Providing Real-Time Information in Ubiquitous Computing Environments

Ding-Jung Chiang
Dept. of Information Management
Northern Taiwan Institute of Science and Technology
Taipei, Taiwan, R.O.C.
Email: djchiang@mis.tsint.edu.tw

Hwei-Jen Lin
Dept. of CSIE
Tamkang University
Taipei, Taiwan, R.O.C.
Email: hjlin@cs.tku.edu.tw

Timothy K. Shih
Dept. of Computer Science
National Taipei University of Education
Taipei, Taiwan, R.O.C.
Email: timothykshih@gmail.com

***Abstract*—In a ubiquitous computing system, users carrying portable devices can access database services from any location without requiring a fixed position in the networking environment. Some examples of strategies supported by databases in mobile computing include location dependent queries, long-lived transactions that require migration of data into the portable devices, form-based transactions, and online information report. In a ubiquitous computing environment, the need for a real-time database management model is strong, because one of the basic requirements in mobile data management is to provide real-time response to transactions of the underlying strategies. However, the resource constraints of ubiquitous computing systems make it difficult to satisfy timing requirements of supported strategies. Low bandwidth, unreliable wireless links, and frequent disconnections increase the overhead of communication between mobile hosts and fixed hosts of the system. There are many situations in which we need to incorporate real-time constraints in broadcasting systems for mobile environments. In this paper, we study broadcast scheduling strategies for push-based broadcast with timing constraints in the form of deadlines. Unlike previously proposed scheduling algorithms for broadcast systems which aim to minimize the mean access time, our goal is to identify scheduling algorithms for broadcast systems that ensure requests meet their deadlines. We present a study of the performance of traditional real-time strategies and mobile broadcasting strategies, and demonstrate that traditional real-time algorithms do not always perform the best in a mobile environment. We propose a multichannel model based on push-based real-time broadcast system and also provide an efficient scheduling algorithm, called dynamic adjustment with time constraint (DATC), which is designed for timely delivery of data to mobile clients.**

***Keywords*—mobile computing,multichannel broadcasting, time constraint**

1. INTRODUCTION

A typical mobile computing system consists of a number of mobile and fixed hosts. A fixed host (FH) is connected with each other via fixed high-speed wired network and constitutes the fixed part of the system. A mobile host (MH) is capable of connecting to the fixed network via a wireless link. Some of the fixed hosts, called mobile support stations (MSS), are augmented with a wireless interface to communicate with mobile hosts. The links between mobile computers and the MSSs can change dynamically. In our model, we assume that all fixed hosts act as mobile support stations (MSS). Each MSS has a database server which enforces strict data consistency. Each mobile host is associated with a coordinator MSS that coordinates the operations of the transactions submitted by that mobile host. Transaction execution descriptions are as the followings:

- Mobile host process (MHP): A mobile transaction exists at the generating mobile host.
- Fixed host process (FHP): A fixed transaction exists at the coordinator MSS of the generating host.
- Fixed cohort process (FCP): FCP at sides where the required data page reside.

Broadcast delivery [11][10] has been proposed and proven to be an efficient way of disseminating data to the mobile client population. This is due to the asymmetric nature of wireless communication, i.e., the downlink bandwidth is much higher than the uplink bandwidth. Associated with broadcast delivery is the problem of how to schedule the broadcasting of the requests to minimize the wait time of the clients. The wait time is also referred to as mean data access time, which is the average amount of time from the arrival of a request, to the time that the requested page is broadcast. With broadcasting, the server can satisfy all pending requests on a data item simultaneously, thus, eliminating the potentially very large overhead of data requests, and saving both the wireless bandwidth and a mobile client's battery energy. Another feature is that it greatly increases the scalability of the broadcast system by keeping the server from being swamped with data requests With the rapid growth of time-constraint information services and business-oriented applications, there is an increasing demand to support quality of service (QoS) in mobile environments. In many situations, user requests are associated with time constraints as a measure of QoS. These constraints can be imposed either by the users or the applications. For example, the timing of buying/selling stocks for a stock holder is very crucial. If the stock information cannot reach a stock holder in time, the information might become useless. For another example, the information about traffic congestion that is caused by a traffic control should also reach a mobile client heading toward this direction timely. If a client receives such information early enough, the client is able to react accordingly to avoid the traffic jam. The value of the information would degrade

