Coded Retransmission in Wireless Networks
Via Abstract MDPs: Theory and Algorithms

Mark Shifrin, Asaf Cohen, Olga Weisman, Omer Gurewitz
Department of Communication Systems Engineering
Ben-Gurion University of the Negev, Beer-Sheva, 84105, Israel
Email: shifrin@tx.technion.ac.il, coasaf@bgu.ac.il, weismano@post.bgu.ac.il, gurewitz@bgu.ac.il

Abstract

Consider a transmission scheme with a single transmitter and multiple receivers over a faulty broadcast channel. For each receiver, the transmitter has a unique infinite stream of packets, and its goal is to deliver them at the highest throughput possible. While such multiple-unicast models are unsolved in general, several network coding based schemes were suggested. In such schemes, the transmitter can either send an uncoded packet, or a coded packet which is a function of a few packets. Sent packets can be received by the designated receiver (with some probability) or heard and stored by other receivers. Two functional modes are considered; the first presumes that the storage time is unlimited, while in the second it is limited by a given Time To Live (TTL) parameter.

We model the transmission process as an infinite-horizon Markov Decision Process (MDP). Since the large state space renders exact solutions computationally impractical, we introduce policy restricted and induced MDPs with significantly reduced state space, and prove that with proper reward function they have equal optimal value function (hence equal optimal throughput). We then derive a reinforcement learning algorithm, for the induced MDP, which approximates the optimal strategy and significantly improves over uncoded schemes. Next, we enhance the algorithm by means of analysis of the structural properties of the resulting reward functional. We demonstrate that our method scales well in the number of users, and automatically adapts to the packet loss rates, unknown in advance. In addition, the performance is compared to the recent bound by Wang, which assumes much stronger coding (e.g., intra-session and buffering of coded packets), yet is shown to be comparable.

Parts of this work will appear at the IEEE International Symposium on Information Theory, ISIT 2015, Hong Kong.
I. INTRODUCTION

Typical wireless access architectures constitute a gateway, or an Access Point (AP), to which all nearby clients are connected by means of a wireless medium. Among the prominent examples for such architecture is the prevailing IEEE 802.11 or LTE infrastructure mode setting. The downlink traffic implied in such topology comprises an AP sending (usually independent) traffic streams to the corresponding users. Furthermore, common wireless standards incorporate reliability mechanisms in order to overcome the inherently poor qualities of the radio channel. For example, IEEE 802.11, like many other network protocols, attains reliability through retransmission.

Network coding [1] refers to the transmission of predefined functions (usually a linear combination) of packets in order to achieve higher throughput, error correction and better security. However, in order for such combinations to be effective, the listening users need to store the relevant parts of the traffic streams. In particular, since transmission over the wireless channel is broadcast in nature, hence can potentially be heard by non-addressees of the dedicated stream, and with the wireless channel tendency to losses, network coding can be exploited for coded retransmissions.

In this work, we address the aforementioned scenario of a single AP sending unicast streams to $K$ corresponding listeners. We assume that all streams are fully backlogged, i.e., there is a packet pending for each receiver at all times (infinite horizon). We also assume a typical Automatic Repeat reQuest (ARQ) mechanism, in which received packets should be acknowledged. Similar to common standards (e.g., 802.11) we assume a per user FIFO discipline, i.e., a packet to a user will be transmitted only after the previous packet to the same user was received correctly. We adopt the decoding and data storage pattern known in literature as instantly decodable network coding [2], specifically, each user stores packets even if not destined to it, yet only uncoded packets are stored at the receivers while coded combinations are discarded. We assume that the data stored at the listeners is known to the AP at all times; this can be achieved by each receiver piggybacking a list of its current stored packets not destined to it, on the user’s upstream traffic (each DATA or ACK sent by the user to the AP).

Using network coding at the AP, the challenge in each downstream transmission to is determine whether to send an ordinary unicast packet to one of the intended receivers, or to send a linear combinations of packets. Note that even under this seemingly moderate setup, in which users
store only uncoded packets, and the AP has at most a single packet pending per user at a time, since each user can potentially store a packet for each other user (i.e., $2^{K-1}$ possibilities per user), the number of different options for stored packets before each transmission-opportunity (termed the state space) is enormous ($2^{(K-1)K}$). Consequently, no efficient solutions optimal in the general case exists [3].

In this paper, we design a computationally feasible, scalable and robust methodology which effectively addresses the aforementioned problem. Furthermore, in addition to the generic problem described above, we also consider a more complicated setup in which the storage time of packets at the receivers is limited by a Time to live (TTL) constraint, i.e., a packet that its due time has expired is dropped. We present a theoretical framework and a model-based learning implementation which allow us to acquire the on-line transmission and retransmission policy under such channel conditions. In particular, we address three specific challenges. First, the fundamental challenge of network coding - deciding what is the most effective linear combination of the data to be transmitted. This problem becomes further complicated, once TTL constraints are introduced. Second, in contrast to most known works, our model presumes infinite data streams for all listeners, rather than limiting the amount of data to a fixed block. Finally, the encoding decisions are made in an environment without prior knowledge of the packet loss probability. As we elaborate in the related work section, previous works in the area mainly considered various optimization problems for multicast transmissions and/or finite horizons (finite block length). However, this is the first work to address all these challenges in a unified framework.

Our main contributions are as follows: we model the transmission process by a Markov decision processes (MDP). Since the original state space is intractable, we utilize state aggregation. State aggregation (sometimes referred to as state abstraction) is a technique to partition the state space such that all states belonging to the same partition subset are aggregated into one meta-state, such that the same policy applies to all states in the meta-state. In contrast to a complex exhaustive search to find the optimal aggregation, we force a state aggregation, based on proved coding concepts. We further introduce a policy restricted MDP and an induced MDP which undergoes a dramatic state space reduction, and show that in case one chooses the appropriate reward function for the induced MDP, the overall reward of both processes will be
equal. Specifically, instead of keeping track of all possible packets (coded and uncoded), we only keep track of two state variables: (i) The size of the maximal group of users in which each member of the group has a packet destined to each other user in the group but its own (i.e., maximal clique; accordingly, in the sequel we will refer to any set of users each having packets of all other users as a clique, and the maximal such set the maximal clique). Note that for each clique, a single coded packet which linearly combines all the packets destined to the users in the clique can be sent, and each user receiving the coded packet can decode its own packet. (ii) The number of users whose packets are not stored by any other user. Note that this abstraction allows us to significantly reduce the state space from $O(2^{K^2})$ to $O(K^2)$. Consequently, we also restrict the action space, such that the only allowed actions are transmitting a packet to one of the users currently not having its packet backlogged at any other user, or transmitting a coded packet to the maximal clique. Hence, we name the MDP which only allows restricted actions based on the aggregation a policy restricted MDP, and the MDP which sees only aggregated states an induced MDP.

Given the transition probabilities, the optimal policy can be read off the Bellman equation for the induced MDP, which has a relatively small state space and thus can be efficiently solved. However, since the transition probabilities are hard to calculate, we learn them using a model-based learning algorithm. Namely, we derive a novel on-line explore and exploit learning algorithm, which iterates between the learning phase and the Bellman equation solution phase in our problem. Hence, we achieve the optimal policy, which, in turn, results in the optimal throughput (under the constraints imposed by the aggregation and state reduction). Note that this approach is independent of the channel conditions, and works equally effectively for any packet loss, including when the packet loss is not stable and fluctuates around some value. We also study the structural properties of the value function, and use these properties to both gain deep understanding on the behavior of optimal policies and accelerate the reinforcement learning (RL) procedure. Specifically, we prove that under mild conditions, there exists a ”threshold type policy”, namely as a function of the maximal clique size, there is only one transition from one optimal action to the other, and once sending a clique is optimal, it continues to be optimal for the larger cliques. We show that our algorithm is both computationally tractable and scalable. At the same time, its performance is comparable to the upper bounds in [4], which are given
for a much stronger coding scheme, including intra-session coding, much larger state space and buffers, and no TTL.

We incorporate the TTL constraint within the aforementioned MDP model and propose two types of state aggregations. We compare our algorithms with known algorithms in the literature via extensive simulations.

A. Related work

Network coding. While the problem of NC has been widely treated in the multicast setting, multiple unicast still provides a rich ground for ongoing research. Coded retransmissions were considered in [5], where, after sending a finite set of packets to all users and receiving acknowledgements, coded retransmissions are calculated and sent in order to complete the missing packets. Hence, this is a finite horizon problem, where a block is sent only when the previous one is completely decoded. [2] continued the above work, seeking to maximize the coding opportunities. Similar to our problem, in [2] users cannot store coded packets. However, [2] fits a multicast scenario rather than multiple unicast. Moreover, the graph required to identify cliques in [2] grows with the stream size, while it is fixed in our scheme. Finite streams and clique structures were also addressed in [6]. Additional strategies for finite streams can be found in [7], [8] and [9].

In [10], the objective was to minimize the delay using random linear NC. Random NC was also applied for mesh networks in [11]. The finite horizon work [12] minimized the delay by linear programming. Network coding for multi-hop wireless network was addressed in [13]. To the best of our knowledge, no previous work analytically treated the setting which includes side information storage time limit (TTL). Practical insights on storage time constraints and imperfect acknowledge delivery are given in [14]. We also mention the MDP based approach for perfect feedback [15] and partially observable MDP for uncertain feedback [16]. Both works, however, are for finite horizon and do not include state aggregation. Thus, the problem of scalability of the solutions with the size of the stream is raised.

Recently, the seminal work in [4] gave codes and bounds for the erasure broadcast channel. The coding strategy therein was proved optimal for up to 3 users, and bounds were given for general $K$ (two users were considered earlier in [17]). The coding scheme therein assumed more
than one packet per user can be coded and overheard (intra-session coding), while we only allow transmitting the first packet per session. Furthermore, the model in [4] allows storing coded packets, at the price of larger buffers and state space, while our model assumes instantly decodable codes. Nevertheless, we use the theoretical upper bound in [4] to evaluate the performance of the schemes suggested herein, and find them comparable despite the much simpler coding in this work. Note also that calculating the regions in [4] is exponentially complex in $K$, while the algorithms suggested herein scale well with the number of users.

To conclude, none of the aforementioned works addressed the problem of multiple unicast with infinite horizon addressed in this paper. Reference [18] attempted to provide heuristic algorithms for a small number of users, yet the algorithms therein show inferior performance compared to the learning-based solutions suggested in this work. In addition, [18] did not consider the channel condition, while our approach is adjustable to the packet loss uncertainty.

**Index Coding and ARQ.** The relation between NC and Index Coding (IC) [19] was formulated in [20]. The most general formulation of the IC problem constitutes a setting of $K$ nodes, each having a set of packets as side information and expecting an optionally distinct set of packets. At the beginning of the communication, all the data is at the base station, and the goal is to find a transmission strategy to satisfy all demands. Therefore, this is, in essence, a finite horizon problem. Of course, similar to previous works, IC, in general, allows for complex coding over all packets in the block and storing of coded packets at the receivers before decoding. In addition, reference [21] treated IC with side information which includes coded packets as well.

Minimization of the overall transmission time was addressed in [22]. The policy described in [22], if considered on a per-node basis, results in a greedy algorithm, maximizing the information gained from a single transmission. In the MDP-based approach herein, however, the transmission policy accounts for the ability to transit to more rewarding states in the future, hence generalizes the greedy approach.

