On Minimizing Broadcast Completion Delay for Instantly Decodable Network Coding

Sameh Sorour and Shahrokh Valaee
The Edward S. Rogers Sr. Department of Electrical and Computer Engineering
University of Toronto
Toronto, ON, M5S 3G4, Canada
Email:{samehsorour, valaee}@comm.utoronto.ca

***Abstract*— In this paper, we consider the problem of minimizing the mean completion delay in wireless broadcast for instantly decodable network coding. We first formulate the problem as a stochastic shortest path (SSP) problem. Although finding the packet selection policy using SSP is intractable, we use this formulation to draw the theoretical properties of efficient selection algorithms. Based on these properties, we propose a simple online selection algorithm that efficiently minimizes the mean completion delay of a frame of broadcast packets, compared to the random and greedy selection algorithms with a similar computational complexity. Simulation results show that our proposed algorithm indeed outperforms these random and greedy selection algorithms.**

***Index Terms*—Wireless Broadcast; Instantly Decodable Network Coding; Online Algorithms; Stochastic Shortest Path Problem; Maximal Weighted Clique Search**

I. INTRODUCTION

Network coding (NC) has shown great abilities to substantially improve transmission efficiency, throughput and delay over broadcast erasure channels. These merits can greatly impact the performance of emerging and proliferating streaming and real-time applications that require content to be received quickly and reliably from a sender over lossy channels, such as satellite imaging, roadside to vehicle safety and streaming communications and internet TV. In these applications, transmissions are subject to packet losses due to channel impairments such as wireless fading and interference. These losses are perceived as packet erasures at higher layers, and are usually modeled by independent erasure channels for different receivers. Consequently, network coding can play a big role in exploiting the diversity of received and lost packets by different receivers to speed up the packet recovery process.

The design of online NC packet selection algorithms that optimize throughput and delay performances over single-hop broadcast erasure channels, has been an intensive area of research [1]–[6] in the past two years. In [1], the authors proposed an online selection algorithm for the three-receiver case, proved its rate optimality and conjectured that it achieves an asymptotically optimal average delay. In [2] and [3], several greedy online NC selection algorithms, minimizing a given notion of unordered delay, were compared for i.i.d. erasure channels. However, these proposed algorithms performed unprioritized packet selection for each NC transmission and did not consider the channel conditions in their selection procedures. [4] proposed a prioritized and channel-aware packet selection algorithm for *instantly decodable network coding (IDNC)*, which reduces the same delay notion in [2] and [3] over Markovian ON-OFF channels. In [7], we proposed an IDNC algorithm based on a maximal clique search over a graph defining all possible instantly decodable packet combinations. However, we assumed that the clique search is done randomly without considering delay minimization nor channel conditions.

On the other hand, [5] and [6] employed Markov decision processes (MDP) [8] to find the optimal NC selection policy that minimizes the distortion of video streams over a finite transmission horizon. However, the dimensionality of the MDP's state and action spaces makes the computation of these optimal policies intractable. In [6], a simulation based dynamic programming algorithm was proposed to reduce the computational complexity. However, the resulting complexity of the proposed algorithm is still intractable.

In this paper, we address the following question: *Given the knowledge of received and lost packets at different receivers and their packet loss rates, how can we design an efficient online packet selection algorithm for IDNC that can minimize the mean completion delay of a frame of packets?* We consider IDNC for its numerous desirable properties. First, IDNC provides instant packet recovery upon appropriate IDNC packet reception, a property that general linear and random NC lack. This property is crucial to reduce decoding delay for the proliferating streaming and real-time applications, which are increasingly employing multiple description source coding [5] and thus are insensitive to in-order delivery. Moreover, IDNC encoding can be implemented using binary XOR, which eliminates complicated operations over large Galois fields as required by linear NC. This XOR encoding also simplifies the decoding process at the receivers as each receiver can simply cancel out the packets it already knows. This eliminates the need for matrix inversion at the receivers, which is a computational bottleneck in linear and random NC [4], thus allowing the design of simple and cost-efficient receivers.

