Performance Improvement in Applying Network Coding to On-demand Scheduling Algorithms for Broadcasts in Wireless Networks

G. G. Md. Nawaz Ali *, Yuxuan Meng *, Victor C. S. Lee *, Kai Liu † and Edward Chan *

* Department of Computer Science
City University of Hong Kong, HKSAR, China
Email: gnawazali2-c@my.cityu.edu.hk, ymeng7@student.cityu.edu.hk, csvlee@cityu.edu.hk, csedchan@cityu.edu.hk
† College of Computer Science, Chongqing University, Chongqing, China
Email: liukai0807@cqu.edu.cn

***Abstract*—Due to its ability to satisfy multiple data items through a single broadcast, on-demand broadcasting has enjoyed wide usage in wireless data dissemination. Recently, applying network coding in broadcast has received much interest, because using this technique a number of items can be served through a single broadcast, which further maximizes the channel bandwidth utilization and improves the overall system performance. A generalized network coding based encoding model has been proposed which helps to migrate a number of existing scheduling algorithms to the network coding assisted version while preserving their original scheduling criteria. However, the proposed system only studies the homogeneous system environment. In this work we have done an extensive simulation based on the proposed generalized encoding model, both in the homogeneous and heterogeneous environment, to analyze the efficiency and adaptability of network coding assisted scheduling algorithms against a number of performance metrics in the real-time environment. Simulation studies reveal some interesting results.**

Keywords-Network coding; on-demand scheduling; wireless broadcast

I. Introduction

In wireless and mobile networks, data broadcasting is an efficient means for data dissemination. The server disseminates data items on the shared broadcast channel, and the clients can be served simultaneously by listening to this channel. In general, there are three broadcasting categories: push-based, pull-based, and hybrid broadcast [1][2]. Pull-based broadcast is also referred to as on-demand broadcast. In on-demand broadcast, which is the focus of this research, the server broadcasts according to the requests sent by clients, by compiling requests in queue and broadcasting requested data items based on the various attributes of pending data items at the server. Clients listen to the broadcast channel, and receive the data items that they need.

Some on-demand data scheduling algorithms have been proposed by researchers [1][3]-[6], which have different application-specific performance objectives. Ahlswede et al. [7] proposed network coding that further improves performance. It utilizes previously received data items cached at clients. Clients requesting different data items can be satisfied simultaneously. The impact of network coding is determined by the encoding decision of the server at each broadcast tick. This paper is based on a generalized encoding framework proposed and described in [8] which incorporates a flexible and adaptive network coding into data scheduling algorithms for on-demand broadcast. Our contribution is in providing extensive simulation experiments to study the performance of scheduling algorithms with and without network coding.

This paper is structured as follows. Section II covers related work in this area. Section III covers the system model used in the paper, followed by coverage of extensive simulation experiments in Section IV. The paper concludes in Section V.

II. Related Works

A number of data scheduling algorithms have been proposed for on-demand broadcast, such as FCFS by Wong [1], Most Requests First (MRF) and Longest Wait First (LWF) in [2], and $R \times W$ by Aksoy et al. [3]. They focused on different metrics, such as reducing access time, deadline miss ratio, or stretch. The well- known EDF algorithm [9] is a scheduling algorithm in real time systems. Xuan et al. [5] proved its good performance in on-demand broadcast. Xu et al. [6] proposed "Slack time Inverse Number of pending requests" (SIN), which can achieve good performance in the real-time environment.

Network coding encodes different data items, broadcasts the encoded data in a broadcast tick, and improves performance. The simplest coding schemes, linear coding is studied by Li et al. [10] which examined the network capacity of multi-cast networks. Park et al. [11] showed that network coding can achieve even 65% higher throughput than conventional multi-cast in a typical ad hoc network scenario. Fragouli et al. [12] formulated an analytical model for energy-efficient broadcast in wireless ad hoc networks with coding .

