DIAS: An Adaptive Push System, based on Distributed Learning Automata

V. Kakali, P. Sarigiannidis, G. I. Papadimitriou, Senior Member IEEE, and A. S. Pomportsis
Department of Informatics, Aristotle University of Thessaloniki
Box 888, 54124 Thessaloniki, Greece
e-mail: vicky198@csd.auth.gr

Abstract
In the field of data broadcasting and delivery information, push systems have appeared able to provide high scalability and client hardware simplicity in wireless environments. Moreover, learning automata may aim in providing adaptive decisions for an environment with unknown client demands. The proposed scheme is composed of distributed learning automata, which provide online estimations regarding the client demands. Clients are organized in groups and may request a set of items. The server side has to choose one item per broadcast. A learning automaton is applied on each group, trying to estimate the clients' demands for each group. In this way, the server side forms a view of the demands of each group and broadcasts items according to the estimated clients' demands. Simulation results show that the novel technique presents lower mean response time than a previous automata-based scheme, meaning that each client is satisfied more regularly.

1. Introduction

Data broadcasting is an efficient way of delivering information over wireless networks (e.g. weather information and news distribution) with pull [1], push [2,3,4] and hybrid [5] being the major approaches of designing broadcast schedules [6]. Moreover, most applications are characterized by locality of demand. Locality of demand means that clients are gathered into groups, each one located at a different region and members of each group have similar demands for information items, different from the demands of clients at other groups (e.g. traffic information).In this paper, the push approach is selected as the main schedule environment. In "pure" push systems (e.g. [2]), the server is assumed to have an a-priori estimate of the demand per information item and broadcasts according to these estimates. Moreover, the adaptive push system of [3] extends the applicability of the push approach to environments characterized by a-priori unknown and dynamic client demands using a learning automaton [7] at the broadcast server in order to provide adaptivity. The adaptive approach presents results that reveal efficient operation in such environments.

The proposed scheme alters the operation of designing the broadcast schedules by applying a distributed learning-automata scheme instead of a centralised one. Moreover, the broadcasts are organised in three phases, named the learning phase, the calculation phase and the schedule phase. The DIstributed Automata Scheme (DIAS) framework targets on accurate estimations regarding the clients' demands. In this manner, it takes into account the locality of item demand per group. This means that the server side tries to determine the

clients demand per group by applying a learning automaton on each group of clients, contrary to a unique centralized learning automaton for the whole system.

The remainder of the paper is organised as follows: Section 2 presents the proposed scheme and simulation results are presented in Section 3. Finally, Section 4 concludes the paper.

2. The DIstributed Automata Scheme (DIAS)

A. The Learning phase

Each Learning Automaton contains the server's estimate $p'_{i,g}$ of the demand probability $p_{i,g}$ for each item i among the set of the items that the server broadcasts for the specific group g .Thus, for each group g , it holds $\sum_{i=1}^{Num} p'_{i,g} = 1$, where Num is the number of items that refers to each group.

During the learning phase, the broadcast follows a round robin manner sending each item once.

After the transmission of item i of group g , each satisfied client transmits a feedback (e.g. one bit), using Code Division Multiple Access (CDMA).

After the reception and decoding of the feedbacks, the server updates the distributed demand probability vector p'_g according to the probability updating scheme that is described below.

When there are satisfied clients, the probability of the transmitted item i is increased while the probabilities of all the rest items are decreased. Considering that the server's k^{th} transmission is item i of group g , the probability vector p'_g is updated according to equation (1):

$$\begin{aligned} p'_{j,g}(k+1) &= p'_{j,g}(k) - L(1-\beta_g(k))\,(p'_{j,g}(k) - a),\ \forall \mathrm{j} \neq \mathrm{i} \\ p'_{i,g}(k+1) &= p'_{i,g}(k) + L(1-\beta_g(k)) \sum_{i \neq j} (p'_{j,g}(k) - a) \end{aligned} \qquad (1)$$

where $p'_{i,g}(k) \in (a,1),\ \forall\ i \in [1,...,Num]$, $L, a \in (0,1)$, $\beta_g(k) = 1 - \dfrac{number\ of\ satisfied\ clients\ of\ g}{number\ of\ clients\ of\ g}$.

The server is aware of the population of each group g using the following mechanism: The server, which has a priori knowledge of the geographic locations (co-ordinates) where his services are offered, sends one control packet for each one of these locations, asking of the clients that belong to this location to send back a feedback. Considering clients equipped with GPS receivers (e.g. PDAs), the client's GPS receiver detects its co-ordinates and the client sends its feedback as an answer to the control packet that refers to its area. L is a parameter that governs the speed of the automaton convergence and parameter a prevents

the probabilities of unpopular items from taking values around zero. A value of $\beta_g(k)$ that equals one represents the case where no client feedback is received.

At the end of the learning phase, the value of the mean number of feedbacks per group is stored in a F vector in order to be used for the calculation phase.