To answer the above question, we first formulate the problem as a *stochastic shortest path (SSP) problem*, which is a special case of MDP that has an absorbing state. Although this formulation suffers from the same curse of dimensionality as in [5], [6], we mainly employ it to draw the theoretical properties of efficient packet selection algorithms. Based on these properties, we propose a simple yet efficient online IDNC selection algorithm, which selects the IDNC packets using a maximum weight vertex search over the graph employed in [7]. Simulation results show that this proposed algorithm achieves a lower mean completion delay compared to the random selection algorithm (that selects served receivers arbitrarily at each step) and the greedy selection algorithm (that serves the maximum number of receivers or maximum requested packets at each step) with similar computational complexity to the random algorithm.

The rest of the paper is organized as follows. In Section II, we introduce the system model and parameters. The IDNC graph is illustrated in Section III. We present the problem formulation using SSP in Section IV and draw from it the properties of efficient IDNC selection algorithms in Section V. Our proposed IDNC selection algorithm is introduced in Section VI and its performance is compared against random and greedy algorithms in Section VII. Finally, Section VIII concludes the paper.

II. System Model and Parameters

The model consists of a wireless sender that is required to deliver a frame (denoted by $\mathcal{N}$) of N source packets to a set (denoted by $\mathcal{M}$) of M receivers. The sender initially transmits the N packets of the frame uncoded in an *initial transmission phase*. Each receiver feedbacks to the sender a positive/negative acknowledgement (ACK/NAK) for each received/lost packet. At the end of the initial transmission phase, two sets of packets are attributed to each receiver i:

- The *Has* set (denoted by $\mathcal{H}_i$) is defined as the set of packets correctly received by receiver i.
- The *Wants* set (denoted by $\mathcal{W}_i$) is defined as the set of packets that are lost by receiver i in the initial transmission phase of the current broadcast frame. In other words, $\mathcal{W}_i = \mathcal{N} \setminus \mathcal{H}_i$.

The sender stores this information in a *state feedback matrix (SFM)* $\mathbf{F} = [f_{ij}]\,,\ \forall\ i \in \mathcal{M}, j \in \mathcal{N}$ such that:

$$f_{ij} = \begin{cases} 0 & j \in \mathcal{H}_i \\ 1 & j \in \mathcal{W}_i\ . \end{cases} \tag{1}$$

After the initial transmission phase, a recovery transmission phase starts. In this phase, the sender exploits the SFM to transmit network coded combinations of source packets. These combined packets must include at most one source packet from the Wants sets of a subset or all the receivers. This NC scheme is referred to as *Instantly decodable network coding*. For each transmission, the receivers send ACK/NAK packets that are used by the sender to update the SFM. This process is repeated until all receivers obtain all the packets. We define the *completion delay* of a frame as the number of recovery transmissions required until all receivers obtain all the packets.

Finally, let $\mathbf{p} = [p_i]$, $i \in \mathcal{M}$ be the packet erasure probability vector. We assume that the entries of $\mathbf{p}$ do not change during the frame transmission period.

III. The Instantly Decodable Network Coding Graph

The IDNC graph is a graph that defines all possible instantly decodable packet combinations. It was first introduced in the context of a heuristic algorithm design solving the index coding problem [9], [10]. The IDNC graph $\mathcal{G}$ is constructed by first generating a vertex v_{ij} in $\mathcal{G}$ for each packet $j \in \mathcal{W}_i$, $\forall\ i \in \mathcal{M}$. Two vertices v_{ij} and v_{kl} in $\mathcal{G}$ are connected if one of the following conditions is true:

- $j = l \Rightarrow$ The two vertices are induced by the loss of the same packet j by two different receivers i and k.
- $j \in \mathcal{H}_k$ and $l \in \mathcal{H}_i \Rightarrow$ The requested packet of each vertex is in the Has set of the receiver that induced the other vertex.

let $\mathcal{K}$ be a maximal clique in $\mathcal{G}$ (a maximal clique is a clique that is not a subset of any larger cliques). From the construction of $\mathcal{G}$, it can be easily inferred that any packet, generated by XORing the source packets identified by the vertices of any maximal clique in $\mathcal{G}$, is an instantly decodable packet. Consequently, IDNC packets can be generated by maximal clique search over $\mathcal{G}$. In the rest of the paper, we say that an IDNC packet *addresses* a receiver if the corresponding clique includes a vertex belonging to this receiver.

