new parameter α_i.
- Perform $RTP(\alpha_i)$ on the arrived part of the input. Change set R.
- Use $RTP(\alpha_i)$ for the jobs arriving during the phase.

It is a straightforward idea to use the value α_i where the cost $RTP(\alpha_i)(I)$ is minimal for the known part of input. We think so that it is NP-hard to find this value thus we used the following sampling algorithm to find the new value of the parameter. The algorithm uses the previous value of the parameter denoted by α^*.

Algorithm CHOOSE

- Generate one element from the intervals $[(i-1)/10, i/10]$ by uniform distribution for $i = 1, \ldots, 10$. Denote this set by S_1. Consider the value α from S_1 where $RTP(\alpha)$ has the smallest cost on I. Denote it by $\bar{\alpha}$.
- Generate one element from the intervals $[\alpha^\star - i/100, \alpha^\star - (i-1)/100]$, $[\alpha^\star + (i-1)/100, \alpha^\star + i/100]$, $[\bar{\alpha} - i/100, \bar{\alpha} - (i-1)/100]$, $[\bar{\alpha} + (i-1)/100, \bar{\alpha} + i/100]$ for $i = 1, \ldots 10$ by uniform distribution. Denote the set of the generated elements by S_2. Return the value α from $S_2 \cup \{\alpha^*\} \cup \{\bar{\alpha}\}$ where $RTP(\alpha)$ has the smallest cost on I.

The basic idea of CHOOSE is to select a good parameter value. We investigate the neighbourhoods of two candidates, one is the previous value of the parameter and a further one is a new value $\bar{\alpha}$ which is the best among several candidates selected independently on the previous value of the parameter.

We compared PAROLE, Greedy and RTP both on randomly generated and on real data. All algorithms gave similar results, but PAROLE had always the best or the second best performance. We summarise these results in Thesis 4.

Thesis 4

We developed a new online algorithm for the solution of the scheduling with rejection problem called PAROLE. We proved by an empirical analysis that the new algorithms give similar results in the average case as the best known algorithms for this problem, but it never gave the worst results among the algorithms used in the tests.

5 Online data clustering algorithms in an RTLS system

In this part we present the results on online data clustering algorithms in an RTLS system which are published in [11]. Real time location systems (RTLS) promise to deliver high precision positions of objects.

A local locating infrastructure is using Angle of Arrival (AoA) and Round-Trip-Time (RTT) measurements to determine the position of objects [4, 5] . In this system the objects are equipped with tags based on the in-house-developed Wireless Smart Item platform (WISMIT). These tags periodically broadcast a radio signal to the infrastructure nodes denoting the beginning of a locating cycle. We will refer to this as a burst. The radio signal carries also user data, aside from tag identification information: an incremental number identifying the burst, referred to as burst-id later on, is also included. A set of infrastructure nodes in proximity of the tag consisting of WISMIT anchors capable of RTT measurements and of Goniometers capable of both RTT and AoA measurements receive this broadcast and initiate location data acquisition for this tag. The measured parameters form a so called burst data set which is at first only available distributed on the infrastructure nodes. The clustering algorithm has to determine when to forward the data to the positioning server.

The clustering algorithm can not simply wait for the arrival of all elements of the burst data set, it needs a more sophisticated technique. Thus the goal of the algorithm is to decide when to pass the collected data to the position calculation. There are two contradicting objectives which should be satisfied. First, the positioning server should receive all the available data from the infrastructure nodes. Moreover the system is not allowed to wait a long time for the incoming data, since the delay decreases the relevance of the determined position. A fairly-good position is better than a position too late. We present an integrated objective function which considers these goals, and defines this clustering problem as a maximisation problem.

To define the objective function we use the following notations. For each positioning time p_j and infrastructure node k let $e_{jk} = 1$ if node k sends data into the positioning calculation, otherwise let $e_{jk} = 0$.
Let

$$r_j = \max_{i=1}^{n}\{Rcpt(x_i) | x_i \in B'_j\}.$$

Now we can define the following objective function, which we have to maximise:

$$f = \sum_{j=1}^{b}\sum_{k=1}^{m} e_{jk}w_k - c\sum_{j=1}^{b}(p_j - r_j).$$

The No Waiting Time Algorithm (NWT in short) sends the first data element for each burst id into the positioning server immediately after it arrives.

The Constant Waiting Time algorithm (CWT) waits time d after the arrival of the first data element for each burst id before sending the data to the location server, if the difference among the arrivals of the data elements with the same burst id (the length of the burst) cannot be greater than a given value d.

We summarise the results on the competitive ratios in Thesis 5.

Thesis 5

Theorem 7 *There exists no algorithm which has smaller competitive ratio than* $\frac{\sum_{k=1}^{m} w_k}{w_1} \geq m$.

Theorem 8 *The competitive ratio of NWT is* $\frac{\sum_{k=1}^{m} w_k}{w_1}$.

