Deadline-aware Broadcasting in Wireless Networks with Network Coding

Pouya Ostovari*, Abdallah Khreishah†, and Jie Wu*
*Department of Computer & Information Sciences, Temple University, Philadelphia, PA 19122
†Department of Electrical & Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102

***Abstract*—Broadcasting with network coding mixes different packets to minimize the number of transmissions, which improves the energy efficiency of wireless networks. On the other hand, delaying the transmissions increases coding opportunities at the intermediate nodes, but increases the delay of the packets. In this paper, we consider these two contradicting factors and study the problem of minimizing the number of transmissions in wireless networks while meeting the deadlines of different packets. We show that this problem is NP-complete; therefore, we provide a heuristic to solve the problem. First, we construct broadcasting trees, each of them rooted at one source. We then specify overlapping conditions based on the constructed trees to determine the number of transmissions each node has to perform without the deadline constraints. Then, we partition the set of packets such that coding is performed among the packets of the same partition, which does not result in deadline misses. Our simulation results show that our technique not only reduces the number of transmissions, but also allows the majority of the nodes to receive their packets on time.**

***Index Terms*—Broadcasting, broadcast tree, network coding, energy efficiency, deadline, NP-completeness.**

I. Introduction

Broadcasting is used frequently in wireless networks to disseminate control messages and data in many applications. Flooding is the simplest broadcasting method in wireless networks, where each node forwards the received packets to its neighbors. Clearly, flooding is not an efficient way for broadcasting due to the unnecessary, redundant transmissions it causes. To perform broadcasting efficiently, many works have targeted decreasing the number of transmissions. These works can be classified into two main categories: probabilistic and deterministic approaches.

In the probabilistic methods, each node forwards the received packets with a given forwarding probability [1]. This probability should be chosen carefully, so that all nodes are able to receive the packets with the restricted number of transmissions. On the other hand, the deterministic approaches use the network topology and neighborhood information to select some forwarding nodes that are responsible for forwarding the received packets. Connected dominating sets (CDS) [2] and pruning approaches [3] belong to this category.

With network coding (NC) [4], [5], intermediate nodes mix packets using mathematical operations, which reduces the number of transmitted packets. NC can be combined with both the probabilistic and deterministic approaches to improve the transmission efficiency in wireless networks. The authors in [6] show that for fixed networks, NC can, at most, offer a constant factor of benefits in terms of energy efficiency. They also propose a probabilistic NC-based broadcasting algorithm. The work in [7] combines the partial dominant pruning (PDP) forwarding approach [3], which is a deterministic approach, with NC. The algorithm uses local, two-hop topology information and makes use of opportunistic listening to reduce the number of transmissions. Using NC with directional antennas is considered in [8]. The work is based on the deterministic forwarding approach that uses directional CDS. It can be noted that the works that use the deterministic approach with NC limit coding to XOR operations and exploit only local coding opportunities.

Most of the works on NC focus on maximizing the throughput [9], achieving fairness [10], or minimizing the energy consumption [8], [6]. While these metrics are important, delay and deadline metrics have received less attention from the community. In this paper, we take a different look at the NC problem by studying the problem of *energy efficient broadcasting in wireless networks subject to deadline constraints*. Delaying the transmissions reduces energy consumption by increasing coding opportunities at the intermediate nodes, but increases the delay of the packets. Thus, it is crucial to specify the degree of NC such that the packets can meet the deadlines. NC has been studied with deadlines in [11], [12], but all of these studies are for single-hop coding and broadcast channels.

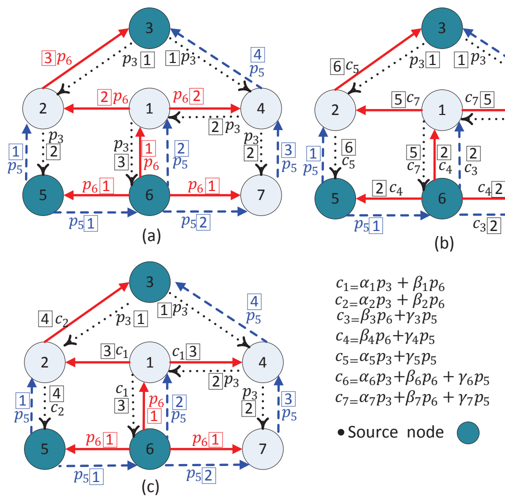
In this paper, we consider multihop wireless networks and we have the following contributions. First, we show that the problem of a minimum energy broadcast, subject to the deadline constraints, is NP-complete. Second, We propose a three-phase algorithm for the problem. Finally, We conduct simulations to show the benefits of our proposed scheme in terms of meeting the deadlines and energy efficiency.

