Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 30 (2007) 957–964
computer communications
www.elsevier.com/locate/comcom

A new high rate adaptive wireless data dissemination scheme

P. Nicopolitidis [a], G.I. Papadimitriou [a], M.S. Obaidat [b,*], A.S. Pomportsis [a]

[a] Department of Informatics, Aristotle University, Box 888, 54124 Thessaloniki, Greece
[b] Department of Computer Science, Monmouth University, West Long Branch, NJ 07764, USA

Available online 18 September 2006

Abstract

With the increasing popularity of wireless networks and mobile computing, data broadcasting has emerged as an efficient way of delivering data to mobile clients having a high degree of commonality in their demand patterns. This paper proposes a push system that continuously adapts to the demand pattern of the client population in order to reflect the overall popularity of each data item. The adaptation is accomplished using a simple feedback from the clients. We propose that the simple feedback is sent only from clients whose distance from the server does not incur a significant timing overhead for the acknowledgment of an item. Simulation results are presented which reveal satisfactory performance in high-speed environments with a priori unknown and dynamic client demands.

Keywords: Adaptive data broadcasting; Learning automata; Propagation delay

1. Introduction

Data broadcasting has emerged as an efficient way for the dissemination of information over asymmetric wireless networks [1]. Examples of data broadcasting are information retrieval applications, like traffic information systems, weather information and news distribution. In such applications, client needs for data items are usually overlapping. As a result, broadcasting stands to be an efficient solution, since the broadcast of a single information item is likely to satisfy a (possibly large) number of client requests.

Communications asymmetry is due to a number of facts:

- *Equipment asymmetry.* A broadcast server usually has powerful transmitters that are not subject to power limitations, whereas client transmitters are usually hindered due to finite battery life. Moreover, it is desirable to keep the mobile clients' cost low, a fact that sometimes results to lack of client transmission capability.
- *Network asymmetry.* In many cases, the available bandwidth for transmission from the server to the clients (downlink transmission) is much more than that in the opposite direction (uplink transmission). Furthermore, there exist extreme network asymmetry cases, where the clients have no available uplink channel (backchannel). In the case of backchannel existence however, the latter is subject to becoming a bottleneck in the presence of a very large client population.
- *Application asymmetry.* This is due to the pattern of information flow. Information retrieval applications are of client–server nature. This means that the flow of traffic from the server to the clients is usually much higher than that in the opposite direction.

The goal pursued in most data delivery proposed approaches is twofold: (a) Determination of an efficient sequence for data item transmissions (broadcast schedule) so that the average time a client waits for an item (mean access time) is minimized and (b) management of the mobile clients' local memory (cache) in a way that efficiently reduces a client's performance degradation when mismatches occur between the client's demands and the server's schedule. This paper focuses on minimization of mean access time over the

* Corresponding author.
E-mail address: obaidat@monouth.edu (M.S. Obaidat).

0140-3664/$ - see front matter
doi:10.1016/j.comcom.2006.08.028

entire client population (overall mean access time) under a priori unknown, dynamic client demands in environments with high transmission speeds.

So far, three major approaches have appeared for designing broadcast schedules. These are:

- The pull-based (also known as on-demand) approach (e.g. [2]). In pull-based systems the server broadcasts information after requests made by the mobile clients via the uplink channel. The server queues up the incoming requests and uses them to estimate the demand probability per-data item. This approach has the advantage of being able to adapt to dynamic client demands, since the server possesses knowledge regarding the demands of the clients. However, it is inefficient from the point of view of scalability. When the client population becomes too large, requests will either collide with each other or saturate the server.
- The push-based approach (e.g. [3,4]). In push-based systems there is no interaction between the server and the mobile clients. The server is assumed to have an a priori estimate of the demand per-information item and transmits data according to this estimate. This approach provides high scalability and client hardware simplicity since the latter does not need to include data packet transmission capability. However, it pays the price of being unable to operate efficiently in environments with dynamic client demands.
- Hybrid approaches (e.g., [5,6]). Hybrid systems employ a combination of push and pull dividing the available downlink bandwidth into two different transmission modes: The periodic broadcast mode, in which the server pushes data periodically to the clients and the on-demand mode which is used to broadcast data explicitly requested by the mobile clients through the uplink channel. Obviously, this approach tries to combine the benefits of pure-push and pure-pull systems.

Most research on push-based systems assumed a priori and static knowledge of the demand pattern. However, today's information retrieval applications are characterized by demand patterns that are likely to be unknown and change with time. Consider a hypothetical scenario in an airport [7]. Users coming to the airport will want information regarding their flight (e.g., exact hour of departure, possible delays, etc). A broadcast server should deliver data according to client demand. For a specific flight, the demand is likely to be in its peak a couple of hours before the flight departure. For example, if our flight departs at 6 PM, early in the day the demand will be very small, as few passengers are likely to come to the airport 5 or 6 h before their flight. At this time, the server should increase the frequency of data items concerning flights leaving in the near future. As the time for the departure approaches, the demand for information regarding our flight will grow due to the increasing number of waiting passengers, and eventually, after a few minutes of the departure of the flight, it will drop again. It can be easily seen that in such an environment the server needs to broadcast information according to the state of the client demand, which is neither a priori known, nor is it static.

