On queueing in coded networks - queue size follows degrees of freedom

Conference Paper · August 2007

DOI: 10.1109/ITWITWN.2007.4318066 · Source: IEEE Xplore

3 authors, including:

Muriel Médard
Massachusetts Institute of Technology
617 PUBLICATIONS **22,620** CITATIONS

On queueing in coded networks – queue size follows degrees of freedom

Jay Kumar Sundararajan, Devavrat Shah, Muriel Médard

Laboratory for Information and Decision Systems,
Massachusetts Institute of Technology, Cambridge, MA 02139.
Email: {jaykumar, devavrat, medard}@mit.edu

***Abstract*— We propose a new queueing mechanism for coded networks with stochastic arrivals and/or lossy links. In this context, earlier work introduced the notion of "virtual queues" which represent the backlog in degrees of freedom. For instance, the work by Ho and Viswanathan defined the achievable rate region for which the virtual queue size is stabilized, using intra-session coding. The queueing scheme that we propose here forms a natural bridge between the virtual queue size and the physical queue size, and thus extends their result to the stability of the physical queues as well. Specifically, we show that the amount of memory used at the transmit buffer in our scheme is upper bounded by the total backlog in the number of linearly independent degrees of freedom. Moreover, our scheme gives an online algorithm for queue update and coding, in the sense that the coding does not happen block by block, but in a streaming manner. The main idea in our scheme is to ensure that the information stored at the sender excludes any knowledge that is common to all receivers. This requires the transmitting node to track the states of knowledge of its receivers. Therefore, if the links are lossy, some form of feedback may be necessary.**

I. Introduction

Network coding was originally introduced in the context of constant rate data sources with periodic arrivals, trying to communicate over a network of error-free links with specified capacities [1], [2]. Later, the models were enhanced to include stochastic arrival processes [3], [4] and losses on links [5]. In these new models, the nodes need to maintain queues, either to account for burstiness or for the losses, and the question of how to manage these queues becomes important.

The schemes proposed in the new models essentially involve transmitting a linear combination of all packets received up to that point of time, with the following property. It must be simultaneously innovative[1] at all receivers except those that already have all the information that the transmitter has. This can be accomplished with high probability, if the transmitter simply stores all received packets and generates random linear combinations of them, provided the field size is large enough.

However, storing all received packets may be undesirable. One way to address this issue may be to use a frame based approach, where at the end of a predetermined coding window, the algorithm takes a break to clear out the existing backlog, after which the buffer is completely empty. This will have a small loss in throughput, since the algorithm does not serve new arrivals while clearing out the backlog. An alternative is to store only those packets that have not yet been decoded by all receivers. However, note that a receiver is guaranteed to decode the original packets only when the number of received innovative packets is equal to the size of the generation used for coding (*i.e.*, when the number of equations equals the number of unknowns). Thus, it is possible that the receivers are unable to decode packets for a long time, even though they may be receiving degrees of freedom at a sufficient rate. This means, the queue may be very long even if only a few degrees of freedom are missing at the receivers. Ideally, one would like a truly online algorithm in which the size of the queue tracks the backlog in the degrees of freedom.

In this paper, we propose a new online coding and queue update algorithm that achieves this goal. In our scheme, we assume that the transmitter gets to know what the receivers have received thus far. The main idea is that the transmitters will now store linear combinations of the original packets. The linear combinations to be stored are chosen such that their span excludes any linear combination that is already known at all the receivers. We prove that this scheme indeed ensures that the amount of information stored tracks the backlog in degrees of freedom. Moreover, our coding scheme is online in the sense that there is no predetermined frame or block with periodic flushing of the buffers at the end of the frame. Instead coding is done in a streaming manner.