II. System Model

A. Setting and Motivation

We consider a multi-hop wireless network with multiple broadcast sessions, where a subset of the nodes are sources while all of the nodes are destinations. Every packet has a deadline to reach each of the destinations. All of the nodes are synchronized and all of the links are reliable. We assume that the nodes have multi-channel multi-radio capability. Thus, all of the nodes can transmit and receive simultaneously, and there is no conflict among the links. Also, we assume that

This research was supported in part by NSF grants ECCS 1128209, CNS 1138963, CNS 1065444, and CCF 1028167.



(a)
(b)
(c)
$c_1=\alpha_1 p_3+\beta_1 p_6$
$c_2=\alpha_2 p_3+\beta_2 p_6$
$c_3=\beta_3 p_6+\gamma_3 p_5$
$c_4=\beta_4 p_6+\gamma_4 p_5$
$c_5=\alpha_5 p_3+\gamma_5 p_5$
$c_6=\alpha_6 p_3+\beta_6 p_6+\gamma_6 p_5$
$c_7=\alpha_7 p_3+\beta_7 p_6+\gamma_7 p_5$
Source node

Fig. 1. Broadcasting (a): without coding (b): coding p_3 and p_6 (c): coding all of the packets together.

each transmission takes one time slot to reach the next hop, and there is no feedback in the network.

NC reduces the number of transmissions but increases the delay. The reason is that each node has to wait until receiving all of the incoming packets to code them together; the sending time of the coded packet should be at least the maximum arriving time of all of the received packets. In Fig. 1(a), nodes 3, 6 and 5 are sources for packets p_3, p_6 and p_5, respectively. The deadline of the packets p_3, p_6 and p_5 are time slots 5, 5 and 6, respectively. The sending time of each packet is shown in the box beside the packet. In this case since there is no coding, all of the packets meet their deadlines, and the number of transmissions is 11. In Fig. 1(b) all of the packets are coded together. Node 1 receives packets p_6, p_3 and p_5 at time slots 2, 3 and 5, respectively. Thus, in order to code these packets together, node 1 has to postpone the transmission of packets to time slot 5. Coding all of the packets together reduces the number of transmissions to 8 but causes a deadline miss as node 3 receives packet p_6 after the packet's deadline. Therefore, we have to find another solution in which we reduce the number of the transmitted packets while meeting the deadlines.

In Fig. 1(c) we code only packets p_3 and p_6 together. The number of transmissions in Fig. 1(c) is equal to 9 which is more than in Fig. 1(b), but in Fig. 1(c) there is no deadline miss. Therefore, an efficient solution for the problem of energy efficient broadcasting with deadline constraints should be partitioning the set of packets such that coding the packets of each partition does not result in deadline misses. Thus, our problem becomes finding the set of partitions that minimize the total number of transmissions such that all of the packets meet their deadlines. We assume that the broadcasting operation is periodic. Thus, our target is to find an efficient solution for the problem and then use that solution in the consecutive rounds.

B. Linear Network Coding

In this paper, we use linear NC. Linear NC is introduced in [5] as it is shown to achieve the capacity for the single multicast session problem. A useful algebraic representation of the linear NC problem is provided in [13].

In linear NC, each node generates and sends a linear combination of the received packets over a finite field. When a node receives an innovative packet, it stores this packet in its packet buffer and the corresponding coefficients vector in its coefficients buffer. An innovative packet is a received packet such that its coefficient vector increases the rank of the matrix formed by the received coefficient vectors. In other words, an innovative packet is a linearly independent packet to the previously received packets. Each forwarder node continues this process. Assume that K single packets are coded together. When a destination node receives K linearly independent coded packets, it will be able to decode all of the coded packets and retrieve all of the single packets.