significantly when the client gets closer to the spot of the control. With data broadcasting approach a broadcast server can serve many mobile clients simultaneously. Therefore, data broadcasting is usually adopted for disseminating data in mobile computing environments. Most of the related current research focuses on a data broadcasting approach, where the transmission of data is done without considering the data items with time constraints. In this study, we present an online scheduling algorithm to maximize the total number of satisfied users in asymmetric communication environments with time requirements. This is achieved by means of dynamic adaptation of the broadcast program to the needs of the users, taking into account the bandwidth constraints inherent in asymmetric communication environments and the deadline requirements of the user requests. The goal of our research is to study broadcast scheduling strategies on the multichannel systems for data broadcast with timing constraints. In such a broadcasting environment, the goal is to determine how well the scheduling algorithms ensure that the database server does not miss deadlines, instead of minimizing wait time. There are several scheduling algorithms for multichannel systems in mobile environments [8][9]. However, we demonstrate that traditional well-known algorithms do not always perform the best in a mobile environment, such as greedy and dynamic programming, when they are applied with time constraints on the multichannel systems in a mobile environment. We propose a model of a multichannel broadcast system for a simulation analysis and also propose an efficient scheduling algorithm called dynamic adjustment with time constraint (DATC). The rest of the paper is organized as follows. In section 2 we discuss related work for real-time strategies and mobile data dissemination. In section 3 we describe our model and propose the DATC approach. In section 4 we present the experimental results and simulation environment. Finally, we conclude the paper in section 5.

2. Related Work

Scheduling of transactions for real-time databases in a non-mobile environment is studied extensively in [1]. A real-time client/server model is considered in which the server assigns priorities to transactions based on several strategies, including Earliest Deadline First (EDF) and Least Slack (LS) first. As its name implies, for EDF the transaction with the earliest deadline is given the highest priority. For LS, the slack time is defined as: $d-(t+E-P)$, where d is the deadline, t is the current time, E is the execution time and P is the processor time used thus far. If the slack time is ≥ 0, it means that the transaction can meet its deadline if it executes without interference. The slack time indicates how long a transaction can be delayed and still meet its deadline. The Least Slack LS differs from EDF because the priority of a transaction depends on the service time it has received. If a transaction is restarted, its priority will change. Simulation results show that the EDF is the best overall policy for real-time database systems in a non-mobile environment. However, when system loads are high, the LS and EDF strategies lose their advantage, even over FCFS, as most transactions are likely to miss their deadlines.

For push-based systems, the longest wait first LWF algorithm has been shown to outperform all other strategies at minimizing wait time [13]. In LWF, the sum of the total time that all pending requests for a data item have been waiting is calculated, and the data item with the largest total wait time is chosen to broadcast next. However, LWF has been recognized as expensive to implement. In [4] a strategy, called requests times wait (RxW), is presented for push-based systems that makes scheduling decisions based on the current state of a queue (instead of access probabilities). The RxW algorithm provides an estimate of the LWF algorithm by multiplying the number of pending requests for a data item times the longest request wait time. In general, the performance of the approximate algorithms has been shown to be close to LWF.

There has been some research work to consider broadcasting for mobile real-time systems [5]. A push-based protocol for organizing broadcast disks for real-time applications, called Adaptive Information Dispersal Algorithm (AIDA), is presented in [3]. In this work, the data must be broadcast periodically to satisfy the timing constraints. The AIDA protocol considers fault-tolerance and the data items are allocated to the broadcast disks to minimize the impact of intermittent failures by utilizing redundancy. AIDA guarantees a lower bound on the probability of meeting timing constraints. Similar work addressing fault-tolerant real-time broadcast disks appears in [2]. In this work, the authors show that designing strategies for real-time broadcast disks is related to pinwheel scheduling. The authors derived a pinwheel algebra, which utilizes rules that can be used to construct fault-tolerant real-time broadcast disks. Their work differs from our work because we assume that we schedule all data items with time constraints using adaptive algorithms under limited bandwidth to minimize miss rate. In the multichannel broadcast disks model, the server periodically repeats a computed broadcast program, based on user access patterns. A broadcast cycle is defined as one transmission of the periodic broadcast program. Deadline constraints have been integrated into the broadcast model in [8]. In order to minimize the total number of deadlines missed by making the most effective use of the available bandwidth, scheduling approach has to focus on critical factors such as access frequency, time constraint, and bandwidth requirements. In [15], scheduling mechanisms for broadcasting data that are to minimize the delay incurred by insufficient channels, but it is reasonable that all clients are satisfied with an expected time to optimize average access time.

