Adaptive Data Broadcasting in Underwater Wireless Networks

Petros Nicopolitidis, *Member, IEEE*, Georgios I. Papadimitriou, *Senior Member, IEEE*, and Andreas S. Pomportsis

***Abstract*—Underwater acoustic networks have recently emerged as a new area of research in wireless networking. These networks can support a large number of applications such as environmental and underwater equipment monitoring. In recent years, there has been substantial work on protocol design for these networks with most efforts focusing on MAC and network layer protocols. Despite being a fundamental networking primitive, data broadcasting has so far received little attention in the context of underwater networks. This paper proposes an adaptive push system for dissemination of data in underwater acoustic wireless networks. Besides achieving adaptation of its broadcast schedule according to the *a priori* unknown needs of the clients, the proposed system also efficiently combats the problem of high latency of the underwater acoustic wireless environment. Simulation results show superior performance of the proposed system in the underwater environment compared to adaptations of existing terrestrial push systems.**

***Index Terms*—Data broadcasting, learning automata, underwater networks.**

I. Introduction

THE underwater environment poses significant challenges for the deployment of wireless networks [1]–[4]. This is because radio waves exhibit poor propagation properties in the water, while optical waves are severely affected by scattering and require high precision in pointing the laser beams. Thus, the enabling technology for the physical layer of underwater is typically implemented via acoustic waves.

Underwater acoustic wireless channels have a number of unique characteristics that affect the design of networking protocols. The most significant are the following.

- Propagation delay that is five orders of magnitude larger than that of radio-based links of the same size. This is attributed to the low speed of sound (1.5 km/s) in the water.
- Bandwidth–distance relationship: There exists a relationship between the maximum coverage area of an underwater acoustic transmitter and both the bandwidth and the frequency band available to this transmitter for broadcasting to clients inside this area. As the coverage area increases, the frequency band that is used must be shifted toward lower frequencies and its available bandwidth reduces [2]. Typical values for this bandwidth range from several kilohertz for areas of a few kilometers to several hertz for areas that span tens of kilometers.
- The bit error rate of the underwater acoustic links is higher than that of the radio-based ones.

In the last years, there has been increasing research interest in the design of protocols for acoustic underwater networks. Many of these protocols are essentially adaptations of existing ones for radio networks. Such typical examples are the underwater variants of the Aloha, medium access with collision avoidance (MACA), and floor acquisition multiple access (FAMA) medium access control (MAC) protocols whose underwater acoustic variants are proposed in [5]–[7], respectively.

Apart from the aforementioned adaptations, new protocols for the acoustic environment have also been proposed. However, all of them target networks with bidirectional links with nodes being primarily oriented to unicast transmission. On the other hand, in situations where multiple clients exhibit common demands for data items from a central server, the paradigm of data broadcasting [8]–[11] can be efficiently applied to increase performance. Such environments comprise a broadcast server that contains a database with data items and a set of clients that demand access to these items with skewed demand patterns, meaning that the majority of the clients demand access to a small specific number of data items. Thus, it is obvious that these data items will be more "popular" than others since broadcasting one of these will satisfy many more clients than the broadcasting of an item that is rarely demanded by the clients.

Despite being a popular networking primitive, data broadcasting has so far received little attention in the context of underwater networks, with the only relevant work presented in [12]–[14]. This paper proposes an adaptive push-based, data broadcasting system for dissemination of data items to multiple underwater clients which have a certain degree of commonality in their demands for data. The proposed system, whose topology is shown in Fig. 1, comprises a base station with acoustic transmission capabilities and a number of clients that listen to the server's broadcasts. The base station also acts as the relay to the terrestrial broadcast server with the two of them communicating over a radio wireless or cable link. The proposed method uses a learning automaton (LA) at the server, which contains a vector that stores the server's estimation of the actual demand probability among the client population for each data item. This vector will be referred to from now on as the item probability estimation vector. The LA continuously adapts to the demand pattern of the client population to reflect the overall popularity of each data item. The adaptation is accomplished using a feedback from the clients which is used at the server to update the item probability estimation vector. Apart from achieving adaptation of its broadcast schedule according to the *a priori* unknown needs of the clients, the proposed system also efficiently

Manuscript received May 22, 2009; revised March 08, 2010 and April 23, 2010; accepted April 23, 2010. Date of publication July 29, 2010; date of current version September 01, 2010.

Associate Editor: R. Spindel.

The authors are with the Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki 54124, Greece (e-mail: petros@csd.auth.gr).

Digital Object Identifier 10.1109/JOE.2010.2049674

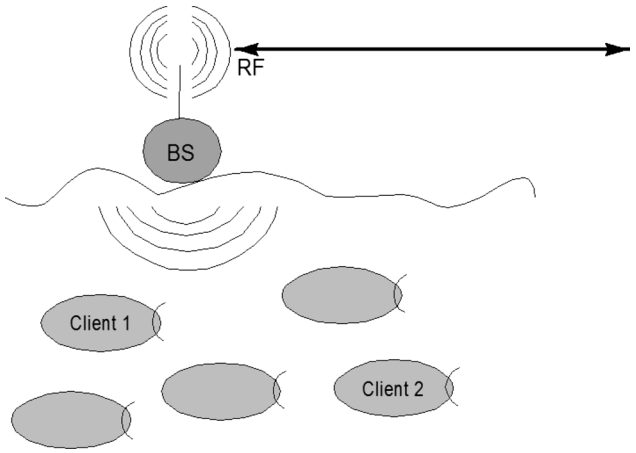

Fig. 1. Underwater push system comprising a base station with acoustic transmission capabilities and a number of clients.

combats the problem of high latency of the underwater acoustic wireless environment.

An example application of adaptive data broadcasting is the case of flight information dissemination in an airport [15]. In such a scenario, 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 overall client demands. For a specific flight, the demand is likely to be in its peak a couple of hours before the flight's departure. For example, if our flight departs at 6 P.M., early in the day, the demand will be very small, as few passengers are likely to come to the airport five or six hours before their flight. At this time of the day, the server should increase the frequency of data items that concern other flights that leave in the near future. As the time for the departure of our flight approaches, the demand for information regarding it will grow due to the increasing number of waiting passengers. Eventually, a few minutes after the departure of the flight, it will drop again. It can easily be seen that in such an environment, overall client demands are neither *a priori* known nor static.

Examples of applications where underwater data broadcasting would be useful are environmental monitoring (e.g., tsunami detection) and support to underwater manned missions, where data items may be broadcast to reach a number of relevant destinations (e.g., monitoring stations or divers, respectively). For example, in the case of support to divers that are equipped with underwater global positioning system (GPS) [16], which is widely available (or another positioning method [17], [18]), and an acoustic navigation system, the proposed system can offer support for the dissemination of information that enables mapping and positioning of the diver. To this end, the broadcast server can disseminate data items that contain graphical (e.g., map, objects) and contextual data (e.g., temperature, currents) regarding the entire area of an underwater operation to the divers. Having knowledge of its current position, the equipment (e.g., [19]) of each diver will expect to receive and thus acknowledge only data items that contain useful information regarding its present position.

The remainder of this paper is organized as follows. Section II overviews terrestrial data broadcasting approaches which the proposed method builds on and then proceeds to overview work done in underwater broadcasting. Section III presents the proposed underwater data broadcasting system. Section IV presents simulation results that reveal the superiority of the proposed system compared to adaptations of existing terrestrial push systems in underwater environments with dynamic client demands. Finally, Section V concludes this paper.

II. Related Work

A. Related Terrestrial Data Broadcasting Systems

There are many papers addressing the problem of efficient data broadcasting. In what follows, we outline some of them. A popular approach in the area of push systems was the broadcast disks model [11]. It proposes a way of superposition of multiple disks spinning at different frequencies on a single broadcast channel. The most popular data are placed on the faster disks and as a result, periodic schedules were produced with the most popular data being broadcast more frequently. This work also proposes some cache management techniques aiming to reduce performance degradation of those clients with demands deviating from the server's schedule. This work was augmented later by dealing with issues such as the impact of changes at the values of the data items at the server between successive broadcasts [20] and addition of a feedback channel to allow users to send explicit requests to the server [21]. The latter approach can be considered as a hybrid system. However, the broadcast disks approach is not adaptive, since it is based on the server's knowledge of a static client demand pattern resulting in predetermined broadcast schedules. Furthermore, it is constrained to fixed sized data items and does not present a way of determining either the optimal number of disks to use or their respective frequencies. Those numbers are selected empirically and as a result, the server may not broadcast data items optimally, even in cases of static demand patterns.