Most works [1][3][4][6] assumed that through one broadcast, only clients requesting the same data item can be satisfied, where the broadcast bandwidth cannot be fully utilized. Chu et al. [13] proposed a multi-data delivery algorithm using the XOR operator to encode the data. However, in their encoded data item at most two items can be encoded. Recently, Zhan et al. [8] proposed a generalized coding framework which can be used to determine the set of request items which can be broadcast to the maximum number of clients, where an encoded items can use maximum coding opportunity.

In this paper, based on the generalized framework in [8], we conduct extensive simulation experiments of a number of

 ISBN: 978-1-61208-346-9

existing scheduling algorithms with coding and without coding to explore the performance in both same and heterogeneous item size to analyze the coding adaptivity efficiency.

III. System Model

In this section we describe the system model as well as the encoding framework used in this research.

A. System Architecture

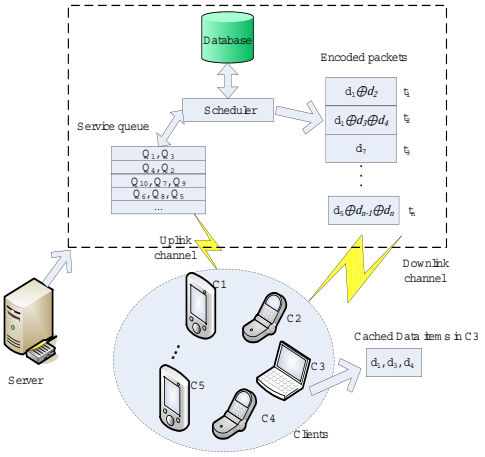


Fig. 1. System Architecture

Our system architecture is the typical on-demand system model for data broadcasting in wireless environment [3] as shown in Figure 1. The system consists of one server and many clients. When a client requires a item and cannot locate it in its local cache, it will send a request through the uplink channel to the server. After sending the request, a client listens to the broadcast channel A client can decode the requested data item from an encoded packet when it has all the other encoded data items in the packet in its cache.

The server has a received queue where it stores the generated requests received from clients. A request is feasible if it has enough slack time to be served. An infeasible request is one whose deadline has missed and will be removed from the received queue. After invoking a certain scheduling algorithm the server retrieves the data items from the local database and according to the clients' cache, it forms the encoded packet for broadcasting through the broadcast channel in the next serving cycle. We use simple XOR operations for encoding and decoding due to its lower overhead functionality [8][14]. The primary goal of real-time scheduling is to serve as many requests as possible before their deadlines and to maximize the broadcast channel bandwidth utilization.

B. Graph Model

A graph model by Zhan et al. [8] can be constructed as follows. The system has a data server s and n clients, $R = \{c_1, c_2, \cdots, c_n\}$. Let $W(c_i)$ be the set of data items requested by client c_i, and $H(c_i)$ be the set of data items cached at client c_i. A database containing data items is in the server, where d_j is the j-th data item, and m is the total number of data items in the database.

TABLE I
The Unified Encoding Model

Scheduler	Input
FCFS	$\phi(t_i,l_i,T_i) = 2^{N-j+1}t_i$, t_i is the j-th largest in $\{t_1, \cdots, t_N\}$, $\chi(\mathbf{t},\mathbf{l},\mathbf{T},\mathbf{x}) = 1$
MRF	$\phi(t_i,l_i,T_i) = 1$, $\chi(\mathbf{t},\mathbf{l},\mathbf{T},\mathbf{x}) = 1$
LWF	$\phi(t_i,l_i,T_i) = t_i$, $\chi(\mathbf{t},\mathbf{l},\mathbf{T},\mathbf{x}) = 1$
$R \times W$	$\phi(t_i,l_i,T_i) = 1$, $\chi(\mathbf{t},\mathbf{l},\mathbf{T},\mathbf{x}) = max_{1\le i\le N}\{x_i \times t_i\}$
EDF	$\phi(t_i,l_i,T_i) = 2^{N-j+1}\frac{1}{T_i}$, $\frac{1}{T_i}$ is the j-th largest in $\{\frac{1}{T_1}, \cdots, \frac{1}{T_N}\}$, $\chi(\mathbf{t},\mathbf{l},\mathbf{T},\mathbf{x}) = 1$
SIN	$\phi(t_i,l_i,T_i) = 1$, $\chi(\mathbf{t},\mathbf{l},\mathbf{T},\mathbf{x}) = max_{1\le i\le N}\{x_i \times \frac{1}{T_i}\}$

