Information Systems 67 (2017) 58–70

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is

Dynamic priority scheduling of periodic queries in on-demand data dissemination systems

Quan Zhou[a], Guohui Li[a], Jianjun Li[a,*], LihChyun Shu[b], Cong Zhang[c], Fumin Yang[a]

[a] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, PR China
[b] National Cheng Kung University, Taiwan, ROC
[c] Wuhan Polytechnic University, Wuhan, Hubei, PR China

ARTICLE INFO

Article history:
Received 15 July 2016
Revised 19 January 2017
Accepted 22 March 2017
Available online 27 March 2017

Keywords:
On-demand
Data dissemination
Bandwidth consumption
Dynamic priority
Periodic queries

ABSTRACT

As an important wireless data broadcast technique, on-demand broadcast has been widely used for dynamic and large-scale data dissemination. An important class of emerging data broadcast applications requires monitoring multiple data items continuously in order to support data-driven decision making. Since wireless bandwidth is a precious shared medium, an important problem to solve is how to disseminate data to periodic queries, so that all the requests can be satisfied while the bandwidth consumption is minimized. In this paper, we first propose a new real-time scheduling algorithm called EDFS, which is a variant of the classic EDF [24] algorithm. Based on EDFS, we propose a novel on-line broadcast scheduling algorithm, called EDFS-BS. To our best knowledge, EDFS-BS is the first dynamic priority based broadcast scheduling algorithm that can be utilized to satisfy the timing constraints of periodic queries. We also propose a bandwidth utilization based schedulability test for EDFS-BS, which is used to ensure timing predictability of a periodic query set. Extensive experiments have been conducted to compare EDFS-BS versus existing solutions with comparable quality. The results show that EDFS-BS outperforms them considerably in terms of wireless bandwidth consumption and query service ratio.

1. Introduction

Since the introductory article by Acharya et al. [1], data broadcast has become a popular data dissemination and access method, due to its distinguishing feature of satisfying all pending requests for the same data item with a single transmission. In general, data broadcast can be classified into two categories: push-based broadcast [3,27] and on-demand broadcast [18,32]. The push-based approach periodically broadcasts a set of predetermined data items, based on some prior knowledge of data access patterns, regardless of individual information requirements. Hence, push-based broadcast is useful for applications with relatively stable access patterns. In contrast, on-demand broadcast is more suitable and widely used for dynamic and large-scale data dissemination. Moreover, for applications that involve queries with time constraints, on-demand broadcast is more suitable than push-based method because it could take each client's timing characteristics and data requirements into consideration while determining the broadcasting period of the data items for the client.

Recently, mobile data services with real-time support have received increasing attention in the research community. A time-critical request is associated with a deadline imposed by either the application or the user, and the results produced in response to the request are useful only if all the required data items can be received before the deadline. Consider the following three real-time data broadcast examples: (1) A basketball fan intending to watch NBA Finals live via his mobile phone receives latest game updates every minute; (2) A stock investor wants to refresh his stock information through his mobile device once every minute to decide whether or not to trade his stock. (3) Before a vehicle is approaching a traffic intersection, the driver inquires the best route to a destination, the navigation system needs to keep monitoring the traffic conditions of different roads, and the most up-to-date values of different pieces of information should be returned in time to help the driver make the decision before the vehicle passing the intersection. One distinct characteristic of such applications is that queries access multiple time varying data items and keep running for a relatively long time. Existing real-time data broadcast research mainly focuses on how to generate broadcast schedules so as to meet the timing constraints of single-item requests [4] or one-shot queries [9,19], with the objective of minimizing queries' average response time or reducing queries' deadline miss ratio.

* Corresponding author.
E-mail address: jianjunli@hust.edu.cn (J. Li).

http://dx.doi.org/10.1016/j.is.2017.03.005
0306-4379/

Table 1
Original queries.

Queries	q_1	q_2	q_3	q_4
Periods	2	3	5	10
Data items	$\{d_1\}$	$\{d_1\}$	$\{d_1, d_2, d_3\}$	$\{d_1, d_2, d_4\}$
unique sets	$\{d_1\}$	$\emptyset$	$\{d_2, d_3\}$	$\{d_4\}$
shared sets	$\emptyset$	$\{d_1\}$	$\{d_1\}$	$\{d_1, d_2\}$

Table 2
Corresponding tasks.

Data items in the *unique* sets	d_1	d_2	d_3	d_4
DC tasks	τ_1	τ_2	τ_3	τ_4
Original periods	2	5	5	10
2-harmonic periods	2	4	4	8

These approaches may not perform as expected, or even not be applicable to the applications mentioned above. It is therefore essential to design new scheduling algorithms to process such periodic queries with real-time requirements.

To our best knowledge, the work in [30] and [23] presented the only two algorithms (RM-UO and UM) that are suitable for supporting periodic queries processing. However, in these two algorithms, each data item accessed by queries generates a Distance-Constrained (DC) task to guarantee that once the time constraint of the generated DC task is satisfied, all the queries that require this data item can obtain it in each of their periods. Since the DC tasks are finally transformed to 2-harmonic tasks in these two algorithms, only fixed-priority real-time scheduling algorithms were utilized to schedule them. In uniprocessor real-time scheduling, a well-known fact is that dynamic priority scheduling is superior to static priority scheduling in terms of processor utilization [13]. It is natural to ask if a similar performance pattern can be observed in the broadcasting environment in terms of wireless bandwidth consumption. This is an important topic to explore as wireless bandwidth is a precious shared medium. In this article, we try to utilize the dynamic priority scheduling algorithm to address the periodic query processing problem.