Theorem 9 *If the difference among the arrivals of the data elements with the same burst id (the length of the burst) cannot be greater than a given value d, and* $\frac{w_1}{c} > d$, *there exists no algorithm which has smaller competitive ratio than* $\min\{\frac{w_1}{w_1 - c\cdot d}, \frac{\sum_{k=1}^{n} w_k}{w_1}\}$.

Theorem 10 *The competitive ratio of CWT is* $\frac{w_1}{w_1 - c\cdot d}$. *if the difference among the arrivals of the data elements with the same burst id (the length of the burst) cannot be greater than a given value d.*

Corollary 1 *In the case when the difference among the arrivals of the data elements with the same burst id (the length of the burst) cannot be greater than a given value d, CWT achieves the smallest possible competitive ratio if $\frac{w_1}{w_1 - c \cdot d} \leq \frac{\sum_{k=1}^{m} w_i}{w_1}$, otherwise NWT achieves the smallest possible competitive ratio.*

We also introduce an algorithm (VWT) trying to get knowledge about d, and present an empirical analysis which compares the algorithms in the average case.

Algorithm VWT uses the following rules to send data to the positioning server:

- If the actual time is $S(j) + W$ and $DS(j)$ is not closed then the algorithm closes set $DS(j)$. It notes that its closing time is $S(j) + W$, and it sends the data to the positioning server. Furthermore let $PD(j) = \max\{0, \min_{a=0}^{MEM-1}\{p_{j-a} - r_{j-a}\} - INC\}$ be the minimal possible decrease amount for W, respecting also INC security time, so that the data of the last MEM bursts would have been left complete. So if $PD(j) \geq DEC$, then W is decreased by $PD(j)$.

- If $DS(j)$ collects the data at time t from all of the infrastructure nodes the algorithm closes set $DS(j)$. It notes that its closing time is t, and it sends the data to the positioning server. Note that this might happen only in the case when $t \leq S(j) + W$. A possible decrease of W can also happen analogue to the first point.

The algorithm uses the following rules to handle the received data x_i.

- If an x_i which belongs to burst j arrives at time t and $DS(j)$ is not closed, then it extends it with the new data element, and checks whether it is the last missing infrastructure node.

- If $DS(j)$ is already closed with closing time at r_j then $W := t - S(j) + INC$.

We analysed the algorithms in the simulation system of the Fraunhofer Institute and the tests show that this new algorithm has significantly better performance than the algorithms with the smallest possible competitive ratio. This is summarised Thesis 6.

Thesis 6

We developed a new online algorithm for the solution of an online clustering problem in the location system. We proved by an empirical analysis that this new algorithm gives better results in the average case then the algorithms with the smallest possible competitive ratio.

References

[1] Albers, S. and Schröder, B. An Experimental Study of Online Scheduling Algorithms. *ACM Journal of Experimental Algorithms*, article 3, 2002.

[2] Bartal, Y., Leonardi, S., Marchetti-Spaccamela, A., Sgall, J. and Stougie, L. Multiprocessor scheduling with rejection, *SIAM Journal on Discrete Mathematics*, 13:64–78, 2000.

[3] A. Borodin, R. El-Yaniv, *Online Computation and Competitive Analysis* Cambridge University Press, 1998.

[4] M. Brugger, T. Christ, F. Kemeth, S. Nagy, M. Schaefer, M.M. Pietrzyk, The FMCW technology-based indoor localization system, *Proceedings of Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS)*, Helsinki, Finnland, 2010 pp. 1–6.

[5] M. Brugger, F. Kemeth, Locating rate adaptation by evaluating movement specific parameters, *Proceedings of 2010 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, Anaheim, USA, 2010 pp. 127–133.

[6] D. R. Dooly, S. A. Goldman, S. D. Scott: On-line analysis of the TCP acknowledgment delay problem. *J. ACM* **48(2)** 243–273, 2001.

[7] Cs. Imreh, T. Németh, *On time lookahead algorithms for the online data acknowledgement problem* in *32nd International Symposium on Mathematical Foundations of Computer Science, LNCS 4708*, Cesky Krumlov, 2007, pp. 288-297.

[8] Cs. Imreh and T. Németh, Parameter learning algorithm for the online data acknowledgment problem, *Optim. Methods Softw.,* **26**, 3 (2011) 397–404.

[9] T. Németh, B. Gyekiczki, Cs. Imreh, Parameter Learning in Lookahead Online Algorithms for Data Acknowledgment, *IEEE International Symposium on Logistics and Industrial Informatics*, 2011

[10] T. Németh and Cs. Imreh, *Parameter learning online algorithm for multiprocessor scheduling with rejection*, Acta Cybernetica 19(1) (2009), pp. 125–133.

[11] T. Németh, S. Nagy, Cs. Imreh Online data clustering algorithms in an RTLS system *Acta Universitatis Sapientiae, Informatica*, **5**, 1 (2013) 5-15.

[12] W. R. Stevens, *TCP/IP Illustrated, Volume I. The protocols*, Addison Wesley, Reading, 1994