Definition 3.1: Given $R = \{c_1, c_2, \cdots, c_n\}$, $D = \{d_1, d_2, \cdots, d_m\}$, $W(c_i) \subseteq D$, $H(c_i) \subseteq D$, $W(c_i) \cap H(c_i) = \emptyset$, we construct a graph $G(V,E)$ as:
$V = \{v_{ij} |\ client\ c_i\ requests\ for\ item\ d_j,\ 1 \le i \le n,\ 1 \le j \le m\}$
$E = \{(v_{i_1 j_1}, v_{i_2 j_2}) |\ j_1 = j_2;\ or\ j_1 \ne j_2,\ d_{j_2} \in H(c_{i_1}),\ d_{j_1} \in H(c_{i_2})\}$
Accordingly, there are two rules to construct the graph $G(V,E)$, by connecting an edge between two vertices in the graph:
the first rule is, $(v_{i_1 j_1}, v_{i_2 j_2})$ with $j_1 = j_2$, which means that if client c_{i_1} and client c_{i_2} require the same data item, there is a link between two vertices $v_{i_1 j_1}$ and $v_{i_2 j_2}$.
The second rule is, $(v_{i_1 j_1}, v_{i_2 j_2})$ with $j_1 \ne j_2$, $d_{j_2} \in H(c_{i_1})$, and $d_{j_1} \in H(c_{i_2})$, which means that if client c_{i_1}'s cache contains the data item being requested by client c_{i_2} and vice versa, there is a link between vertices $v_{i_1 j_1}$ and $v_{i_2 j_2}$.

A clique is a subset of the vertices in an undirected graph, such that every two vertices in the subset are connected by an edge in graph theory. To apply the network coding to broadcast, needs to find the maximum clique δ_{max} of the graph $G(V,E)$. By broadcasting the set of requested data items in δ_{max} with coding, the maximum number of clients is served in each broadcast tick.

C. Encoding Framework

Besides satisfying a number of requests by each broadcast unit, scheduling also should pay attention to some other applications specific requirements such as the longest waiting time or current stretch, minimal slack time, etc. A generalized encoding framework is proposed in [8] as follows:

The vertices in the graph are noted as $v_1, v_2, \cdots, v_N$, where N is number of vertices and v_i corresponds to a request q_i. Three weights, t_i, l_i and T_i, are associated with vertex v_i, where they are respectively the waiting time, the size of data item and the slack time of the request q_i corresponding to v_i.

$x_i \in \{0,1\}$ is to denote whether vertex v_i is selected or not in the optimal clique C, whereas $x_i = 1$ means v_i is selected. Here $\mathbf{T} = \{T_1, T_2, \cdots, T_N\}$, $\mathbf{x} = \{x_1, x_2, \cdots, x_N\}$, $\mathbf{t} = \{t_1, t_2, \cdots, t_N\}$, $\mathbf{l} = \{l_1, l2, \cdots, l_N\}$. $\phi(t_i,l_i,T_i)$ is the weight function of attribute(s) associated with v_i and $\chi(\mathbf{t},\mathbf{l},\mathbf{T},\mathbf{x})$ is the optimized function in different applications.

 ISBN: 978-1-61208-346-9

TABLE II
SIMULATION PARAMETERS

Parameter	Default	Range	Description
ClientNum	300	—	Number of clients
λ	20	—	Mean request generation rate
DBSIZE	1000	—	Size of the database
SIZEMIN, *SIZEMAX*	1, 10	—	Min. and Max. data item size
CacheSize	60	30-180	Client cache size
θ	0.6	0.0-1.0	Zipf distribution parameter
μ^-, μ^+	120, 200	—	Min. and Max. laxity

maximize

$$\psi(C) = \chi(\mathbf{t},\mathbf{l},\mathbf{T},\mathbf{x}) \sum_{i=1}^{N} \phi(t_i,l_i,T_i)x_i \quad (1)$$

subject to

$$(x_i + x_j) \leq 1, (v_i,v_j) \notin E(G), 1 \leq i \neq j \leq N, x_i = \{0,1\}$$

The weight function $\phi(t_i,l_i,T_i)$ and the optimized function $\chi(\mathbf{t},\mathbf{l},\mathbf{T},\mathbf{x})$ are used with different settings to meet various application requirements.