3. Broadcasting with Time Constraints

We now describe a framework to support the push-based broadcast scheduling problem with time constraints. In this section, the real-time scheduling problem, system architecture and solving mechanism are introduced.

3.1. System Architecture

Server side: We assume that there are K channels in a broadcast area, each channel denoted $C_i, 1 \leq i \leq K$. A database is made up of N unit-sized items, denoted $d_j, 1 \leq j \leq N$. Each item is broadcast on one of these channels, so channel C_i broadcasts N_i items, $1 \leq i \leq K, \sum_{i=1}^{K} N_i = N$. Each channel cyclically broadcasts its items. Time is slotted into units called ticks. The size of data item is fixed and equal to one tick. Each data item is denoted $d_i\,(id_i, t_i, p_i)$ by the following parameters[6]:

- id: identifier of data item.
- t_i: relative deadline, i.e. the maximum acceptable delay for its processing.
- p_i: access probability for d_i.

Requests are for single item and assumed to be exponentially distributed.

Client side: Each client can require one data item per request associated with a time constraint. When a client needs a data item, it first tunes in the broadcast channels to retrieve the contents of channel. By the information of channels, the client can determine whether he can get the data item from the broadcast channels. If the needed data item is in the broadcast data set, the client tunes in the broadcast channels and retrieves the desired data item. Otherwise, the client sends a request to the server via the uplink channel, and listens to the broadcast channels to retrieve the data pages.

3.2. Problem Formulation

We formulate our problem to make it a resolvable problem as follows. Given a number of data items N to be broadcast in multiple broadcast channels K. Each data item is associated with a time constraint. Every access of a client is only one data item. Expected delay, w_i,is the expected number of ticks a client must wait for the broadcast of data item d_i. Average expected delay is the number of ticks a client must wait for an average request and is computed as the sum of all expected delays, multiplied by their access probabilities:

$$Average\ Expected\ Delay(W) = \sum_{i=1}^{N} w_i p_i \qquad (1)$$

,where w_i is expected delay[14] and p_i is access probability for data item d_i respectively. With time constraints, a request for data item d_i has missed its deadline when timing fault(expected delay for data item d_i exceeds its time constraint $t_i < W$) occurred at some time slot. The miss rate of all data items is defined as follows:

$$Miss\ Rate = \sum_{j=1}^{K} \sum p_i \qquad (2)$$

Our goal is to broadcast all data items with time constraints on multiple broadcast channels that minimizes the miss rate.

3.3. Design of Algorithm DATC

We provide an algorithm to generate a valid broadcast program so as to minimize miss-rate. If miss-rate is zero, let average access time minimized. We formulate our problem and make appropriate assumptions to make it a resolvable problem as follows.

Let each item contains two attributes: access probability and time constraint. Given a database D with its size $|D| = N$ and the number of channels = K, we aim to allocate each item in D into K channels, such that N items are cyclically broadcast on multi-channel, the miss rate can be written as: $\sum_{i=1}^{K} \left(\sum_{d_i \in c_i} p_i \right)$. Given an example using above the assumptions, we make a comparison between greed algorithm [12] and our algorithm DATC.

Example 1. *Given K = 3 broadcast channels, consider a set of N = 10 data items, $\{d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_9, d_{10}\}$, with the following skewed access probabilities and time constraints:*

d_1	d_2	d_3	d_4	d_5
0.199	*0.140*	*0.114*	*0.099*	*0.089*
8	*3*	*3*	*10*	*9*
d_6	d_7	d_8	d_9	d_{10}
0.081	*0.075*	*0.070*	*0.066*	*0.062*
2	*6*	*10*	*8*	*7*