This paper enhances the adaptive push-based system of [8] to enable efficient operation in high-speed data broadcasting environments. It suggests the use of a Learning Automaton at the server, which continuously adapts to the demand pattern of the client population in order to reflect the overall popularity of each data item. The adaptation is accomplished using a simple feedback from the clients. Using this approach, information items are transmitted according to client demands, which can be initially unknown to the server and time varying. Contrary to [8], we propose that the simple feedback is sent only from clients whose distance from the server does not incur a significant timing overhead for the acknowledgment of an item. This acknowledgement has the form of a short feedback pulse. The acknowledging nodes' pulses add at the server that uses the received energy to update the Automaton. The item probabilities estimated by the Automaton converge near the actual overall item demand probabilities of the client population, making this approach attractive for dissemination applications with dynamic demand patterns.

The remainder of this paper is organized as follows: Section 2 presents the proposed system. Section 3 presents simulation results, which reveal satisfactory performance in high-speed environments with a priori unknown and dynamic client demands. Finally, Section 4 summarizes and concludes the paper.

2. The adaptive wireless data broadcasting scheme

Learning Automata ([9–11]) are structures that can acquire knowledge regarding the behavior of the environment in which they operate. In the area of data networking Learning Automata have been applied to several problems, including the design of self-adaptive MAC protocols ([12–15]).

In the fixed rate adaptive wireless push system [8], which enhanced the non-adaptive one of [4], the server is equipped with an S-model Learning Automaton that contains the server's estimate p_i of the demand probability d_i for each data item i among the set of the items the server broadcasts. Clearly, $\sum_{i=1}^{N} d_i = 1$, where N is the number of items in the server's database. At each cycle, the server selects to transmit the item i that maximizes the cost function $G(i) = (T - R(i))^2 \frac{d_i}{l_i}\left(\frac{1+E(l_i)}{1-E(l_i)}\right)$, $1 \leqslant i \leqslant N$, where T is the current time, $R(i)$ the time when item i was last broadcast, l_i is the length of item i and $E(l_i)$ is the probability that an item of length l_i is erroneously received. For items that haven't been previously broadcast, $R(i)$ is initialized to -1. If the maximum value of $G(i)$ is shared by more than one item, the algorithm selects one of them arbitrarily. Upon the broadcast of item i at time T, $R(i)$ is changed so that $R(i) = T$.

After the transmission of item i, the broadcast server awaits for an acknowledging pulse from every client that was waiting item i. The aggregate received pulse power is used at the server to update the Automaton. The probability distribution vector p maintained by the Automaton estimates the demand probability d_i (and thus the popularity) of each information item i. For the next broadcast, the server chooses which item to transmit by using the updated vector p.

When the transmission of an item i does not satisfy any waiting client, the probabilities of the items do not change. However, following a transmission that satisfies clients, the probability of item i is increased. The following Liner Reward-Inaction (L_{R-I}) probability updating scheme [9] is employed after the transmission of item i (assuming it is the server's kth transmission).

$$p_j(k+1) = p_j(k) - L(1-b(k))(p_j(k) - a), \quad \forall j \neq i$$
$$p_i(k+1) = p_i(k) + L(1-b(k))\sum_{i \neq j}(p_j(k) - a) \tag{1}$$

where $p_i(k)$ takes values in $(a,1)$ and L, a take values in $(0,1)$. The role of parameter is to prevent the probabilities of non-popular items from taking values very close to zero in order to increase the adaptivity of the Automaton. This is because if the probability estimate p_i of an item i approaches zero, then $G(i)$ will take a value very close to zero. However, item i, even if unpopular, still needs to be transmitted since some clients may request it. Furthermore, the dynamic nature of client demands might make this item popular in the future. $b(k)$ represents the environmental response after the server's kth transmission. Upon reception of the sum of the acknowledging client pulses, this sum is normalized in the interval $[0,1]$. A value of $b(k)$ that equals 1 represents the case where no client acknowledgment is received. Thus, the lower the value of $b(k)$, the more clients were satisfied by the server's kth transmission.

Using the probability updating scheme of (1), the item probabilities estimated by the Automaton converge near the actual demand probabilities for each item. This makes this approach attractive for dissemination applications with dynamic client demands. This convergence is schematically shown in Fig. 1, which plots the convergence of an item probability estimate towards the actual overall demand probability for that item in a simulation of a sample scenario with a priori unknown client demands that change after some time from about 0.1 to about 0.5. It is evident that convergence of the Automaton item probability estimate to the overall client demand for this item is achieved.