For networks with random arrivals, earlier work by Ho *et al.* [3] defined the achievable rate region for backpressure algorithms using intra-flow network coding. Their work considered the stability of "virtual queues", which correspond to the backlog in degrees of freedom. This work was extended to inter-session coding by Eryilmaz and Lun [4]. Our scheme thus forms a natural bridge between the virtual queue size and the physical queue size and thus can be used to extend these results to the stability of the physical queues as well.

In addition, the bridge between the virtual queue size and the physical queue size means that various results from traditional queueing theory such as the transform based analysis for the queue size of M/G/1 queues, or even a Jackson network type of result can be extended to study the actual memory

This work is supported by the DARPA ITMANET grant and by the NSF grant CNS-0627021 (NeTS:NBD: XORs in the Air: Practical Wireless Network Coding).

[1]A linear combination is said to be *innovative* at a receiver if its coefficient vector is linearly independent of the coefficient vectors of all previously received coded packets.

used by nodes in networks with network coding, as opposed to just the backlog in degrees of freedom.

We present our algorithm in the specific context of a packet erasure broadcast channel with feedback. However, the idea is applicable in more general networks. Note that, in networks with perfect links and random arrival processes, there is no need for feedback, since the transmitter can track the states of knowledge of the receivers simply using the schedules from the past. An example of such a network is the multicast switch with intra-flow network coding which was studied in [6].

The rest of the paper is organized as follows. Section II describes the packet erasure broadcast setting. Section III contains an intuitive description of the proposed queueing algorithm followed by the formal specification. The main result that the actual queue size is upper bounded by the total backlog in degrees of freedom at the transmitter, is proved in Section IV. Finally, Section V gives the conclusions.

II. The Setting

Consider the following communication setup. A stream of packets arrives according to a stochastic arrival process at a sender S that wants to broadcast this stream to a set of receivers $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$, over a packet erasure broadcast channel.

Time is assumed to be slotted. In each slot t, the channel can accept as input, a packet of fixed size. The packet is then conveyed with no errors to an arbitrary subset $D(t) \subseteq \mathcal{R}$ of the receivers. Those receivers that do not receive the packet, can detect the erasure.

All arrivals occur at the mid-point of a time slot, while all transmissions occur at the beginning of a slot. We assume that, before the beginning of a slot, the sender gets to know which receivers received the transmission without erasure in the previous slot.

It is well known that a scheme that employs network coding at the sender will achieve the capacity in the above setup. The backlog in terms of the number of degrees of freedom between the sender and any of the receivers can be proved to be stable, provided the arrival rate is below the capacity.

In this work, we focus on the queueing strategy used by the sender in such a scheme assuming causal channel state feedback. We present an online coding and queue update algorithm that will allow us to translate stability in the backlog of degrees of freedom to stability in the physical queue size at the sender.

Although the rest of this work focuses on the broadcast situation outlined above, the online algorithm is quite general and can be applied in several other situations.

III. The Queue Update Algorithm

The sender maintains one queue. The key idea that we introduce is that the contents of the queue may themselves be linear combinations of the packets from the original stream. In every time slot, it computes one linear combination of the packets in the queue and transmits it over the channel. The following gives the details of the proposed algorithm.

A. An intuitive description of the algorithm

In this work, we only use linear codes. Every transmitted packet is a linear combination of some set of packets from the incoming stream. Now, if a node has received a collection of such linear combinations with coefficient vectors, say, $\mathbf{v_1}, \mathbf{v_2}, \ldots \mathbf{v_k}$, then it can compute other linear combinations whose coefficient vectors are in the span of $\{\mathbf{v_1}, \mathbf{v_2}, \ldots \mathbf{v_k}\}$. As a result, the state of knowledge of any node can be represented by a vector space – a node is said to "know" a vector space V if it can compute any linear combination of the original packets, whose coefficient vector is in V.

In order to guarantee the highest possible throughput, we require the following property to hold in every time slot – the sender should transmit a linear combination that is innovative to every receiver that receives it, except those which already know everything that the sender knows.