Index coding in a scenario where each packet should be transmitted to all was compared to an ARQ scheme in [23]. It was shown that as the number of users $K$ grows, the number of transmissions with NC is constant, while it is logarithmic in $K$ in the case of ARQ. ARQ schemes were also analyzed in [24] and implemented in [25], where the authors considered a broadcast network and the queue size at the sender side as the primary performance metric. As for unicast
scenarios, the finite horizon scheme [20] optimized the number of decoding operations, rather than the number of transmissions.

**State aggregation.** As a road-map paper for the state aggregation methods see [26]. This work defined 5 abstraction methods, where the most relevant to our setting is $\pi^*$-abstraction. We partially adopt their definitions of aggregated and detailed (ground) states and the corresponding abstraction function. $\pi^*$-abstraction can be suboptimal compared to the original MDP [27]. However, our approach is different from [26], since we do not attempt to perform a search to find the aggregation which would preserve optimality, but rather, based on key principles in coding and re-transmission, define a robust MDP abstraction, in order to acquire the smallest states space and action space.

**II. Model description**

We now describe the model in more detail. While the sender has an infinite set of packets per receiver, we assume only one such packet (the first in line for that receiver) is active per receiver. The sender does not transmit new packets for a receiver until the active one is received correctly and acknowledged. This restriction conforms with Stop-and-Wait protocols which are utilized in wireless standards, e.g., 802.11. We assume that when a receiver overhears or decodes a packet destined to another, it is able to store it. We assume the AP always knows the states of the receivers using status updates sent by each receiver.

We first treat the case where the stored packets are never outdated. Denote by $M$ the space of $K \times K$ binary matrices, where each $s \in M$ represents a possible state. In particular, each line $i : i \in \{1, \cdots, K\}$ constitutes a vector of indicators such that $s_{ij} = 1$ if and only if user $j$ has a packet designated for user $i$. The state of the system at each time instant $t$ is represented by $s(t) \in M$. A user which successfully decodes its packet, denote it as $k$, immediately sends an acknowledgment. Hence, in this case, $s(k, i) = 0, \forall i$ is set. Setting the entire row $k$ to zero is motivated by the simple reasoning that users that stored the packet prior to the successful transmission can now discard it. The sender can now send the next packet for that user. In case the destination fails to receive its packet, we set $s_{k,k'} = 1$ if the packet is heard by user $k'$ and $s_{k,k'} = 0, k \neq k'$, otherwise.

Next, we formulate the state model for the TTL constrained case. For simplicity, we assume a
system of identical users, i.e., all users have a similar TTL limit. Let TTL equivalent of $T$ time slots be given. Denote the TTL of a stored packet, at some given time slot, as $\tau \in \{1, \cdots, T\}$, and convene to $\tau = 0$ in the case that no packet is stored. Every time slot, for every packet stored by a user, $\tau$ is reduced by 1. Once $\tau = 0$, the corresponding packet is dropped. Denote by $\mathbf{M}^{\text{TTL}}$ the space of $K \times K$ matrices, where each $s \in \mathbf{M}^{\text{TTL}}$ represents a matrix of TTL values associated with undecoded packets held by the receivers. In particular, each line $i : i \in \{1, \cdots, K\}$ constitutes a vector of TTL parameters, such that $s_{ij} = \tau$, if and only if user $j$ has a packet destined to user $i$, and there are $\tau$ time slots left till the packet expires. We assume that the acknowledgments conform to the assumptions similar to those of the scenario without TTL constraint. In case the destination fails to receive its packet, we set $s_{k,k'} = T$ if the packet is heard by user $k'$ and $s_{k,k'} = 0, k \neq k'$, otherwise. In the case the stored packet is retransmitted, and user $k'$ already has its packet stored, the corresponding TTL is updated by setting $\tau = T$.

The action space comprises the possible packet combinations the AP can send at any given time slot. Denote the action at transmission slot $t$ as $a(t)$. The listeners attempt to decode the transmission and the next state $s(t+1)$ is determined.

The setting described above can be seen as a framework including state-space, action-space and the transition probabilities. Due to the Markov property, we deduce that the problem can be formulated as an MDP, with the objective to maximize the transmission throughput. Hence, we define an appropriate stochastic reward $r(s(t+1), a(t), s(t))$, associated with transitioning from state $s(t)$ to state $s(t+1)$ after taking the action $a(t)$, such that positive reward is accumulated for each successfully decoded packet. For example, if a coded packet of $n$ packets is sent, and $m \leq n$ of them are successfully decoded by their intended receivers, we have $r(s(t+1), a(t), s(t)) = m$. Failing to decode gives no reward. Storing a packet at the receiver which is not the addressee gives no reward. However, note that it may increase the potential number of packets decoded in the future (that is, transition to a state with a higher potential value).

In the next section, we bring the technical definition of the MDP and state aggregation, in order to utilize it for the described model. For general definitions and theory of MDP the reader is referred to [28].
III. MDP WITH RESTRICTED ACTION SPACE AND INDUCED MDP

In this section, we introduce the general notation which lays the ground for the state aggregation. We follow the concepts of abstract MDPs in [26], yet adjust our notation and forthcoming analysis to fit our model and results throughout the rest of the paper.

Consider the MDP of the original model. Denote it as $M_0(S, R, P, A, \gamma)$, where $S$ is the state-space, $R$ is a reward function, $P$ are transition probabilities and $A$ is the action space. We consider both long run average cost and discounted cost with $\gamma$ as the discount factor. We term every state $s \in S$ as a detailed state, since it includes a detailed description of which packet was received and by which users.

Denote the set of all admissible policies as $\mathcal{U}$, where a control policy $U \in \mathcal{U}$ is said to be admissible if it only sees current and past values of the states. We denote by $p(s'|s, a)$ the probability to proceed to state $s'$, being in state $s$ and acting with action $a$, and by $r(s', a, s)$ stochastic reward function attained from such instance. We further denote by $r_0(s, a) = \sum_{s'} r(s', a, s)p(s'|s, a)$ the average reward of being in state $s$ and taking action $a$. The discounted infinite horizon cost associated with a given policy $\pi$ and initial state $s_0$ is given by

$$J_\pi(s_0) = \mathbb{E}\left[ \sum_{t=0}^{\infty} \gamma^t r(s'_{t+1}, a^\pi_t, s_t) \mid s_0 \right].$$

The long run average cost associated with policy $\pi_a$ is

$$J_{\pi_a} = \lim_{N \to \infty} \frac{1}{N} \mathbb{E}\left[ \sum_{t=0}^{N} r(s'_{t+1}, a^\pi_t, s_t) \right].$$

(1)

Note that there is no conditioning on the initial state in (1) since it is well-known that in the average case and under mild conditions on the MDP [28] it has no impact on the cost. For simplicity, in this section and further, we only refer to the discounted case. We mention again the average case in section [V] and in the appendices. The value function for the discounted case is given by $V(s_0) = \sup_{\pi \in \mathcal{U}} J^\pi(s_0)$.

We now define the restricted and induced MDPs. These definitions will allow us to work with much simpler MDPs in our communication problem, yet retain the intuitive notation of network coding and, of course, the near-optimal performance. The policy restricted MDP is defined by the state aggregation, such that we restrict using the same policy to all states mapped to a fixed...
value (termed aggregated state) by a given mapping. The induced MDP is formed by the atomic states, induced by the aforementioned aggregated states, and has similar action rules. Hence, it has significantly smaller state space. We now make these definitions concrete.

Definition 3.1. A policy restricted MDP denoted by $\hat{\mathcal{M}}_1 = \mathcal{P}(\hat{\mathcal{M}}_0, \phi, \hat{A})$, is defined by

(I) A restricted set of policies $\mathcal{U}_1 \in \mathcal{U}$, such that for all $\pi_1 \in \mathcal{U}_1$, $\pi_1(s) = \pi_1(\phi(s))$ for some suitable mapping $\phi: S \mapsto \hat{S}$, where $\hat{S} = \bigcup_i \hat{s}_i$ and $\hat{s}_i = \{s_j | \phi(s_j) = s_i, s_j \in S\}$ and

(II) Action space restriction $\hat{a}(s_1) = \hat{a}(s_2)$ iff $\phi(s_1) = \phi(s_2)$, for all $\hat{a} \in \pi_1$.

In another words, we define a policy over $\hat{S}$, which consists of aggregated states. In particular, all detailed states in some aggregated state give the same resulting state $\phi(s)$ in $\hat{S}$. Moreover, we enforce the same policy for all states for which $\phi(s)$ is equal. The corresponding value function is given by $V_{\hat{\mathcal{U}}_1}(s_0) = \sup_{\pi \in \mathcal{U}_1} J^\pi(s_0)$. Fix some $\pi_1 \in \mathcal{U}_1$. Then, with respect to this policy, the probabilities are calculated as follows. First, $p(\hat{s} | s, \hat{a}) = \sum_{s' \in \hat{s}} p(s' | s, \hat{a})$. We assume the existence of stationary probabilities for being in a detailed state, conditioned on an aggregated state. Denote these probabilities as $p(s | \hat{s})$. Second, observe that

$$p(s' | \hat{s}, \hat{a}) = \sum_{s''} p(s' | s'' = \hat{s}, \hat{a})p(s'' | \hat{s}) = \sum_{s''} p(s' | s'', \hat{a})p(s'' | \hat{s}),$$

therefore,

$$p(s' | \hat{s}, \hat{a}) = \sum_{s' \in \hat{s}} \sum_{s''} p(s' | s'', \hat{a})p(s'' | \hat{s}).$$

Consequently, once there is a policy restriction, one can define a probability space which includes only the aggregated states and the corresponding transition probabilities $p(\hat{s}' | \hat{s}, \hat{a})$, omitting the transitions between the detailed states. We next define the induced MDP.

Definition 3.2. MDP $\hat{\mathcal{M}} = \mathcal{I}(\mathcal{M}_0, \phi, \hat{A})$ is induced by policy restricted $\mathcal{M}_1$ on $\mathcal{M}_0$, if

(I) Each state $\hat{s} \in \hat{S}$ uniquely relates to some $\hat{s} \in \hat{S}$; Denote this relation as $\hat{s} \sim \hat{s}$.

(II) For all $\hat{s} \sim \hat{s}$, then the actions $\hat{A}(\hat{s})$ available in $\hat{s}$ are equivalent to $\hat{A}(\hat{s})$. Denote this relation as $\hat{A} \sim \hat{A}$.