The probability updating scheme of Eq. (1) is of O(N) complexity. Therefore the adaptive push method does not increase the computational complexity of the static one The bucketing described in [4] would not further reduce computational complexity in the adaptive method, as complexity would remain of O(N) due to the probability updating scheme.

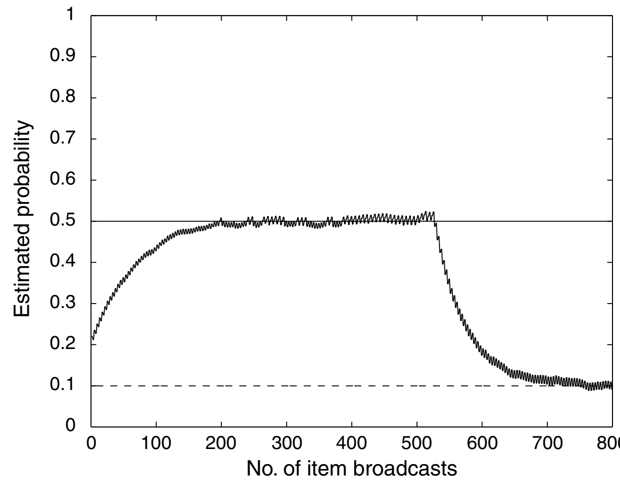

Fig. 1. Convergence of automaton estimation of the demand of an item.

However, the above-described system will suffer from a performance decrease in environments where either the coverage area (and thus the maximum propagation delay) is large, or the transmission speed is relatively high. Both these conditions translate into an increased *Max_Pr_delay*/*Tr_time* ratio, where *Max_Pr_delay* is the maximum one-way propagation delay in the system (thus the propagation delay from the server's antenna to a client located at the border of the service area) and *Tr_time* is the duration of an item's transmission. The significance of the increased *Max_Pr_delay*/*Tr_time* is because the server is able to proceed to the broadcast of item $i+1$ only after a time period *Tr_time* + 2∗*Max_Pr_delay* + *Feedback_time* has elapsed since the initiation of the broadcast of item i, so as to wait for possible reception of pulses from clients at the border of the service area of the system. *Feedback_time* is the transmission duration of the short feedback pulse and is considered very small. Thus, it can be seen that when the *Max_Pr_delay*/*Tr_time* ratio is large the server will experience a significant timing overhead for the acknowledgment of each item. To this end, we propose that feedback is sent only from clients located inside a circular area *ACK_Area* with the server's antenna at the circle's center. Every other client that is located outside *ACK_Area* will never acknowledge an item unless the client moves inside *ACK_Area* as explained later. The radius of *ACK_Area* corresponds to a maximum propagation delay *Pr_delay*, which is very small, compared to *Tr_time*.

The server notifies the clients that acknowledgements can be transmitted only by those that are inside *ACK_Area* by periodically transmitting via a control item both the radius of *ACK_Area* and the power at which it transmits all items. By measuring the power of reception of the control item and using the information regarding the power at which the server transmitted the item, every client can find if is current location is inside *ACK_Area* and thus if it will acknowledge data item receptions by feedback pulses.

The normalization procedure in the calculation of $b(k)$ suggests the existence of a procedure to enable the server to possess an estimate of the number of clients inside *ACK_Area*. This is made possible by broadcasting a control packet that forces every client in *ACK_Area* to respond with a power-controlled feedback pulse. The broadcast server will use the power of the received pulses S to estimate the number of clients in *ACK_Area*. Then, upon reception of an aggregate feedback pulse of power Z after the server's kth broadcast, $b(k)$ is calculated as Z/S. This estimation process enables newly arriving clients in *ACK_Area* to acknowledge item receptions after they have responded to the estimation process. It will take place at regular time intervals with the negligible overhead of broadcasting a unit-length item.

As the signal strength of each client's pulse at the server suffers a $1/d^n$ type path loss (with a typical $n = 4$ [16]), the feedback pulses of clients must be power controlled. To this end, every information item will be broadcast including information regarding the signal strength used for its transmission and acknowledging clients set the power of their feedback pulse to be the inverse of the ratio (signal strength of the received item)/(signal strength of the item transmission). Using this form of power control, the contribution of each client's feedback pulse at the server will be the same regardless of the client's distance from the antenna.

The estimation of the number of clients inside *ACK_Area* at regular time intervals means that between two consecutive estimations, the broadcast server may for some time possess imprecise knowledge regarding the number of clients within *ACK_Area*. For this issue, we argue that clients arriving inside *ACK_Area* are allowed to send feedback for received items only after they have joined the process for the estimation of the number of clients within *ACK_Area*. This can be considered as a kind of registration that aims to maintain the precision of the environmental response in cases of increasing number of clients between consecutive estimations. On the other hand, a decreasing number of clients between consecutive estimations leads to a lower value of $\beta(k)$ and thus $L\beta(k)$. Consequently, this can be seen as a temporary reduction of the adaptation parameter L which will lead to slower convergence of the automaton.