It is clear that in order to guarantee this property, the packets that have been decoded by all receivers need not be retained at the sender. But, our proposal is more general than that. The key intuition is that we can satisfy the above property even if we restrict the sender's transmission to be chosen from a subspace that is independent[2] of the subspace representing common knowledge available at all the receivers.

In other words, **the sender need not use for coding (and hence need not store) any information that has already been received by all the receivers**. We do not restrict the queue to store only uncoded packets, but instead allow it to store linear combinations of the packets. Therefore, at any point in time, the queue simply stores a basis for the coset space corresponding to the space of common knowledge of all the receivers.

Now, in order to know what needs to be stored and what can be safely dropped, the sender must keep track of the state of knowledge of every receiver using the causal channel state information. A receiver's state of knowledge is essentially the span of the coefficient vectors of the innovative packets it has received so far. In an online situation, tracking the receivers' states of knowledge could potentially get complicated by the fact that the dimension of the vector space representing the state of knowledge of the receivers keeps growing in time as more and more innovative packets are received. However, this issue can be avoided by noting that once the commonly known information is dropped from the queue, the states of knowledge can also be updated to exclude the common part. In other words, we are interested in tracking only that part of the state of knowledge that is not yet known at all receivers. To summarize, after every time slot, the algorithm recomputes the common knowledge and removes it from the states of knowledge, and also removes the corresponding information from the queues. We show that the size of such an incremental version of the state of knowledge is stable, provided the backlog in degrees of freedom is stable. This means that the queue size will also be stable.

[2]A subspace S_1 is said to be *independent* of another subspace S_2 if $S_1 \cap S_2 = \{\mathbf{0}\}$.

Another problem is that the coefficient vectors might have a very large number of components, since a coded packet could, in general, be a linear combination of all original packets received thus far. We can circumvent this problem by representing coded packets by their coefficient vectors with respect to the current queue contents, and not with respect to the original packets. And as claimed above, we can show that the queue size is stable. So, the size of the coefficient vectors is also stabilized. We call the new representation *local coefficient vectors* as opposed to the *global coefficient vectors* which are in terms of the original packets. Since linear independence of the local coefficient vectors carries over to the original global vectors, the operations performed by the algorithm on local vectors can be mapped to the corresponding operations on the global vectors. Also, it suffices to ensure that the transmitted packet is innovative in terms of the local vectors.

These ideas are explained in a concrete manner below. After the following remark, we first present the algorithm that runs at the sender, and then present the theorems that prove that with this algorithm, the actual queue size at the sender is upper bounded by the total backlog in degrees of freedom at the sender for all the receivers.

Remark 1 (A note on the overhead): Note that such a change of representation from global to local coefficient vectors results in the sender losing track of the mapping between the contents of its queue and the original uncoded packets. However, for the receivers to be able to decode, they will need the global coefficient vectors. Therefore, we assume that the global vectors are stored in the packet headers as usual and conveyed to the receivers. Now, the length of these global vectors is simply the current size of the coding window. Since the coding window size is not predetermined, the sender will have to include it in the header. At any point, the coding window includes all arrivals since the most recent emptying of the sender's virtual queue. This means that the size of the coding window (and hence the overhead for sending the coefficient vectors) is closely related to the busy period of the virtual queue. If we can show a bound on the busy period of the virtual queues using their stability, then this would give us a bound on the coding window size and hence on the overhead needed for storing the global vectors.

B. Algorithm at sender

All operations in the algorithm occur over a finite field of size $q > n$.

1) Initialize sets $B, B_1, \ldots, B_n$ to the empty set. These sets will hold the bases of the vector spaces that represent the states of knowledge of the sender and the receivers in that order. The representation is in the form of local coefficient vectors (*i.e.*, with respect to the current contents of the queue) and not global ones (*i.e.*, with respect to the original packets).
2) Initialize the vector $\mathbf{g}$ to the zero vector. This will hold the coefficients of the transmitted packet in each slot. (Again, the coefficients are in terms of the current contents of the queue, not the original packets.)

