Acta Univ. Sapientiae, Informatica, **5,** 1 (2013) 5–15

DOI: 10.2478/ausi-2014-0001

Online data clustering algorithms in an RTLS system

Tamás NÉMETH
University of Szeged
Institute of Informatics
email: tnemeth@inf.u-szeged.hu

Sándor NAGY
Fraunhofer Institut for Integrated Circuits, Locating and Communication Systems Department
email: sandor.nagy@iis.fraunhofer.de

Csanád IMREH
University of Szeged
Institute of Informatics
email: cimreh@inf.u-szeged.hu

Abstract. This paper proposes and evaluates solutions for an online clustering problem and gives a mathematical model for it. The problem at hand occurs often in the fusion of data streams for example in real time locating systems. The goal is to gather as much incoming information from several sources as possible but also minimize the delay before the next processing step can be executed. The key characteristic is that the data is available in a bursty fashion, in the special case of an RTLS according to the locating cycles. After an introduction of the background a general mathematical model for the problem is given, and then two basic algorithms referred to as NWT and CWT are analyzed by the method of competitive analysis. Each turning out to deliver an optimal solution under different constraints. Then an experimental evaluation follows based on a simulation involving the CWT and the algorithm referred to as VWT. The later is giving a configuration free solution for the problem.

Computing Classification System 1998: F.1.2
Mathematics Subject Classification 2010: 68W27
Key words and phrases: online algorithms, clustering, locating systems

1 Introduction

The problem for this paper emerged in the real time locating system developed by the Fraunhofer IIS, but it can be generalized. We will do so in the second section by giving a mathematical model for it. To draw a context around the problem here we explain where it originates from, and why the specific constraints are posed.

Real time location systems (RTLS) promise to deliver high precision positions of objects with manifold applications in transport and logistics, ambient assisted living, or emergency mission support and also in the entertainment and sports like the Chip-in-the-Ball technologies. The mentioned locating system of the Fraunhofer IIS is a radio based system working with a local locating infrastructure using Angle of Arrival (AoA) and Round-Trip-Time (RTT) measurements to determine the position of objects (see [2] and [3] for a more detailed description of this RTLS system). 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. After the measurement is done the individual parts of the burst data set, the data elements are sent over an Ethernet connection to the central positioning server through the TCP/IP network stack utilizing UDP as a transport protocol. UDP packets were chosen in contrast to a TCP stream because of its lower overhead and characteristics: if a datagram was lost, no additional time is spent for detecting the loss and retransmitting it. There is also no guaranty for order-reserving delivery.

After receiving sufficiently enough data, ideally the whole burst data set, the positioning server can determine the position of the object. For this the raw location data is going through the following algorithmical steps in the server:

1. Raw data filter
2. Clustering

3. Burst filter
4. Position calculation
5. Position filter

Most of the steps are meant for reducing measurement noise and errors.

The focus of this paper lies on the second step: the clustering component. It is in charge for gathering as much of the burst data set as possible and forward it as quickly as possible to the burst filtering and position calculation. For this it has to work in an online manner by clustering the data collected by the infrastructure nodes. It has also to deal with transmission errors or disturbances like the temporary coverage of an infrastructure node, and network packet losses and with variable network and processing delays as well. Therefore 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 maximization problem. Since collecting the data and the position calculation can be done parallel and independently for several tags, we suppose that there is only one tag present and the infrastructure nodes collect data only from this tag. We note that this problem is similar to the online data acknowledgment problem, where the goal is to determine the sending time of the acknowledgments (see [1] and [4]). One of the presented algorithms uses ideas used in the field of data acknowledgment problems.

As next we present the mathematical model of the problem, giving the objective function and we define competitiveness. Typically, the quality of an online algorithm is judged using competitive analysis. This method will be used in Section 3 to analyze the efficiency of our algorithms. We will present there two analyses: one without constraints on the data arrival times, and one with constraints to better describe the system behavior and deliver stronger results.