In [7], we employed an IDNC algorithm that randomly selects a maximal clique for each transmission. When the ACK/NAK of this transmission is received by the sender, it updates the graph and the procedure is repeated until all receivers correctly detect all packets. One can intuitively suppose that the best selection strategy is to select any maximum clique in $\mathcal{G}$ at each step (a maximum clique is a clique that has the largest number of vertices). We refer to this approach as greedy selection. In this paper, we aim to find a more strategic clique selection approach in order to minimize the mean completion delay over both the random and greedy selection algorithms.

IV. Problem Formulation using SSP

A. The SSP Problem

The stochastic shortest path (SSP) problem is a special case of the infinite horizon MDP, which can model decision based stochastic dynamic systems with a terminating state. In SSP, the different possible situations that the system could encounter are modeled as states $s \in \mathcal{S}$ (where $\mathcal{S}$ denotes the state space of SSP). In each state $s \in \mathcal{S}$, the system must select an action a from an action space $\mathcal{A}(s) \subseteq \mathcal{A}$ that will charge it an immediate cost $c(s,a)$ (where $\mathcal{A}$ denotes the action space of SSP). The terminating condition of the system can be thus represented as a zero-cost absorbing goal state (s_g). Once an action a is taken at state s, the system can move to a state s' with probability $P_a(s,s')$, which only depends on the current state and the taken action. An SSP policy $\pi = [\pi(s)]$ is a mapping from $\mathcal{S} \rightarrow \mathcal{A}$ that specifies a given action to each of the states. The optimal policy π^* of an SSP is the one that minimizes the cumulative mean cost until the goal state is reached.

The algorithms solving SSPs define a value function $V_\pi(s)$ as the expected cumulative cost until absorption, when the system starts at state s and follows policy π. It can be recursively expressed $\forall\ s \in \mathcal{S}$ as:

$$V_\pi(s) = c(s, \pi(s)) + \sum_{s' \in \mathcal{S}(s,a)} P_{\pi(s)}(s, s')\, V_\pi(s')\ , \quad (2)$$

where $\mathcal{S}(s, a)$ is the set of successor states to s when action a is taken (i.e. $\mathcal{S}(s, a) = \{s' | P_a(s, s') > 0\}$). Consequently, the optimal policy at state s can be defined $\forall\ s \in \mathcal{S}$ as:

$$\pi^*(s) = \arg \min_{a \in \mathcal{A}(s)} \left\{ c(s, a) + \sum_{s' \in \mathcal{S}(s,a)} P_a(s, s')\, V_{\pi^*}(s') \right\} . \quad (3)$$

B. Problem Formulation

The packet selection problem that minimizes the mean completion delay for IDNC can be formulated as an SSP problem as follows:

1) State Space $\mathcal{S}$:
States are defined by all possibilities of SFM $\mathbf{F}(s)$ that may occur during the recovery transmission phase. For state s, the matrix represents the content of Has and Wants sets in s (i.e. $\mathcal{H}_i(s)$ and $\mathcal{W}_i(s)\ \forall\ i \in \mathcal{M}$) as defined by (1). Note that, based on this state definition, the cardinality of the state space $|\mathcal{S}| = O\left(2^{MN}\right)$.

2) Action Spaces $\mathcal{A}(s)$:
For each state s, the action space $\mathcal{A}(s)$ consists of all possible maximal cliques in graph $\mathcal{G}(s)$ constructed from the Has and Wants sets of different receivers in state s. Defining $\mathcal{C}(s)$ as the set of maximal cliques in $\mathcal{G}(s)$, the cardinality of state s action space $|\mathcal{A}(s)|$ is equal to $|\mathcal{C}(s)|$.