Push-based systems are also proposed in [9]. This work proposes broadcast schedules based on the so-called square root rule. Assuming that the instances of each item are equally spaced in the broadcast, it shows that the client mean response time is minimized when the server broadcasts an item i with frequency proportional to the square root of the factor d_i/l_i, where d_i is the overall client demand probability for item i and l_i is this item's length. In [22], the same result is reached for equal l_i's through numerical experiments and also examines the outcoming algorithm performance on a pull system. This work is carried out to a further extent in [23], where it is argued that in addition to the mean client response time, the variance of the response time should also be taken into account by the broadcast scheduler. The proposed algorithm provides a balance between minimization of the mean client response time and the variance.

The above push-based approaches are practical only for fairly stable environments since no mean of updating the probabilities of data items is proposed. The broadcast disks approach is based on the server's knowledge of a static demand pattern resulting

in predetermined, nonadaptive, broadcast schedules. The aforementioned approaches assume that the access probabilities of data items are available and are thus limited in terms of robustness to changing demand. The method proposed in [8] and [10] proposes a push-based system that is adaptive to dynamic client demands. It uses a LA at the broadcast server to provide adaptivity to [9] while maintaining its computational complexity. Using feedback from the clients, the automaton continuously adapts to the overall client population demands to reflect the overall popularity of each data item. It is shown via simulation that, contrary to the method of [9], the adaptive system provides superior performance in an environment with *a priori* unknown client demands, which can also change over time with the nature of these changes being unknown to the broadcast server. Further extensions of [8] address performance improvement [24], [25] and fairness [26] in environments with locality of client demands.

B. Underwater Broadcasting Systems

Despite being a popular networking primitive, data broadcasting has so far received little attention in the context of underwater networks with the only relevant work presented in [12] and [14]. Casari and Harris [12] propose three reliable broadcast protocols based on forward error correction (FEC) capabilities where clients can also act as forwarders of the server's broadcast to clients farther away. This functionality is based on the bandwidth–distance relationship of the underwater acoustic environments and is thus accomplished by employing specific frequency bands where the signals are not expected to travel long distances. This results in a reduction of the number of transmissions required to complete the broadcast, which in turn reduces both the overall energy consumption and the total time it takes to complete the broadcast. The performance of the proposed solutions is compared against two versions of a traditional radio-based reliable multicast protocol.

Casari *et al.* [13], [14] take a different approach, by using Fountain codes to enhance the efficiency of the data dissemination process in face of poor channel conditions. The paper thus proposes a broadcasting scheme based on these codes, which can provide a tradeoff between different performance metrics such as delay, advancement per hop, and transmission power.

III. The Proposed Push System

A. Learning Automata

Learning automata [27]–[29] are mechanisms that can be applied to learn the characteristics of a system's environment. A LA is an automaton that improves its performance by interacting with the random environment in which it operates. Its goal is to find the optimal action among a set of A actions, so that the average penalty due to the environment is minimized. This means that there exists a feedback mechanism that notifies the automaton about the environment's response to a specific action. The operation of a LA constitutes a sequence of time cycles that eventually lead to minimization of average penalty. The LA uses a vector $p(n) = \{p_1(n), p_2(n), \ldots, p_A(n)\}$, which represents the probability distribution for choosing one of the actions $a_1, a_2, \ldots, a_A$ at time cycle n. Obviously, $\sum_{i=1}^{A} p_i(n) = 1$. The core of the operation of the LA is the probability updating algorithm, also known as the reinforcement scheme which uses the environmental response $\beta(n)$ triggered by the action α_i selected at cycle n to update the probability distribution vector p. After the update is finished, the automaton selects the action to perform at time cycle $n+1$, according to the updated probability distribution vector $p(n+1)$. A general reinforcement scheme has the form of the following formula, where $a(n)$ is a variable that holds the action that was selected from the set $a_1, a_2, \ldots, a_A$ at time cycle n:

$$\begin{aligned} p_i(n+1) &= p_i(n) - (1-\beta(n))\, g_i\left(p(n)\right) + \beta(n) h_i\left(p(n)\right), \\ &\qquad \text{if} \quad \alpha(\mathrm{n}) \neq \alpha_i \\ p_i(n+1) &= p_i(n) + (1-\beta(n)) \sum_{j \neq i} g_j\left(p(n)\right) \\ &\quad - \beta(n) \sum_{j \neq i} h_j\left(p(n)\right), \qquad \text{if} \quad \alpha(\mathrm{n}) = \alpha_i. \end{aligned} \tag{1}$$

The cycle n is defined as the time period in which the LA chooses one of the actions $a_1, a_2, \ldots, a_A$, executes it, receives the environmental response $\beta(n)$ triggered by the action α_i, and updates the probability distribution vector p. The functions g_i and h_i are associated with reward and penalty for action α_i, respectively, and $\beta(n)$ is a metric of the environmental response, normalized in $[0, 1]$. The lower the value of $\beta(n)$ is, the more favorable the response becomes. When $\beta(n)$ takes continuous values after normalization in the interval $[0, 1]$, the automaton is known as an S-model. In the area of data networking, learning automata have been applied to several problems, including the design of self-adaptive MAC protocols for wired and wireless platforms (e.g., [30]–[32]) and routing [33], [34].

B. The Broadcasting Algorithm—Basics and Motivation

In data broadcasting, the primary criterion for performance optimality is the minimization of the mean response time of the client population. The latter is defined as the mean time that elapses between the instant that a demand for an item is made at an arbitrary client and the instant when the client receives the demanded item. To optimize performance, broadcast schedules must be periodic [35], and the variance of spacing between consecutive instances of the same item must be reduced [36]. Based on the above, the broadcast scheduling of many push systems (e.g., [9]) are based on the following arguments.

1) Schedules with minimum overall mean client response time are produced when the intervals between successive instances of the same item are equal.
2) Under the assumption of equally spaced instances of the same items the minimum overall mean client response time occurs when the server broadcasts an item i with frequency being proportional to the factor $\sqrt{(d_i/l_i)((1+E(l_i))/(1-E(l_i)))}$, where d_i is the demand probability for item i, l_i is the item's length, and $E(l_i)$ is the probability that an item of length l_i is received with an unrecoverable error.

Vaidya and Hameed [9] reveal that a broadcast algorithm based on the above arguments minimizes the mean response time of the system (optimized performance). Thus, the broadcasting algorithm used in this paper is also based on the

above arguments and operates as follows. The broadcast server is equipped with an S-model linear reward-inaction LA that contains the server's estimate p_i of the actual demand probability d_i of the client population for each data item i among the set of the items the server broadcasts. Clearly, $\sum_{i=1}^{N} p_i = \sum_{i=1}^{N} d_i = 1$, where N is the number of items in the server's database. As in terrestrial instances of the proposed approach [8]–[10], [24]–[26], at each cycle, the server selects to transmit the item i that maximizes the cost function $G(i) = (T - R(i))^2(p_i/l_i)((1 + E(l_i))/(1 - E(l_i)))$, where T is the current time, $R(i)$ is the time when item i was last broadcast, l_i is the length of item i, and $E(l_i)$ is the probability of item with length l_i being erroneously received. For items that have not been previously broadcast, R is set 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 $R(i) = T$.

The server will probe the data items to receive feedback from them so as to estimate the demand probability for each data item. We note here that the feedback will concern individual items, thus feedback is never solicited to acknowledge groups of data items. For the feedback transmission, code-division multiple access (CDMA), which has also been used in the underwater acoustic environment [37]–[39], is chosen. On the other hand, data item broadcasts use narrowband modulation over the entire available bandwidth for data broadcasting. Each client that was waiting for the item that was broadcast sends its feedback to the base station using a user-specific high-speed code (long code). At the receiver end (base station), signals are separated by using a correlator, which only accepts signal energy from the specific client's long code and despreads its spectrum. Other co-user signals remain spread because their spreading algorithm is uncorrelated with the desired signal's algorithm and they appear as noise.