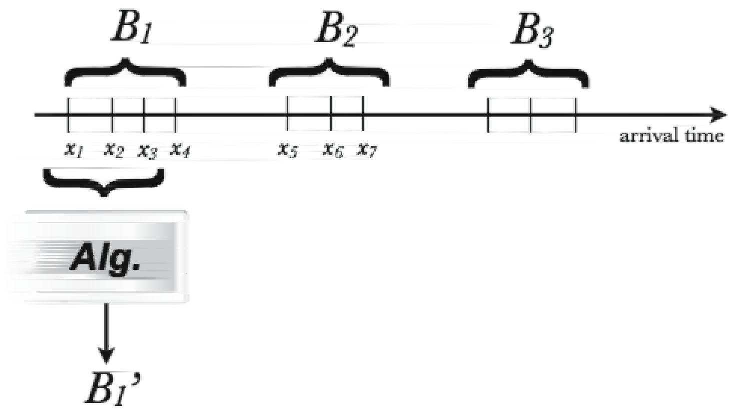

Figure 1: The clustering of the input signals

2 The mathematical model of the clustering problem

The input of this clustering problem is a list X: $x_1, \ldots, x_n$ of collected data elements. We will use the following notation: $\mathtt{Burst}(x_i)$ denotes the burst id which identifies to which locating cycle the data belongs to. $\mathtt{Node}(x_i)$ gives the infrastructure node which collected the data. $\mathtt{Rcpt}(x_i)$ is the reception time of the data. The difference between the reception times of the last and the first data elements of a burst is called the length of the burst.

We note that in the application the data has further attributes also (e.g.: AoA values, RTT values) which are substantial for the position calculation, but are not used in the clustering algorithm. $B_j = \{x_i | \mathtt{Burst}(x_i) = j\}$ is the burst data set, $B'_j \subseteq B_j$ is the subset of the burst data set which is sent into the positioning. The point of time when the algorithm decides to send B'_j to the positioning is denoted by p_j ($j = 1, \ldots, b$, where b is the number of bursts). These sets are presented on Figure 1.

If the infrastructure nodes have different technical properties (position, accuracy) then the collected data can have different importance. Therefore we assign a weight $w_k > 0$ to the infrastructure node k which describes the importance of the data collected by the node. Without loosing generality we can assume that $w_1 = \min_{k=1}^{m} w_k$ where m is the number of infrastructure nodes.

To evaluate the algorithms first we have to define an objective function which measures their efficiency. We have to take into account both objectives of the algorithm: loose minimal amount of data, minimize the delay of starting the positioning part. 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 maximize:

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

The first part considers the total importance of the data which are sent to the positioning server, the second part is the sum of the unnecessary latencies multiplied by c, the cost of the unity latency. We assume $c > 0$.

An online algorithm for a maximization problem is ρ-competitive if the optimal gain is never more than ρ times the gain of the online algorithm.

In a more formal way: for an arbitrary online algorithm $\mathcal{A}$ and an input sequence X the gain of the solution given by $\mathcal{A}$ is denoted by $\mathcal{A}(X)$. Moreover for an input sequence X let $OPT(X)$ denote the gain of the optimal offline solution. Then an online algorithm $\mathcal{A}$ is called ρ-competitive if $OPT(X) \leq \rho \cdot \mathcal{A}(X)$ for any input sequence X.

We note that it would also be possible to define the problem as a minimization problem using an objective function which is the sum of the weighted number of the lost data elements and the latencies. But in this case the optimal offline algorithm has cost 0. This would make it impossible to use competitive analysis to study our algorithm. Therefore we decided to develop the maximization version of the objective function.

3 Competitive analysis

3.1 Analysis without constraints

First we do not introduce any further constraints on the input sequence X. For this case we show that no constant competitive algorithm exists as the following lower bound dictates.

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

Proof. Consider the following sequence. Let the first data element with burst id 1, and infrastructure node 1 arrive at time 0. If the online algorithm chooses $p_1 \geq \frac{w_1}{c}$, then the sequence is ended, and the online gain is non-positive, the offline algorithm can use $p_1^{opt} = 0$, and its gain is w_1. Therefore in this case the algorithm is not competitive. Now suppose that $p_1 < \frac{w_1}{c}$. Then $m-1$ further data elements arrive each of them having burst id 1 and infrastructure node ids $2, \ldots, m$ at time $\frac{w_1}{c}$. In this case the online algorithm has a gain of at most w_1 and the optimal offline algorithm uses $p_1^{opt} = \frac{w_1}{c}$, and is $\sum_{k=1}^{m} w_k$.
□

3.1.1 No Waiting Time Algorithm

The previous lower bound shows that in the worst case data arrives after sending the data to the positioning server. Therefore the online algorithm has no reason to wait for further data after the arrival of the first. The No Waiting Time Algorithm (NWT in short) follows this idea, it sends the first data element for each burst id into the positioning server immediately after it arrives. The competitive ratio of this algorithm is determined below.