4. Experimental Results

4.1. Simulation Environment

In our real-time broadcast simulation model, bandwidth is not explicitly modelled. Instead, similar to previous work [4], we use broadcast ticks as a measure of time. The greatest advantage of this approach is that the results are not limited to any particular bandwidth and/or data item size. Rather, it aims to capture the fundamental characteristics of the systems. The model simulates a one-hop wireless network. All data items are stored in a data server in a fixed location. Mobile clients need to send requests to the server via an uplink back-channel before the requested page can be broadcast. The arrival of requests generated by mobile clients follows a Poisson process and the inter-arrival time is exponential with mean λ. Each request has a request id, arrival time and deadline. For each page, a queue is maintained to store the information about requests on the object. We assume the results produced after a deadline are useless (firm deadlines), so all requests that have missed their deadlines are discarded. Mobile clients are responsible for re-sending requests when link errors occur. We also assume a deadline can capture the mobility of clients who are no longer able to receive the broadcast. In our model, since newly generated data requests are sent to the server immediately, the request generating time is equal to the time the server receives it (assuming network delay is ignored). We also ignore the overheads of request processing

Algorithm *DATC(int N, int K, float P, int T)*
($*$ N: number of items, K: numbers of channels, P: access probabilities, T: time constraints $*$)
Input: set of N unit sized items ordered by popularity, K channels.
Output: K partitions to minimize miss-rate.

```
1.  Partition_number = 1;
2.  while (Partition_number < K)  do
3.      for each partition k with data items i through j  do
4.          (* Find the best point s to split in partition k *)
5.          for (s = i; s ≤ j; s = s + 1)  do
6.              (* Initialize the best split point for this partition as the first data item. If we find a better one subsequently, update
                the best split point. *)
7.              if ((s = i) or (Local_change > C_ij^5))
8.                  (* C_ij^5 is computed as the expected delay of a data item in a channel of size j − i + 1. *)
9.                  (* Initialize the best solution as the one for the first partition. If we find a better one subsequently, update
                    the best solution. *)
10.               then Local_S = s
11.                   Local_change = C_ij^5
12.             if ((k = 1) or (Global_change > Local_change))
13.               then Global_change = Local_change
14.                   Global_S = Local_S
15.                   Best_part = k
16.             Split partition Best_part at point Global_P
17.             Partition_number = Partition_number + 1
18.             for data items with time constraints on each channel  do
19.                 Order data items by time constraints
20.                 Earliest deadline first to broadcast
```

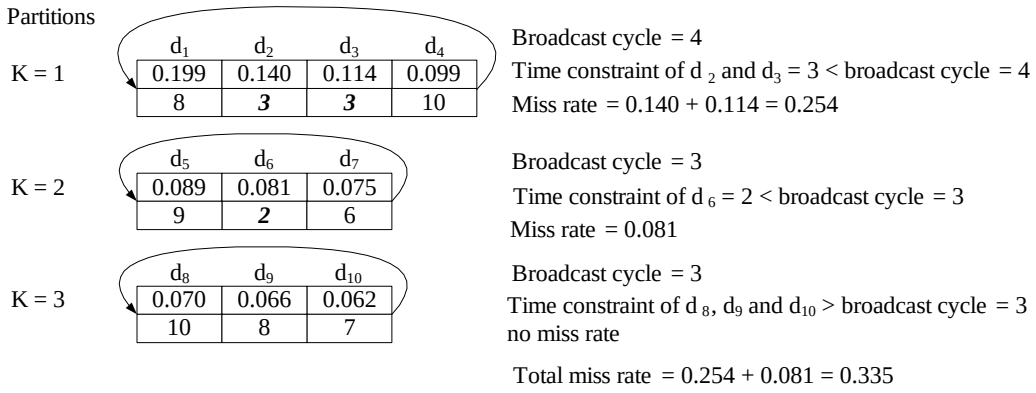

Fig. 1. Broadcasting 10 data items with time constraints to partition 3 disjoint subsets and the miss rate = 0.335.

at the server, because the main purpose of the model is to compare the scheduling power of various strategies. We assume requests generated by mobile clients are read only, and no update request is allowed. Concurrency control issues are not our main concern, and thus, not considered. At each tick of the simulation clock, the following occurs. A simulated request generator generates requests with exponential inter-arrival time. The information about each request id, arrival-time and deadline is recorded. The request is then inserted to the corresponding queue. The server checks the deadlines of all the arrived requests, and discards those requests that have missed their deadlines. Then the server selects a page to broadcast by applying a scheduling strategy and starts to broadcast the selected page. All requests requesting the page are satisfied when the broadcast is finished. A client can request multiple pages and a page can be requested by multiple mobile clients at a time. We assume that data demand probabilities p_i follow the Zipf [7] distribution in which:

$$p_i = \frac{(1/i)^\theta}{\sum_{i=1}^{M} (1/i)^\theta}, \quad (i = 1, 2, 3, \ldots, M)$$

where p_i represents the i'th most popular page. The Zipf distribution allows the pages requested to be skewed. Figure 2 shows the results of our simulation comparing the DACT strategy to the strategy EDF for uniformly distributed dead-