3. Performance evaluation

We used simulation in order to assess the performance increase offered by the proposed system (denoted by *Adaptive Push – variable ACK area* in the Figures) to the adaptive push one of [8] (denoted by *Adaptive Push* in the Figures). In the adaptive push system of [8] every client that receives an item, irrespective of the client's distance from the server, must acknowledge it via a feedback pulse. In the proposed system, we set the server to request acknowledgements only from clients which are inside an *ACK_Area* such that the ratio *Pr_delay*/*Tr_time* is less than D, where *Pr_delay* is the maximum propagation delay in the *ACK_Area* from the client to the server antenna and *Tr_time* is the item transmission time. In both systems the duration of the feedback pulse transmission is considered to be very small in comparison to *Tr_time*.

3.1. Server model

We consider a broadcast server having a database of N equally sized items. The server is initially unaware of the demand for each item, so initially every item has a probability estimate of $1/N$. Client demands are a priori unknown to the server. Item broadcasts are subject to reception errors, with unrecoverable errors per-instance of an item occurring according to a Poisson process with rate λ, as in [4].

3.2. Client model

We consider *ClNum* clients that have no cache memory, an assumption also made in other similar research (e.g. [3,4,8]). Every client accesses items in the interval [1, *Range*], which can be a subset of the items that are broadcast. All items outside this range have a zero demand probability at the client. This item range consists of an integral number of R regions of size *RSize* items. Items inside the same region are demanded with the same probability of

$$d_i = c\left(\frac{1}{i}\right)^{\theta}, \text{ where } c = 1/\sum_k \left(\frac{1}{k}\right)^{\theta}, \quad k \in [1, \ldots, R] \qquad (2)$$

and θ is a parameter named access skew coefficient. This is the Zipf distribution used in modeling of client demands in other papers as well ([8,4,17,18]). For $\theta = 0$, the Zipf distribution reduces to the uniform distribution. As the value of θ increases, the Zipf distribution produces increasingly skewed demand patterns. The Zipf distribution can thus efficiently model applications that are characterized by a certain amount of commonality in client demand patterns. For a specific client with $Range = 200$ and $RSize = 1$, the demand probabilities per item for different values of θ are shown in Fig. 2.

To simulate some "noise" in client demands, we introduce parameters *Dev* and *Noise*. These parameters determine the percentage of clients that deviate from the initial demand pattern described above and the degree of that deviation, respectively. For every client, a coin toss, weighted by *Dev*, is made. If the outcome of the toss states that the client is to deviate from the initial demand pattern, then a new demand pattern for this client is generated. This pattern is produced in the following way: with probability *Noise* the demand probability of each item in the client's demand pattern database is swapped with that of another item that is selected in uniform manner from the interval $[1, \ldots, N]$.

Client placement takes place on *LP* different circles, which are equally separated and are located outside the antenna's near field, with the circle at the maximum

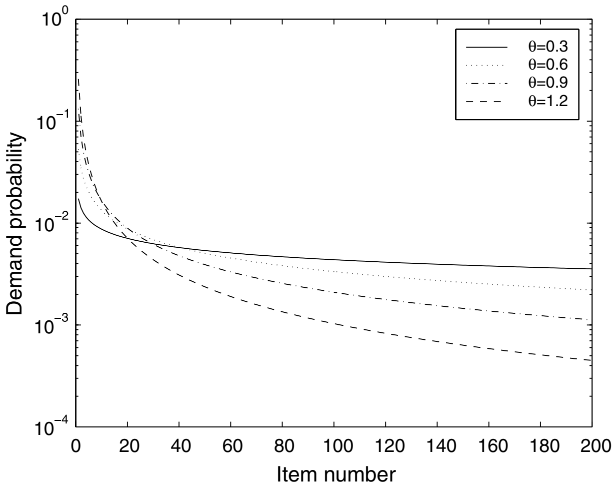

Fig. 2. Demand probability produced by the Zipf distribution for different values of θ.

distance corresponding to the coverage border of the system. The probability of a client to be located on a certain circle is proportional to the perimeter of the circle. Client placement is independent of the client demand patterns.

3.3. The simulation environment

The simulation is carried out until at least *Num* requests are satisfied at each client, meaning that overall, at least *Num*∗*ClNum* requests have been served. Figs. 3–10 display the results of certain simulation experiments for various values of *Dev* and θ. The results were obtained with the following parameter values being constant:

- $N = 200$.
- $ClNum = 10000$.
- $Num = 1000$.
- $D = 0.1$.
- $Noise = 0.5$.