3) State-Action Transition Probabilities:
To define the state-action transition probability $P_a(s, s')$ for an action $a = \mathcal{K}(s) \in \mathcal{C}(s)$, we first introduce the following two sets:

$$\mathcal{X} = \left\{ i \in \mathcal{M} \;\middle|\; |\mathcal{W}_i(s)| > |\mathcal{W}_i(s')| \right\} \quad (4)$$
$$\mathcal{Y} = \left\{ i \in \mathcal{M} \;\middle|\; |\mathcal{W}_i(s)| = |\mathcal{W}_i(s')| \text{ and } \exists\, v_{ij} \in \mathcal{K}(s) \right\} . \quad (5)$$

The first set includes the receivers whose Wants sets have decreased from state s to state s'. This means that these receivers have successfully received an IDNC packet, which addressed them by one of their missing packets. The second set includes receivers that have been addressed by the IDNC packet but their Wants sets did not change due to their loss of this IDNC packet. Based on the definitions of these sets, $P_a(s, s')$ can be expressed as follows:

$$P_a(s, s') = \prod_{i \in \mathcal{X}} (1 - p_i) \cdot \prod_{i \in \mathcal{Y}} p_i\ . \quad (6)$$

4) State-Action Costs:
The mean completion delay is defined in SSP terms as the expected number of transitions in the process before arriving to the goal state. Since any transition (due to any action) takes one packet transmission, the cost payed by the process is one time-slot. Consequently, the costs of all actions in all states should be set to 1. In other words, $c(s, a) = 1\ \forall\ a \in \mathcal{A}(s), s \in \mathcal{S}$.

C. SSP Solution Complexity

The optimal policy of an SSP problem can be computed using the famous policy iteration and value iteration algorithms. The complexity of these algorithms are $\Theta(|\mathcal{S}|^n |\mathcal{A}(s_s))$, where s_s is the starting state and the value of n depends on the employed version of the algorithms. According to the dimensions of $\mathcal{S}$ and $\mathcal{A}(s)$ described in Section IV-B, we conclude that obtaining the optimal policy is impossible in real-time for typical values of M and N. Even the simulation based technique proposed in [6] will not be able to compute the optimal policy in real-time since its complexity still scales with $|\mathcal{S}|$.

V. Properties of Efficient IDNC Selection Policy

In this section, we explore the properties of the SSP formulation described in Section IV-B and draw some properties that characterize an efficient policy minimizing the mean completion delay. From Section IV-B, we can infer that the SSP formulation has the following two properties:

- *Non-singleton acyclicity*: This property arises from the fact that no state can be revisited once the process moves to a next state. Indeed, if some packets are received by some receivers when an action is taken at a given state, there is no means of going back with these receivers not having these packets. However, a state can revisit itself (singleton cycles) if all the receivers addressed by the taken action do not receive the IDNC packet.
- *Non-increasing successor value functions*: Since there are no cycles of size more than one, successor states of a given state are all closer to the goal state than itself. Consequently, the expected cost to absorption starting from a given state is always greater than or equal to the expected costs to absorption starting from all its successor states.

These two properties can be employed to draw the properties of the optimal policy π^* minimizing the mean completion delay as follows. If the system is at state s, we have:

$$\begin{aligned} \pi^*(s) &= \arg \min_{a \in \mathcal{A}(s)} \left\{ 1 + \sum_{s' \in \mathcal{S}(s,a)} P_a(s, s')\, V_{\pi^*}(s') \right\} \\ &= \arg \min_{a \in \mathcal{A}(s)} \left\{ \sum_{s' \in \mathcal{S}(s,a)} P_a(s, s')\, V_{\pi^*}(s') \right\} \\ &= \arg \min_{a \in \mathcal{A}(s)} \left\{ \mathbb{E}_a \left(V_{\pi^*}(s') \right) \right\} , \end{aligned} \quad (7)$$

where $\mathbb{E}_a$ is the expectation operator over the different transition possibilities when action a is taken. Thus, the optimal action at state s is the action minimizing the expectation of the optimal value functions of the successor states. But since all successors of state s are closer to the goal state (thus having less mean completion delays) except for itself, the optimal action at state s is the one that has high probabilities of moving to states with the least expected residual completion delay. To evaluate the expected residual completion delays of successor states, we can exploit the results in [11], [12]. In these references, the mean completion time for a set of receivers to successfully receive a sufficiently large number N of packets using instantly decodable network coding can be expressed as:

$$MCT = \frac{N}{1 - \max_{i \in \mathcal{M}} \{p_i\}} \ . \tag{8}$$

This expression was derived in [11], [12] while assuming that, for large N, the receiver with the worst channel condition will always have the largest Wants set during the entire recovery transmission phase. The expression shows that the worst channel receiver is the one controlling the completion time and must be addressed by one of its missing packets in each recovery transmission. If this receiver was not addressed in each recovery retransmission, the mean completion time would have been larger. Consequently, this expression is approximately the optimal mean completion time for large N and one worst channel receiver, which is achieved when the worst channel receiver is addressed in each IDNC packet.

Extending this argument to the case of multiple worst channel receivers, an efficient policy would be the one addressing the maximum number of such receivers in each instantly decodable packet. Now extending this argument to the case when N is not large, an efficient policy would be the one addressing the maximum number of receivers, having larger individual mean completion times, in each instantly decodable packet. We define the individual mean completion time ($\tau_i(s)$) for receiver i at state s as the expected time for this receiver to receive all its missing packets if addressed in all future transmissions. Based on this definition, $\tau_i(s)$ can be expressed as:

$$\tau_i(s) = \frac{|\mathcal{W}_i(s)|}{1 - p_i} \ . \tag{9}$$

Having this argument illustrated, we are now ready to present our proposed packet selection algorithm to minimize the broadcast mean completion delay.

VI. Proposed Algorithm

We previously stated that the IDNC packet selection in state s is performed by a maximal clique selection from the state's IDNC graph $\mathcal{G}(s)$. According to the properties illustrated in the previous section, an efficient IDNC packet selection algorithm should select, at each visited state, the maximal clique that includes the maximum number of vertices belonging to receivers having the largest $\tau_i(s)$ values. One way to do so is to list all maximal cliques in $\mathcal{G}(s)$ and select the clique satisfying this property. However, maximal clique listing is in general computationally complex. Consequently, it is preferable to design a simpler algorithm that performs a maximal clique selection through a maximum weight vertex search. In this search, the weights of vertices must reflect the required property illustrated in Section V.

Algorithm 1 Maximum Weight Vertex Search Algorithm

Require: $\mathbf{F}(s)$ and $\tau_i(s)$
  Initialize $\mathcal{K}^*(s) = \varnothing$.
  Construct $\mathcal{G}(s)$ and $\mathbf{A}(s)$.
  while $\mathcal{G}(s) \neq \varnothing$ **do**
    Compute $\Delta_{ij}(s)$ and $w_{ij}(s)$ using (11) and (12).
    Select $v^* = \arg\max_{v_{ij} \in \mathcal{G}(s)} \{w_{ij}(s)\}$.
    Add v^* to $\mathcal{K}^*(s)$
    Set $\mathcal{G}(s) \leftarrow \mathcal{G}_{v^*}(s)$ and $\mathbf{A}(s) \leftarrow \mathbf{A}_{v^*}(s)$
  end while

To design the vertices' weights, we first define $a_{ij,kl}(s)$ as the adjacency indicator of vertices v_{ij} and v_{kl} in $\mathcal{G}(s)$ such that:

$$a_{ij,kl}(s) = \begin{cases} 1 & v_{ij} \text{ is connected to } v_{kl} \text{ in } \mathcal{G}(s) \\ 0 & \text{otherwise .} \end{cases} \tag{10}$$

We then define the weighted degree $\Delta_{ij}(s)$ of vertex v_{ij} as:

$$\Delta_{ij}(s) = \sum_{\forall v_{kl} \in \mathcal{G}(s)} a_{ij,kl}(s) \, \tau_k(s) \ . \tag{11}$$