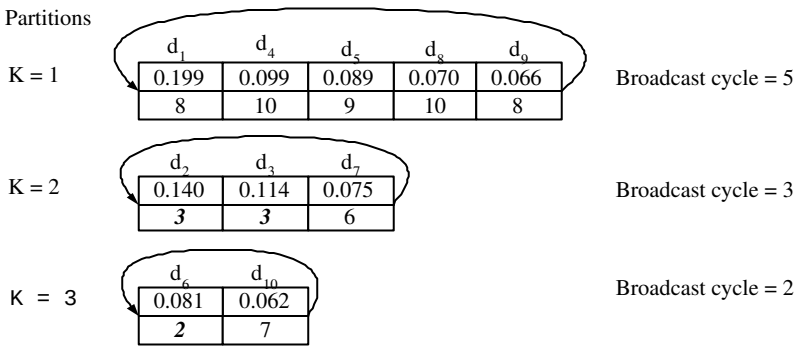

Fig. 2. Using DATC algorithm to partition 3 disjoint subsets and the miss rate = 0.

Symbol	Description	Default	Range	Unit
DBSIZE	Total number of data pages stored in server	100	100-10000	pages
λ	Mean request arrival rate (exponential)	-	2-60	requests/tick
θ	Request skewness (Zipf)	1.0	0.0-1.0	-
$MinSlack$	Minimum slack time	1.0	-	ticks
$MaxSlack$	Maximum slack time	-	20-300	ticks
$\lambda_{deadline}$	Parameter of exponential deadline distribution	-	10-300	ticks

TABLE I
SIMULATION PARAMETERS

lines.

4.2. Simulation Parameters

We compare the DACT approach with the algorithms described in Section 2: EDF, LSF, RxW, and LWF. Figure 3 illustrates the distribution of miss rate. We only choose the push-based algorithms to compare the results since we believe these push-based algorithms better adapt to the dynamic changes of the intensity and distribution of system workloads. The push-based (access probabilities, broadcast histories, etc.) off-line algorithms are not considered due to the fact that they are mainly for fairly stable systems. We implement the simulation model described in the previous section using C++. In each experiment, we run the simulation for 5000 time units, and we use an average of 20 runs of each simulation as the final result. The Parameters used in this simulation are summarized in Table 1. The default total number of data pages stored in the server, referred to as DBSIZE, is 100 pages. Client requests reach the system with exponential inter-arrival time with mean λ, and λ is varied in our simulation from 2 - 60. It is assumed that each data request requires 1 broadcast tick to broadcast. An open system model is used to simulate the system for extremely large, highly dynamic populations. Data access follows a skewed Zipf distribution with parameter Θ to control the skewness. The minimum slack time is 10, with the maximum slack time ranging from 20 to 300. This variation in maximum slack time allows us to vary the tightness in the deadlines. In addition to a uniform distribution of deadlines, an exponential distribution is utilized with lambda ranging from 10 to 300. After doing a large number of experiments with various factors that affect the performance, we come up with an overall performance comparison between the previous algorithms and our scheduling algorithm DATC in Table 2. We grade the level of performance from 1 to 5. The higher the degree is, the better the performance is. On the contrary, overhead with higher degree shows that the algorithm gets more cost in Table 2.