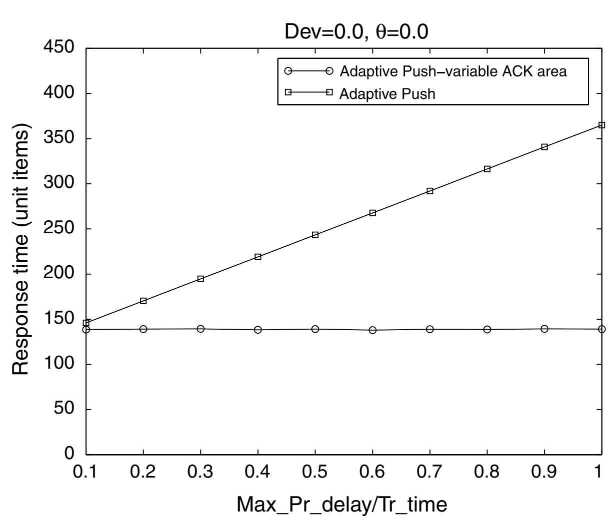

Fig. 3. Response time versus *Max_Pr_delay*/Tr_time ratio. $Dev = 0.0$, $\theta = 0.0$.

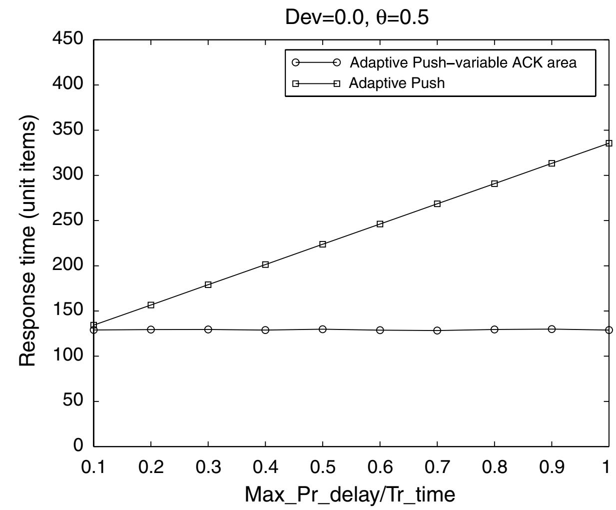

Fig. 4. Response time versus *Max_Pr_delay*/*Tr_time* ratio. $Dev = 0.0$, $\theta = 0.5$.

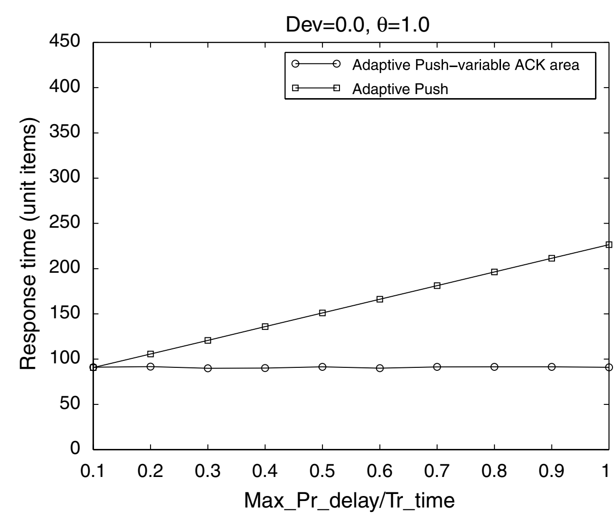

Fig. 5. Response time versus *Max_Pr_delay*/*Tr_time* ratio. $Dev = 0.0$, $\theta = 1.0$.

- $L = 0.15$.
- $a = 10\text{-}4$.
- $\lambda = 0.1$.
- $LP = 100$.
- $Range = 200$.
- $RSize = 1$.

In each Figure, two plots are presented that compare the performances (Overall Mean Access Times, also known as Response Times) of the proposed adaptive push that utilizes the variable Acknowledging area and the adaptive push one of [8] for various values of the ratio *Max_Pr_delay*/*Tr_time*.

3.4. Simulation results

The main conclusions that can be drawn from Figs. 3–10 are the following:

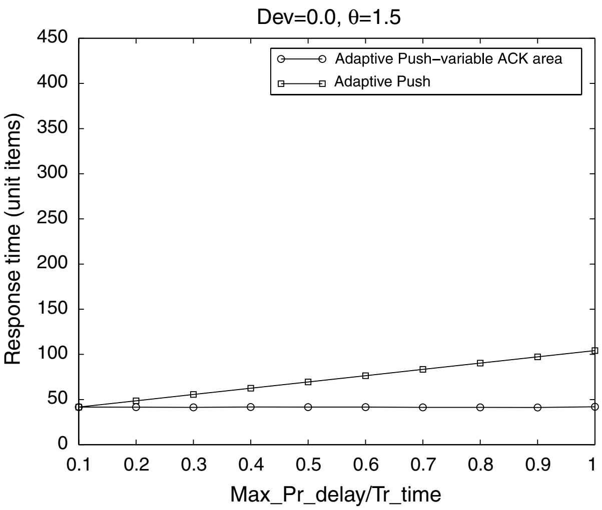

Fig. 6. Response time versus *Max_Pr_delay*/*Tr_time* ratio. $Dev = 0.0$, $\theta = 1.5$.