The strategies for various scheduling algorithms with network coding in a unified model described in [8] are summarized in Table I, where each of the scheduling disciplines tries to maximize the functions $\psi(C)$ in (1).

IV. PERFORMANCE EVALUATION

A. Simulation Setup

Our simulation model is implemented in CSIM19 using the default parameters shown in Table II which are based on [8]. A closed system model is used. The item access pattern follows the zipf distribution, where skewness is controlled by the parameter θ. For a real-time environment, we set the relative deadline (RD) of a request R_i as follows:

$$RD_i = (1 + uniform(\mu^-, \mu^+)) * T_i^{serv} \quad (2)$$

where μ^- *and* μ^+ are the minimum and maximum laxity for calculating the relative deadline, respectively, and T_i^{serv} is the service time of R_i. The deadline of a request R_i is Dl_i, which is computed by:

$$Dl_i = AT_i + RD_i \quad (3)$$

where AT_i is the arrival time of R_i.

The channel bandwidth is 1.0. For the homogeneous environment each item size is 1.0, but for the heterogeneous environment different item sizes are generated using the random item size distribution (RAND) [15].

B. Performance Metrics

1) Deadline Miss Ratio (DMR): The ratio of the number of requests which missed their deadline over all submitted requests..
2) Average Encode Length (AEL): The average number of data items encoded in each encoded packet. A high AEL means more clients can decode their expected data items from an encoded broadcast packet.
3) Saved Bandwidth Ratio (SBR):

$$SBR = \frac{\#Satisfied\ Item - \#Broadcast\ Item}{\#Satisfied\ Item} \quad (4)$$

where *Satisfied Item Number* means the number of data items satisfied and *Broadcast Item Number* the number of encoded items being broadcast.

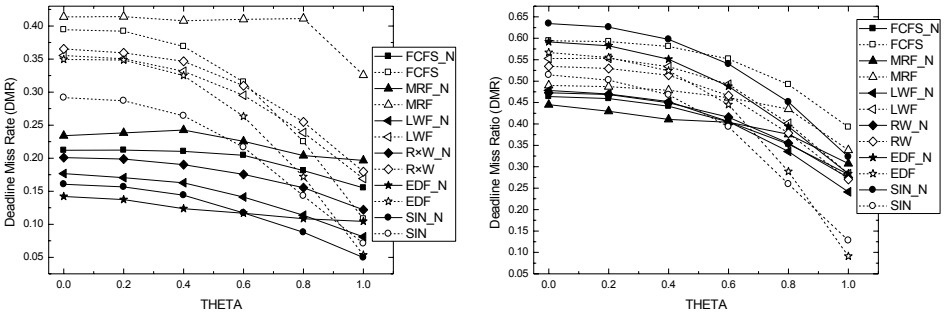
C. Performance Analysis

In this section, we discuss the performance analysis of coding and non-coding versions of different existing scheduling algorithms under both the same and different item size distribution. Simulation experiments continue until 98% confidence interval has been achieved. The different scheduling algorithms can be divided into two types: algorithms such as SIN and EDF which target time constraints or urgency (TC) and those that do not focus on time constraints (NTC).