(III) The transition probabilities are defined on similar probability space and comply with $p(\hat{s}' | \hat{s}, \hat{a}) = p(s' | \hat{s}, \hat{a})$, for all $\hat{s}', \hat{s}, \hat{a}$, for which $\hat{s} \sim \hat{s}$. 
Note that an induced MDP sees no detailed states. Namely, each state of the induced MDP stands for distinct aggregation of detailed states in a policy restricted MDP. Note that if one takes a sequence of detailed states \( \{ s_0, s_1, s_2, \cdots \} \) and applies \( \phi \) to it, the resulting sequence \( \{ \phi(s_0), \phi(s_1), \phi(s_2), \cdots \} \) is not necessarily Markovian. This is because \( \phi \) is non-injective surjective function. That is, it is not a bijection for the reason the injective property does not hold. However, as we show in the sequel, one can construct transition probabilities from \( \phi(s_i) \) to \( \phi(s_j) \), i.e. the aggregated states, such that the resulting process is Markovian. Note further that the actions defined in (II) above, in practical cases are, in fact, the same actions as in \( \bar{A} \). Denote \( \hat{U} \) defined over \( \hat{A} \). The discounted infinite horizon cost associated with some policy \( \hat{\pi} \in \hat{U} \) is given by \( J^{\hat{\pi}}(\hat{s}_0) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \hat{r}(\hat{s}'_{t+1}, a_t, \hat{s}_t)|\hat{s}_0] \). The corresponding value function is given by \( V_{\hat{U}}(\hat{s}_0) = \sup_{\hat{\pi} \in \hat{U}} J^{\hat{\pi}}(\hat{s}_0) \).

We aim to set the appropriate reward function for the induced MDP such that its value function will be comparable to that of the policy restricted. The relation between \( J(M_0, \phi, \hat{A}) \) and \( P(M_0, \phi, \bar{A}) \) is given by the following proposition.

**Proposition 3.1.** For an MDP \( M_0(S, R, P, A, \gamma) \), a policy restricted MDP \( M_1(S, R, P, \bar{A}, \gamma) \) such that \( M_1 = P(M_0, \phi, \bar{A}) \), and an induced MDP \( \hat{M}(\hat{S}, \hat{R}, \hat{P}, \hat{A}, \gamma) \), such that \( \hat{M} = I(M_0, \phi, \hat{A}) \), with given initial states \( \hat{s}_0 \sim \bar{s}_0 \), there exists a reward function \( \hat{R} \), such that \( V_{\hat{U}}(\hat{s}_0) = V_{\bar{U}}(s_0) \), where \( \{ s_0 | \phi(s_0) = \bar{s}_0 \} \).

See Appendix A for the proof. Intuitively, one sees that the reward of an induced MDP may be interpreted as the suitably weighted sum of the rewards of the corresponding policy restricted MDP, normalized by the sum of the weights. Note that these weights are found by the transition probabilities to the detailed states which compose the corresponding destination aggregated state, \( \bar{s}' \), for which the relation \( \bar{s}' \sim \hat{s}' \) holds. The key point is, that with the proper reward function, the induced MDP achieves the same value function as the restricted one. Note that since \( \hat{U}_1 \subset U \), in general, we have \( V_{\hat{U}}(\hat{s}_0) = V_{\bar{U}}(s_0) \leq V_{\hat{U}}(s_0) \).

We point out the generality of our approach in this paper. Namely, the only parts that determine the aggregation method and, as we show in following sections, the entire solution and algorithms, are the choice of the aggregation function \( \phi \) and of the actions space formed by \( \bar{a}(\phi(s)) \). In particular, they allow to uniquely define \( P(M_0, \phi, \bar{A}) \) and \( I(M_0, \phi, \hat{A}) \).
In the rest of the paper, for the sake of simplicity, we will omit the relation $\sim$, where the process being related to is clear. Hence, in these cases we will refer to both induced and policy restricted MDPs with slight abuse of notation, by merely using $\bar{A}$ and $\bar{S}$ signs.

IV. STATE AGGREGATION AND REINFORCEMENT LEARNING BASED SOLUTION

Having set the ground, we can now provide the formal definition of the state aggregation and restricted policy for the communication problem, where the action is only based on the maximal clique size and the number of empty lines in the state matrix. State aggregation may be equivalently applied for both average and discounted costs. In this section, we mainly focus on the discounted case. We first define the clique structure and associate it with clique transmission. Denote a directed graph $G$ with set of edges $\Gamma$, based on $s$, $\Gamma(s) = \{e_{ij} = \{v_i, v_j\}|s(i, j) = 1\}$, where we assign vertex $v_j$ to user $j$. Consider a set of vertices forming a clique of size $a$, i.e.,

$$C_a = \{v_{i_1}, \ldots, v_{i_a} : s(i_m, j_n) = 1, \forall j \neq i; \ i, j \in \{1, \ldots, a\}\},$$

Then, any such clique can be transmitted as a coded message and, potentially, can be simultaneously decoded by all $v \in C_a$. Denote the size of the maximal clique in $s$ by $L(s)$. Note that while finding a maximal clique is hard in general, graphs resulting from the state matrix in our setting are random and have cliques of logarithmic size [13], hence $L(s)$ can be found efficiently. Denote by $E(s)$ the number of empty lines in $s$. We bring two examples of this state-space aggregation. The first example merely gives the matrices of two different states that map to the same aggregated states. In Appendix B, the second example demonstrates the application of Proposition 3.1 for the calculation of the rewards for the induced MDP.

**Example 4.1.** Consider a communication network consisting of 5 users. Observe the two following states:

$$s_1 = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \quad s_2 = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Note that $s_1$ contains a clique of size 3 with users 2, 3, 4 and a clique of size 2 with users 1, 2. The state $s_2$ contains the 2 cliques of size 3 with users 1, 2, 3 and users 1, 2, 5. There are no empty lines in either state. Next, consider the aggregation such that the state is defined by the couple $\{L(s), E(s)\}$. Indeed, we consider only the maximal size of the clique and the number
of empty lines. Then, both states above pertain to the same aggregated state denoted by \((3, 0)\).

Hence, the same policy in the induced MDP will be applied to them. Note that, of course, once an action is decided, e.g., sending the maximal clique, the actual combination sent depends on the detailed state, that is, sending \(p_2 \oplus p_3 \oplus p_4\) for \(s_1\) or \(p_1 \oplus p_2 \oplus p_5\) for \(s_2\).

Recall that we define the policy restriction in order to find the approximate transmission policy. Denote the MDP referring to the setting described above as \(M_0(M, R, P, A, \gamma)\). Define \(\phi : M \to \{N \times N\}\), such that \(\phi(s) = \{L(s), E(s)\}\). When the state \(s\) will be clear from the context, we will omit \(s\). Observe that the number of unique pairs \(\{L, E\}\) is upper bounded by \(J = (K + 1)K\). Define a state aggregation by the set \(\bar{s}(L', E') = \{s : L(s) = L', E(s) = E'\}\). We say that policy \(\pi_1\) is restricted, if having \(\phi(s') = \phi(s'')\), for any two distinct states \(s'\) and \(s''\) renders \(\pi_1(s') = \pi_1(s'')\). Assuming the storage time is unlimited, we apply restricted policy \(\pi_1\) with actions \(\bar{A} \in \{1, 2\}\), where \(\bar{a} = 1\) stands for sending the maximal coded clique, and \(\bar{a} = 2\) stands for sending an uncoded packet corresponding to a randomly chosen empty line. Note that the action space defined here is not the only plausible option. For example, one may define sending the empty line which has a largest potential to increase the maximal clique. However, our approach is to choose the simplest action possible, as far as both the reinforcement learning (RL) complexity and the convergence time are concerned. We provide an additional explanation on this issue in section \(\text{VI}\). Denote \(M_1 = \mathcal{P}(M_0, \phi, \bar{A})\). Next, define \(\hat{s} \sim \bar{s}\), where \(\bar{s} \in \bar{S}\), and \(\hat{A} \sim \bar{A}\). Denote the corresponding induced MDP \(M_2 = I(M_0, \phi, \hat{A})\).

A. Reinforcement learning for \(M_2\)

This far, we have not explicitly defined the reward \(\hat{R}\) and the transition probabilities \(p(\hat{s}|\hat{s}', \hat{a})\), from the set \(\hat{P}\). Theoretically, one can construct the reward using Proposition \(3.1\). There are two major obstacles. First, the large state space implies that the calculation is computationally difficult. Second, observe that in \(5\) one has to calculate \(p(s''|\bar{s})\). However, these probabilities are policy dependent, while our objective is to find an approximate policy, within the policy restriction imposed by \(M_1\). Considering these difficulties, in practical systems, we offer to use model-based learning in order to estimate the corresponding transition probabilities and the reward functions. Several flavors of learning algorithms are known, e.g., MBIE \([29]\), E3 \([30]\) and R-Max \([31]\). Since our main concern is implementation simplicity, we use RL \([32]\), using...
an $\epsilon$-greedy policy approach. However, the distinctive difficulty of our learning problem is expressed in highly differentiated access frequencies among the various states. For this reason, some algorithms, such as [31], cannot converge in a reasonable time. Indeed, the initiation of all values to a maximal reward proved to be ineffective in our case.

Hence, we derived a new, modified algorithm, which iterates between two steps. Specifically, we utilize the policy restricted $\mathcal{M}_1$ as an input model to perform the estimation. Next, we update the policy, by finding the exact solution for $\mathcal{M}_2$, using the newly learned reward functions and transactions probabilities obtained during the learning step. To this end, we have $\mathcal{M}_1 = \mathcal{M}_1(\mathcal{M}, R, P, \bar{A}, \gamma)$ and $\mathcal{M}_2 = \mathcal{M}_2(\hat{\mathcal{M}}, \hat{R}, \hat{P}, \bar{A}, \gamma)$. Since the state-space of $\mathcal{M}_2$ is very small, compared to that of $\mathcal{M}_1$, we can directly apply value iteration on the corresponding Bellman equation, that is,

$$V(\hat{s}) = \max \left\{ E_{\hat{s}'}[r(\hat{s}', \hat{a} = 1, \hat{s}) + \gamma V(\hat{s}')], E_{\hat{s}'}[r(\hat{s}', \hat{a} = 2, \hat{s}) + \gamma V(\hat{s}')] \right\}. \quad (4)$$

The solution $V(s)$ to Equation (4) serves to find the new policy, which is utilized in order to learn the new transition probabilities. Then, the policy for the next step is update by solving again the Bellman equation. This learning procedure continues until sufficient convergence in $V(s)$ is achieved or until the policy is unchanged. The outcome of this is the optimal policy for the induced MDP and the nearly-optimal corresponding $V$. Note that after a finite number of iterations, we learn the reward which nearly corresponds to the one given in Proposition 3.1. Thus, the optima of $\mathcal{M}_2$ and $\mathcal{M}_1$ coincide. The algorithm, termed Algorithm A, is described formally in the frame above, where $\pi_R$ is a random policy with actions $\hat{a} = 1$ or $\hat{a} = 2$ with probability $1/2$ each, when the choice is feasible.

$\pi_R$ is used during the initialization stage of the algorithm, where the transition probabilities and reward function $\hat{R}$ of $\hat{\mathcal{M}}$ are initialized, and it is used with probability $\varepsilon_k$ in step $k$. The choice of $N_k$ may be equal for all $k$ or may depend on $k$. For smaller $N_k$ the policy updates are more frequent. Note that $\varepsilon_k$ approaches zero as $k$ increases.

For the average cost long run case, the algorithm should be altered by correspondingly adjusting the learning phase (see, e.g., [33]) and the update phase (see, e.g., [28]). We discuss the implementation details and results in Section VI.
B. State aggregation with a TTL constraint

Under a TTL-constraint, a similar MDP formulation can be used. Denote it by \( \mathcal{M}^{\text{TTL}} = \mathcal{M}^{\text{TTL}}(\mathcal{M}^{\text{TTL}}, R, P, A, \gamma) \). We define the aggregation function which maps a detailed state to the corresponding aggregated state. Then, we will be able to design model-based learning, similarly to the case with no TTL constraint. We propose two possible aggregations and compare between them. We name the two methods Aggregation I and Aggregation II and introduce them as follows.