The decoding process is done using Gaussian elimination for solving a system of linear equations. In [14], it is shown that selecting the coefficients, in a distributed manner at random, achieves the capacity asymptotically with respect to the finite field size.

III. Heuristic

In the appendix, we prove that the problem of energy-efficient broadcasting, subject to the deadline constraints, is NP-complete. Therefore, in this section, we propose a *deadline-aware network coding* (DANC) heuristic to solve the problem. For simplicity, we assume that each packet has the same deadline to reach all of the destinations. Our algorithm contains following three phases:

- Constructing broadcasting trees: This phase ensures the decodability of the coded packets at the destination nodes. This phase is done once in the initializing phase.
- Partitioning the set of packets: The purpose of this phase is to guarantee meeting all the deadlines. This phase is done once in the initializing phase.
- Performing coding: In this phase, the relay nodes do the actual coding. This phase is repeated periodically.

By using broadcasting trees, each node receives enough linearly independent packets, so the nodes are able to decode the coded packets. If we allow all of the packets to be coded together, we can decrease the number of transmissions, but the delay increases and some packets may miss their deadlines. Therefore, we partition the set of packets such that coding the packets of each partition does not result in deadline misses.

We assume that the broadcasting operation is periodic; we run the first two phases only once, then the third phase runs periodically. Thus, the complexity for the first two phases is not a major issue (however it is polynomial), though their performance is important because they decide the operations of the third phase.

A. Constructing Broadcasting Trees

We use broadcasting trees to broadcast the packets. A broadcasting tree is a spanning tree rooted at one source node to reach all of the other nodes. Fig. 2(b) shows three broadcasting trees. If a node is a non-leaf node in more

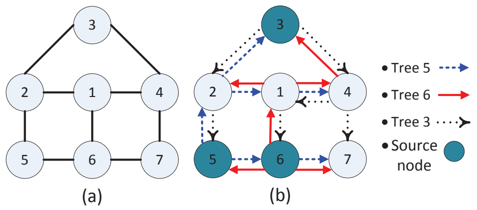


Fig. 2. (a): A given topology. (b): Two broadcasting trees

than one broadcasting tree, it has the opportunity to code the received packets in order to send fewer packets. Assume that there are K sources we will have K broadcasting trees, so each destination node receives K coded packets, each of them from a different broadcasting tree. In order to ensure the decodability of the packets at the destination nodes, the K received packets have to be linearly independent.

We define the overlap-degree of node u to v as the the number of trees that use link (u, v), and we represent it as $\delta(u, v)$. Also, we define the maximum overlap-degree of node u as $\Delta(u) = \max_v(\delta(u, v))$. In Fig. 2(b), $\delta(6, 1) = 1$, $\delta(6, 5) = 1$, and $\delta(6, 7) = 2$, so $\Delta(6) = 2$. In order to guarantee decodability, node 6 has to send two linearly independent coded packets to node 7. Each of these packets has to contain both of the packets p_4 and p_6. To make sure that the coefficient vectors in the buffers of all of the nodes achieve full rank, the number of transmissions at node u has to be at least equal to $\Delta(u)$. Consequently, if we can reduce Δ of each node, we can reduce the total number of transmissions. If we select the coefficients in a distributed manner at random, destination nodes will receive K linearly independent coded packets with high probability, almost 1 [14].

Our heuristic sequentially constructs broadcasting trees, each of them rooted at a source node. First, this approach sorts the sources in increasing order of the deadline of their packets. Then, in each iteration, our algorithm starts from a new source and traverses the network using the BFS algorithm. During traversal, each node not in the tree selects a node in the tree as its parent based on the following two rules:

- $Rule_1$: Node v selects the parent u that has the maximum number of effective neighbors.
- $Rule_2$: Node v selects the parent u where selecting that node does not increase $\Delta(u)$.

Effective neighbors of node u are the neighbors that do not have a parent in the tree. While constructing the broadcasting trees, we give more priority to $Rule_1$ over $Rule_2$. The reason is that a node with the maximum number of effective neighbors can cover more nodes by a single transmission. Algorithm 1 describes our algorithm.