   At the beginning of every time slot, do:
3) *Incorporate new arrivals*

   Let a be the number of new arrivals that occurred in the middle of the previous slot. Add these new packets to the end of the queue. Let $b := |B|$. Replace B by $\{\mathbf{e_1}, \mathbf{e_2}, \ldots, \mathbf{e_{a+b}}\}$. ($\mathbf{e_j}$ is the unit vector with a 1 in the j^{th} position.) To make the number of components of all vectors consistent (*i.e.*, equal to $a + b$), append a zeros to every vector in each B_j, and also to $\mathbf{g}$.
4) *Incorporate channel state feedback:*

   If this is the first time slot, then skip this step. Otherwise, for every receiver $j = 1$ to n, do:

   > If there was a successful transmission to receiver j in the previous slot, append $\mathbf{g}$ to B_j.

5) *Separate out the knowledge that is common to all receivers:*

   Compute the following:

   B_Δ := Any basis of $\cap_{j=1}^{n} span(B_j)$.

   B' := Completion of B_Δ into a basis of $span(B)$.

   B'' := $B' \backslash B_\Delta$.

   B'_j := Completion of B_Δ into a basis of $span(B_j)$ in such a way that, if we define $B''_j := B'_j \backslash B_\Delta$, then the following holds: $B''_j \subseteq span(B'')$. Lemma 1 proves that this is possible.
6) *Update the queue contents:*

   Let $\mathbf{y_1}, \mathbf{y_2}, \ldots \mathbf{y_K}$ represent the current contents of the queue. The queue size K is equal to $(a + b)$. Replace the contents of the queue with packets $\mathbf{y'_1}, \mathbf{y'_2}, \ldots \mathbf{y'_{K'}}$ of the form $\sum_{i=1}^{K} h_i \mathbf{y_i}$ for each $\mathbf{h} \in B''$. The new queue size is thus $K' = |B''|$.
7) *Recompute local coefficient vectors with respect to the new queue contents:*

   For every vector in B''_j find the corresponding coordinate vector in the new basis B'', and call the resulting set of vectors the new B_j. Update the value of B to $\{\mathbf{e_1}, \mathbf{e_2}, \ldots, \mathbf{e_{K'}}\}$.
8) *Transmission:*

   If B is not empty, update $\mathbf{g}$ to be any vector that is in $span(B)$ but not in $\cup_{\{j:B_j \subsetneq B\}} span(B_j)$.

   Compute the packet $\sum_{i=1}^{K'} g_i \mathbf{y'_i}$ and transmit it on the packet erasure broadcast channel. Lemma 2 shows that such a $\mathbf{g}$ exists provided the field size exceeds n. If B is empty, set $\mathbf{g}$ to 0 and transmit nothing.
9) Go back to step 3.

Lemma 1: In step 5 of the algorithm above, it is possible to complete B_Δ into a basis B'_j of each $span(B_j)$ such that B''_j is within the span of B''.

Proof: We show that any completion of B_Δ into a basis of $span(B_j)$ can be changed to a basis with the required property.

Let $B_\Delta = \{\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_m}\}$. Suppose we complete this into a basis C_j of $span(B_j)$ such that:

$$C_j = B_\Delta \cup \{\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c}_{|\mathbf{B_j}|-\mathbf{m}}\}$$

Now, we claim that at the beginning of step 5, $span(B_j) \subseteq span(B)$ for all j. This can be proved by induction on the slot number, using the way the algorithm updates B and the B_j's. Intuitively, it says that any receiver knows a subset of what the sender knows.