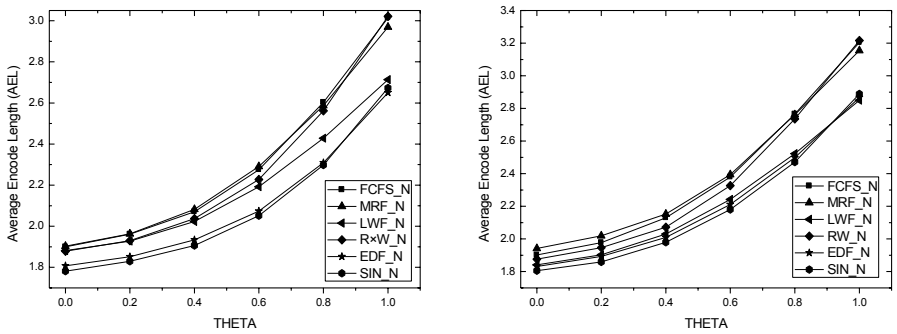
1) Impact of Skewness Parameter θ: Figure 2 shows the impact of data item access pattern θ. When $\theta = 0.0$, item access pattern is uniformly distributed. But with increasing θ, the probability for accessing popular data items increases. In serving such a popular item, a number of requests can be served concurrently, which explains why the performance of all algorithms improves with increasing θ. From Figure 2(a) (I), for the same item size distribution, network coding assisted TC algorithm SIN_N and EDF_N show better performance than NTC network coding assisted algorithms MRF_N, FCFS_N, $R \times W\_N$ and LWF_N. To see the impact of considering urgency as a request selection, consider Figure 3 (I). This figure shows the percentage of served requests over total submitted requests against the slack time (ST) of requests in the system under default settings. The percentage value in the y-axis against the value '0-1/4 of RD' in the x-axis means percentage number of requests served when these requests have remaining deadlines i.e., slack times (ST) less than 1/4 of the original assigned relative deadlines (RD) of the respective requests. Similarly, there are total four categories. To understand this metric, let us have an example. Suppose a request R_i's relative deadline RD_i is 12. So, initially R_i's slack time ST_i is also 12. As time passes ST_i also decreases. Now, if R_i is served when $ST_i = 10$, it will fall in the 4th category (that is 3/4 - RD = 9 - 12). Similarly, if R_i is served when $ST = 7$ or $ST = 1$ it will fall in the 3rd and 1st category respectively. Accordingly, if more number requests served in the 1st and subsequent categories, the algorithm is more aware of the scheduling of deadline urgent requests. From Figure 3 (I) we can see that, in SIN_N more requests are served in the first category when compared to MRF_N, in which more requests are served in the last category. It proves that MRF_N does consider the slack time of the requests, which is why fewer requests are served in the first category, hence urgent requests may not meet their deadlines resulting in higher DMR in MRF_N.

On the contrary, from Figure 2(a) (II), surprisingly TC algorithms SIN_N and EDF_N can not retain their supremacy in terms of DMR for RAND item size distribution. Recalling

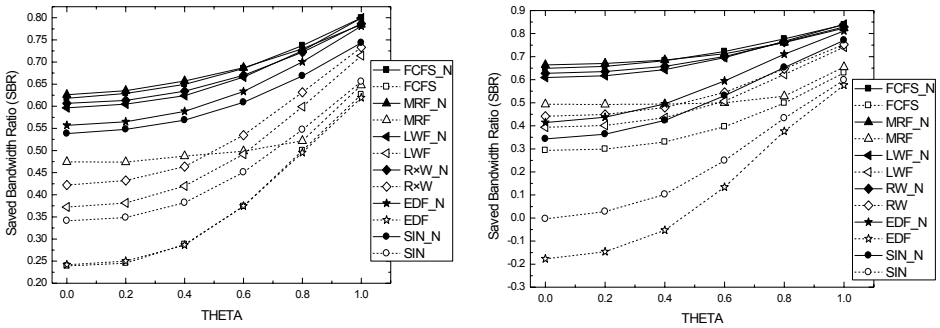
 ISBN: 978-1-61208-346-9



(a) DMR for (I) Same, and (II) RAND item size distribution



(b) AEL for (I) Same, and (II) RAND item size distribution



(c) SBR for (I) Same, and (II) RAND item size distribution

Fig. 2. Impact of skewness parameter (θ).