The number of mobile clients that can be supported in the system is given by the capacity of CDMA [40], which is given by

$$ClNum = \frac{W_{FB}/U_{FB}}{E_b/N_o} + 1 \tag{2}$$

where *ClNum* is the number of mobile clients supported, W_{FB} represents the transmission bandwidth for the feedback, U_{FB} represents the feedback transmission bit rate, and E_b/N_o is the bit energy-to-noise power spectral density in linear scale. As far as power control on the returning feedbacks is concerned, every item will be broadcast having piggybacked on it the actual power P_t at which it was transmitted. Based on this information and the power P_r measured at item reception, each client will set the transmission power of its feedback to P_t/P_r. Using this form of power control, client feedbacks will arrive at the same power at the server irrespective of the clients' distances from the server. Another option would be to piggyback the actual position of the server within broadcast items, so that clients can perform power control on the returning feedbacks by obtaining their position (and thus their distance from the server) through their onboard positioning system (e.g., [16]–[18]). However, the increased latency that the feedback packets will experience

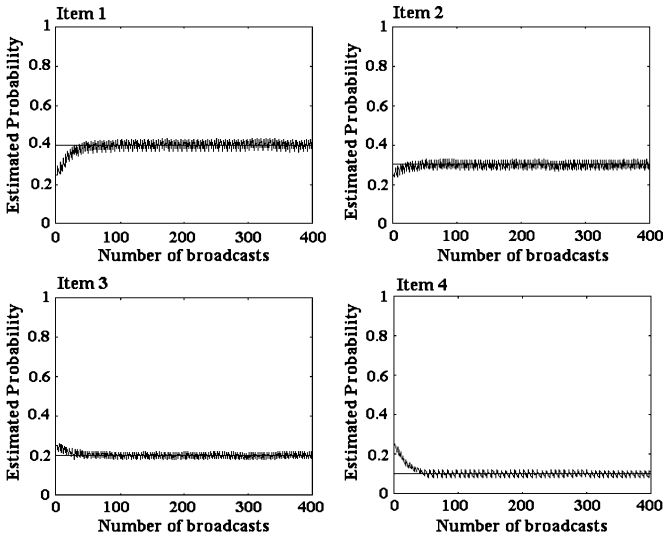

Fig. 2. Convergence of the automaton estimation of the item demand probabilities.

over the high-propagation delay of underwater acoustic links will pose a significant performance bottleneck for the system of [8], as this waits for client feedbacks after each item broadcasts for a time interval that equals the maximum round trip time plus the feedback transmission time. This problem can be clarified with the following example. Consider a server that acoustically transmits items to underwater clients located inside the server's coverage area (a circle with a 10-km radius). This gives a maximum round trip time (*MAX_RTT*) of about 13 s. Typical communication parameters for distances up to 10 km are reported in [2] and are also used in the section where we evaluate the performance of the proposed approach. According to these, we can assume an available data bandwidth of 10 kHz and a 2.4-kb/s data rate. Moreover, we assume that the server broadcasts $N = 500$ different 1000-b-long data items. If we use a small 200-Hz part of the available bandwidth for realizing the system's feedback, we can use the rest of the available bandwidth to broadcast items. Thus, with 9.8 kHz available, we can broadcast using rates around 2.35 kb/s which yield a 0.42-s item transmission time. Moreover, to support an adequate number of clients, (2) mandates the use of a small feedback user data rate. Thus, if we employ a feedback data rate of 0.5 b/s the one-bit feedback of the clients in the system of [8] lasts 2 s.

Regarding the tradeoff between the amount of bandwidth dedicated to data and feedback, one can see that for the same number of supported clients, if we dedicate more bandwidth to the feedback, the feedback rate is increased and the data rate is decreased. Thus, the system would become one that broadcasts at a lower speed but shows increased learning rate of the client demands due to the increased feedback rate. However, in a slowly changing environment, as usually found in data broadcasting contexts, the faster rate of learning the demand probabilities would not provide additional performance increase (as shown in [41]), thus the net effect would be the increase of the response time (thus performance decrease) due to the decreased data rate. The propagation delay would not affect the above tradeoff as the proposed system continuously

broadcasts data items and is not burdened by the propagation delay.

It can be easily seen that, in the above discussion, the stop-and-wait approach of [8] for feedback collection after each item broadcast will underutilize the acoustic links, since utilization will equal to *Itm_time/(MAX_RTT+ Itm_time + Fdb_time)*= $0.42/(0.42 + 2 + 13)$ =2.7%, where *Itm_time + Fdb_time* are the durations of the data items and feedback transmissions, respectively.

C. The Proposed Underwater Push System

To solve the low-utilization problem described above, we can change the system to a continuous one by making the server immediately proceed to the next item broadcast after completing the previous one. In this case, the server will demand feedback once every $2/0.42 = 5$ item broadcasts so as to give clients enough time for their previous feedback transmission to be completed. However, with this approach, a new problem arises. The increased latency of the links will pose a problem for feedback collection because feedback will now be received asynchronously relative to the data item transmissions. This means that there will exist situations where the server will need clarification regarding the data item that an incoming feedback acknowledges. In the context of the previous example, if we continuously broadcast items, then a new feedback will be demanded every *Fdb_time* second. For the broadcast of a certain item P, the feedback can reach the server at most *MAX_RTT* + *Itm_time* + *Fdb_time* = 15.4 s after the start of the item's broadcast. This of course corresponds to an acknowledging client that is located at a 10-km distance from the server. However, in this time interval, the server will have probed up to $15.4/2 = 8$ more data items. For these items, the server can very well have received feedback if these were destined for clients closer to the server than the ones that acknowledged item P.

It can be easily seen from the above discussion that a continuous broadcast over the acoustic links that utilizes one-bit feedback by the clients will result in ambiguities regarding the item that each incoming feedback acknowledges. This problem is solved by increasing the length of the feedback so that it contains the ID of the item it acknowledges.

To achieve this, to collect feedback from the clients, the server uses an in-band channel over the transmitted data items. This channel is realized by piggybacking two binary numbers A and B on the header of each item i, $1 \leq i \leq N$. A is a one-bit number, while B comprises a $\lceil \log_2 N \rceil$ number of bits. When the server wants to collect feedback from the clients that are currently waiting to receive item j, it sets on the header of the broadcast item i, $A = 1$, and $B = j$. When the server does not request feedback after the broadcast of item j, it notifies the clients for this by setting $A = 0$. Thus, each user's feedback will be made of $1 + \lceil \log_2 N \rceil$ number of bits, where N is the number of items the server broadcasts.

After the transmission of a probing request, the server will continuously broadcast items for a time interval equal to the duration of the feedback transmission, which is known *a priori* since the feedback is $1 + \lceil \log_2 N \rceil$ bits and the feedback transmission speed U_{FB} is known. After each such interval, the server will transmit the next item i in the schedule with a piggybacked probe for another item j. Of course, if *Itm_time* $\geq$ *Fdb_time*, the server will be making probes with each broadcast item. Thus, as opposed to [8], the server will continuously broadcast items rather than employ a stop-and-wait approach. With a continuous broadcast of items and variable client placements, the server will receive feedbacks from clients while transmitting data items, thus the use of full-duplex acoustic modems is needed, as already proposed in other acoustic underwater protocols [4], [5], [42].

To select the item to probe j, we separately apply the cost function G to produce the sequence of probed items by assuming that time elapses only when transmission of items with probing requests is made. This means that the server needs to store a separate vector R' that stores the time when each item was last probed by the server.

The item ID on the incoming feedback is read by the server and used to update the estimate of the corresponding items. The probability distribution vector p maintained by the LA estimates the demand probability d_i of each information item i. Until the next probability update takes place the server will use the updated vector p to calculate the cost G of each item in the process of choosing which one to broadcast.

When the probing of an item j does not yield client feedback because no client was waiting for j, the probabilities of the items do not change. However, following a probe for j that yields client feedback, the probability of j is increased. The following linear reward-inaction (L_{R-I}) probability updating scheme is employed after the probing of item j (assuming it is the server's kth probability update):

$$\begin{aligned} p_m(k+1) &= p_m(k) - L\left(\frac{p_m(k)-a}{ClNum}\right) \qquad \forall j \neq m \\ p_j(k+1) &= p_j(k) + \left(\frac{L}{ClNum}\right)\sum_{m \neq j}(p_m(k)-a) \end{aligned} \quad (3)$$

where $p(k)$ takes values in $(\alpha, 1)$ and L, α take values in $(0, 1)$. α prevents the probabilities of nonpopular items from taking values very close to zero to increase the adaptivity of the LA. This is because if the probability estimate p_j of an item j approaches zero, then $G(j)$ will take a value very close to zero. However, item j, 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. To obtain *ClNum*, the server can broadcast a control item that forces every client to respond with feedback and then waits for *MAX_RTT* + *Itm_time* + *Fdb_time* to collect the feedback.