B. The calculation phase

The broadcasts of the next phase are formed based on the formula below:

$S_g = \frac{F_g}{\sum_{i=1}^{G} F_g} \cdot G \cdot Num$ **,**where S_g is the number of the future broadcasts dedicated to each group , G is the number of system's groups and Num is the number of items that refers to each group .

C. The schedule phase

According to S vector, a number of broadcasts are scheduled equal to $sum(S)$ as follows:

while (sum(S)>0)
maximum = max(S)
selected group = the index of maximum in vector S
broadcast to the "selected group", the item i that maximizes the cost function presented below
update $p'_{selected\ group}$
S(selected group) = S(selected group) – 1
end while

The server selects to broadcast to the "selected group" the item i that maximizes the cost function $CF(i)$ [2]

$$CF(i) = (T - R(i))^2 \frac{p'_{i,g}}{l_i} \frac{(1+E(l_i))}{(1-E(l_i))},\ 1 \le i \le Num \qquad (2)$$

where T is the current time, $R(i)$ is the time when item i was last broadcast, $p'_{i,g}$ is the server's estimated demand probability for item i, l_i is the item's length, $E(l_i)$ is the probability than an item of length l_i is received with an unrecoverable error and Num is the number of the data items of group g . For items that have not been previously broadcast $R(i)$ is initialized to -1 .

3. The Simulation Environment

A client population of $NumCl$ is considered. Clients are gathered into G groups each one of which is located at a different location. In order to model groups with different group sizes, we compute the size of each group via the Zipf distribution.

$$c\left(\frac{1}{g}\right)^{\theta}, \quad \text{where } c = \frac{1}{\sum_k \left(\frac{1}{k}\right)^{\theta}},\ k \in [1..G] \qquad (3)$$

where $1 \le g \le G$, θ is a parameter named access skew coefficient. For $\theta = 0$, the Zipf distribution reduces to a uniform distribution of group sizes. For large values of θ, the Zipf distribution produces increasingly skewed patterns.

Any client belonging to group g is interested in the same subset D_g of server's data items. All items outside this subset have a zero demand probability at the clients of the group. Moreover, $D_i \neq D_j, \forall i, j \in [1..G],\ i \neq j$. Each one of the D_g subsets comprises of Num items and the client's demand probability $p_{i,g}$ for each item in place i in that subset is computed via the above mentioned Zipf distribution. The item length, l, is considered to be the same for each item and equal to the unit, $l = 1$.

Also, a server equipped with one omni-directional antenna and a number of G (one for each of the groups) learning automata are assumed. Each learning automaton contains its probability estimate of the Num equally-sized items that concern each group (geographic location). Finally, $E = 1 - e^{-\lambda}$ is the probability that an item is received with an unrecoverable error (Poisson process).

The simulation runs until each server broadcasts N items. The overhead due to the duration of the feedback pulses and the signal propagation delay is defined via the parameter Ouh. The simulation results presented in this section are obtained with the following values to the parameters: $NumCl = 1000$, $N = 100000$, $Ouh = 10^{-3}$, $\lambda = 0.1$, $L = 0.15$, and $\alpha = 10^{-4}$.

Network	G	Num	θ
N_1	$\in$ [5...20]	5	0.9
N_2	10	$\in$ [5...20]	0.9
N_3	20	5	$\in$ [0.0 ... 1.0]

Table 1. The characteristics of the different simulation environments.

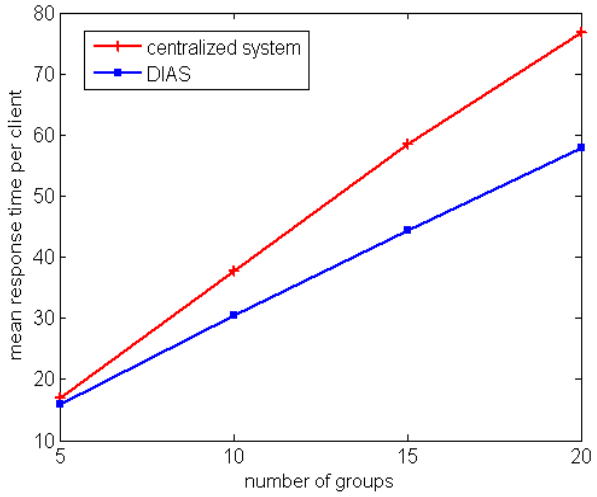

Fig.1. The mean response time versus number of groups for network N_1.