1) Aggregation I: Define \( \phi_I : \mathcal{M}^{\text{TTL}} \to \{N \times N \times N\}, \phi_I(s) = \{F, C, E\} \), Where \( F \) is the lowest strictly positive TTL in \( s \), \( C \) is the size of the maximal clique, which contains the row with \( \tau = F \), and \( E \) is the number of empty lines in \( s \). Note that \( C \) is not necessary equal to \( L(s) \), the maximal clique in \( s \). Denote the action space by \( \bar{A}^I = \{1, 2\} \). For all \( \bar{a} \in \bar{A}^I \), \( \bar{a} = \bar{a}(\phi_I(s)) \) we have \( \bar{a} \in \{1, 2\} \), where \( \bar{a} = 1 \) stands for sending a coded clique \( C(s) \), which contains a line with \( \tau = F \), and \( \bar{a} = 2 \) stands for sending an uncoded packet corresponding to a randomly chosen empty line from \( E(s) \). We define a restricted policy space \( \bar{U}^I \) over the action space \( \bar{A}^I \). Note that \( \phi_I(s') = \phi_I(s'') \) for any two distinct states \( s' \) and \( s'' \) means \( \pi^I(s') = \pi^I(s'') \) for any \( \pi^I \in \bar{U}^I \). Denote \( \mathcal{M}^I_1 = \mathcal{P}(\mathcal{M}_0, \phi_I, \bar{A}^I) \), such that \( \mathcal{M}^I_1 = \mathcal{M}^I_1(\mathcal{M}^{\text{TTL}}, R, P, \bar{A}^I, \gamma) \). Next, define \( \bar{s} \sim \bar{s} \), where \( \bar{s} \in \bar{S} \), and \( \bar{A}^I \sim \bar{A}^I \). Denote the corresponding induced MDP as \( \mathcal{M}^I_2 = \mathcal{J}(\mathcal{M}_0, \phi_I, \bar{A}^I) \).

The basic approach for finding an approximately optimal policy under Aggregation I, is by harnessing Algorithm A. The corresponding Bellman equation is written similarly to what appears in (4), where the solution is found by substituting the relevant aggregated states.
2) Aggregation II: We define the second aggregation type in order to compare it to another purely heuristic algorithm which is defined in the sequel, and to Aggregation I as well. Define \( \phi_{II} : \mathcal{M}^{TTL} \rightarrow \{N \times N \times N\} \), \( \phi_{II}(s) = \{F, L, E\} \), where \( E \) is the number of empty lines in \( s \), \( F \) is the lowest strictly positive TTL in \( s \), and \( L = L(s) \), the size of the maximal clique in \( s \). Note that there is no knowledge about the size of the maximal clique containing the line with \( \tau = F \), as in Aggregation I. Denote the action space \( \bar{A}^{II} = \{1, 2, 3\} \). For all \( \bar{a} \in \bar{A}^{II} \), \( \bar{a} = \bar{a}(\phi_{II}(s)) \) we have \( \bar{a} \in \{1, 2, 3\} \), where \( \bar{a} = 1 \) stands for sending a coded maximal clique \( C(s) \), which contains a line with \( \tau = F \); \( \bar{a} = 2 \) stands for sending an uncoded packet corresponding to a randomly chosen empty line, and \( \bar{a} = 3 \) stands for sending a \( L(s) \), maximal coded clique in \( s \). Note that the action \( \bar{a} = 1 \) presumes no prior knowledge about the size of \( C(s) \). Thus, the decision in this case is myopic as far as the size of clique being sent is concerned. The learning in the case of Aggregation II is performed by utilizing algorithm A. We compare by simulations both aggregation types, with an alternative heuristic policy in Section VII.

V. STUDY OF THE PROPERTIES OF \( V \)

In this section, we present an in-depth view of the suggested abstract MDP-based approach by exploring the properties of the value function found through the reinforcement learning procedure. Our primary objective is to understand the structure of the value function at different initial state points. Namely, we aim to isolate properties of \( V(s) \) related to each one of the model parameters. This, in turn, will allow us to incorporate these properties in the main learning algorithm, resulting in improved speed and precision of convergence. Moreover, it will give us better understanding of how each of the parameters (e.g., clique size) affects the results, and how the overall coding process should behave as a function of these parameters.

In particular, in some cases, we will observe a threshold type policy in one of the parameters. That is, a policy in which there is at most one switching state from one optimal action to the second. Such a property is desirable as once the switching point is found, we may set the actions to their optimal values without the need to iterate until the ultimate convergence. Furthermore, in most cases, such a threshold policy will give a fundamental and rigorous reasoning to very intuitive results, e.g., if sending a coded clique is beneficial for some \( L(s) \), it is definitely beneficial for any \( l > L(s) \).
For simplicity, we demonstrate the proof of the existence of a threshold-type policy for the 1-dimensional aggregation defined below.

3) One-dimensional aggregation: As an alternative to the multi-dimensional aggregation patterns, we introduced an even more coarse abstraction. Namely, define $f : \mathbb{M} \rightarrow \{\mathbb{N}\}$, such that $f(s) = \{L(s)\}$, that is, the size of the largest clique. Define a state aggregation by the set $\bar{s}(L') = \{s : L(s) = L'\}$. While oversimplified, and as such resulting in maybe inferior performance, this aggregation and the induced MDP serve as a good example for which we can investigate the value function and gain important insights.

Proposition 5.2 below proves the existence of a threshold policy under an average cost. Let $\pi_a$ be a maximizer over all $\pi$ in (1). That is: $\pi_a = \arg\max \pi \lim_{N \to \infty} \frac{1}{N} \mathbb{E}\left[ \sum_{t=0}^{N} r(s_{t+1}^*, a_t^*, s_t) \right]$

**Proposition 5.2.** There exists a constant $k$, $k \in \{2, \ldots, K\}$ such that for $0 \leq i < k$, $\bar{a}(\bar{s}(i)) = 2$, yet for $k \leq i \leq K$ we have $\bar{a}(\bar{s}(i)) = 1$.

That is, send the maximal clique (a coded packet) if and only if its size is at least $k$. Otherwise, send an empty line (an uncoded packet). The equivalent statement for the discounted case requires knowing the transition probabilities, which are hard to calculate. The connection between average and discounted costs, however, is well-known and is described by the Blackwell optimality condition \[28\]. In particular, Blackwell optimal policy is optimal for the average cost as well. Yet, as seen from the proof of Proposition 5.2, the optimal policy for the average cost, in this case, is not unique. Hence, the opposite is not necessarily true. Nevertheless, we address this in the simulations.

To understand the proposition intuitively, observe that the probability to increase the maximal clique size is rapidly decreasing as the clique size grows. Thus, intuitively, the cumulative reward which is associated with sending an empty line will have a marginal component which is decreasing and a constant component related to the immediate reward of decoding the transmitted uncoded packet. This is contrasted to the cumulative reward which is associated with sending a maximal clique. Clearly, in the latter case, the reward is constantly growing in the size of the clique. In other words, we keep sending empty lines while the probability to increase the clique size is still significant, yet, when this probability decreases until it is beneficial to collect the reward of a clique, we send the clique.
We will need the following notation for the proof of Proposition 5.2. We say that a state \( s \) is **recurrent under the policy** \( \mu \) if when starting at state \( s \) and acting according to \( \mu \), the probability to return to \( s \) is 1. A state which is not recurrent under \( \mu \) is **transient under** \( \mu \).

**Proof.** [Proposition 5.2] Consider a policy \( \pi_* \), which is optimal for the average long run cost, 
\[
\pi_* = \arg \max_{\pi} J_\pi,
\]
where \( J_\pi \) is given in (1). Denote a set of states \( S_1 \subset S \) such that \( s_i \in S_1 \) if \( a_{\pi_*}(s_i) = 1 \). Denote a state \( s_m \), such that \( s_m \in S_1 \) and \( L(s_m) < L(s_i), \forall i, s_i \in S_1 \). Namely, \( S_1 \) is the set of states for which sending a clique is optimal, and \( s_m \) is the state with the minimal clique in \( S_1 \) - the minimal clique size for which it is optimal to send a clique. We have the following claim.

**Claim 1.** Any state \( s_i \) such that \( L(s_i) > L(s_m) \) is transient under \( \pi_* \).

**Proof.** We use the fact that nodes do not use **coded** packets in order to decode packets not intended to them. Namely, nodes store only uncoded packets intended for other users. Hence, clique transmissions cannot increase the clique size, and, moreover, decrease it with some non-zero probability (note that transmission of an empty line can increase the clique size, yet by at most 1). Therefore, at \( s_i \) the state can be only decreased. Consequently, for any \( s_i \) such that \( L(s_i) > L(s_m) \), \( s_i \) is transient under \( \pi_* \). \( \square \)

Denote the set \( S_r \) such that \( s_i \in S_r \) if \( L(s_i) \leq L(s_m) \), and denote \( S_t = S \setminus S_r \). Now see that by the claim above, \( s_m \) is the only recurrent state in \( S_1 \). Define \( n_m \), the first time under \( \pi_* \) to be in \( s_m \). We have

\[
J_{\pi_*} = \lim_{N \to \infty} \frac{1}{N} \left[ \sum_{n=0}^{n_m-1} r_{\pi_*}(s(n), a(n)) + \sum_{n=n_m}^{N} r_{\pi_*}(s(n), a(n)) \right].
\]

Observe that all states encountered for \( n > n_m \) are recurrent. That stems from the fact that after \( n_m \), the process stays in \( S_r \). Since \( n_m \) is finite a.s., the first sum (once normalized by \( N \)) goes to zero. Next, define policy \( \pi_m \) which acts similarly to \( \pi_* \) for all \( j \) such that \( L(s_j) \leq L(s_m) \) (that is, all recurrent states) yet sets \( a(s_j) = 1 \) otherwise. That is, a threshold policy. Denote by \( n_t \) the first time to hit \( s_m \) under \( \pi_m \). Observe that

\[
\lim_{N \to \infty} \frac{1}{N} \sum_{n=n_m}^{N} r_{\pi_*}(s(n), a(n)) = \lim_{N \to \infty} \frac{1}{N} \sum_{n=n_t}^{N} r_{\pi_m}(s(n), a(n)) = V
\]
Thus $\pi_m$ is also an optimal policy. Note that the relation between $n_l$ and $n_m$ is not essential, since both are finite.

It is left to show that the policy which always sends empty lines, that is, sends no cliques at all is suboptimal. Denote such a policy as $\pi_e$. However, in such a policy the expected reward at each step is given by $1 - p$, and any other policy which sends a clique at any step outperforms $\pi_e$ by some $\epsilon > 0$. This accomplishes the proof of the proposition.

The technique demonstrated in the 1-D case can be extrapolated to more complex aggregations. However, the proofs in these cases will involve treatment of significantly more complex Bellman equations. Alternatively, one may merely assume the existence of a threshold policy, based on observations from simulations. The main advantage of having the threshold-type policy proof/observation is the possibility to enhance algorithm A, as we explain next. Assume there exists a threshold policy in $E$, as was presented in Aggregation I. Namely, once for some $E = i$, there is a switch from optimal action 2 (transmission of an empty line) to action 1 (transmission of a clique), then we deduce that 1 is optimal for all $E < i$, while 2 is optimal for all $E \geq i$.