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT SCHEDULING ALGORITHMS

	DMR	ART	AS	Overhead
LWF	1	2	3	5
LSF	1	1	2	5
RxW	3	5	4	4
EDF	3	3	1	4
DATC	5	5	4	3

DMR = Deadline Miss Rate
ART = Average Response Time
AS = Average Stretch

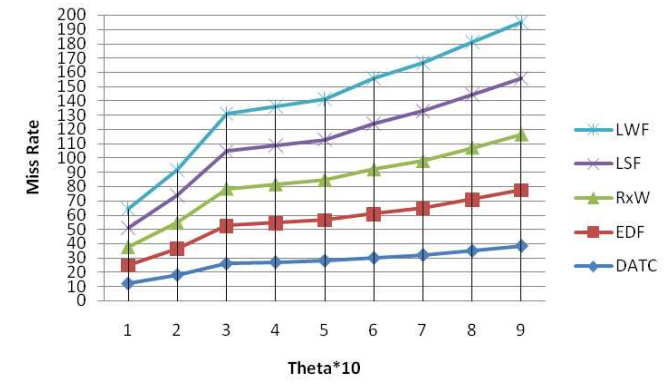

Fig. 3. Deadlines missed

5. Conclusion

In this paper, we present an analysis model as well as a simulation model for push-based real-time broadcast systems. As this paper demonstrates, the traditional well-proven strategies, like EDF and LS in the non-mobile real-time environment, do not perform efficiently in a broadcast system environment. We propose an efficient scheduling algorithm, called dynamic adjustment with time constraint (DATC), which is designed for timely delivery of data to mobile clients. We compare DATC with traditional client/server based real-time scheduling strategies and mobile non-real-time broadcast strategies. The proposed DATC strategy is shown to generally outperform the existing real-time strategies, with different deadline distributions. In the future we plan to improve the DATC strategy by reducing its scheduling time complexity. Other concentrations include investigation of real-time scheduling algorithms that can handle transmission errors, update requests, unfixed page size and multi-hop communication.

References

[1] R. Abbott and H. Garcia-Molina. Scheduling real-time transactions: A performance evaluation. *ACM Transactions on Database Systems*, 17:513–560, 1992.

[2] S. Baruah and A. Bestavros. Pinwheel scheduling for fault-tolerant broadcast disks in real-time database systems. In *Proceedings of the 13th International Conference on Data Engineering.*, pages 543–551, April 1997.

[3] A. Bestavros. Aida-based real-time fault-tolerant broadcast disks. In *Proceedings of Real-Time Technology and Applications Symposium.*, pages 49–58, 1996.

[4] Michael Franklin Demet Aksoy. Scheduling for large-scale on-demand data broadcasting. In *In Proceedings of the INFOCOM Conference*, pages 651–659, March 1998.

[5] J. Fernandez-Conde and K. Ramamritham. Adaptive dissemination of data in time-critical asymmetric communication environments. In *Proceedings of the 11th Euromicro Conference on Real-Time Systems*, pages 195–203, 1999.

[6] C. Kaiser Z. Mammeri Francis Cottet, J. Delacroix. Scheduling in real-time systems. John Wiley and Sons, Ltd, 2002.

[7] G.K.Zipf. Human behaviour and the principle of the least effort. In *Proceedings of the 25th International Conference on Distributed Computing Systems*, Reading,MA, 1949. Addison-Wesley.

[8] K. A. Hua K. Prabhakara and J.-H. Oh. Multi-level multi-channel air cache designs for broadcasting in a mobile environment. In *Proceeding of the 16th International Conference on Data Engineering.*, pages 167–176, February 2000.

[9] D. L. Lee Q. L. Hu and W.-C. Lee. Dynamic data delivery in wireless communication environments. In *Proceedings of International Workshop on Mobile Data Access*, pages 218–229, November 1998.

[10] M. Franklin S. Acharya and S. Zdonik. Dissemination-based data delivery using broadcast disks. *IEEE Personal Communications*, 2(6):50–60, December 1995.

[11] M. Franklin S. Acharya, R. Alonso and S. Zdonik. Broadcast disks: data management for asymmetric communication environments. In *Proceeding of ACM SIGMOD*, pages 199–210, March 1995.

[12] C. E. Leiserson T. H. Cormen and R. L. Rivest. Introduction to algorithms. The MIT, 1992.

[13] John W. Wong. Broadcast delivery. *Proceedings of the IEEE*, 76(12):1566–1577, Dec. 1988.

[14] Wai Gen Yee and Shamkant B. Navathe. Efficient data access to multi-channel broadcast programs. In *Proceedings of the 12th International Conference on Information and Knowledge Management.*, pages 153–160, 2003.

[15] Chao-Chun Chen Yu-Chi Chung and Chiang Lee. Time constrained service on air. In *Proceedings of the 25th International Conference on Distributed Computing Systems*, pages 739–748, June 06-10 2005.