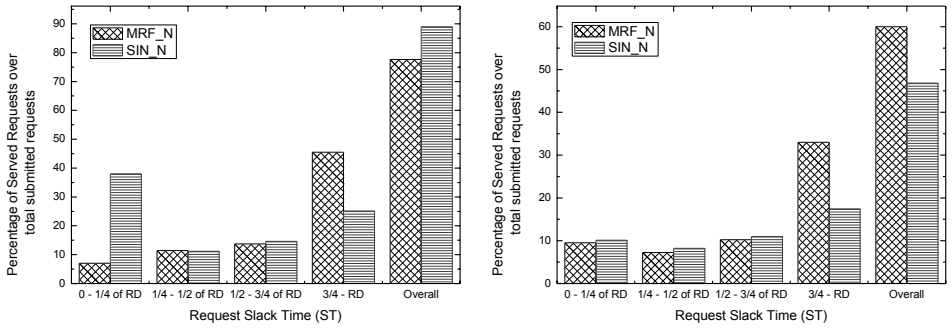
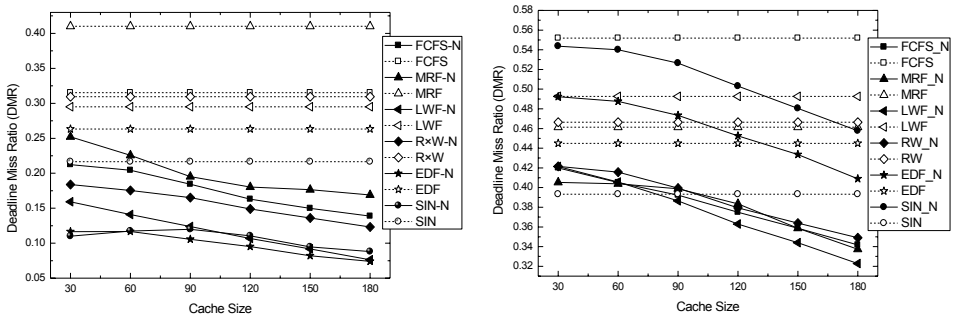
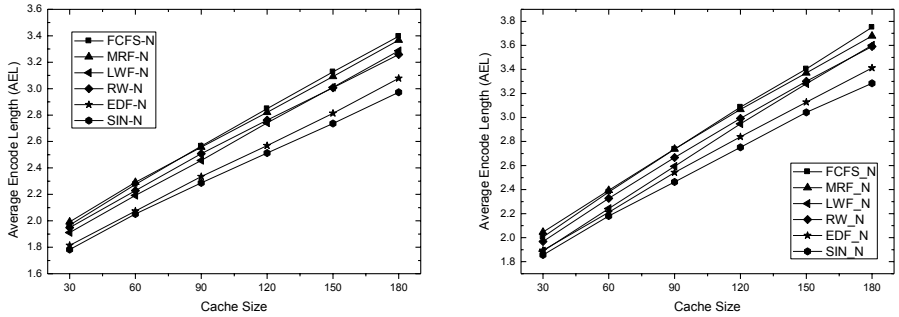


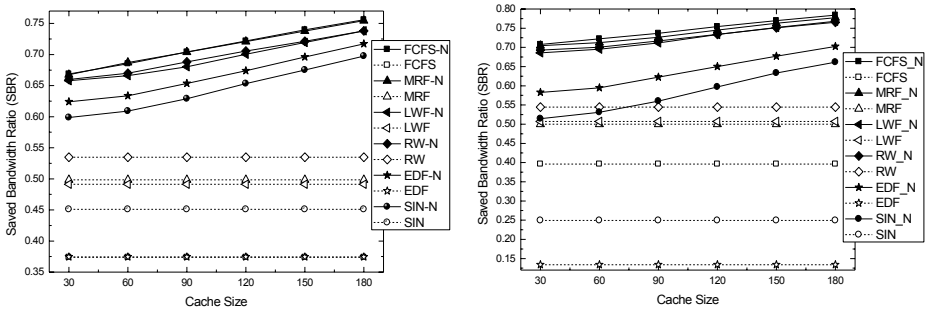
Fig. 3. Distribution of serving requests over total submitted requests for(I) Same, and (II) RAND item size distribution.