Therefore, for each vector $\mathbf{c} \in C_j \backslash B_\Delta$, $\mathbf{c}$ must also be in $span(B)$. Now, since $B_\Delta \cup B''$ is a basis of $span(B)$, we can write $\mathbf{c}$ as $\sum_{i=1}^{m} \alpha_i \mathbf{b_i} + \mathbf{c'}$ with $\mathbf{c'} \in span(B'')$. In this manner, each $\mathbf{c_i}$ gives a distinct $\mathbf{c'_i}$. It is easily seen that $C'_j := B_\Delta \cup \{\mathbf{c'_1}, \mathbf{c'_2}, \ldots, \mathbf{c'}_{|\mathbf{B_j}|-\mathbf{m}}\}$ is also a basis of the same space that is spanned by C_j. Moreover, it satisfies the property that $C'_j \backslash B_\Delta \subseteq span(B'')$. ■

Lemma 2: ([6]) *Let $\mathcal{V}$ be a vector space with dimension k over a field of size q, and let $\mathcal{V}_1, \mathcal{V}_2, \ldots \mathcal{V}_n$, be subspaces of $\mathcal{V}$, of dimensions $k_1, k_2, \ldots, k_n$ respectively. Suppose that $k > k_i$ for all $i = 1, 2, \ldots, n$. Then, there exists a vector that is in $\mathcal{V}$ but is not in any of the $\mathcal{V}_i$'s, if $q > n$.*

Proof: See [6] for the proof. ■

IV. Connecting the physical and virtual queue sizes

Let $H(t)$ be a matrix whose rows are the *global* coefficient vectors of the queue contents after incorporating the slot t arrivals, *i.e.*, at the end of step 3 in time slot $(t+1)$. Let $a(t)$ denote the number of arrivals in slot t, and let $A(t)$ be the total number of arrivals up to and including slot t. Thus, $A(t) = \sum_{t'=0}^{t} a(t')$. Note that each row of $H(t)$ is in $\mathbb{F}_q^{A(t)}$.

Let $B_\Delta(t)$ (resp. $B''(t)$, $B'_j(t)$ and $B''_j(t)$) denote the matrix whose rows are the vectors in B_Δ (resp. B'', B'_j and B''_j) at the end of step 5 in time slot t. Let $B_j(t)$ be a matrix containing as rows, the vectors in B_j at the end of step 3 in slot $(t+1)$. Also, let $\mathbf{g}(t)$ denote the vector $\mathbf{g}$ at the end of step 3 in time slot $(t+1)$, *i.e.*, the local coefficient vector of the packet transmitted in slot t, with $a(t)$ zeros appended.

Lemma 3: The rows of $H(t)$ are linearly independent for all t.

Proof: The proof is by induction on t.

Basis step: In the middle of time slot 0, the first batch of $a(0)$ arrivals occur. So, $H(0) = I_{a(0)}$ and hence the rows are independent.

Induction hypothesis: Assume $H(t-1)$ has linearly independent rows.

Induction step: The queue is updated such that the linear combinations corresponding to local coefficient vectors in B'' are stored, and subsequently, the $a(t)$ new arrivals are appended. Thus, the relation between $H(t-1)$ and $H(t)$ is:

$$H(t) = \begin{bmatrix} B''(t)H(t-1) & 0 \\ 0 & I_{a(t)} \end{bmatrix}$$

Now, $B''(t)$ has linearly independent rows, since the rows form a basis. The rows of $H(t-1)$ are also linearly independent by hypothesis. Hence, the rows of $B''(t)H(t-1)$ will also be linearly independent. Appending $a(t)$ zeros and then adding an identity matrix block in the right bottom corner does not affect the linear independence. Hence, $H(t)$ also has linearly independent rows. ■