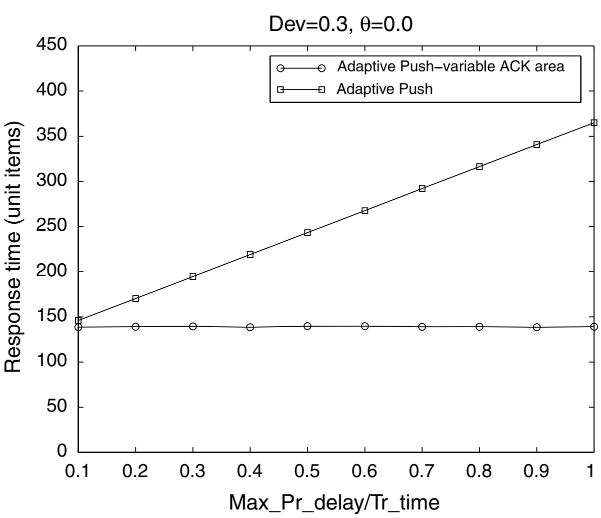

Fig. 7. Response time versus *Max_Pr_delay*/*Tr_time* ratio. $Dev = 0.3$, $\theta = 0.0$.

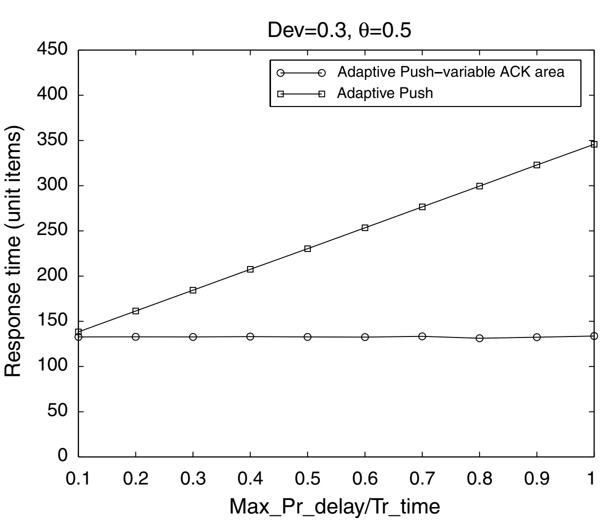

Fig. 8. Response time versus *Max_Pr_delay*/*Tr_time* ratio. $Dev = 0.3$, $\theta = 0.5$.

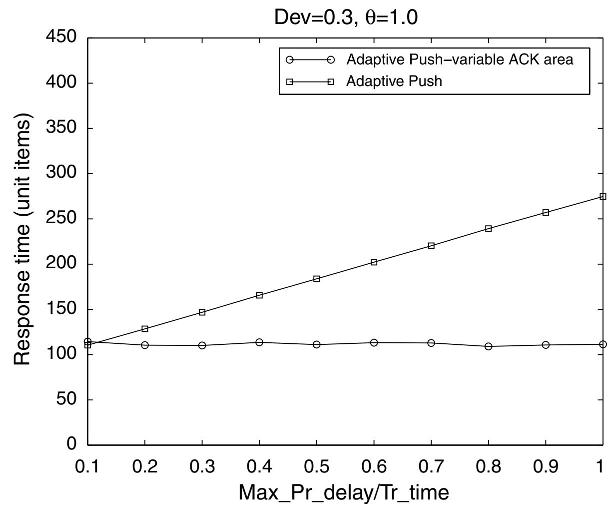

Fig. 9. Response time versus *Max_Pr_delay*/*Tr_time* ratio. $Dev = 0.3$, $\theta = 1.0$.

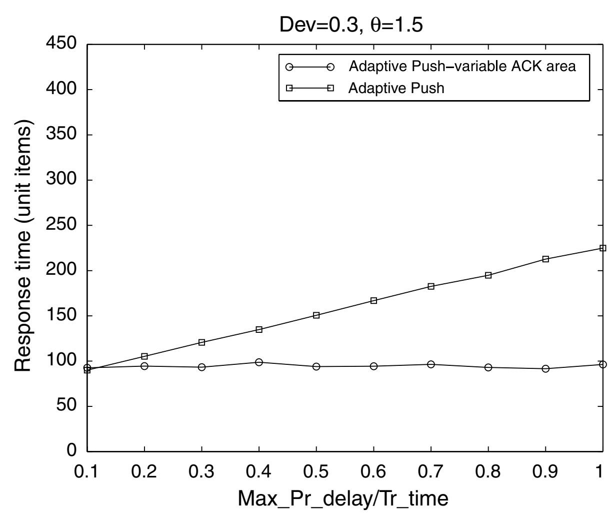

Fig. 10. Response time versus *Max_Pr_delay*/*Tr_time* ratio. $Dev = 0.3$, $\theta = 1.5$.