Using the probability updating scheme of (3), the item probabilities estimated by the LA converge near the actual demand probabilities for each item [8]. This makes this approach attractive for dissemination applications with dynamic client demands. This convergence is schematically shown in Fig. 2, which plots the item probability estimates versus the overall actual demand probabilities, in a simulation of a sample database comprising four items. The client demand pattern is unknown to the server, so initially, the four items have an equal demand probability estimate. It is clearly seen that convergence of the item probabilities estimated by the automaton to the actual overall demand pattern of the client population is achieved.

The proposed approach described above constitutes the first proposal for underwater adaptive broadcasting protocols. Protocols proposed so far [12]–[14] are nonadaptive as they do not address adaptation to user needs. The most appropriate of these as a performance criterion for the proposed approach would be [14]. In this approach, the server would send all the N items of the server database regardless of the actual user interest. Thus, differently from the nonadaptive broadcast scheme of [9], in the scheme of [14], the server would not serve broadcast items circularly in a round-robin fashion, but initiate the broadcast once and for all and not step to the subsequent one until the item presently being served has reached all nodes.

One can easily compute the upper and lower differences for the exchanged traffic and client response time between the proposed system and that of [14]. Assume an error-free environment (this will slightly favor the system of [14] as it lifts the need for feedback and item redundancy) and that the clients are interested in a subset $S < N$ of the server's items which are continuously broadcast, a fact that obviously hides the propagation delay for the system of [14] as well. Furthermore, after reception of a demanded item, clients will not demand another one unless the entire client population has been served in that round. The lowest difference for the aforementioned metrics will occur at zero values of the data skew parameter. Due to the adaptation mechanism, the proposed system will transmit the S requested items uniformly at random most of the time and thus S item broadcasts will be needed to satisfy the first request of each client. N broadcasts will be needed by the system of [14] as this initiates the broadcast once and for all each time. This results in the amount of traffic saved being equal to $1-S/N$ in favor of the proposed system. The corresponding client mean response time in the proposed system would be equal to the duration of broadcasting the S demanded items (thus $S\times$*Itm_time*) [8]–[11] and is attributed to the continuous nature of broadcasting. For the method of [14], the client mean response time will be equal to the duration of broadcasting the N items (thus $N\times$*Itm_time*) [8]–[11]. These yield a time saving of $1-S/N$ in favor of the proposed system at zero data skew parameter values.

As the value of the data skew parameter increases, the above differences will also increase in favor of the proposed system. The most significant differences for the aforementioned metrics will occur for high values of the data skew parameter ($\geq$ 1.5). At these values, due to the adaptation mechanism, the proposed system will almost exclusively transmit the first item of the server database, as this will be the only one demanded by the clients at such values of data skew parameter. Thus, typically, one item broadcast will be needed to satisfy the first request of each client. N broadcasts will be needed by the system of [14] as this initiates the broadcast once and for all each time. This results in the amount of traffic saved being equal to $1-1/N$ in favor of the proposed system. The corresponding client mean response time in the proposed system would be equal to the duration of broadcasting the single demanded item (thus *Itm_time*) [8]–[11] and is attributed to the continuous nature of broadcasting. For the method of [14], the client mean response time will be equal to the duration of broadcasting the N items (thus $N\times$*Itm_time*) [8]–[11]. These yield a time saving of $1-1/N$ in favor of the proposed system at high data skew parameter values.

D. The Proposed Underwater Push System With Bandwidth Saving

From the above discussion, it is obvious that the continuous underwater acoustic broadcasting system sacrifices part of the scarcely available bandwidth for the implementation of the feedback system. However, as will be seen in Section IV, it exhibits significantly better performance compared to adaptations of existing terrestrial push systems. Nevertheless, with a simple modification, we can implement feedback with less bandwidth and thus devote more bandwidth to data broadcasts.

To this end, to collect feedback from the clients, the server uses an in-band channel over the transmitted data items, which is now realized by piggybacking a number k of $\lceil \log_2(M+1) \rceil$ of bits in the header of each item i, $1 \leq i \leq N$, where $M < N$ must be large enough to code the maximum number of probes that can result in ambiguous feedbacks for a system with a certain coverage area. Thus, as M is less than the number of items N, if we keep the number of supported clients *ClNum* and the feedback duration constant, we can utilize a smaller feedback user rate U_{FB}. This obviously translates via (2) to less bandwidth needed for the feedback operation. For example, for the system mentioned in Section III-C, with a 10-km coverage range (thus *MAX_RTT* = 13 s) with *Itm_time* and *Fdb_time* being 0.42 and 2 s, respectively, the maximum number of ambiguous feedbacks is *(MAX_RTT + Itm_time + Fdb_time)/Fdb_time* $= 8$ and thus we need to use $\lceil \log_2 9 \rceil = 4$ b. This is obviously a reduction compared to the $1+\lceil \log_2 N \rceil = 1+\lceil \log_2 500 \rceil = 10$ b we needed to code the feedbacks with the previous algorithm and translates to dedicating for item broadcasting an additional 60% of the bandwidth that was used for feedback by the previous algorithm.

The algorithm, whose pseudocode is shown in Fig. 3, operates as follows. We start from a value of $k = 1$ for the probe counter k, $1 \leq k \leq M$. If we do not want to probe any item, we piggyback a value of zero on the broadcast item i. To probe item j with the broadcast of item i:

- we piggyback a probe for k on the header of i;
- we use the vector $\mathrm{map}[]$ to mark the item which the current probe for k refers to by setting $\mathrm{map}[k] = j$;
- we then increase k by 1 (upon overflow, we set $k = 1$);
- upon the reception of a feedback that contains a number v, $1 \leq v \leq M$, we identify via $\mathrm{map}[]$ the data item that this probe acknowledges and we proceed to use (3) to perform the probability updating accordingly.

It has to be noted that the modification described in this section would lead to less improvement in the presence of significant preamble length. From this point of view, this modification provides for the upper limit of the bandwidth that can be saved when acknowledged items are differentially rather than absolutely encoded on the returning feedbacks.

IV. Performance Evaluation

Using simulation, we compared the proposed system against the one in [8] and the static round-robin (flat) broadcast. The flat

```
k=1;
M=ceil(Fdb_time /(MAX_RTT+Itm_time+Fdb_time));
while(server broadcasts items)
{
        map[k]=item_to_probe;
        transmit item i with piggybacked probe for k;
        if (k<M)
                k++;
        else k=1;
        if (feedback_received()==true)
        {
                v=feedback_ received;
                update_probability(map[v]);
        }
}
```

Fig. 3. Pseudocode of the bandwidth-saving version of the proposed push algorithm.

approach also derives from the method in [9] for equiprobable, fixed-length items.

We consider a broadcast server having a database of N equally sized items. We assume that the items contain time-varying information, thus between successive broadcast of the instance of the same item, the actual information it contains changes. The server is initially unaware of the demand for each item, so every item has a probability estimate of $1/N$. We consider *ClNum* clients that have no cache memory due to the time-varying nature of item contents. Clients access items in the interval [1, *Range*]. This range consists of R regions of size Rsize each that contain equiprobably demanded items. The demand probability $d(i)$ of an item in region i is $d(i) = c(1/i)^{\theta}$ where $c = 1/R\text{size}\sum_k (1/k)^{\theta}$, $k \in [1 \dots R]$ and θ is a parameter named access skew coefficient. This is the Zipf distribution used in modeling of client demands [8]–[11], [20]–[26]. As the value of θ increases, Zipf produces increasingly skewed demand patterns.

To simulate some disagreement in the client demand patterns, we introduce the 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. 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 another item that is selected in a uniform manner from the interval $[1 \dots N]$. Each client is placed at x kilometers from the server, with x being uniformly selected from the interval $[1 \dots$ *MAX_DISTANCE*$]$.

We did not take into account reception errors as our goal is to assess the relative performance increase of the proposed approach compared to applications of [8] and flat broadcast in the underwater environment due to the use of the novel feedback scheme. In the presence of channel errors, some data items and feedbacks will be lost. Regarding the data items, this loss will be common for all methods. However, lack of feedback or feedback errors lead to a wrong estimation of the users' interest in the data being broadcast. While flat broadcast may be quite transparent to this, adaptive broadcast policies would be more negatively affected. Specifically, the loss of feedbacks would lead to small inaccuracies for the estimation of client demands. This inaccuracy will actually be a small random reduction in the number of incoming feedbacks for a certain data item and thus it will translate in small random reductions of the learning rate L. These reductions will be common for both the proposed approach and that of [8] and thus will not change their relative performance. Regarding the relative performance of the proposed approach and that of the flat (nonadaptive) scheme, despite that these random reductions will temporarily reduce the learning rate of the proposed system, this will continue to be superior to the flat one because the latter cannot adapt to the client's demands for data items when the data skew parameter takes nonzero values.