Fig. 3(b) shows a constructed broadcasting tree. The depth of nodes 6 and 7 in the constructed tree is 3, but the depth of the shortest path tree is 2. If the depth of a constructed broadcasting tree is more than the deadline of the packet of its source node, we will reconstruct that tree by adding a new rule to the algorithm. $Rule_3$: node u selects a parent with the minimum depth. We give more priority to this rule than the previous rules to guarantee meeting the deadline. The output of the new algorithm is a shortest path tree. The constructed

Algorithm 1 Constructing broadcasting trees

```
for each source node $u$ in ascending order of deadlines do
    Add node $u$ to tree $T(u)$
    while there is a node $\notin T(u)$ do
        Select the next node $v \notin T(u)$ using BFS algorithm
        Select node $w \in T(u)$ as $v$'s parent based on $Rule_1$
        and $Rule_2$
        Select $w$ as parent of its neighbors $\notin T(u)$
```

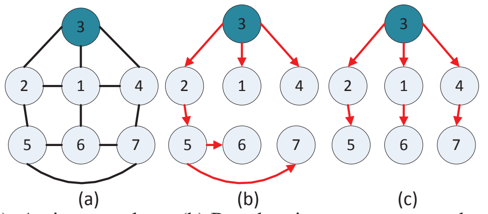


Fig. 3. (a): A given topology. (b):Broadcasting tree constructed using $Rule_1$ and $Rule_2$. (c): Shortest path tree.

tree is shown in Fig. 3(c). The number of transmissions in Fig. 3(b) is 3 and in Fig. 3(c) is 4. Therefore, we cannot start from $Rule_3$, and we only use $Rule_3$ if we find that using $Rule_1$ and $Rule_2$ does not guarantee meeting the deadline.

Assume that in Fig. 2 we have constructed tree 5, and we want to construct tree 6. First, node 1 selects node 6 as its parent, and nodes 5 and 7 connect to node 6. Then, node 2 can select node 1 or 6 as its parent. Node 2 selects node 1 which has more (two) effective neighbors. If node 3 selects node 2, $\Delta(2)$ increases, so node 3 selects node 4 as the parent.

B. Partitioning the Set of Packets

So far, we have discussed the first phase which guarantees the decodability of the packets at the destination nodes. However, it does not guarantee meeting the deadlines. To prevent missing the deadlines, we have to decide which packets to code together. For this purpose, we use a greedy heuristic to partition the set of packets into different partitions, such that coding all of the packets of each set together does not result in deadline misses.

Our *Deadline-Aware Network Coding* (DANC) heuristic uses the constructed broadcasting trees. First, the algorithm sorts the list of the packets in increasing order of their deadlines (each packet belongs to the root of one tree). Then, the algorithm places the first packet of the list to the first partition. After that, the algorithm finds which packets can be added to the partition without causing deadline misses. The algorithm finds the remaining partitions using the same operation. The detailed algorithm is shown in Algorithm 2.

To compute the receiving times of the packets, we use the Receiving Time (RT) algorithm. First, for each relay node u, the RT algorithm finds the set of packets in partition P that node u is a relay node of. We represent this set as $R_P(u)$. Using the BFS algorithm, the RT algorithm traverses the trees of a given partition P simultaneously. If all of the traversal trees that their respective packets are in $R_P(u)$ have reached node u, the algorithm assigns the maximum arriving time of the trees, plus one (each transmission takes one time slot to

Algorithm 2 Partitioning the set of packets

```
Sort list of packets $L$ in increasing order of deadlines
while $L \neq$ empty do
    $i \leftarrow i+1$, Create new partition $P_i$
    Transfer the first packet of $L$ to $P_i$
    for each packet $p$ of $L$ in ascending order do
        Using The RT algorithm, compute receiving times of the packets in $P_i \cup \{p\}$
        if no deadline misses then
            Delete $p$ from $L$, Add $p$ to $P_i$
Return $\{P_1, .., P_i\}$
```

Algorithm 3 Performing coding

```
On receiving packet $p$ by node $u$
if $u \in$ relay nodes of $p$ then
    Find the partition $P$ such that $p \in P$
    wait until receiving all of the packets $\in R_P(u)$
    send $\Delta_P(u)$ random combination of the packets $\in R_P(u)$
```

reach the next hop), to the receiving time of the corresponding packets by the children nodes of node u.