Theorem 2 *The competitive ratio of NWT is* $\frac{1}{w_1}\sum_{k=1}^{m} w_k$.

Proof. Consider an arbitrary sequence which contains b bursts. For each burst the algorithm gains the value of the first data element, therefore its gain is at least $b \cdot w_1$. On the other hand the optimal gain cannot be more than $b \cdot \sum_{k=1}^{m} w_k$. □

Corollary 3 *NWT achieves the smallest possible competitive ratio in the general model.*

3.2 Latency limited analysis

In this subsection we consider restricted inputs better modeling the system functions. We suppose that 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. In this subsection we will assume that $\frac{w_1}{c} > d$, otherwise the lower bound proof of the general case also works in this case and we obtain that NWT is an algorithm with the smallest possible competitive ratio.

Theorem 4 *There exists no algorithm which has smaller competitive ratio than* $\min\left\{\frac{w_1}{w_1 - c \cdot d}, \frac{1}{w_1}\sum_{k=1}^{n} w_k\right\}$.

Proof. Consider the following sequence. Let the first data element with burst id 1, and infrastructure node 1 arrive at time 0. If the online algorithm chooses $p_1 \geq d$, then the sequence is ended, and the online gain is $w_1 - c \cdot p_1 \leq w_1 - c \cdot d$, the offline algorithm can use $p_1^{opt} = 0$, and its gain is w_1. Therefore in this case the algorithm is not better than $\frac{w_1}{w_1 - c \cdot d}$-competitive. Now suppose that $p_1 < d$. Then $m - 1$ further data elements arrive each having burst id 1 and infrastructure node ids $2, \ldots, m$ at time d. In this case the online algorithm has a total gain of at most w_1 and the optimal offline algorithm uses $p_1^{opt} = d$, and its gain is $\sum_{k=1}^{m} w_k$. □

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.

Theorem 5 *The competitive ratio of CWT is* $\frac{w_1}{w_1 - c \cdot d}$.

Proof. Consider an arbitrary sequence X which contains b bursts. By the limited latency constraint it follows that for each burst id the algorithm collects all data. Therefore its total gain is at least $G - b \cdot c \cdot d$, where G is the sum of the gains of all the incoming data. On the other hand the optimal gain cannot be more than G, and $G \geq b \cdot w_1$. Therefore the competitive ratio of CWT is at most

$$\frac{G}{G - b \cdot c \cdot d} \leq \frac{w_1}{w_1 - c \cdot d}.$$

□

Corollary 6 *If* $\frac{w_1}{w_1 - c \cdot d} \leq \frac{1}{w_1}\sum_{k=1}^{m} w_i$ *then CWT achieves the smallest possible competitive ratio with the latency constraint, otherwise NWT achieves the smallest possible competitive ratio.*

4 Experimental evaluation

As it is shown in Section 3 some very simple algorithm can achieve the best possible competitive ratio but its efficiency depends on the knowledge of the parameter d. In this section we first introduce a more sophisticated algorithm trying to get knowledge about d, and later present an empirical analysis which compares the algorithms in the average case.

4.1 Variable Waiting Time Algorithm

In this algorithm each burst id j has a starting time denoted by $S(j)$ which is the reception time of the first data element having burst id j. Furthermore each burst has a dataset $DS(j)$ which contains the collected data for the burst. (This will be the set B_j' after releasing it.) The algorithm uses a waiting time variable t which is the time while the algorithm waits data of burst j. The algorithm tries to learn the best value of this variable. Algorithms which use similar parameter learning idea are presented for the online data acknowledgment problem and for the scheduling problem with rejection in [5] and [6]. The algorithm also uses two other parameters DEC and INC, the first gives a lower bound on the possible change of the variable W in the descending direction, INC is a security distance which gets added additionally if the value of W has to be increased. MEM is a parameter which determines how long the algorithm should keep record about recent bursts.

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 $W := t - S(j) + INC$.

4.2 Empirical analysis

To analyse the algorithms in average case we used the simulation tool developed at the Fraunhofer Institute. In this simulation tool a virtual object with a tag is moving on a configurable trajectory. It generates RTT and AoA data for 3 Goniometers (collecting both RTT and AoA data) and 8 anchors (collecting only RTT data). The following properties of the system are determined randomly in our simulation:

- each data element sent by the virtual infrastructure nodes is lost with a given probability (it is defined separately for the RTT and AoA data)
- the length of the time span between two bursts and between the data elements of a burst are both normally distributed.