(a) DMR for (I) Same, and (II) RAND item size distribution



(b) AEL for (I) Same, and (II) RAND item size distribution



(c) SBR for (I) Same, and (II) RAND item size distribution

Fig. 4. Impact of Cache Size.

from the network coding view, for different item sizes, the size of the encoded packet will be the maximum item size in the selected clique. Now, a TC algorithm uses slack time for selecting the optimal clique. Unlike productivity based algorithm, it may select a clique having smaller clique size with urgent request as a clique vertex. Regardless of the selected clique size, the service time of the encoded packet will be as large as the service time of the largest data item in the selected clique, hence the service time is increased, which makes it more difficult to meet the deadline. This statement is supported by Figure 3 (II), which shows that unlike same item size distribution, SIN_N does not outperform MRF_N in the first category (0 - 1/4 of RD), and overall requests serving percentage also lower than MRF_N. Nevertheless, the non network coding version of the algorithms do not have this clique size problem, and hence show superior performance for skewed access pattern (Figure 2(a) (II)). On the other hand, the coding version of a NTC based algorithm typically tries to serve a clique having maximum clique size, hence although the service time is increased, many requests can be served concurrently. Therefore, although such an algorithm in heterogeneous item size environment has worse performance than the same item size environment, it still has a better performance than its corresponding non-coding version as well as the network coding version.

Except for FCFS_N, all the NTC algorithms use item productivity as their request selection criterion, i.e. use size of a clique ((in other words, number of vertices in the clique)), but ignore the urgency of a request. However, FCFS_N selects the optimal clique, including the weight of a corresponding vertex, waiting time of all the vertices in a clique is counted, which indirectly supports the clique size. For example, it is

 ISBN: 978-1-61208-346-9

more likely that a clique having larger size, might have more total weight than that of a smaller size clique. So, the network coding version of FCFS, i.e. FCSF_N, has similar performance to other WTC algorithms. It is evident from Figure 2(a) (I), network coding version of a algorithm has better performance than its non-network coding version. An exception is EDF, when $\theta = 1.0$; EDF_N shows better DMR than EDF. This is because in the skewed item access pattern the non-network coding version of an algorithm can gain more than network coding environment.

From Figures 2(b) (I) and (II), with increasing θ, average encode length (AEL) of all the algorithms increases, because with increased access pattern more requests ask for the popular data items, hence being a requested item of a client could be a cached item of another client with a higher probability, which helps to form a bigger clique and increase the coding opportunity as well. NTC algorithms have better AEL and saved bandwidth ratio (SBR) than TC algorithms irrespective of item size distribution as shown in Figures 2(b) (I), and 2(b) (II), 2(c) (I) and 2(c) (II). Regarding SBR, regardless of item size distribution, coding version of an algorithm has better SBR than its non-coding version. This is because, as the client gradually accumulates more items in the cache, the client can exploit its cache in the coding version to satisfy more requests. This is clearly not possible in the non-coding version of the algorithms.

2) Impact of Cache Size: Figure 4 shows the impact of the cache size of a client for both same and different item size distribution. Since the non-coding version of an algorithm does not consider client's cache, changes in cache size has no impact on performance. Increasing cache of a client provides more room to store more items, which increases the probability of one client's cached item to be another client's requested item. This is the key to increase the coding flexibility and AEL which provides more opportunity for performance gain. For this reason DMR declines with increasing cache size for both same and rand item size distributions (Figures 4(a) (I), 4(a) (II)), AEL increases (Figures 4(b) (I), 4(b) (II)), and SBR increases (4(c) (I), 4(c) (II)).