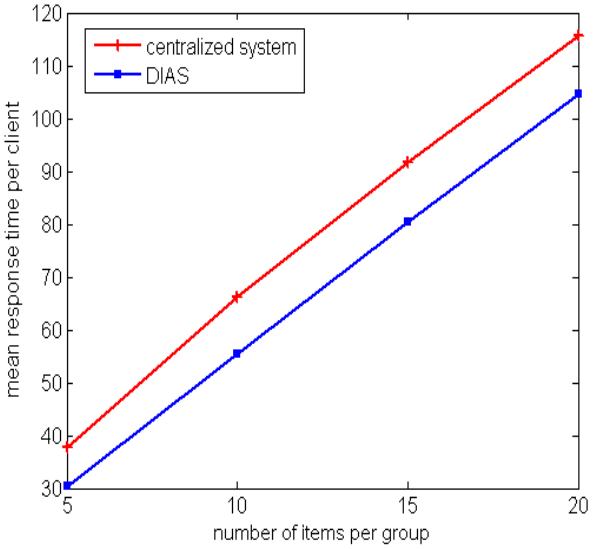

Fig.2. The mean response time versus number of data items per group for network N_2.

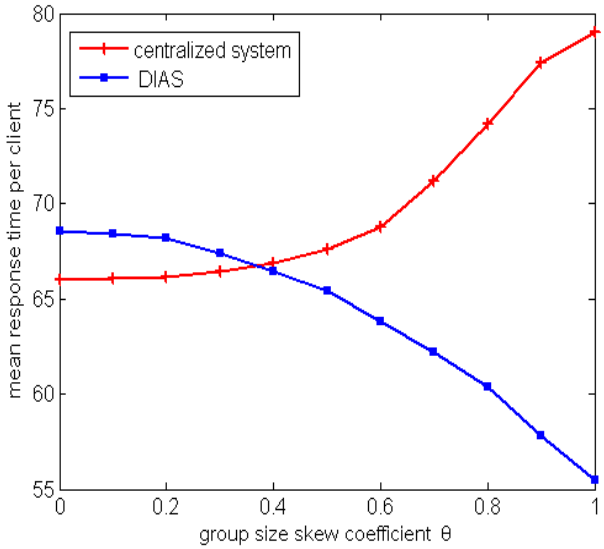

Fig.3. The mean response time versus group size skew coefficient θ for network N_3.

Figures 1-3 display the mean response time of the networks N_1, N_2 , N_3, versus different number of groups, different number of items per group and different values of size skew coefficient θ, respectively. In each of the figures, we compare the proposed DIAS scheme to the centralized one. Fig.1, depicts clearly the increased performance that the proposed distributed system achieves against the centralized one as the number of groups of the clients is increased. Due to the large number of groups, the system of the centralized automaton is unable to adapt efficiently to the many unlike demands compared to the DIAS where each automaton is dedicated to the specific demands of its group. Fig. 2 confirms the superiority of the proposed DIAS for various numbers of items per group. Fig. 3 reveals the performance that DIAS achieves against the centralized system as the population of the groups of clients is getting more unequal (the group size skew coefficient θ is increased). This is presumed as the unique centralized automaton is unable to make accurate estimations and thus to adapt efficiently in a such unequal environment (large values of θ). On the other hand, the DIAS framework achieves accurate estimations regarding the clients' demands using a dedicated automaton for each group that is updated separately by taking into account the locality of item demand per group.

4. Conclusion

A novel push system was presented that adopts a distributed learning automaton operation. The core idea is based on distributed automata, which operate separately for each

group of clients. The distributed broadcast scheduling manner serves more efficiently, in terms of mean response time, than the centralized scheduling system. This improvement leads to more satisfied clients for the same number of broadcasts than the centralized one.

References

[1] D.Aksou and M.Frankin, "RxW: A Schedualing Approach for Large-Scale On-Demand Data Broadcast", ACM/IEEE Transactions on Networking vol.7, no.6, pp.846-860, December 1999.

[2] N.H.Vaidya and S.Hameed, "Scheduling Data Broadcast In Asymmetric Communication Environments", Wireless Networks, vol.2, no.3, pp.171-182, May 1999.

[3] P. Nicopolitidis, G. I. Papadimitriou and A. S. Pomportsis, " Using Learning Automata for Adaptive Push - Based Data Broadcasting in Asymmetric Wireless Environments ", IEEE Transactions on Vehicular Technology, vol.51, no.6, pp.1652-1660, November 2002.

[4] P. Nicopolitidis, G. I. Papadimitriou and A. S. Pomportsis, "Multiple Antenna Data Broadcasting for Environments with Locality of Demand", IEEE Transactions on Vehicular Technology, vol.56, no.5, pp.2807-2816, September 2007.

[5] K. Stathatos, N. Roussopoulos and J. S. Baras, "Adaptive Broadcasts in Hybrid Networks", In Proceeding of VLDB 1997, pp.326-335.

[6] Shyam Kapadia, Bhaskar Krishnamachari, "Distributed Computing in Sensor Systems", chapter title "Comparative Analysis of Push-Pull Query Strategies for Wireless Sensor Networks", pp. 185-201, Springer Berlin / Heidelberg, 2006.

[7] M. A. L. Thathachar, P. S. Sastry, "NETWORKS OF LEARNING AUTOMATA, Techniques for Online Stochastic Optimization", Kluwer Academic Publishers, 2004.