Heuristically, the threshold structure may be deduced from policy convergence results of the algorithm for other cases. For example, one can see the policy tendency from cases with lower packet loss, or with smaller state space, e.g., smaller number of users, smaller TTL, etc. Assume the existence of a threshold policy in one of the parameters (e.g. $F, C, E$). Now, at step 4 of the algorithm, in case the policy in some (possibly rarely visited) state is not yet clear at some point of the algorithm run, correct it according to the already known (or conjectured) threshold rule. This method will accelerate the overall convergence. Another useful property of $V$, which gives good understanding of its behavior, is its slope. We show both upper and lower bounds on this slope in Appendix C. Similarly, the bounds can be useful for the manual calibration of the value function in order to speed up the convergence.

VI. Simulation results

In this section, we evaluate the suggested transmission strategy through extensive MATLAB simulations. Our simulation results provide insight on the impact of each of the mechanisms described throughout the paper on the inclusive strategy. Specifically, we thoroughly examine the effect of different parameters such as TTL and packet loss probabilities on different aspects.
of the decision processes such as the effect on the value function or on the policy structure. In addition we evaluate our algorithm and compare the different aggregations suggested. In our simulations we consider a single cell comprising an AP and $K$ receivers. Since our results relate to the traffic from the AP to the users our simulations only consider the downstream traffic. We assume that all $K$ users have pending traffic waiting to be transmitted. An i.i.d Bernoulli channel error is assumed, where each packet transmission is received or dropped by each user independently with probability $1 - p$ and $p$, respectively, and is independent between different transmission attempts. At each transmission attempt the AP converts the detailed state into the aggregated state and transmits a coded/uncoded packet according to the current policy. In all cases compared, the AP activates the learning routines considering the discounted infinite horizon cost. Thus, it computes the values attained by value functions for all possible initial states. We later use the same policy for calculating the long run average cost. Note that based on the Blackwell optimality argument (e.g., [28]), if $\gamma \to 1$, under mild conditions the policy which is optimal for the discounted problem is optimal for the average cost problem as well. The number of iterations for each phase (learning and improvement) is set in accordance with the specific configuration.

A. Results without a TTL constraint

We start by evaluating the policy resulted from our learning algorithm, for the proposed aggregation in the case of no TTL constraint (Section IV). We compare our results with the bounds obtained in [4]. The aggregation for the TTL-unconstrained case constitutes a 2-dimensional state space, namely, the size of the maximal clique $C$ and the number of empty lines $E$ (Section IV). The action space comprises two possible actions, transmitting to a user that its packet was not received by any user (empty line in the state matrix) and transmitting to the maximal group of users in which each member of the group has a packet destined to every other user in the group (maximal clique in the state matrix). Our simulation results clearly depict that the optimal policy is the same regardless of the packet loss probability. In particular, the optimal policy is defined by transmitting a random empty line whenever there are empty lines ($E > 0$) and transmitting to the maximal clique otherwise. Accordingly, the obtained policy is a threshold-based policy. The intuition behind this strategy is clear: the reward associated with both possible actions,
transmitting a random empty line or transmitting the maximal clique, is time independent, i.e., the expected reward is the same if the transmission occurs now or in one of the following transmission opportunities. Moreover, since any empty line is not included in any clique all the more so in the maximal clique, yet transmitting an empty line can potentially increase the size of a clique without incurring any penalty for delaying the current maximal clique transmission, it is worthy to fill in the state matrix such that no empty lines are left, and only then to transmit the maximal clique. Note that this policy coincides with the one heuristically suggested in [18] denoted as the semi-greedy algorithm. Accordingly, the simulation results imply that under the restricted action space described above, the semi-greedy algorithm [18] is optimal, as long as no TTL constraints are applied. Moreover, for the simple case of 2-users system, these results achieve the sum-capacity which is found according to [17] and [4]. Figure 1 compares the performance in a 5-users network with aggregation to the bound in [4], derived for systems with much stronger coding capabilities. We denote it as the Wang upper bound. Note that to calculate the bound one needs to solve 120 inequalities, hence the graph has small discrepancies. For larger systems, such calculations may be too complex.

B. Results for TTL constrained aggregations

Next we evaluate the performance of the suggested transmission strategy under TTL constraints.

We simulated Aggregation I (Section IV), aiming to examine the structure of the value function for all feasible states. Namely, we try to to get a grip on the effect of different parameters on $V$. Our objective was to identify simple observation properties such as monotonicity, convexity and threshold-type structure. Such properties can be potentially utilized for the RL convergence speed-up. This will allow to successfully operate larger systems. We start by examining a system with $K = 5$ receivers. We set $\gamma = 0.99$. The results are depicted in Figure 2. The $Y-axis$ depicts the value attained by each state, $V(F; C; E)$, (denoted by asterisks). Each value corresponds to the given initial state. $X-axis$ relates to an enumeration of the states, $\{1, 2, \cdots \}$. Note that the asterisks form
groups of monotoneous patterns of values. In particular, the states are assigned numbers which grow first in TTL ($F$), next with maximal clique size ($C$) and finally they grow with the number of empty lines ($E$). For example, state 1 refers to the state in which there are no empty lines, maximal clique size 1 and $TTL = 9$, State 2 relates to the values of the state in which there are no empty lines, the maximal clique size contains the line with the greatest $TTL$ is 8, state 96 which is the last state refers to the state in which there are 5 empty lines (i.e. the empty matrix).

Note that for the common policy that only allows uncoded transmissions the value is fixed $1 - \frac{p}{1 - \gamma} = 1 - 0.25 = 0.75$, which is below the scale of the graph, i.e., the value for all states is higher than the one for the common uncoded ARQ retransmissions.

We emphasized the evolvement of the value function when only a single parameter varies while holding the other two are fixed. Specifically, in order to understand the effect of empty line on the policy token we emphasize by the dotted (red) line the states in which the $TTL$ and the size of its corresponding clique are constant and equal to two ($F = 2$, $C = 2$) and the number of empty lines is varied ($0 \leq E \leq 3$). This can be intuitively explained by the property that lines which are non-empty contain some information that potentially can be exploited in future transmissions, while the empty lines contain no information whatsoever. In addition, in order to demonstrate the value function dependence on the clique size, we emphasize the states in which $TTL$ is fixed and equals 2 ($F = 2$), number of empty lines is fixed (we show two different values), and the clique size varies. Observe $V(2; C; 0)$ and $V(2; C; 1)$ which are represented by the solid cyan and the solid magenta lines, for $E = 0$ and $E = 1$, respectively. As expected, both lines have an increasing pattern with $C$, i.e., the greater the maximal clique which corresponds to the line with lowest $TTL$, the greater the value function. By observation, one can also assume that the value function has a convex increasing form in $C$ (cyan and magenta lines) and convex decreasing in $E$ (the red line).

We explore next the dependence of the policy found for Aggregation I on various parameters,
Fig. 3. Approximately optimal policy, for a system with $K = 5$ users and $TTL = 5$. 1 stands for sending the clique containing the oldest line, while 2 stands for sending a random empty line. Note the dependence of the policy on the packet loss, e.g. in states 7, 9, 11, 21, 28, 29 (These states are marked in red). The impact of the parameter $F$ can be seen from states $\{F, 3, 2\}$, (states 20, 21, 22), for example. Note that the clique is always sent in the cases where $F = 1$, i.e., the oldest line in this clique is about to expire. In the cases where $F > 1$, the policy depends on the packet loss, and generally tends to change to 2 once the oldest line is about to expire. 

at packet loss range of 5% to 35%. The results are shown in Figure 3. For reference convenience, the first column denotes the state enumeration. Recall, that 1 stands for sending the maximal clique containing the oldest line, while 2 stands for transmitting a random empty line. This results clearly demonstrate that the algorithm overcomes the packet loss uncertainty and converges to the optimal policy in accordance with the channel condition. In addition, the threshold-type property, which can highly accelerate the RL procedure, in the $F$, the TTL, of the oldest line can be seen in the table (see states (20-22), (27-29).) The proof of this property is hard to accomplished, as it relies on the transition probabilities, which are hard to attain. As explained in Section IV they are approximated by RL. Accordingly, simulation-based exploration is imminent in order to identify structural properties. Alternatively, one can attempt to state the threshold property for the average long run case, as we proved for the 1-D case in Section V. Note that as long as all three dimensions of $V$ are viewed, the thresholds are expected to form three-dimensional surfaces.

We conclude the observations above by proposing an effective speedup for Algorithm $A$. The
proposed enhancement stems from simulation results and by the previously discussed properties of value function in section V. First, in order to successfully operate a larger system, one can solve a (trial) system with small number of users with same aggregation and the same channel conditions. Next, the resulting optimal policy can be extrapolated in order to get the policy for the desired system, for example, threshold and monotonicity patterns, as we examined above. In particular, define an approximating policy $\pi^0_X$ using an assessment based on the policy found from a smaller system and the observed properties. Heuristically, this policy should allow a randomization around conjectured threshold states. Next, an adjustment of $\hat{V}_i$ and that of $\pi^{k+1}_X$ is heuristically performed. Again, this improvement can be done using the estimated properties of the value function, or can be combined within the regular run of the reinforcement learning as it appears in Algorithm A.

In order to evaluate the effect of TTL on the policy, we compare both Aggregation I and Aggregation II with the greedy and semi-greedy algorithms proposed in [16]. Specifically, the greedy algorithm aims at maximizing the instantaneous reward received for each transmission opportunity. Hence, the policy according to the greedy algorithm is to transmit the maximal clique for each transmission opportunity. Whenever there is no clique (i.e., $C \leq 1$) transmit a random empty line. The semigreedy (SG) policy is defined in the subsection above. Figure 3 (left and middle) compares the value function of the discounted infinite horizon cost with a zero matrix as the initial state for the various policies, and Figure 3 (right) compares the value functions for the average cost long run of the different policies, where we used the policy found in the discount case.

Figure 4 (left) clearly depicts that as expected under the TTL constraints the semi-greedy algorithm works poorly, and almost coincide with the uncoded policy, as it does not take into account lines which can be discarded, hence misses clique transmission opportunities just for trying to fill the matrix with non-empty lines. Moreover, in system where the number of users is greater than TTL, the AP will never be able to fill the state matrix with non-empty lines and the aforementioned semi-greedy algorithm coincides with the uncoded algorithm which sends only uncoded packets. Accordingly, we devised an alternative heuristic algorithm, termed modified semi-greedy (MSG). MSG differs from SG in that whenever there is a line in which the TTL is going to expire on the next slot (i.e., $TTL = 1$) the AP transmits the maximal
clique containing the oldest line. The results of the MSG heuristic are also depicted in Figure 4. Note that MSG is oblivious to the channel conditions and acts identically for any packet loss (Figure 4 left). Further note that even though both policies rely on the same parameters to make a decision, i.e., both perform based on the triplet \{oldest line, maximal clique size, number of empty lines\}, Aggregation II outperforms the MSG algorithm at all packet loss values. Which can be explained by that MSG, while being effective as a simple heuristic algorithm, neglects the channel condition, i.e., MSG provides only a single retransmission opportunity for a packet before it gets obsolete, regardless the loss probability. This is opposed to Aggregation II which effectively adjusts the policy to the channel packet loss with no prior knowledge on the packet loss (\(p\)), based on the on-line learning. Accordingly, the advantage of Aggregation II becomes more prominent at higher packet loss values, as can be seen in Figure 4.