In Fig. 1, the deadlines of the packet p_3, p_6 and p_5 are 5, 5 and 6, respectively. First, we add packet p_3 to partition P_1. Then we code packet p_6 with p_3 and compute the receiving times of the packets. The sending time of the packets are shown in Fig. 1(c). Because all of the nodes receive both packets on-time, we add packet p_6 to partition P_1. Next, we code packet p_5 with the packets in P_1. Fig. 1(b) shows the sending time of the packets. We cannot add packet p_5 to partition P_1, as nodes 1 and 5 receive packets p_3 and p_6 after their deadlines. Therefore, the partitions are {3,6} and {5}.

C. Performing Coding

We extend $\Delta(u)$ to $\Delta_P(u)$. $\Delta_P(u)$ represents the maximum overlap-degree of node u for the packets in partition P. From the first two phases, each node u knows which packets it has to forward, and also it knows $\Delta_P(u)$ of each partition which specifies the number of transmissions of the coded packets of that partition. When a relay node u receives a packet p, it finds the partition P such that $p \in P$. Then, node u waits until receiving all of the packets of partition P so that the node is a relay node of $(R_P(u))$. Assuming that $\Delta_P(u) = m$, node u sends m random linear combinations of the packets. The relay nodes perform similar operations for the other partitions.

IV. Extensions

A. Deadlock Detection and Recovery

Since there is more than one source in our problem, it is likely that deadlock occurs in the network. When there is a circular waiting among the processes (nodes) to access the resources (packets), a deadlock happens. Fig. 4(a) shows a deadlock between two trees. In this figure, nodes 3 and 6 are sources. Node 4 receives a packet from node 3 and waits to receive the other packet from node 1. On the other hand, node

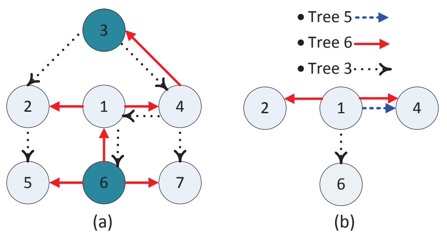


Fig. 4. (a): A cycle between two trees. (b): Coding optimization.

1 waits to receive a packet from node 4. As a result, we have a deadlock in this network.

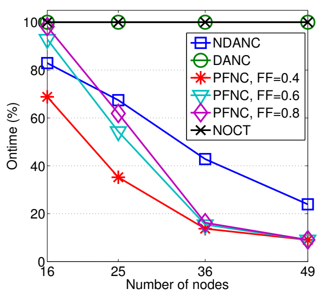
We resolve the deadlock problem in the partitioning phase. To address the deadlock problem, we use a distributed deadlock detection and recovery scheme [15]. We allow deadlocks to happen, then we resolve them. To resolve that deadlock, at least one node among the nodes that cause the deadlock has to forward a packet without waiting for other packets. In Fig. 1, node 4 can break the deadlock by forwarding packet 3.

Using convolutional codes [16] is another way to resolve the deadlocks. However, the complexity of convolutional codes limits their applicability. In our heuristic, we use linear coding which is less complex than convolutional codes and can be implemented in a decentralized way. We also use deadlock detection and recovery to resolve the deadlocks.

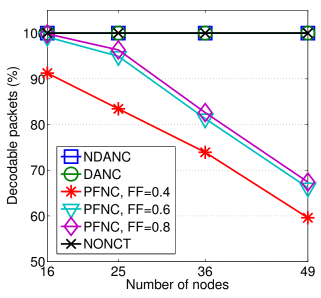
B. Coding Optimization

Assume that in Fig. 4(b), the deadlines are set such that all of the packets can be coded together. Also, assume that node 1 has received packet p_5. In our former coding algorithm, node 1 has to wait to receive all of the packets. However, node 1 does not need to wait for other packets, and it can send packet p_5 immediately. Then, the next transmission covers packets p_3 and p_6. Therefore, node u can send the packets with the same number of transmissions and less delay. Only, we need to ensure that the sent packets are linearly independent and collectively cover all of the packets.