Thus, a large weighted vertex degree reflects its connection to a large number of vertices belonging to receivers with large values of $\tau_i(s)$. We finally define the vertex weight $w_{ij}(s)$ as:

$$w_{ij}(s) = \tau_i(s) \, \Delta_{ij}(s) \ . \tag{12}$$

Consequently, a vertex v_{ij} has a large weight when it both belongs to a receiver with large $\tau_i(s)$ value and is connected to a large number of vertices having large $\tau_i(s)$ values. Let $\mathcal{G}_v(s)$ be the subgraph in $\mathcal{G}(s)$ only including all vertices connected to vertex v. We finally define $\mathbf{A}(s)$ and $\mathbf{A}_v(s)$ as the adjacency matrices of $\mathcal{G}(s)$ and $\mathcal{G}_v(s)$, respectively.

Based on these definitions, we can introduce our proposed packet selection algorithm as follows. The algorithm operates only for visited states. In each visited state s, the algorithm computes a maximal clique $\mathcal{K}^*(s)$ in $\mathcal{G}(s)$ as depicted in Algorithm 1. At first, $\mathcal{K}^*(s)$ is an empty set. The algorithm starts by selecting the maximum weight vertex in $\mathcal{G}(s)$ to be the source node in $\mathcal{K}^*(s)$. For each of the following iterations, the algorithm first recomputes the new vertex weights within the subgraph connected to all previously selected vertices in $\mathcal{K}^*(s)$, then adds the new maximum weight vertex to it. The algorithm stops when there is no further vertex connected to all vertices in $\mathcal{K}^*(s)$. We refer to this algorithm as the *maximum weight vertex search algorithm*. Once the clique is computed, the sender forms and sends an IDNC packets by XORing the source packets identified by the vertices in $\mathcal{K}^*(s)$. According to the received feedback, a new state is visited and the process is re-executed until the absorbing goal state is reached.

VII. Simulation Results

In this section, we present simulation results comparing the performance of our proposed algorithm to:

- Random clique search algorithm employed in [7].

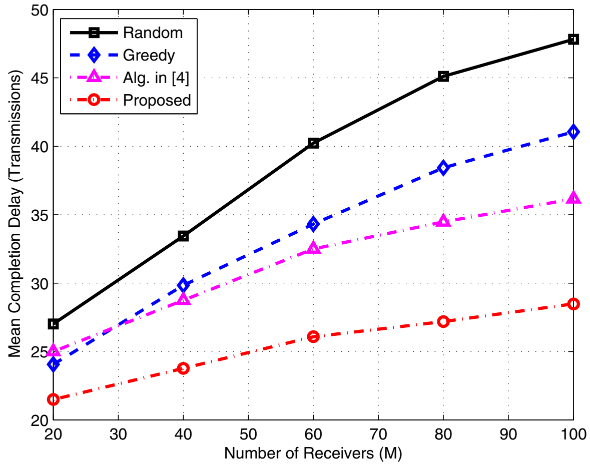

Fig. 1. Comparison of Mean Completion Delays against M

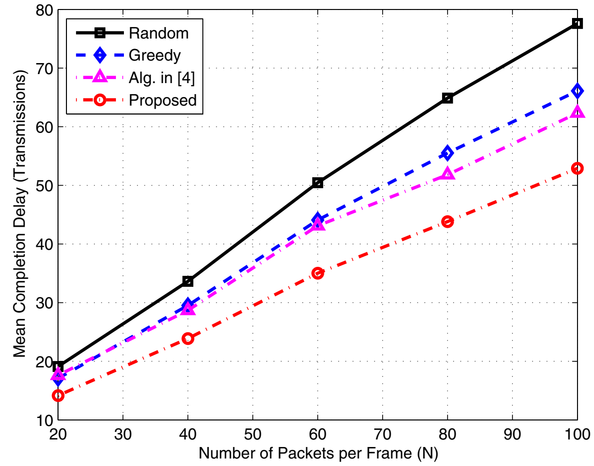

Fig. 2. Comparison of Mean Completion Delays against N

- Greedy clique selection algorithm, which selects a maximum clique in $\mathcal{G}$ for each IDNC packet.
- Selection algorithm proposed in [4], which selects and combines the packets that are most wanted by different receivers while being instantly decodable. In terms of clique selection, this algorithm works similar to our proposed algorithm with packet weighted vertices. A vertex weight is larger if both its corresponding packet is requested by a larger number of receivers and is connected to a larger number of vertices having the same property.