Next, observe that when the number of users is greater than TTL, the effect of the surplus of the number of users is negligible. This stems, since action space that the empty line is chosen at random, from the fact that at most \(E = TTL\) lines can have non-zero entries at all times. Indeed, we see that \(K = 10\) leads to almost no improvement in performance compared to the \(TTL = 5\) case (the corresponding lines in the middle graph are almost coincide). Hence, we conjecture that for the case where \(K > TTL\), further state-space minimization could be done. However, once one increases the \(TTL\) parameter the performance improvement is tangible. These results are seen on the middle graph as well. Finally we compare the average cost long run simulation results (Figure 4 right). Relying on Blackwell optimality, we used the same policies we found
for the discounted case. One sees the same performance gradation as for the discounted cost.

APPENDIX

A. Proof of Proposition 3.1

Proof. We prove by constructing a reward function \( \hat{R} = \{ \hat{r}(\hat{s}', \hat{a}, \hat{s}) \} \). Let the reward associated with \( M_1 \) be \( r_1(s', \bar{a}, \bar{s}) \). By definition of \( r_1 \),

\[
\mathbb{E}r_1(s', \bar{a}, \bar{s}) = \sum_{s'' \in \bar{s}} [r(s', \bar{a}, s'') ] p(s''|\bar{s}),
\]

(5)

Partitioning all states in \( S \) to the aggregated states, we have:

\[
r_1(\bar{s}, \bar{a}) = \sum_{\bar{s}'} \sum_{s' \in \bar{s}'} r_1(s', \bar{a}, \bar{s}) p(s'|\bar{s}, \bar{a}).
\]

(6)

Similarly to \( r_1(\bar{s}, \bar{a}) \) in \( M_1 \), define \( \hat{r}(\hat{s}, \hat{a}) \) in \( \hat{M} \):

\[
\hat{r}(\hat{s}, \hat{a}) = \sum_{\hat{s}'} \hat{r}(\hat{s}', \hat{a}, \hat{s}) p(\hat{s}'|\hat{s}, \hat{a})
\]

(7)

Thus, we wish to find \( \hat{r}(\hat{s}', \hat{a}, \hat{s}) \) such that

\[
\mathbb{E}r_1(\bar{s}, \bar{a}) = \mathbb{E}\hat{r}(\hat{s}, \hat{a}).
\]

(8)

Since both the summation in (5) and the outer summation in (7) are over all aggregated states, (8) will be achieved by taking:

\[
E\hat{r}(\hat{s}', \hat{a}, \hat{s}) p(\hat{s}'|\hat{s}, \hat{a}) = \mathbb{E} \sum_{s' \in \bar{s}'} r_1(s', \bar{a}, \bar{s}) p(s'|\bar{s}, \bar{a}).
\]

That is,

\[
\mathbb{E}\hat{r}(\hat{s}', \hat{a}, \hat{s}) = \sum_{s' \in \bar{s}'} \mathbb{E}r_1(s', \bar{a}, \bar{s}) p(s'|\bar{s}, \bar{a})
\]

(9)

with the mapping \( \hat{s} \sim \bar{s} \) and \( \hat{a} \sim \bar{a} \). Hence, we have the desired result:

\[
V^{\hat{r}}_{U}(\hat{s}_0) = \sum_{n=0}^{\infty} \gamma^n r(\bar{s}, \bar{a}) = \sum_{n=0}^{\infty} \gamma^n r_1(\bar{s}, \bar{a}) = V^{r_1}_{U}(s_0)
\]
B. State aggregation example

Example 1.2. The following demonstrates state aggregation (as it was defined by Aggregation I in Section IV and results of Proposition 3.1). Consider the case of 4 users. Each line holds the packets of user \( i \). We exemplify the detailed states where \( L = 3, N = 1 \). These states are aggregated into the state \( \bar{s}_{3,1} \). Possible cliques are demonstrated in the detailed states \( s_1, s_2, s_3, s_4 \) below. Observe that these states contain only minimal number of 1-s.

\[
\begin{align*}
 s_1 &= \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \\
 s_2 &= \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \\
 s_3 &= \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \\
 s_4 &= \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}
\end{align*}
\]

See that in \( s_1 \), there are 8 additional options for the last column. In particular, observe the following four states with the same empty line and the same clique as in \( s_1 \).

\[
\begin{align*}
 s_5 &= \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \\
 s_6 &= \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \\
 s_7 &= \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \\
 s_8 &= \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}
\end{align*}
\]

The same holds for \( s_2, s_3 \) and \( s_4 \). Concluding, the state \( \bar{s}_{3,1} \) aggregates 32 detailed states.

There are two possible actions, denote them \( \bar{a} = 1 \) and \( \bar{a} = 2 \), which stand for transmitting the clique or transmitting (the only) empty line. Note that the encoded message for \( s_1 \) contains the bits 1, 2, 3, for \( s_2 \) it contains the bits 2, 3, 4, for \( s_3 \) it contains the bits 1, 3, 4 and for \( s_4 \) it contains the bits 1, 2, 4. The probabilities \( p(s = s_1|\bar{s} = \bar{s}_{3,1}) \) stand for the probability to be in a specific detailed state which belongs to the aggregated state \( \bar{s}_{3,1} \). The rest of the example concentrates on the state \( s_5 \in s_{3,1} \) and action \( \bar{a} = 1 \), i.e., the clique is transmitted. Assume the action results in the detailed state \( s_a \).

\[
\begin{align*}
 s_a &= \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \\
 s_9 &= \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}
\end{align*}
\]

Clearly, \( s_a \in \bar{s}_{0,3} \). Denote by \( p \) the packet loss probability. For simplicity, we assume the same \( p \) for all users. This transition occurs with probability \( p(s_a|\bar{a} = 1, s_5) = p(1 - p)^2 \). That is, two of the users in the clique (1 and 3) successfully decoded the encoded bit, while user 2 failed to do so. See that the same transition can happen from state \( s_9 \). That is, the clique containing
encoding of 2, 3, 4 was transmitted, and user 2 failed to decode. This transition occurs with probability \( p(s_a|\bar{a} = 1, s_a) = p(1 - p)^2 \) as well. We sum up over all such detailed states (as in (2)):

\[
p(s_a|\bar{a} = 1, \bar{s} = \bar{s}_{3,1}) = \sum_{s_i \in \bar{s}_{3,1}} p(s_a|\bar{a} = 1, s_i)p(s_i|\bar{s}_{3,1}),
\]

This summation counts over all 32 detailed states in \( \bar{s}_{3,1} \). Clearly, some of the probabilities, e.g., \( p(s_a|\bar{a} = 1, s_2) \) are zero, hence do not contribute to the summation. For calculation convenience, we assume that in these cases \( r(s_a, \bar{a} = 1, s_i) = 0 \). We calculate the average reward associated with the transition from \( s_{3,1} \) to \( s_a \), according to (5): \( Er_1(s_a, \bar{a} = 1, \bar{s}_{3,1}) = \sum_{s_i \in \bar{s}_{3,1}} r(s_a, \bar{a} = 1, s_i)p(s_i|\bar{s}) \). Note that transition to state \( s_a \), acting \( \bar{a} = 1 \) from \( \bar{s}_{3,1} \), is only possible when 2 of 3 encoded bits were successfully decoded. Thus, the reward for these cases is equal to 2, while for the other cases it is zero. The set containing possible aggregated next states, assuming the clique size is 2, is \( \bar{S}' = \{ \bar{s}_{3,1}, \bar{s}_{2,2}, \bar{s}_{0,3}, \bar{s}_{0,4} \} \), where the components refer to the events of successfully decoding of 0, 1, 2 and 3 bits correspondingly. In order to calculate \( r_1(\bar{s}_{3,1}, \bar{a}) \), we first summarize over all possible outcomes \( r_1(\bar{s}_{3,1}, \bar{a} = 1) = \sum_{s_i} r_1(s_i, \bar{a} = 1, \bar{s}_{3,1})p(s_i|\bar{a} = 1, \bar{s}_{3,1}) \). Substituting the expected values and the probabilities we found above, and arranging according to the aggregated states, we have:

\[
r_1(\bar{s}_{3,1}, \bar{a} = 1) = \sum_{s_i \in \bar{s}_{3,1}} Er_1(s_i, \bar{a} = 1, \bar{s}_{3,1})p(s_i|\bar{a} = 1, \bar{s}_{3,1})
+ \sum_{s_i \in \bar{s}_{2,2}} Er_1(s_i, \bar{a} = 1, \bar{s}_{3,1})p(s_i|\bar{a} = 1, \bar{s}_{3,1}) + \sum_{s_i \in \bar{s}_{0,3}} Er_1(s_i, \bar{a} = 1, \bar{s}_{3,1})p(s_i|\bar{a} = 1, \bar{s}_{3,1}) +
\sum_{s_i \in \bar{s}_{0,4}} Er_1(s_i, \bar{a} = 1, \bar{s}_{3,1})p(s_i|\bar{a} = 1, \bar{s}_{3,1}) = \sum_{s_i, s_j \in \bar{s}} Er_1(s_i, \bar{a} = 1, \bar{s}_{3,1})p(s_i|\bar{a} = 1, \bar{s}_{3,1})
\]

We now turn to the induced MDP \( \bar{M} \). We will assume that the states and the actions of \( \bar{M} \) are equivalent to those of \( M_1 \). Continuing with the example, assume that \( \bar{s} = \bar{s}_{3,1} \) and \( \bar{a} = 1 \). We find the reward associated with transition to \( \bar{s}_{0,3} \), \( R(\bar{s}_{0,3}, \bar{a} = 1, \bar{s}_{3,1}) \). In order to do so, equate component-wise \( R(\bar{s}, \bar{a}) \) and \( r_1(\bar{s}_{3,1}, \bar{a} = 1) \) as follows:

\[
r(\bar{s}_{0,3}, \bar{a} = 1, \bar{s}_{3,1})p(\bar{s}_{0,3}|\bar{s}_{3,1}, \bar{a} = 1) = \sum_{s_i \in \bar{s}_{0,3}} Er_1(s_i, \bar{a} = 1, \bar{s}_{3,1})p(s_i|\bar{a} = 1, \bar{s}_{3,1})
\]

It is left to calculate the probability \( \hat{p}(\bar{s}_{0,3}|\bar{s}_{3,1}, \bar{a} = 1) \). We equate this transition probability to \( p \) and use (3):

\[
p(\bar{s}_{0,3}|\bar{s}_{3,1}, \bar{a} = 1) = p(\bar{s}_{0,3}|\bar{s}_{3,1}, \bar{a} = 1) = \sum_{s' \in \bar{s}_{0,3}} \sum_{s \in \bar{s}_{3,1}} p(s'|ar{a} = 1, s)p(s|\bar{s}_{3,1})
\]
Finally, the solutions for all possible $R(s', \hat{a} = 1, \hat{s}_{3,1})$ are found from

$$r_{\hat{s}_{0,3}, \hat{a} = 1, \hat{s}_{3,1}} = \frac{\sum\limits_{s_i \in \hat{s}_{0,3}} E r_1(s_i, \hat{a} = 1, \hat{s}_{3,1}) p(s_i|1, \hat{s}_{3,1})}{\sum\limits_{s' \in \hat{s}_{0,3}} \sum\limits_{s \in \hat{s}_{3,1}} p(s'|\hat{a} = 1, s)p(s|\hat{s}_{3,1})}$$

and

$$r_{\hat{s}_{2,3}, \hat{a} = 1, \hat{s}_{3,1}} = \frac{\sum\limits_{s_i \in \hat{s}_{2,3}} E r_1(s_i, \hat{a} = 1, \hat{s}_{3,1}) p(s_i|1, \hat{s}_{3,1})}{\sum\limits_{s' \in \hat{s}_{2,3}} \sum\limits_{s \in \hat{s}_{3,1}} p(s'|\hat{a} = 1, s)p(s|\hat{s}_{3,1})}$$

which finishes the example.