To reduce the coding delay while preserving the number of transmissions of each node u to be equal to $\Delta(u)$, we define the following rule. Node u can perform a transmission if for each children node v of u, at least one of the following conditions is true. $Condition_1$: the coded packet contains a new packet p_i, such that there is a link from nodes u to node v in tree T_i. $Condition_2$: node u has sent all of the packets that there is a link in their respective trees to node v. $Condition_3$: node u has transmitted less than $\Delta(u) - \delta(u, v)$ packets. $Condition_1$ means that node u has a new packet for node v. When node v has received all of the necessary packets from node u, $Condition_2$ is true; node u does not need to send any more packets to node u. $Condition_3$ is related to the example in Fig. 4(b). It means that $\Delta(u) - \delta(u, v)$ of the transmissions do not need to contain a new packet for node v. As a result, these transmissions can contain packets for only the children of u, such that the maximum overlap-degree of node u is to that node (node 4 in Fig. 4(b)).

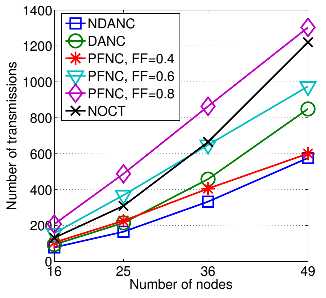


(a)

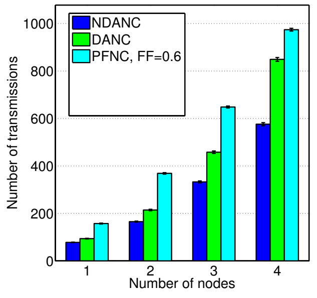


(b)

Fig. 5. (a): On-time received packets. (b): Total number of decodable packets.



(a)



(b)

Fig. 6. (a): Total number of transmissions. (b): Total number of transmissions and confidence interval with confidence level equal to 95%.

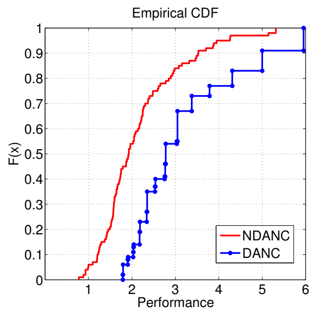


Fig. 7. Empirical CDF of performance of our approaches compared with probabilistic approach with forwarding factor 0.4.

V. Simulation Results

In addition to the DANC approach, we simulate a *Non Deadline-Aware Network Coding (NDANC)* approach. In the NDANC approach, we put all of the packets in the same partition and allow all of the packets to be coded together. We evaluate the DANC and NDANC approaches by comparing the number of transmissions, the on-time received packets, and the decodable packets. To find the number of decodable packets, we ignore the deadlines and compute the number of received packets that can be decoded at the destination nodes.

We implemented a simulator in MATLAB environment to compare our proposed methods with the *Probabilistic Forwarding with Network Coding (PFNC)* approach in [6]. In [6], when a node receives an innovative packet, it sends a coded packet, of the innovative packets it has in its buffer, with a given probability. The value of this probability is called the *Forwarding Factor* (FF). We also simulate a deterministic, non-coding protocol. In this protocol, we use broadcasting trees to broadcast the packets, and we call it the *Non-Coding Tree* (NONCT) approach in the plots.

We perform our simulation on grid topologies with 16, 25, 36 and 49 nodes, and with random deadlines in the range of r and $6r$, where r is the diameter[1] of the network. For each grid size, we run the simulation for 100 cases.

In the first experiment, we compare the number of on-time received packets. As is shown in Fig. 5(a), the DANC method guarantees meeting the deadlines. Also, in the NONCT approach, all nodes receive the packets on-time because there is no coding. The number of on-time received packets increases as we increase the forwarding factor since, by increasing FF, each node forwards more packets. Because all of the packets are coded together in the NDANC and PFNC approaches, the delay increases as the number of nodes increases; as a result, more packets miss their deadlines.

In the next experiment, we ignore the deadline constraints and compare the number of decodable packets at the nodes. Fig. 5(b) shows that in the DANC, NDANC, and NONCT approaches, all of the nodes can decode all of the packets, as using broadcasting trees guarantees decodability. In contrast, some of the received packets in the probabilistic approach are not decodable. By increasing the forwarding factor, due to the increase in the number of transmissions, the number of decodable packets in the probabilistic approach increases.