Finally, as we used, according to (2), an available bandwidth for the feedback capable of supporting up to *ClNum* mobile clients, multiple-access and the CDMA feedback storm issues do not arise for the feedback.

The simulation is carried out until at least *NumReq* requests are satisfied at each client, meaning that overall, at least *NumReq* $\times$ *ClNum* requests have been served.

The results were obtained with the following parameter values being constant:

- $N = 500$;
- *NumReq* $= 5000$;
- $L = 0.15$;
- $\alpha = 10^{-4}$;
- *Noise* $= 0.5$;
- item size $= 10^3$ b;
- $E_b/N_o = 5$.

As far as the number of feedback bits is concerned, the nonadaptive method does not use feedback. We use the one-bit feedback for the adaptive method of [8]. Ten bits are needed for the feedback of the proposed method. The number of bits used for the feedback of the bandwidth saving version is mentioned later on.

For the nonadaptive push system, since it does not employ the feedback, we utilize the entire 10-kHz available bandwidth for data transmission, thus for this system $U_{\text{DATA}} = 2400$ b/s [2]. For the proposed adaptive system, we use 20% of the available bandwidth for realizing the system's feedback, thus for this system $W_{\text{DATA}} = 8$ kHz, $W_{FB} = 2$ kHz, and consequently, $U_{\text{DATA}} = 1900$ b/s. Moreover, as far as the feedback of the system is concerned, we set $U_{FB} = 5$ b/s. The above parameters via (1) allow for *ClNum* $= 81$. For the adaptive system of [8] which utilizes one-bit feedback, for the same feedback duration, we can support 81 clients with $W_{FB} = 200$ Hz and $U_{FB} = 0.5$ b/s. Thus, for this system, $W_{\text{DATA}} = 9.8$ kHz, and consequently, $U_{\text{DATA}} = 2352$ b/s. Finally, the values of these parameters for the bandwidth saving version are mentioned later on.

In Figs. 4–13, we present simulation results for the following two network environments:

- Network N1: *Range* $= 500$, $R\text{size} = 1$;
- Network N2: *Range* $= 100$, $R\text{size} = 2$.

In Figs. 4–13, we plot the performance of the three systems versus the data access skew coefficient θ. The performance metric we use, which is commonly used in data broadcasting [8]–[11], [20]–[26], is the client mean response time, which

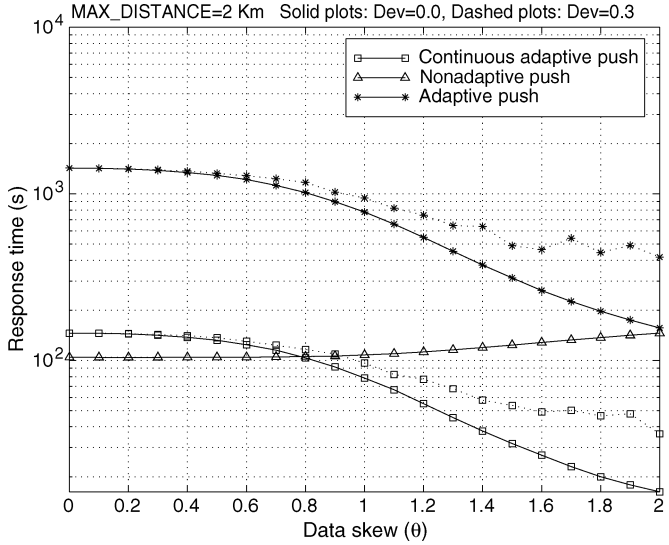

Fig. 4. Mean response time versus access data skew coefficient for the three systems in Network N1 for a server coverage area of 2-km radius.

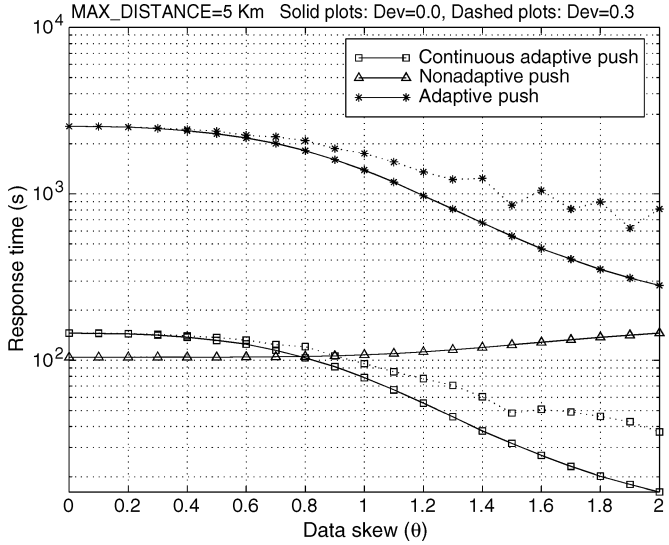

Fig. 5. Mean response time versus data access skew coefficient for the three systems in Network N1 for a server coverage area of 5-km radius.

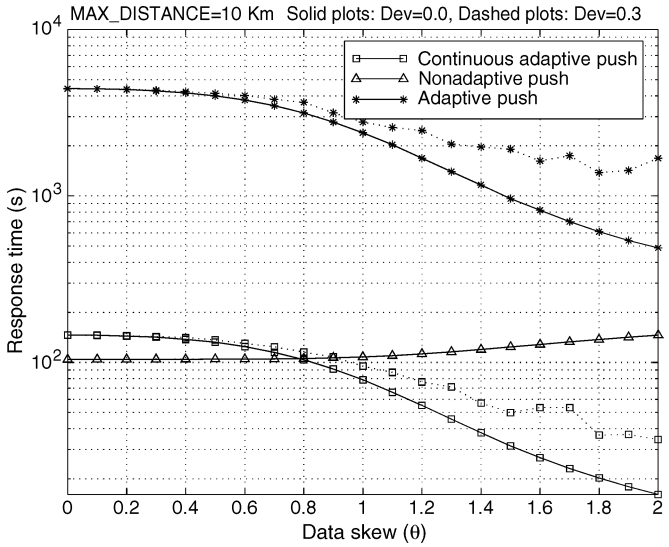

Fig. 6. Mean response time versus data access skew coefficient for the three systems in Network N1 for a server coverage area of 10-km radius.

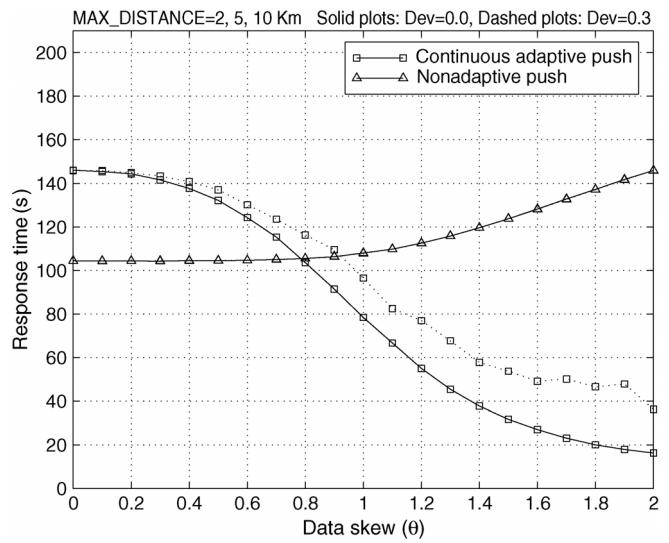

Fig. 7. Mean response time versus data access skew coefficient for the proposed and the nonadaptive system in Network N1.

is the mean time a client waits to receive a requested item. In Figs. 4–13, the plots termed "nonadaptive push," "adaptive push," and "continuous adaptive push" refer to the flat broadcast, to the method of [8], and to the proposed method, respectively. The solid plots correspond to the performance of the systems with $Dev = 0$, whereas the dashed ones correspond to the performance of the systems with $Dev = 0.3$.

Figs. 4–7 display the results for $Range = 500$ and $Rsize = 1$. Specifically, Figs. 4–6 show, for a maximum coverage area of the server (marked *MAX_DISTANCE*) of radius 2, 5, and 10 km, respectively, the relative performance of the three systems with a logarithmic (base 10) scale used for the Y-axis. Fig. 7 shows the relative performance of the proposed system and the flat one. This plot appears only once as the performance of the two systems is the same regardless of the coverage area of the server.