### C. Proof of Bounds

We prove low and upper bounds on the slope of $V(s)$, discounted infinite horizon cost. Denoting $p^c_k$, the probability to increase $L(s)$ from $k$ to $k + 1$ when transmitting an empty line, see that $p^c_k < p$, that is, incrementing the clique is conditioned on the transmission being unsuccessful. Denote by $p^c_{k,i}$, $0 \leq i \leq k$, the transition probability from state $k$ from to state $i$, when acting by the transmission of the clique (i.e. $a = 1$). Note that $p^c_{k,0}$ is formally given by $p^c_{k,0} = p(s' = i|s = k, a = 1)$. Define operator $T$, corresponding to the Bellman equation, acting on $V$

$$TV(k) = \max\{[p^c_k \gamma V(k + 1) + (1 - p^c_k) \gamma V(k) + (1 - p)]\} \sum_{i=0}^{k} p^c_{k,i} \gamma V(i) + (1 - p)k\}, \quad (10)$$

with boundary conditions

$$TV(0) = \{ [p^c_0 \gamma V(1) + (1 - p^c_0) \gamma V(0) + (1 - p)]\}$$

$$TV(K) = \sum_{i=0}^{K} p^c_{K,i} \gamma V(i) + (1 - p)K.$$ 

The immediate rewards are explained as follows. The reward for transmission of an empty line is given by the probability of a successful transmission, that is $1 - p$. In the case a clique of size $k$ is transmitted, we have $k$ potential i.i.d rewards, which gives $(1 - p)k$. To simplify the notation, denote $\tilde{S}(k) = \gamma \sum_{i=0}^{k} p^c_{k,i} V(k - i) + (1 - p)k$ and $\tilde{E}(k) = p^c_k \gamma V(k + 1) + (1 - p^c_k) \gamma V(k) + (1 - p)k$.

Let $S$ be the set of functions from $\{0, 1, \ldots, K\}$ to $\mathbb{R}$ that are nondecreasing, and have slope bounded from above by $d_k$, that is

$$V(k + 1) - V(k) \leq d, \quad k \in \{0, 1, \ldots, K - 1\}, \quad (11)$$
and bounded from below as follows:

\[ V(k) - V(k-i) \geq i - c, \quad \text{where } i \in \{1, \ldots, K-1\}, \ k \in \{i, i+1, \ldots, K\}. \quad (12) \]

Lemma 1.1 below asserts that \( T \) preserves \( S \), and acts on it as a strict contraction. The combination of these two assertion implies that \( V(s) \) is in \( S \) (see the discussion below), that is, it possesses the corresponding properties (12) and (11).

**Lemma 1.1.** There exist constants \( c \) and \( d \), such that one has \( TS \subset S \). Moreover, there exists a constant \( \alpha \in (0, 1) \) such that

\[ \|TU - TW\| \leq \alpha\|U - W\| \text{ for every } U, W \in S. \]

**Discussion.** The main difficulty of the proof below stems from the ambiguity regarding the transition probabilities. That is, the precise calculation of these probabilities is computationally infeasible, especially for large number of users, \( K \). We solved this by reinforcement learning on the practical side. On the analytical side, we make several assumptions and estimations, which we justify throughout the proof. To this end, the proof is primarily built on the assumption that \( V \in S \) and possesses all the corresponding properties. We exploit this assumption in order to prove that operator \( T \), acting on \( S \), preserves these properties, that is \( TV \in S \). Now note that the map defined by operator \( T \) in (10), acting on a complete metric space \( S \), with \( T : \mathbb{R}^{|S|} \to \mathbb{R}^{|S|} \) of value functions, is a strict contraction, \[34\] Theorem V.18. Therefore, \( T \) has a unique fixed point which solves \( TU = U \). On the other hand, \( V \) is the unique solution to the same (Bellman) equation in the space of all functions. As a result, \( V = U \). Whence, in case we start the converging procedure with initial function which preserves (11) and (12), by iteratively activating the operator \( T \), we end up with solution which preserves the aforementioned property.

1) **Proof of Lemma 1.1.** Denote by \( p_{k,i,j}^c \) the probability \( p_{k,i}^c \), conditioned that the largest fully disjoint clique with the clique of size \( k \), prior the transmission, was of size \( j \). Note that \( j \leq k \). Denote the probability of having such a disjoint clique as \( p_{k,j}^c \) (by total probability \( p_i^k = \sum_j p_{i,j}^k p_{k,j} \cdot )

By Equation (17) and Lemma 1.2 (see the end of this section) it holds either \( p_{k,i}^c = p_{k,i,0}^c + a_1 = p_i(1-p)^{k-i}(\binom{k}{i}) + a_1 \), for some nonnegative \( a_1 \), or \( p_{k,i}^c = 0 \). (Note, that \( a_1 = 0 \) in the case there
were no other cliques of size $k - i$ prior to the encoded transmission.)

See that by multiple application of (11) and (12)

$$\bar{S}_k \leq \gamma V_0 + \gamma \sum_{i=0}^{k} i * p_{k,i}^c d + (1 - p)k$$

$$= \gamma V_0 + \gamma \sum_{i=0}^{k} i * p_{k,i,0}^c d + (1 - p)k + a_2(k) = \gamma V_0 + \gamma p k d + (1 - p)k + a_2(k)$$  \hspace{1cm} (13)

and

$$\bar{S}_k \geq \gamma V_k - \gamma \sum_{i=0}^{k} (k - i) * p_{k,i}^c d + (1 - p)k$$

$$= \gamma V_k - \gamma \sum_{i=0}^{k} (k - i) * p_{k,i,0}^c d + (1 - p)k - b_2(k) = \gamma V_k - \gamma (1 - p)kd + (1 - p)k - b_2(k)$$  \hspace{1cm} (14)

where $a_2(k)$ and $b_2(k)$ stand for summations of all compensation constants $a_1(k,i)$, in both cases above.

We use the contraction property in the remaining part of the proof. Since, by assumption, $V$ satisfies (11) and (12), we only have to show that

$$\max\{\bar{S}(k+1), \bar{E}(k+1)\} - \max\{\bar{S}(k), \bar{E}(k)\} \leq d$$ \hspace{1cm} (15)

$$\max\{\bar{S}(k-i), \bar{E}(k-i)\} - \max\{\bar{S}(k), \bar{E}(k)\} \leq -i + c$$ \hspace{1cm} (16)

We analyze all the possible options within the curly brackets, as follows.

1.

$$TV(k+1) - TV(k) = \bar{S}(k+1) - \bar{S}(k)$$

$$TV(k-i) - TV(k) = \bar{S}(k-i) - \bar{S}(k)$$

Applying Lemma 1.3 it immediately follows that $TV(k+1) - TV(k) < d$ and $TV(k-i) - TV(k) \geq -i + c$ in this case.

2.

$$TV(k+1) - TV(k) = \bar{E}(k+1) - \bar{E}(k)$$

$$TV(k-i) - TV(k) = \bar{E}(k-i) - \bar{E}(k)$$

In order to prove the second case we should comply with the expressions for $d$ and $c$ found in the first case. Note that $p_{k+1}^c < p_k^c$. That is, the probability to increase the size of the maximal clique then acting by sending an empty
line decreases with the state size. Hence,

\[ \hat{E}(k+1) - \hat{E}(k) = p_{k+1}^c \gamma V(k+2) + (1 - p_{k+1}^c) \gamma V(k+1) - p_k^c \gamma V(k+1) - (1-p_k^c) \gamma V(k) \]

\[ = p_{k+1}^c \gamma V(k+2) + (1 - p_{k+1}^c - p_k^c) \gamma V(k+1) - (1 - p_k^c) \gamma V(k) \]

\[ \leq \gamma dp_{k+1}^c + (1 - p_k^c) \gamma V(k+1) - (1 - p_k^c) \gamma V(k) \leq \gamma dp_{k+1}^c + d(1 - p_k^c) \gamma < d \gamma < d \]

and

\[ \hat{E}(k-i) - \hat{E}(k) = p_{k-i}^c \gamma V(k-i+1) + (1 - p_{k-i}^c) \gamma V(k-i) - p_k^c \gamma V(k+1) - (1 - p_k^c) \gamma V(k) \]

\[ \leq [p_{k-i}^c \gamma V(k-i+1) - p_{k-i}^c \gamma V(k-i)] + \gamma V(k-i) + [(1 - p_k^c) \gamma V(k+1) - (1 - p_k^c) \gamma V(k)] - \gamma V(k+1) \]

\[ \leq \gamma dp_{k-i}^c + \gamma V(k-i) - (1 - p_k^c)d \gamma - \gamma V(k+1) \leq \gamma dp_{k-i}^c + (1 - p_k^c)d \gamma - \gamma i - \gamma + \gamma c < -i + c \]

See that for \( \gamma \) close enough to 1 the last assertion is true.

3.

\[ TV(k+1) - TV(k) = \tilde{S}(k+1) - \hat{E}(k) \]

\[ TV(k-i) - TV(k) = \tilde{S}(k-i) - \hat{E}(k) \]

Using the proof of case 1:

\[ \tilde{S}(k+1) - \hat{E}(k) \leq \tilde{S}(k+1) - \tilde{S}(k) \leq d \]

\[ \tilde{S}(k-i) - \hat{E}(k) \leq \tilde{S}(k-i) - \tilde{S}(k) \leq -i + c \]

4.

\[ TV(k+1) - TV(k) = \hat{E}(k+1) - \tilde{S}(k) \]

\[ TV(k-i) - TV(k) = \hat{E}(k-i) - \tilde{S}(k) \]

Using the proof of case 2:

\[ \hat{E}(k+1) - \tilde{S}(k) \leq \hat{E}(k+1) - \hat{E}(k) \leq d \]

\[ \hat{E}(k-i) - \tilde{S}(k) \leq \hat{E}(k-i) - \hat{E}(k) \leq -i + c \]
There are additional combinations, such as $\tilde{E}(k + 1) - \tilde{S}(k)$ and $\tilde{S}(k - i) - \tilde{S}(k)$, however their proof is straightforward using same considerations as above. It is trivially seen that all the cases hold for the boundary conditions as well.