V. Conclusion and Discussion

In this paper, based on the generalized encoding model proposed in [8], we conducted extensive simulations for same and different item size distributions in the real-time environment. We analyze the simulation performance of a number of existing scheduling algorithms both in their coding and non-coding versions against a number of performance metrics to examine the efficiency and adaptability of coding assisted scheduling algorithms. Simulation results show that network coding assisted algorithms improve their performance with increased cache size and skewed access pattern. Generally speaking, NTC algorithms have superior performance in AEL and SBR than TC algorithms irrespective of data item size distribution. In addition, regardless of item size distribution, all the coding assisted algorithms outperform their corresponding non-coding version in SBR. On the other hand TC based algorithms perform better in DMR in the real-time setting for the same item size distribution. However, surprisingly, in the different item size distribution their performance decline dramatically. A plausible reason is that in the coding version, the scheduling algorithm needs to serve the maximum sized item of the selected clique irrespective of the clique size in each broadcast, hence a clique may have urgent vertex but also may have smaller clique size. On the contrary, the non-coding version simply selects the best item based on the underlying scheduling principle.

We are exploring the use of other metrics in deepening our understanding of the behavior of different algorithms when network coding is used, and will hopefully present the results in a future paper.

References

[1] J. W. Wong and M. H. Ammar, "Analysis of broadcast delivery in videotex system," Journal of IEEE Transactions on Computers, vol. 34, no. 9, September 1985, pp. 863–866.

[2] J. W. Wong, "Broadcast delivery," Proceedings of the IEEE, vol. 76, no. 12, 1988, pp. 1566–1577.

[3] D. Aksoy and M. Franklin, "$R \times W$: A scheduling approach for large-scale on-demand data broadcast," Journal of IEEE/ACM Transactions on Networking (TON), vol. 7, no. 6, December 1999, pp. 846–860.

[4] S. Acharya and S. Muthukrishnan, "Scheduling on-demand broadcasts: new metrics and algorithms," in Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking (MobiCom'98), 1998, pp. 43–54.

[5] P. Xuan, S. Sen, O. Gonzalez, J. Fernandez, and K. Ramamritham, "Efficient and timely dissemination of data in mobile environments," in Proceedings of the 3rd IEEE Real Time Technology and Applications Symposium(RTAS'97), Montreal, Canada, 1997.

[6] J. Xu, X. Tang, and W. Lee, "Time-critical on-demand data broadcast algorithms, analysis and performance evaluation," IEEE Transactions on Parallel and Distributed Systems, vol. 17, no. 1, 2006, pp. 3–14.

[7] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow," IEEE Transactions on Information Theory, vol. 46, no. 4, July 2000, pp. 1204–1216.

[8] C. Zhan, V. Lee, J. Wang, and Y. Xu, "Coding-based data broadcast scheduling in on-demand broadcast," IEEE Transactions on Wireless Communications, vol. 10, no. 11, November 2011, pp. 3774–3783.

[9] C. Liu and J. Layland, "Scheduling algorithms for multiprogramming in hard real-time traffic environments," Journal of the Association for Computing Machinery (ACM), vol. 20, no. 1, 1973, pp. 46–61.

[10] S.-Y. Li, R. Yeung, and C. Ning, "Linear network coding," IEEE Transactions on Information Theory, vol. 49, no. 2, February 2003, pp. 371–381.

[11] J.-S. Park, D. Lun, F. Soldo, M. Gerla, and M. Médard, "Performance of network coding in ad hoc networks," in Proceedings of the IEEE Military Communications Conference, Washington, DC, October 2006, pp. 1–6.

[12] C. Fragouli, J. Widmer, and J.-Y. Le Boudec, "A network coding approach to energy efficient broadcasting: From theory to practice," in Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM'06), Barcelona, Spain, April 2006, pp. 1–11.

[13] C.-H. Chu, D.-N. Yang, and M.-S. Chen, "Multi-data delivery based on network coding in on-demand broadcast," in Processings of the 9th International Conference on Mobile Data Management (MDM'08), Beijing, April 2008, pp. 181–188.

[14] J. Chen, V. C. S. Lee, and C. Zhan, "Efficient processing of real-time multi-item requests with network coding in on-demand broadcast environments," in Proceedings of the 15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA '09), Beijing, August 2009, pp. 119 – 128.

[15] J. Xu, Q. Hu, W. Lee, and D. L. Lee, "Performance evaluation of an optimal cache replacement policy for wireless data dissemination," IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 1, January 2004, pp. 125–139.

 ISBN: 978-1-61208-346-9