Consider a simple example shown in Table 1. There are four queries with periods 2, 3, 5 and 10. The data items they need to access are $\{d_1\}$, $\{d_1\}$, $\{d_1, d_2, d_3\}$ and $\{d_1, d_2, d_4\}$, respectively. Under RM-UO, the data items each query requires will be divided into the *unique* set and the *shared* set. Specifically, each data item in the *unique* set of a query has the shortest period among all the queries requiring it. The *shared* set of a query consists of the rest data items accessed by it. The data set division result for the above example is shown in Table 1. After the data set division operation, the DC task set (presented in Table 2) is generated, where one DC task corresponds to one data item in the *unique* sets. Note here the *shared* sets are ignored, since the data items in the *shared* set of a query can be shared from the *unique* set of other queries. After obtaining the DC task set, RM-UO invokes the `Sr` [16] algorithm to transform the DC task set into a 2-harmonic[1] task set with periods 2, 4, 4 and 8, respectively. Since the bandwidth utilization of the first three 2-harmonic tasks ($\frac{1}{2} + \frac{1}{4} + \frac{1}{4}$) has reached the upper bound 100%, RM-UO fails to provide service for the fourth query. Moreover, since there are no tasks that can be merged by either MQM or RQM [23], the UM algorithm is also unable to provide service for the fourth query.

[1] We call a task set $\mathcal{T} = \{\tau_1, \ldots, \tau_n\}$ 2-harmonic if the period of any task τ_k, denoted by T_k, is 2^x ($x \in 0 \cup N^+$) times of other task periods which are not greater than T_k. For example, the task set with periods $T_1 = 3$, $T_2 = 6$, $T_3 = 6$, $T_4 = 24$ is 2-harmonic.

The reason why the fourth query cannot be serviced because both RM-UO and UM treat all the tasks as DC tasks, and transform them into 2-harmonic tasks to schedule. Obviously, when transforming to 2-harmonic tasks (shortening the periods of the DC tasks), the utilization of the task set will be increased, which has adverse effects on the schedulability of the task set. We argue that, however, it is not necessary to treat all the generated tasks as DC ones. Instead, some generated tasks can be viewed as real-time tasks, while the objective of processing periodic queries can still be achieved. In this way, the real-time tasks can be scheduled under EDF [24] to save more bandwidth, and consequently, more queries can be served. As an illustration, Fig. 1 gives a feasible schedule for the above example, in which all the four queries $\{q_1, q_2, q_3, q_4\}$ can get services in each of their periods. As can be seen, the time interval between two adjacent broadcasts of d_2 exceeds 5 – its original period, which means there is no need to treat τ_2 as a DC task. In fact, in this example, only τ_1 should be treated as a DC task and the remaining ones can be considered as real-time tasks. Consequently, there is no need to shorten the periods of τ_2, τ_3 and τ_4. In this case, the total bandwidth utilization of the final task set ($\frac{1}{2} + \frac{1}{5} + \frac{1}{5} + \frac{1}{10}$) happens to be 100%, which means the task set is schedulable under EDF, also indicating that all the queries can get their services.

Inspired by the above observation, we propose a novel broadcast scheduling algorithm to provide services for periodic queries. The main contributions of our work can be summarized as follows.

- We propose EDFS, a variant of the classic EDF scheduling algorithm, which can be used to schedule real-time tasks in the data broadcast environment. We also provide a necessary and sufficient schedulability test for EDFS.
- Based on EDFS, we propose, to our knowledge, the first dynamic priority based broadcast scheduling algorithm, EDFS-BS, which comprehensively considers the real-time characteristics of the tasks, the sharing feature of the broadcast data items and the continuity of the broadcast services. We also analyze the schedulability of EDFS-BS and provide a bandwidth utilization based schedulability test for it.
- Extensive experiments have been conducted to compare EDFS-BS with existing solutions with comparable quality. The experimental results demonstrate that EDFS-BS can result in significant performance improvement as compared to its competitors, in terms of service ratio and bandwidth consumption.

The rest of this paper is organized as follows: Section 2 gives our system model, along with some basic assumptions. Section 3 introduces the EDFS algorithm. Based on EDFS, we detail the design of EDFS-BS, analyze the schedulability and describe the implementation issues of it in Section 4. Experimental results are shown and discussed in Section 5. Section 6 reviews related work. Finally, Section 7 draws a conclusion of this paper with a brief discussion on future work.

2. Model and assumptions

2.1. System model

We consider a typical on-demand broadcast model as depicted in Fig. 2. In this model, mobile devices issue queries to the server through an up-link channel and the server accepts queries and schedules data items to be accessed by these queries. Note that we assume mobile devices issue queries *periodically*, and each query instance has a deadline, which is the starting time of the query instance following the current instance. Moreover, a query is allowed to require multiple data items. A query instance is completed *if and only if* it obtains all the required data items before its deadline. As in [30], when the server receives a query request, it checks