Fig. 6(a) shows the total number of transmissions for different approaches. Both of our heuristics have fewer transmissions than the probabilistic approach. The reason is that the redundant transmissions are removed. Only the PFNC approach with $FF = 0.4$ has less transmissions than the DANC approach, but the number of on-time received packets of the PFNC approach is between 30 to 90 percent of the DANC approach. The DANC method has more transmissions than the NDANC approach since, in the NDANC approach, we code all of the packets together. As it is anticipated, the number of transmissions of the NONCT approach is more than the DANC and NDANC approaches, and the NDANC approach has the fewest number of transmissions. The confidence intervals of the NDANC, DANC, and PFNC approach with confidence level equal to 95% are shown in Fig. 6(b).

Fig. 7 shows the performance of our heuristics. For each simulation run, we calculate the ratio of the number of on-time received packets in our heuristics and in the probabilistic approach with a forwarding factor of 0.4, and we show the empirical CDF of the results. It can be seen that in less than only 3% of the cases, the number of on-time received packets of the NDANC method is less than that of the PFNC approach, while the performance of the DANC approach is always better than that of the PFNC approach. In about 10% of the cases, the performance of the DANC approach is between five and six times that of the PFNC approach, and in about 45% of the cases, the performance of the DANC approach is more than three times that of the PFNC approach.

[1]The diameter of the network is the distance in terms of hop count between the farthest nodes of the network.

VI. Conclusion

We study the problem of energy-efficient broadcasting with deadline constraints. We prove that this problem is NP-complete. Thus, we propose a deadline-aware heuristic to solve this problem. We use the concept of broadcasting trees to select forwarder nodes. Our DANC heuristic classifies the packets into sets, such that coding all of the packets of each set does not result in a deadline miss. Our heuristic also works for the case when packets do not have deadline constraints. In wireless networks with periodic broadcasting, our protocol computes the coding decision once, and based on that, each node determines its responsibility in future rounds.

References

[1] Y. Sasson, D. Cavin, and A. Schiper, "Probabilistic broadcast for flooding in wireless mobile ad hoc networks," in *IEEE WCNC 2003*, vol. 2, Mar 2003, pp. 1124–1130.

[2] J. Wu and H. Li, "On calculating connected dominating sets for efficient routing in ad hoc wireless networks," in *DIALM*, 1999.

[3] W. Lou and J. Wu, "On reducing broadcast redundancy in ad hoc wireless networks," *IEEE Transactions on Mobile Computing*, vol. 1, no. 2, pp. 111– 122, Apr-Jun 2002.

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

[5] S. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.

[6] C. Fragouli, J. Widmer, and J. L. Boudec, "Efficient broadcasting using network coding," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 450–463, Apr 2008.

[7] L. Li, R. Ramjee, M. Buddhikot, and S. Miller, "Network coding-based broadcast in mobile ad-hoc networks," in *IEEE INFOCOM*, May 2007.

[8] S. Yang and J. Wu, "Efficient broadcasting using network coding and directional antennas in MANETs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 2, pp. 148–161, Feb 2010.

[9] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "Xors in the air: practical wireless network coding," in *ACM SIGCOMM*, 2006.

[10] A. Khreishah, C. Wang, and N. Shroff, "Cross-layer optimization for wireless multihop networks with pairwise intersession network coding," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 5, pp. 606–621, 2009.

[11] X. Li, C.-C. Wang, and X. Lin, "Throughput and delay analysis on uncoded and coded wireless broadcast with hard deadline constraints," in *IEEE INFOCOM*, 2010.

[12] C. Zhan and Y. Xu, "Broadcast scheduling based on network coding in time critical wireless networks," in *IEEE International Symposium on Network Coding*, June 2010.

[13] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782– 795, Oct 2003.

[14] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.

[15] J. Wu, *Distributed System Design*. CRC Press, 1999.

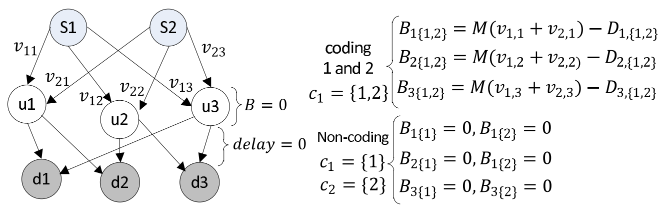
[16] E. Erez and M. Feder, "Convolutional network codes for cyclic networks," in *Proc. NETCOD 2005 Workshops*, 2005.