- We observe that the performance of the adaptive push system of [8] worsens (the Response time increases) as the ratio *Max_Pr_delay*/*Tr_time* increases. This is due to the fact that an increase of the propagation delay compared to the item transmission time means that for each item transmission, the server of the adaptive push system of [8] must wait a relatively longer proportion of the item transmission time so as to collect the pulses of every acknowledging client in the entire system service area. This translates to an increased overhead per item transmission, which is seen that significantly degrades the performance of the adaptive push system of [8] when *Max_Pr_delay* approaches *Tr_time*.
- The above-mentioned deficiency for the adaptive push system of [8] is not observed in the proposed push system. This is due to the fact that in the proposed system the server will wait to receive feedback only from those clients located at such distances so that the ratio *Pr_delay*/*Tr_time* is less than 0.1, where *Pr_delay* is

the maximum propagation delay from such a client to the server. This of course translates into a fixed time overhead for the acknowledgement of each item transmission. This fixed overhead, which is independent of the *Max_Pr_delay*/*Tr_time* ratio, is the reason that the performance of the proposed system is stable and does not depend on the value of the *Max_Pr_delay*/*Tr_time* ratio. Furthermore, the performance improvement over the adaptive push system of [8] becomes significant when *Max_Pr_delay* becomes comparable to *Tr_time* (e.g., more than 100% better when *Max_Pr_delay*/*Tr_time* = 1).
- The performances of both schemes improve with increasing θ. This is expected behavior [8], as the Learning Automaton adaptation mechanism manages to learn the actual demand probabilities of the various information items and use these values on the selection of the item to broadcast.
- This performances of both schemes decrease for increasing values of *Dev*. This is again expected [8] as a non-zero value of *Dev* reduces the commonality between client demand patterns.

4. Conclusion

This paper proposed a push system that can efficiently operate in high speeds in environments with dynamic client patterns. Adaptation is accomplished using a simple feedback from the clients. The simple feedback is sent only from clients whose distance from the server does not incur a significant timing overhead for the acknowledgment of an item. Simulation results are presented which reveal satisfactory performance in high-speed environments with a priori unknown and dynamic client demands.

References

[1] P. Nicopolitidis, M.S. Obaidat, G.I. Papadimitriou, A.S. Pomportsis, Wireless Networks, John Wiley and Sons Ltd., 2003.
[2] D. Aksou, M. Franlkin, Scheduling for large-scale on-demand data broadcasting, in: Proceedings of IEEE Infocom, 1998, pp. 651–659.
[3] S. Acharya, M. Franklin, S. Zdonik, Dissemination-based data delivery using broadcast disks, IEEE Personal Communications vol. 2 (no. 6) (1995) 50–56.
[4] N.H. Vaidya, S. Hameed, Scheduling data broadcast in asymmetric communication environments, ACM/Baltzer Wireless Networks 5 (3) 171–182.
[5] K. Stathatos, N. Roussopulos, J.S. Baras, Adaptive data broadcast in hybrid networks, in: Proceedings of VLDB, 1997, pp. 326–335.
[6] J. Fernandez, K. Ramamritham, Adaptive dissemination of data in time-critical asymmetric communication environments, in: Proceedings of 11th IEEE Euromicro Conference on Real-Time Systems, 1999, pp. 195–203.
[7] T. Imielinski, S. Vishwanathan, Adaptive wireless information systems, in: Proceedings of SIGDbS, 1994.
[8] P. Nicopolitidis, G.I. Papadimitriou, 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) (2002) 1652–1660.
[9] K.S. Narendra, M.A.L. Thathachar, Learning Automata: An Introduction, Prentice Hall, 1989.
[10] G.I. Papadimitriou, A new approach to the design of reinforcement schemes for learning automata: stochastic estimator learning algorithms, IEEE Transactions on Knowledge and Data Engineering vol. 6 (no. 4) (1994) 649–654.
[11] G.I. Papadimitriou, Hierarchical discretized pursuit nonlinear learning automata with rapid convergence and high accuracy, IEEE Transactions on Knowledge and Data Engineering vol. 6 (no. 4) (1994) 654–659.
[12] G.I. Papadimitriou, A.S. Pomportsis, Learning automata-based tdma protocols for broadcast communication systems with bursty traffic, IEEE Communication Letters vol. 4 (no. 3) (2000) 107–109.
[13] G.I. Papadimitriou, A.S. Pomportsis, Self-adaptive TDMA protocols for WDM star networks: a learning-automata-based approach, IEEE Photonics Technology Letters vol. 11 (no. 10) (1999) 1322–1324.
[14] G.I. Papadimitriou, D.G. Maritsas, Learning automata-based receiver conflict avoidance algorithms for WDM broadcast-and-select star networks, IEEE/ACM Transactions on Networking vol. 4 (no. 3) (1996) 407–412.
[15] P. Nicopolitidis, G.I. Papadimitriou, A.S. Pomportsis, Learning-automata-based polling protocols for wireless LANs, IEEE Transactions on Communications vol. 51 (no. 3) (2003) 453–463.
[16] J.B. Andersen, T.S. Rappaport, S. Yoshida, Propagation measurements and models for wireless communication channels, IEEE Communications Magazine vol. 33 (no. 1) (1995) 42–49.
[17] S. Acharya, M. Franklin, S. Zdonik, Dissemination-based data delivery using broadcast disks, IEEE Personal Communications vol. 2 (no. 6) (1995) 50–60.
[18] C.J. Su, L. Tassiulas, Broadcast scheduling for information distribution, in: Proceedings of the IEEE Infocom, May 1997, pp. 109–117.