Figs. 8–11 display the results for $Range = 100$ and $Rsize = 2$. Specifically, Figs. 8–10 show, for a maximum coverage area of the server (marked *MAX_DISTANCE*) of radius 2, 5, and 10 km, respectively, the relative performance of the three systems with a logarithmic (base 10) scale used for the Y-axis. Fig. 11 shows the relative performance of the proposed system and the flat one. Again, this plot appears only once as the performance of the two systems is the same regardless of the coverage area of the server.

Finally, Figs. 12, and 13 plot the performance of the proposed algorithm and that of the bandwidth-saving version for a maximum coverage area of the server (marked *MAX_DISTANCE*) of radius 2, 5, and 10 km in Networks N1 and N2, respectively, with both figures obtained for $Dev = 0$. By allowing the same number of clients with the other systems (thus $ClNum = 81$) and by calculating the number of feedbacks that can be ambiguous, the bandwidth saving version utilizes the following parameter values:

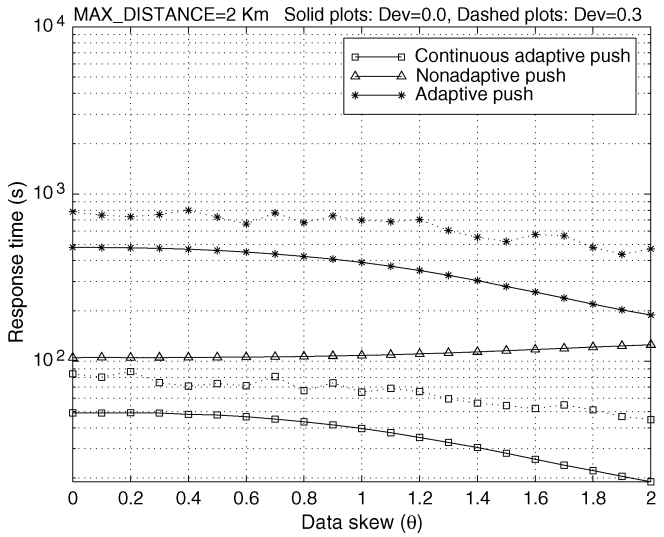

Fig. 8. Mean response time versus data access skew coefficient for the three systems in Network N2 for a server coverage area of 2-km radius.

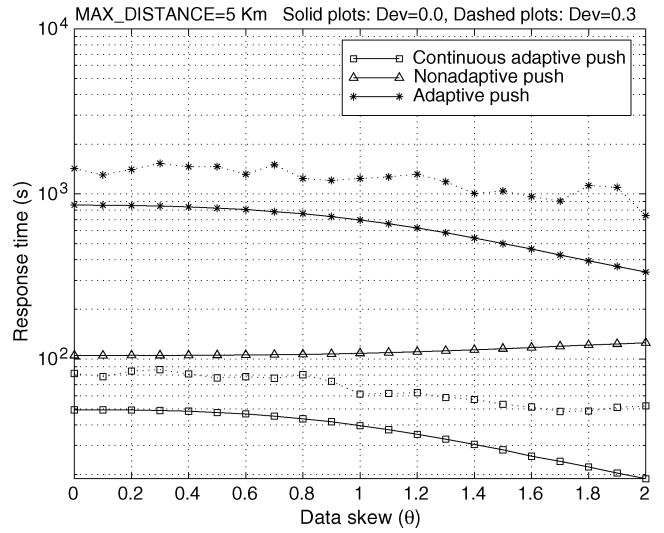

Fig. 9. Mean response time versus data access skew coefficient for the three systems in Network N2 for a server coverage area of 5-km radius.

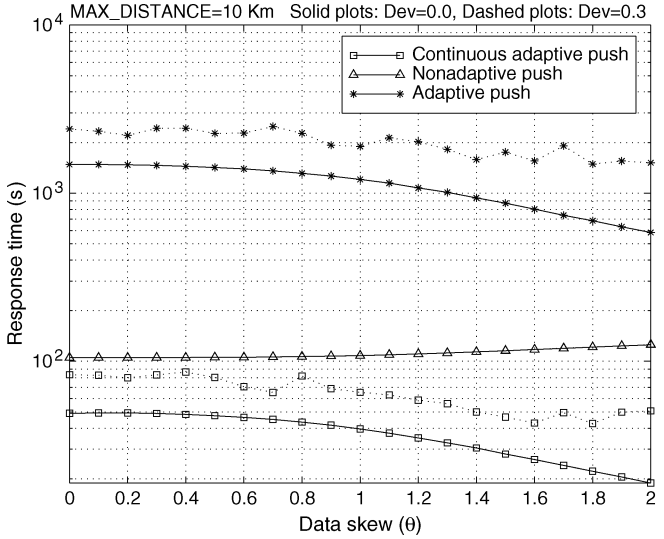

Fig. 10. Mean response time versus data access skew coefficient for the three systems in Network N2 for a server coverage area of 10-km radius.

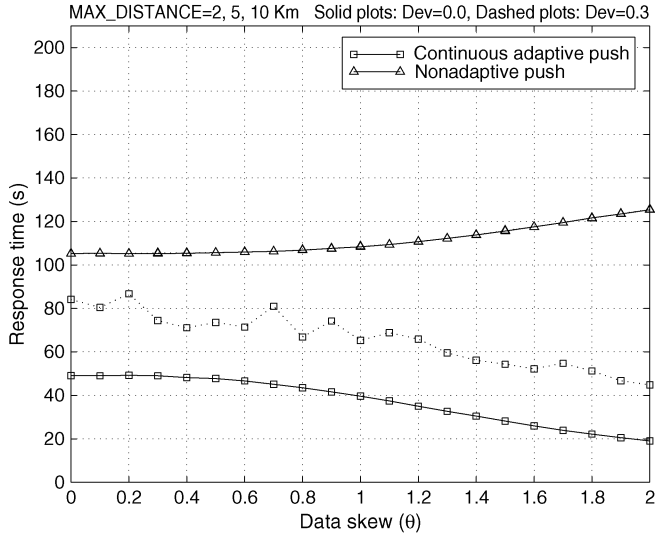

Fig. 11. Mean response time versus data access skew coefficient for the proposed and the nonadaptive system in Network N2.

- *MAX_DISTANCE* = 2 km: two feedback bits, $W_{\mathrm{DATA}} =$ 9.6 kHz, $U_{\mathrm{DATA}} =$ 2304 b/s, $W_{FB} =$ 400 Hz, $U_{FB} =$ 1 b/s;
- *MAX_DISTANCE* = 5 km: three feedback bits, $W_{\mathrm{DATA}} =$ 9.4 kHz, $U_{\mathrm{DATA}} =$ 2256 b/s, $W_{FB} =$ 600 Hz, $U_{FB} =$ 1.5 b/s;
- *MAX_DISTANCE* = 10 km: four feedback bits, $W_{\mathrm{DATA}} =$ 9.2 kHz, $U_{\mathrm{DATA}} =$ 2208 b/s, $W_{FB} =$ 800 Hz, $U_{FB} =$ 2 b/s.

The main conclusions that can be drawn from Figs. 4–13 follow.

- The adaptive systems show better performance as the data skew parameter increases while the nonadaptive system shows worse performance. These are both, because in the adaptive systems, the LA adaptation mechanism learns the actual demand probabilities of the client population, whereas such an adaptation is not present in the nonadaptive system. Specifically, as the data skew parameter increases to larger values, the vast majority of the clients tend to demand the first few items of the server's database. As the adaptation mechanism learns this, these items are broadcast with higher frequency, thus leading to a higher performance (lower mean response time) for the clients. In the nonadaptive system, the same phenomenon caused by the increasing values of the data skew parameter is not accompanied by a corresponding change in the frequencies of the broadcast items due to the lack of adaptivity. Thus, as the server's items are transmitted in a flat manner, for low values of the data skew parameter, the clients demand items in a flat manner as well, with resulting client mean response times being approximately half the period of broadcasting the server's database [8]–[11]. However, for increasing values of the data skew parameter as mentioned

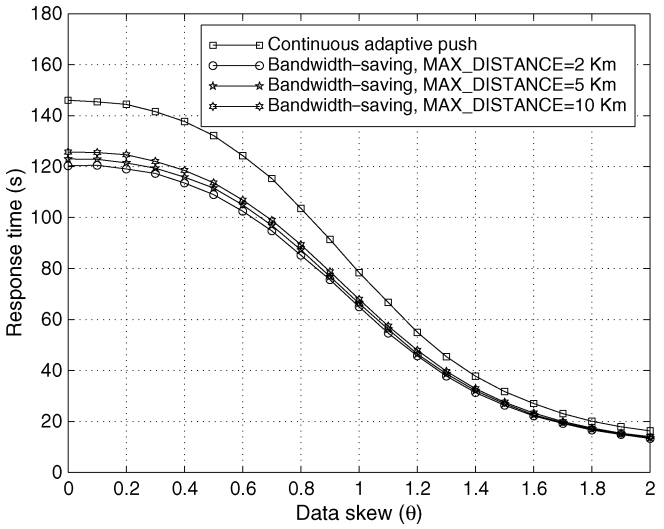

Fig. 12. Mean response time versus data access skew coefficient for the proposed system and the bandwidth-saving variant in Network N1.