Appendix

Theorem 1: The problem of energy-efficient broadcasting, subject to the deadline constraints, is NP-complete.

Proof: In order to show that the problem is NP-complete, we need to show that it is NP and NP-hard.

It is easy to show that this problem is NP; if we are given the topology, the set of sources and destinations, and the energy consumption at every node, we can verify in polynomial time, using the BFS algorithm, that these parameters solve the problem.



S1
S2
v_{11}
v_{21}
v_{12}
v_{22}
v_{13}
v_{23}
u1
u2
u3
$B = 0$
$delay = 0$
d1
d2
d3
coding 1 and 2
$c_1 = \{1,2\}$
$B_{1\{1,2\}} = M(v_{1,1} + v_{2,1}) - D_{1,\{1,2\}}$
$B_{2\{1,2\}} = M(v_{1,2} + v_{2,2}) - D_{2,\{1,2\}}$
$B_{3\{1,2\}} = M(v_{1,3} + v_{2,3}) - D_{3,\{1,2\}}$
Non-coding
$c_1 = \{1\}$
$c_2 = \{2\}$
$B_{1\{1\}} = 0, B_{1\{2\}} = 0$
$B_{2\{1\}} = 0, B_{1\{2\}} = 0$
$B_{3\{1\}} = 0, B_{3\{2\}} = 0$

Fig. 8. A reduction from an instance of a vector packing problem to an energy-efficient broadcast with deadline constraints. (B_{jc} represents the decoding delay for the packets of the set c at node d_j.)

In order to show that it is NP-hard, we need to provide a polynomial time reduction from a well known NP-complete problem. We choose the *vector packing problem* as the known NP-complete problem. In vector packing, we have K vectors, each with N positive integers. The i-the vector can be represented by $V_i = [v_{i1}, \ldots, v_{iN}]$. We also have identical bin vectors. Each bin vector contains N integers and can be represented by $[b_1, \ldots, b_N]$. The problem is packing the vectors in as few bins as possible. The constraint is that the sum of the vectors in each bin cannot exceed the size of the bin. Formally, the problem can be described as minimizing L, subject to: $\sum_{i \in l} v_{ij} \leq b_j, \quad \forall l \in \{1, \ldots, L\}, \forall j \in \{1, \ldots, N\}$, where $i \in l$ means that the i-th vector is packed in the l-th bin.

The reduction is as follows. For every instance of the vector packing problem, we generate an instance of our problem according to the following rules. First, we place K sources, N intermediate nodes $u_i, i \in \{1, \ldots, N\}$, and N destination nodes in the graph. We connect each source to all intermediate nodes and each intermediate node to K different destinations, such that each destination node has K input links from K different intermediate nodes. Then, we set the delay of the link between s_i and u_j to $v_{ij}, \forall i, j$, the delay for the links connecting the u and d nodes to zero, and all of the transmission costs to zero except for the u nodes where we set the cost to 1 per sent packet.

Let $M = \frac{\max_{ij}(v_{ij})}{\min_{ij}(v_{ij})}$ and let D_{jc} represent the delay for receiving the $|c|$ linearly independednt packets from the set c at node j when we choose to code the packets in the set c. We set the decoding delay for the packets of the set c at node d_j as $M \sum_{i \in c} v_{ij} - D_{ic}$. Note that due to the multiplication by M, the decoding delay is always ≥ 0. Therefore, the total delay at node d_j to receive and decode the packets in the set c would be $M \sum_{i \in c} v_{ij}$. Also, the total cost of transmissions is proportional to the number of partitions where coding is allowed. This is due to setting the cost of all transmissions to zero except the cost of transmissions at the u nodes.

After doing this reduction, if we set the deadline of packet p_i to reach d_j to Mb_j, it is easy to see that the vector packing problem is solvable iff the minimum cost deadline-aware problem is solvable on the constructed graph. ■

Fig. 8 shows a reduction from a vector packing problem with three 2-dimensional vectors to an energy-efficient broadcasting problem with deadline constraints.