Petros Nicopolitidis received the B.S. and Ph.D. degrees in computer science from the Department of Informatics, Aristotle University of Thessaloniki, in 1998 and 2002, respectively. Since 2004 he is a Lecturer at the same Department. His research interests are in the areas of wireless networks and mobile communications. He is coauthor of the book Wireless Networks (New York: Wiley, 2003).

Georgios Papadimitriou received the Diploma and Ph.D. degrees in Computer Engineering from the University of Patras, Greece in 1989 and 1994 respectively. From 1989 to 1994 he was a Teaching Assistant at the Department of Computer Engineering of the University of Patras and a Research Scientist at the Computer Technology Institute, Patras, Greece. From 1994 to 1996 he was a Postdoctorate Research Associate at the Computer Technology Institute. From 1997 to 2001, he was a Lecturer at the Department of Informatics, Aristotle University of Thessaloniki, Greece. Since 2001 he is an Assistant Professor at the Department of Informatics, Aristotle University of Thessaloniki, Greece. His research interests include optical networks, wireless networks, high speed LANs and learning automata. Prof. Papadimitriou is Associate Editor of five scholarly journals, including the IEEE Transactions on Systems, Man and Cybernetics-Part C, the IEEE Transactions on Broadcasting, and the IEEE Communications Magazine. He is co-author of the books "Multiwavelength Optical

LANs'' (Wiley, 2003) and ''Wireless Networks'' (Wiley, 2003) and co-editor of the book ''Applied System Simulation'' (Kluwer, 2003). He is the author of more than 100 refereed journal and conference papers. He is a Senior Member of IEEE.

Professor Mohammad S. Obaidat is an internationally well-known academic, researcher, and scientist. He received his Ph.D. and M.S. degrees in Computer Engineering with a minor in Computer Science from The Ohio State University, Columbus, Ohio, USA. Dr. Obaidat is currently a full Professor of Computer Science at Monmouth University, NJ, USA. Among his previous positions are Chair of the Department of Computer Science and Director of the Graduate Program at Monmouth University and a faculty member at the City University of New York. He has received extensive research funding. He has authored or co-authored five books and over threee hundred (300) refereed scholarly journal and conference articles. Dr. Obaidat has served as a consultant for several corporations and organizations worldwide and is editor of many scholarly journals including being the Chief Editor of the International Journal of Communication Systems published by John Wiley. In 2002, he was the scientific advisor for the World Bank/UN Workshop on Fostering Digital Inclusion. He was an IEEE (Institute of Electrical and Electronics Engineers) distinguished visitor/speaker and has been serving as a National ACM (Association for Computing Machinery) distinguished lecturer since 1995. Recently, Dr. Obaidat was awarded the distinguished Nokia Research Fellowship and the Distinguished Fulbright Scholarship.

Andreas S. Pomportsis received the B.S. degree in physics and the M.S. degree in electronics and communications from the University of Thessaloniki, Greece, and the Diploma degree in electrical engineering from the Technical University of Thessaloniki, Greece. In 1987, he received the Ph.D. degree in computer science from the University of Thessaloniki. Currently, he is a Professor at the Department of Informatics, Aristotle University of Thessaloniki, Greece. He is coauthor of the books Wireless Networks (New York: Wiley, 2003) and Multiwavelength Optical LANs (New York: Wiley, 2003). His research interests include computer networks, learning automata, computer architecture, parallel and distributed computer systems, and multimedia systems.