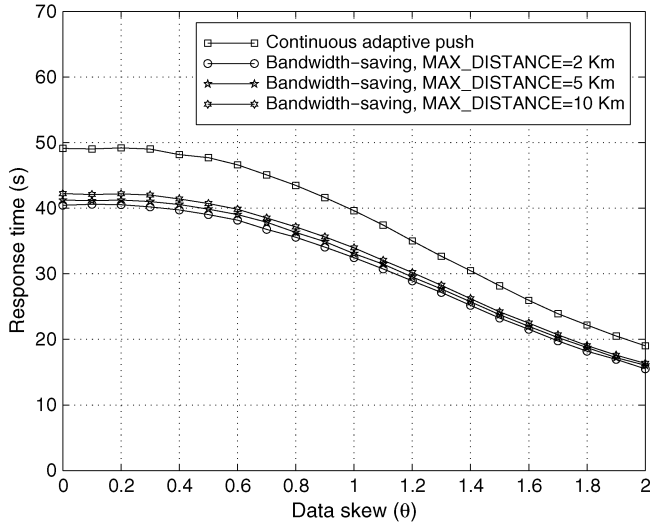

Fig. 13. Mean response time versus data access skew coefficient for the proposed system and the bandwidth-saving variant in Network N2.

above, a client selects only among very few items. Thus, it generally has to wait more than half the period of the broadcast to obtain the requested item, which in turn leads to a lower performance (higher mean response time).

- The proposed approach significantly outperforms that of [8] and the performance difference increases for increasing coverage of the server. Moreover, its absolute performance remains the same regardless of the coverage area of the server. This is because the proposed method for feedback collection allows a continuous transmission of data items, which hides the increased transmission latency that will only burden the system for the first item transmission. In contrast, the method of [8] waits after each item transmission for the sum of twice the maximum propagation delay plus the feedback duration plus the item transmission time to receive feedback, which of course poses a huge time overhead.
- For small values of θ, the performance of the proposed system when $Range = N = 500$ is a little worse than that of the flat scheme (see Figs. 3–6). This is because 1) the flat system does employ feedback and is thus not impaired with the aforementioned performance bottleneck; 2) in the proposed system, small values of θ reduce the demand pattern to a uniform distribution over the server's data items, which in turn leads to the flat broadcast of the server's items; 3) due to the need of the proposed protocol for a feedback channel, the available bandwidth for broadcasting for the proposed system is 80% of that used by the flat broadcasting system. However, for increasing θ, the performance of the proposed system increases significantly compared to that of the flat approach.
- The performance improvement of the proposed approach decreases for $Dev = 0.3$, due to the greater skew in client demands [8], however, it keeps significantly better than that of the other two approaches.
- When $Range = 100 < N$, the flat approach is well outperformed by the proposed method even in the case of small θ, since it notifies the server to broadcast frequently only the demanded items (see Figs. 8–11). Thus, it would be useful even in cases of applications that access subsets of the server's database with demand patterns close to being uniform.
- As can be seen from Figs. 12 and 13, the bandwidth-saving version of the proposed system leads to a further performance increase.

V. Conclusion

This paper proposes an adaptive push system for acoustic information dissemination of data in underwater clients. Apart from achieving adaptation of its broadcast schedule according to the *a priori* unknown needs of the clients, the proposed system also efficiently combats the problem of high latency of the underwater acoustic wireless environment. Simulation results show superior performance compared to existing approaches for terrestrial push systems. Moreover, the performance of the proposed system is not affected by increasing the coverage area of the broadcast server.

References

[1] D. Pompili and I. F. Akyildiz, "Overview of networking protocols for underwater wireless communications," *IEEE Commun. Mag.*, vol. 47, no. 1, pp. 97–102, Jan. 2009.

[2] I. F. Akyildiz, D. Pompili, and T. Melodia, "Underwater acoustic sensor networks: Research challenges," *Ad Hoc Netw.*, vol. 3, no. 3, pp. 257–279, Mar. 2005.

[3] G. G. Xie and J. H. Gibson, "A network layer protocol for UANs to address propagation delay-induced performance limitations," in *Proc. MTS/IEEE OCEANS Conf. Exhib.*, 2001, vol. 4, pp. 2087–2094.

[4] J. G. Proakis, E. M. Sozer, J. A. Rice, and M. Stojanovic, "Shallow water acoustic networks," *IEEE Commun. Mag.*, vol. 39, no. 11, pp. 114–119, Nov. 2001.

[5] N. Chirdchoo, W.-S. Soh, and K.-C. Chua, "Aloha-based MAC protocols with collision avoidance for underwater acoustic networks," in *Proc. IEEE Int. Conf. Comput. Commun.*, Anchorage, AK, May 2007, pp. 2271–2275.

[6] H-H. Ng, W-S. Soh, and M. Motani, "MACA-U: A media access protocol for underwater acoustic networks," in *Proc. IEEE Global Telecommun. Conf.*, New Orleans, LA, Dec. 2008, DOI: 10.1109/GLOCOM.2008.ECP.165.