To see that $V(k)$ is non-decreasing in $k$ we use the following argumentation. Denote the aggregated state of having a maximal clique of size $k$ as $s_k$, $s_k \in \tilde{S}$. Define function $g_k : s_k \rightarrow s_{k-1}$, $k > 1$, such that for each $s_k$, $g_k$ acts by deleting a random line from the maximal clique of size $k$, i.e. updating all entries of the chosen line to 0. We aim to compare $V(s(k)) = V(k)$ and $V(g_k(s(k)))$. By simple coupling argumentation one defines two processes and sees that $V(s(k)) \geq V(g_k(s(k)))$. We skip the trivial details. Finally the contraction property of operator $T$ follows from the well known results on MDP. See \[35\], for example. This accomplishes the proof of the lemma.

**Lemma 1.2.** For $j > 2$, that is disjoint clique exists,

$$p_{k,i,j}^c = 0, \quad j > i$$

$$p_{k,i,0}^c \leq p_{i,j}, \quad i \leq j$$

**Proof.** Trivially, in case the disjoint clique is larger than $j$, the probability to have clique smaller than $j$ is zero. Therefore, the first assertion trivially holds, $p_{k,i,j}^c = 0 \quad j > i$.

Next, see that for all $i$,

$$p_{k,i,0}^c = p^i(1 - p)^{k-i}\binom{k}{i}.$$ (17)

The sum of all transition probabilities from state $k$ acting $\bar{a} = 1$, for all $j$ is 1:

$$\sum_{i=0}^{k} p_{k,i,j}^c = 1$$

Hence, the second assertion holds.

**Lemma 1.3.** One has constants $d$ and $c$ such that

$$\tilde{S}(k + 1) - \tilde{S}(k) < d$$

$$\tilde{S}(k - i) - \tilde{S}(k) > -i + c$$

For all $k$ and $i < k$.

**Proof.** We prove by finding such constants. Substitute (11) and (12), using inequalities (13) and (14), and perform
algebraic simplifications. Write

\[ \dot{S}(k) - \dot{S}(k-1) = \gamma \sum_{i=0}^{k} p_{k,i}^c V_i + (1 - p)(k) - \gamma \sum_{i=0}^{k-1} p_{k-1,i}^c V_i + (1 - p)(k-1) \]

\[ \leq \gamma V_0 + \gamma kdp + (1 - p)(k) + a_2(k) - \gamma V_{k-1} + (1 - \gamma d)(1 - p)(k-1) + b_2(k-1) \]

\[ \leq d_k \gamma k + d\gamma p - d\gamma - p + 1 + a_2(k) + b_2(k-1) + (1 - k) \gamma + c_\gamma \leq d \]

and

\[ \dot{S}(k-i) - \dot{S}(k) \leq \gamma V_0 + \gamma p d(k-i) + (1 - p)(k-i) + a_2(k-i) - \gamma V_k + (1 - \gamma d)(1 - p)k + b_2(k) \]

\[ \leq -d\gamma ip + d + k\gamma k + pi + \gamma c - \gamma i - k + a_2(k-i) + b_2(k) \leq -i + c \]

Next, for simplicity, assume equalities for both inequalities above and write

\[
\begin{aligned}
d\gamma k + d\gamma p - d\gamma - p + 1 + (1 - k) \gamma + c_\gamma &= d \\
-d\gamma ip + d + k\gamma k + pi + \gamma c - \gamma i - k + a_2(k-i) - b_2(k) &= -i + c
\end{aligned}
\]

Solving for \(d\) and \(c\) we have the following expressions

\[ c = A(\gamma k + \gamma p - \gamma - 1)b_2(k) - A(\gamma k + \gamma p - \gamma - 1)a_2(k-i) \]

\[ + A(p(\gamma^2 ik - \gamma^2 i + \gamma^2 k - \gamma ik - \gamma k + i)) \]

\[ d = A\gamma a_2(k-i) - A\gamma b_2(k) + A(\gamma ip - \gamma^2 - \gamma k + \gamma p - p + 1) \]  \( (18) \)

Where \(1/A = \gamma^2 ip + \gamma^2 p - \gamma^2 - \gamma k - \gamma p + 1\). Observe that \(1/A \approx ip - k\) as \(\gamma \to 1\). The rightmost part of \(d\) in \(19\) is essentially independent of \(i\) and \(k\), and is less than 1 for all \(k, i\). Consequently, the assumption \(d\) is independent of \(k\) is plausible. One the other hand, \(c\) has very low positive values, comparatively to that of \(i\). Hence, the constants \(d\) and \(c\) above satisfy the lemma.

\[ \blacksquare \]

**REFERENCES**

[1] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, “Network information flow,” *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[2] S. Sorour and S. Valaee, “On densifying coding opportunities in instantly decodable network coding graphs,” in *IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2012, pp. 2456–2460.

[3] R. Dougherty and K. Zeger, “Nonreversibility and equivalent constructions of multiple-unicast networks,” *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 5067–5077, 2006.
[4] C.-C. Wang, “On the capacity of 1-to-broadcast packet erasure channels with channel output feedback,” IEEE Transactions on Information Theory, vol. 58, no. 2, pp. 931–956, 2012.

[5] S. Sorour and S. Valaee, “An adaptive network coded retransmission scheme for single-hop wireless multicast broadcast services,” IEEE/ACM Transactions on Networking (TON), vol. 19, no. 3, pp. 869–878, 2011.

[6] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, “Wireless broadcast using network coding,” IEEE Transactions on Vehicular Technology, vol. 58, no. 2, pp. 914–925, 2009.

[7] L. Keller, E. Drinea, and C. Fragouli, “Online broadcasting with network coding,” in Workshop on Network Coding, Theory and Applications. (NetCod.), Jan 2008, pp. 1–6.

[8] X. Xiao, Y. Lu-Ming, W. Wei-Ping, and Z. Shuai, “A wireless broadcasting retransmission approach based on network coding,” in IEEE International Conference on Circuits and Systems for Communications. (ICCSC.), 2008, pp. 782–786.

[9] R. A. Costa, D. Munaretto, J. Widmer, and J. Barros, “Informed network coding for minimum decoding delay,” in IEEE International Conference on Mobile Ad Hoc and Sensor Systems. (MASS.), 2008, pp. 80–91.

[10] A. Eryilmaz, A. Ozdaglar, and M. Medard, “On delay performance gains from network coding,” in Annual Conference on Information Sciences and Systems, 2006, pp. 864–870.

[11] Y. Lin, B. Liang, and B. Li, “Slideor: Online opportunistic network coding in wireless mesh networks,” in IEEE Conference on Computer Communications (INFOCOM), 2010, pp. 1–5.

[12] P. Sadeghi, R. Shams, and D. Traskov, “An optimal adaptive network coding scheme for minimizing decoding delay in broadcast erasure channels,” EURASIP Journal on Advances in Signal Processing, pp. 4:1–4:14, 2010.

[13] S. Rayanchu, S. Sen, J. Wu, S. Banerjee, and S. Sengupta, “Loss-aware network coding for unicast wireless sessions: design, implementation, and performance evaluation,” in ACM SIGMETRICS Performance Evaluation Review, vol. 36, no. 1, 2008, pp. 85–96.

[14] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Méard, and J. Crowcroft, “Xors in the air: practical wireless network coding,” in ACM SIGCOMM Computer Communication Review, vol. 36, no. 4, 2006, pp. 243–254.

[15] D. Nguyen, T. Nguyen, and X. Yang, “Multimedia wireless transmission with network coding,” in IEEE Packet Video Workshop, 2007, pp. 326–335.

[16] D. Nguyen and T. Nguyen, “Network coding-based wireless media transmission using POMDP,” in IEEE Packet Video Workshop., 2009, pp. 1–9.

[17] L. Georgiadis and L. Tassiulas, “Broadcast erasure channel with feedback-capacity and algorithms,” in Workshop on Network Coding, Theory, and Applications, (NetCod.). IEEE, 2009, pp. 54–61.

[18] A. Cohen, E. Biton, J. Kampeas, and O. Gurewitz, “Coded unicast downstream traffic in a wireless network: analysis and wifi implementation,” EURASIP Journal on Advances in Signal Processing, no. 1, pp. 1–20, 2013.

[19] Y. Birk and T. Kol, “Coding on demand by an informed source (iscod) for efficient broadcast of different supplemental data to caching clients,” IEEE/ACM Transactions on Networking (TON), vol. 14, no. SI, pp. 2825–2830, 2006.

[20] S. El Rouayheb, A. Sprintson, and C. Georgiades, “On the index coding problem and its relation to network coding and matroid theory,” IEEE Transactions on Information Theory, vol. 56, no. 7, pp. 3187–3195, 2010.

[21] M. Dai, K. Shum, and C. W. Sung, “Data dissemination with side information and feedback,” IEEE Transactions on Wireless Communications, vol. 13, no. 9, pp. 4708–4720, 2014.

[22] D. E. Lucani, F. H. Fitzek, M. Méard, and M. Stojanovic, “Network coding for data dissemination: it is not what you
know, but what your neighbors don’t know,” in International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks. (WiOPT), 2009, pp. 1–8.

[23] M. Ghaderi, D. Towsley, and J. Kurose, “Reliability gain of network coding in lossy wireless networks,” in IEEE Conference on Computer Communications. (INFOCOM), 2008.

[24] J. K. Sundararajan, D. Shah, and M. Médard, “Arq for network coding,” in IEEE International Symposium on Information Theory. (ISIT), 2008, pp. 1651–1655.

[25] J. K. Sundararajan, P. Sadeghi, and M. Médard, “A feedback-based adaptive broadcast coding scheme for reducing in-order delivery delay,” in Workshop on Network Coding, Theory, and Applications. (NetCod), 2009, pp. 1–6.

[26] L. Li, T. J. Walsh, and M. L. Littman, “Towards a unified theory of state abstraction for MDPs,” in International Symposium on Artificial Intelligence and Mathematics, 2006.

[27] N. K. Jong and P. Stone, “State abstraction discovery from irrelevant state variables,” in International Joint Conference on Artificial Intelligence, (IJCAI), 2005, pp. 752–757.

[28] D. P. Bertsekas, Dynamic programming and optimal control. Athena Scientific Belmont, MA, 1995.

[29] A. L. Strehl and M. L. Littman, “An analysis of model-based interval estimation for markov decision processes,” Journal of Computer and System Sciences, vol. 74, no. 8, pp. 1309–1331, 2008.

[30] M. Kearns and S. Singh, “Near-optimal reinforcement learning in polynomial time,” Machine Learning, vol. 49, no. 2-3, pp. 209–232, 2002.

[31] R. I. Brafman and M. Tennenholtz, “R-max-a general polynomial time algorithm for near-optimal reinforcement learning,” The Journal of Machine Learning Research, vol. 3, pp. 213–231, 2003.

[32] R. S. Sutton and A. G. Barto, Introduction to reinforcement learning. MIT Press, 1998.

[33] S. Mahadevan, “Average reward reinforcement learning: Foundations, algorithms, and empirical results,” Machine learning, vol. 22, no. 1-3, pp. 159–195, 1996.

[34] M. C. Reed and B. Simon, Methods of Modern Mathematical Physics: Functional analysis. 1. Access Online via Elsevier, 1980, vol. 1.

[35] M. L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 1994.