To compare these algorithms to our proposed algorithm, we define the improvement factor (IF) of our proposed algorithm compared to algorithm X as:

$$\text{IF} = \frac{\text{MCD}_X - \text{MCD}_{\text{prop}}}{\text{MCD}_X} \times 100\ . \tag{13}$$

In our simulations, we assume that the packet erasure probabilities of different receivers change from frame to frame in the range [0.05,0.3].

Figure 1 depicts the comparison of mean completion delays achieved by the different algorithms against M, for $N = 40$. Results show that our proposed algorithm achieves an improvement factor of $20 - 40\%$, $10 - 30\%$ and $14 - 21\%$ compared to the random, greedy and packet weighted [4] algorithms, respectively, as M increases.

Figure 2 depicts the same comparison against N for $M = 40$. Results show that our proposed algorithm achieves an improvement factor of $26 - 32\%$, $17 - 21\%$ and $15 - 19\%$ compared to the random, greedy and packet weighted [4] algorithms, respectively.

VIII. Conclusion

In this paper, we aimed to design an IDNC packet selection algorithm that minimizes the mean completion delay in wireless broadcast. We first formulated the problem as an SSP problem and showed that an efficient algorithm should service the maximum number of receivers with larger individual mean completion times. Based on this property, we proposed a simple online selection algorithm that employs a maximum weight vertex search approach over the IDNC graph to determine the best packets to encode at each state. The vertices' weights were designed to reflect the property inferred from SSP. Simulation results show that our proposed algorithm indeed outperforms the random and greedy selection algorithms.

References

[1] J. Sundararajan, D. Shah, and M. Medard, "Online network coding for optimal throughput and delay - the three-receiver case," *International Symposium on Information Theory and Its Applications (ISITA'08)*, pp. 1–6, Dec. 2008.

[2] L. Keller, E. Drinea, and C. Fragouli, "Online broadcasting with network coding," *Fourth Workshop on Network Coding, Theory and Applications (NetCod'08)*, pp. 1–6, Jan. 2008.

[3] E. Drinea, C. Fragouli, and L. Keller, "Delay with network coding and feedback," *IEEE International Symposium on Information Theory (ISIT'09)*, pp. 844–848, June 2009.

[4] P. Sadeghi, D. Traskov, and R. Koetter, "Adaptive network coding for broadcast channels," *Fifth Workshop on Network Coding, Theory and Applications (NetCod'09)*, pp. 1–6, June 2009.

[5] D. Nguyen, T. Nguyen, and X. Yang, "Multimedia wireless transmission with network coding," *Packet Video Workshop (PV'07)*, pp. 326–335, Nov. 2007.

[6] D. Nguyen and T. Nguyen, "Network coding-based wireless media transmission using POMDP," *Packet Video Workshop (PV'09)*, pp. 1–9, May 2009.

[7] S. Sorour and S. Valaee, "Adaptive network coded retransmission scheme for wireless multicast," *IEEE International Symposium on Information Theory (ISIT'09)*, June 2009.

[8] M. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, N. U. John Wiley & Sons, Inc. New York, Ed., 1994.

[9] M. Chaudhry and A. Sprintson, "Efficient algorithms for index coding," *IEEE Conference on Computer Communications Workshops (INFOCOM'08)*, pp. 1–4, April 2008.

[10] S. El Rouayheb, M. Chaudhry, and A. Sprintson, "On the minimum number of transmissions in single-hop wireless coding networks," *IEEE Information Theory Workshop (ITW'07)*, pp. 120–125, Sept. 2007.

[11] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Wireless broadcasting using network coding," *Third Workshop on Network Coding, Theory and Applications (NetCod'07)*, pp. 1–6, January 2007.

[12] ——, "Wireless broadcast using network coding," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 2, pp. 914–925, Feb. 2009.