[7] M. Molins and M. Stojanovic, "Slotted FAMA: A MAC protocol for underwater acoustic networks," in *Proc. MTS/IEEE OCEANS Conf.*, Boston, MA, Sep. 2006, DOI: 10.1109/OCEANSAP.2006.4393832.
[8] P. Nicopolitidis, G. I. Papadimitriou, and A. S. Pomportsis, "Using learning automata for adaptive push-based data broadcasting in asymmetric wireless environments," *IEEE Trans. Veh. Technol.*, vol. 51, no. 6, pp. 1652–1660, Nov. 2002.
[9] N. H. Vaidya and S. Hameed, "Scheduling data broadcast in asymmetric communication environments," *Wireless Netw.*, vol. 5, no. 3, pp. 171–182, May 1999.
[10] P. Nicopolitidis, G. I. Papadimitriou, and A. S. Pomportsis, "Continuous-flow wireless data broadcasting," *IEEE Trans. Broadcast.*, vol. 55, no. 2, pp. 260–269, Jun. 2009.
[11] S. Acharya, M. Franklin, and S. Zdonik, "Dissemination-based data delivery using broadcast disks," *IEEE Personal Commun.*, vol. 2, no. 6, pp. 50–60, Dec. 1995.
[12] P. Casari and A. F. Harris, III, "Energy-efficient reliable broadcast in underwater acoustic networks," in *Proc. 2nd Workshop Underwater Netw.*, Montreal, QC, Canada, Sep. 2007, pp. 49–56.
[13] P. Casari, M. Rossi, and M. Zorzi, "Towards optimal broadcasting policies for HARQ based on fountain codes in underwater networks," in *Proc. IEEE/IFIP 5th Annu. Conf. Wireless on Demand Netw. Syst. Services*, Garmisch-Partenkirchen, Germany, Jan. 2008, pp. 11–19.
[14] P. Casari, M. Rossi, and M. Zorzi, "Fountain codes and their application to broadcasting in underwater networks: Performance modelling and relevant tradeoffs," in *Proc. 3rd ACM Int. Workshop Underwater Netw.*, San Francisco, CA, Sep. 2008, pp. 11–18.
[15] T. Imielinski and S. Vishwanathan, "Adaptive wireless information systems," in *Proc. Special Interest Group Database Syst. Conf.*, 1994, pp. 19–41.
[16] C. Bechaz and H. Thomas, "GIB system: The underwater GPS solution," in *Proc. 5th Eur. Conf. Underwater Acoust.*, Lyon, France, May 2000.
[17] Kenneth and D. Frampton, "Acoustic self-localization in a distributed sensor network," *IEEE Sens. J.*, vol. 6, no. 1, pp. 166–172, Feb. 2006.
[18] A. Mahajian and M. Walworth, "3D position sensing using the differences in the time-of-flights from a wave source to various receivers," *IEEE Trans. Robot. Autom.*, vol. 17, no. 1, pp. 91–94, Feb. 2001.
[19] KORD DEFENCE PTY LTD [Online]. Available: http://wetpc.com.au/html/technology/wearable.htm
[20] S. Acharya, M. Franklin, and S. Zdonik, "Disseminating updates on broadcast disks," in *Proc. 22th Int. Conf. Very Large Data Bases*, Sep. 1996, pp. 354–365.
[21] S. Acharya, M. Franklin, and S. Zdonik, "Balancing push and pull for data broadcast," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Tucson, AZ, May 1997, pp. 183–194.
[22] C. J. Su and L. Tassiulas, "Broadcast scheduling for information distribution," in *Proc. IEEE 16th Annu. Joint Conf. Comput. Commun. Soc.*, Kobe, Japan, May 1997, pp. 109–117.
[23] N. H. Vaidya and S. Jiang, "Response time in data broadcast systems: Mean, variance and trade-off," in *Proc. Workshop Satellite Based Inf. Services*, Oct. 1998, pp. 52–59.
[24] P. Nicopolitidis, G. I. Papadimitriou, and A. S. Pomportsis, "Exploiting locality of demand to improve the performance of wireless data broadcasting," *IEEE Trans. Veh. Technol.*, vol. 55, no. 4, pp. 1347–1361, Jul. 2006.
[25] P. Nicopolitidis, G. I. Papadimitriou, and A. S. Pomportsis, "Multiple antenna data broadcasting for environments with locality of demand," *IEEE Trans. Veh. Technol.*, vol. 56, no. 5, pp. 2807–2816, Sep. 2007.
[26] V. Kakali, G. I. Papadimitriou, P. Nicopolitidis, and A. S. Pomportsis, "A new class of wireless push systems," *IEEE Trans. Veh. Technol.*, vol. 58, no. 8, pp. 2529–4539, Oct. 2009.
[27] K. S. NarendraM. A. L. Thathachar, *Learning Automata: An Introduction*. Englewood Cliffs, NJ: Prentice-Hall, 1989, ch. 1–9.
[28] G. I. Papadimitriou, "A new approach to the design of reinforcement schemes for learning automata: Stochastic estimator learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 6, no. 4, pp. 649–654, Aug. 1994.
[29] G. I. Papadimitriou, "Hierarchical discretized pursuit nonlinear learning automata with rapid convergence and high accuracy," *IEEE Trans. Knowl. Data Eng.*, vol. 6, no. 4, pp. 654–659, Aug. 1994.
[30] P. Nicopolitidis, G. I. Papadimitriou, and A. S. Pomportsis, "Learning-automata-based polling protocols for wireless LANs," *IEEE Trans. Commun.*, vol. 51, no. 3, pp. 453–463, Mar. 2003.
[31] G. I. Papadimitriou and A. S Pomportsis, "Learning automata-based TDMA protocols for broadcast communication systems with bursty traffic," *IEEE Commun. Lett.*, vol. 4, no. 3, pp. 107–109, Mar. 2000.
[32] G. I. Papadimitriou and A. S. Pomportsis, "Self-adaptive TDMA protocols for WDM star networks: A learning-automata-based approach," *IEEE Photon. Technol. Lett.*, vol. 11, no. 10, pp. 1322–1324, Oct. 1999.
[33] A. A. Economides, P. A. Ioannou, and J. A. Silvester, "Decentralized adaptive routing for virtual circuit networks using stochastic learning automata," in *Proc. IEEE 7th Annu. Joint Conf. Comput. Commun. Soc.*, New Orleans, LA, Mar. 1988, pp. 613–622.
[34] A. A. Economides, "Learning automata routing in connection-oriented networks," *Int. J. Commun. Syst.*, vol. 8, no. 4, pp. 225–237, Jul.–Aug. 1995.
[35] M. H. Ammar and J. W. Wong, "On the optimality of cyclic transmission in teletext systems," *IEEE Trans. Commun.*, vol. COM-35, no. 1, pp. 68–73, Jan. 1987.
[36] R. Jain and J. Werth, "Airdisks and airRAID: Modeling and scheduling periodic wireless data broadcast," Rutgers—The State Univ. New Jersey, Piscataway, NJ, Tech. Rep. DIMACS TR: 95-11, 1995.
[37] D. Pompili, T. Melodia, and I. F. Akyildiz, "A CDMA-based medium access control for underwater acoustic sensor networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 4, pp. 1899–1909, Apr. 2009.
[38] H.-P. Tan and W. K. G. Seah, "Distributed CDMA-based MAC protocol for underwater sensor networks," in *Proc. 32nd IEEE Conf. Local Comput. Netw.*, Dublin, Ireland, Oct. 2007, pp. 26–36.
[39] A. Kebkal, K. Kebkal, and M. Komar, "Data-link protocol for underwater acoustic networks," in *Proc. IEEE OCEANS Europe Conf.*, Brest, France, Jun. 2005, pp. 1174–1180.
[40] K. S. Gilhousen, I. M. Jacobs, R. Padovani, A. J. Viterbi, L. A. Weaver, and C. E. Wheatley, III, "On the capacity of a cellular CDMA system," *IEEE Trans. Veh. Technol.*, vol. 40, no. 2, pp. 303–312, May 1991.
[41] P. Nicopolitidis, G. I. Papadimitriou, and A. S. Pomportsis, "An adaptive wireless push system for high-speed data broadcasting," in *Proc. IEEE Workshop Local Metropolitan Area Netw*, Chania, Greece, Sep. 2005, DOI; 10.1109/LANMAN.2005.1541528.
[42] J. L. Talavage, T. E. Thiel, and D. Brady, "An efficient store-and-forward protocol for a shallow-water acoustic local area network," in *Proc. OCEANS Conf.*, Brest, France, Sep. 1994, pp. 1883–1888.

Petros Nicopolitidis (M'08) received the B.S. and Ph.D. degrees in computer science from the Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece, in 1998 and 2002, respectively.

From 2004 to 2009, he was a Lecturer at the Department of Informatics, Aristotle University of Thessaloniki, where since 2009, he has been an Assistant Professor. He has published more than 60 papers in international refereed journals and conferences. He is coauthor of the book *Wireless Networks* (New York, NY: Wiley, 2003). His research interests are in the areas of wireless networks and mobile communications.

Dr. Nicopolitidis has been an Associate Editor for the *International Journal of Communication Systems* since 2007.

Georgios I. Papadimitriou (M'90–SM'02) received the Diploma and Ph.D. degrees in computer engineering from the University of Patras, Patras, Greece, in 1989 and 1994, respectively.

From 1989 to 1994, he was a Teaching Assistant at the Department of Computer Engineering, 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, Thessaloniki, Greece. From 2001 to 2006, he was an Assistant Professor at the Department of Informatics, Aristotle University of Thessaloniki. Since 2006, he has been an Associate Professor at the Department of Informatics, Aristotle University of Thessaloniki. He is the coauthor of the books *Multiwavelength Optical LANs* (New York, NY: Wiley, 2003) and *Wireless Networks* (New York, NY: Wiley, 2003) and the coeditor of the book *Applied System Simulation* (Norwell, MA: Kluwer, 2003). He is the author of more than 100 refereed journal and conference papers. His research interests include optical networks, wireless networks, high-speed local area networks (LANs), and learning automata.

Prof. Papadimitriou is an Associate Editor of five scholarly journals, including the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, the IEEE TRANSACTIONS ON BROADCASTING, and the IEEE COMMUNICATIONS MAGAZINE.

Andreas S. Pomportsis received the B.S. degree in physics and the M.S. degree in electronics and communications from the University of Thessaloniki, Thessaloniki, Greece, in 1978 and 1981, respectively, the Diploma degree in electrical engineering from the Technical University of Thessaloniki, Thessaloniki, Greece, in 1985, and the Ph.D. degree in computer science from the University of Thessaloniki, 1987.

Currently, he is a Professor at the Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece. He is the coauthor of the books *Wireless Networks* (New York, NY: Wiley, 2003) and *Multiwavelength Optical LANs* (New York, NY: Wiley, 2003). His research interests include computer networks, learning automata, computer architecture, parallel and distributed computer systems, and multimedia systems.