We generated the following 6 inputs. In tests A and B the number of bursts was 68 and 262 and during these bursts 952 and 3668 data elements arrived, the average length of the bursts was 2.04 and 2.6. In tests C and D we studied slightly shorter bursts, the average length was 0.99 and 1.06. In test C we used 131 bursts with 1800 data elements, in test D 138 bursts was used with 1932 data elements. Finally in tests E and F much longer bursts were used with average length 34.07 and 35.9. Test E was smaller it contained 127 bursts with 1742 incoming data elements, test F was large it contained 879 bursts with 12079 data elements.

We considered 3 constant waiting time algorithms for the solution of the problem. CWT(1) used a smaller constant which is close to the average burst lengths of C and D, CWT(2) used a larger constant which was still smaller then the average length in tests E and F, and $CWT(3)$ used a constant which was close to the average lengths in tests E and F. In Table 1 we collected the ratios which were received by dividing the gain of the algorithm by the optimal gain. We used the values $w_i = 0.07$ for each i and $c = 0.01$ in the objective function.

	Test A	Test B	Test C	Test D	Test E	Test F
CWT(1)	0.930	0.883	0.967	0.964	0.142	0.194
CWT(2)	0.730	0.736	0.715	0.721	0.794	0.759
CWT(3)	0.660	0.666	0.643	0.651	0.976	0.936
VWT	0.976	0.985	0.978	0.984	0.985	0.914

Table 1: The test results

Based on the above results we can draw the following conclusions:

- It is clear that the efficiency of the CWT algorithms depends highly on the average length of the bursts. CTW(1) gives good results for tests A, B, C, D but has extremely poor performance on tests E and F. CWT(2) and CWT(3) gave good results on tests E and F but their results were weak on the other tests. This means that these algorithms can be used only in the cases where we have some a priori information about the data. But we note that in these cases they work well: their performance is better than 0.9.

- The VWT algorithm performs also well with unknown average burst-lengths, it delivers good results in each testcase. The minimal ratio of the algorithms to the optimal gain is 0.914, the average ratio is 0.97.

5 Summary and open questions

In this paper we defined an online optimization model for the clustering problem which appears in real time location systems. We presented optimal online algorithms in the sense that they achieve the smallest possible competitive ratio and a more sophisticated algorithm which is designed to learn the average length of the bursts. We showed by an empirical analysis that this learning algorithm is useful if we have no a priori information about the data.

There are some further interesting questions related to our problem. It would be interesting to study other, more difficult objective functions. Furthermore, in this model we supposed that at the arrival of the data element we receive the id of the burst where the data element was sent. In some application we do not receive this information, it would be interesting to study this scenario as well.

Acknowledgement

This research was partially supported by the TÁMOP-4.2.1/B-09/1/KONV-2010-0005 program of the Hungarian National Development Agency. Cs. Imreh was supported by the Alexander von Humboldt Foundation, completed most of this work at the Humboldt University at Berlin, and is grateful to Professor Suzanne Albers for the kind hospitality. Tamás Németh completed some of this work while visiting the Fraunhofer Institute in Nuremberg, and is grateful for the kind hospitality.

References

[1] S. Albers,H. Bals, Dynamic TCP acknowledgement: Penalizing long delays, *SIAM J. Discrete Math.* **19,** 4 (2005) 938–951. ⇒7

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

[3] M. Brugger, F. Kemeth, Locating rate adaptation by evaluating movement specific parameters, *Proc. 2010 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, Anaheim, USA, 2010, pp. 127–133. ⇒6

[4] D. R. Dooly, S. A. Goldman, S. D. Scott, On-line analysis of the TCP acknowledgment delay problem, *J. ACM* **48,** 2 (2001) 243–273. ⇒7

[5] Cs. Imreh, T. Németh, Parameter learning algorithm for the online data acknowledgment problem, *Optim. Methods Softw.*, **26,** 3 (2011) 397–404. ⇒12

[6] T. Németh, Cs. Imreh, Parameter learning online algorithm for multiprocessor scheduling with rejection, *Acta Cybernet.*, **19,** 1 (2009) 125–133. ⇒12

Received: March 8, 2013 • Revised: May 14, 2013