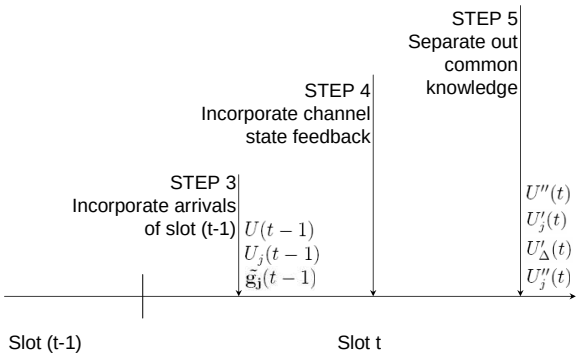

Fig. 1. The main steps of the algorithm, along with the times at which the various $U(t)$'s are defined

Define the following:

$$\begin{array}{lcl} U''(t) & := & \text{Row span of } B''(t)H(t-1) \\ U'_j(t) & := & \text{Row span of } B'_j(t)H(t-1) \\ U'_\Delta(t) & := & \cap_{j=1}^{n} U'_j(t) \\ U''_j(t) & := & \text{Row span of } B''_j(t)H(t-1) \\ U(t) & := & \text{Row span of } H(t) \\ U_j(t) & := & \text{Row span of } B_j(t)H(t) \end{array}$$

Note that $U''(t)$, the $U'_j(t)$'s , $U'_\Delta(t)$ and the $U''_j(t)$'s are all subspaces of $\mathbb{F}_q^{A(t-1)}$, whereas $U(t)$ and the $U_j(t)$'s are all subspaces of $\mathbb{F}_q^{A(t)}$. Figure 1 shows the points at which these $U(t)$'s are defined in the slot.

The fact that $H(t)$ has full row rank (proved above in Lemma 3) implies that the operations performed by the algorithm in the domain of the local coefficient vectors can be mapped to the corresponding operations in the domain of the global coefficient vectors:

1) The intersection subspace $U'_\Delta(t)$ is indeed the row span of $B_\Delta(t)H(t-1)$.
2) Let $R_j(t)$ be an indicator (0-1) random variable which takes the value 1 iff the transmission in slot t is successfully received without erasure by receiver j and in addition, receiver j does not have all the information that the sender has. Let $\tilde{\mathbf{g}}_\mathbf{j}(t) := R_j(t)\mathbf{g}(t)H(t)$. Then,
$$U'_j(t) = U_j(t-1) \oplus span(\tilde{\mathbf{g}}_\mathbf{j}(t-1)) \qquad (1)$$
where $\oplus$ denotes direct sum of vector spaces. The way the algorithm chooses $\mathbf{g}(t)$ guarantees that if $R_j(t)$ (and hence $\tilde{\mathbf{g}}_\mathbf{j}(t)$) is non-zero, then $\tilde{\mathbf{g}}_\mathbf{j}(t)$ will be outside the corresponding $U_j(t)$, *i.e.*, it will be innovative. This fact is emphasized by the direct sum in this equation.
3) Because of the way the algorithm performs the completion of the bases in the local domain in step 5, the following properties hold in the global domain:
$$U(t-1) = U'_\Delta(t) \oplus U''(t) \qquad (2)$$
$$U'_j(t) = U'_\Delta(t) \oplus U''_j(t) \quad \text{and,} \qquad (3)$$
$$U''_j(t) \subseteq U''(t), \quad \forall j = 1, 2, \ldots, n \qquad (4)$$

From the above properties, we can infer that $U_1''(t) + U_2''(t) + \ldots U_n''(t) \subseteq U''(t)$. After incorporating the arrivals in slot t, this gives:

$$U_1(t) + U_2(t) + \ldots U_n(t) \subseteq U(t) \tag{5}$$

Now, in order to relate the queue size to the backlog in number of degrees of freedom, we define the following vector spaces which represent the cumulative knowledge of the sender and receivers:

- $V(t)$:= the vector space representing the knowledge of the sender after incorporating the arrivals in slot t. This is simply equal to $\mathbb{F}_q^{A(t)}$
- $V_j(t)$:= the vector space representing the knowledge of receiver j at the end of slot t
- $V_j'(t)$:= the vector space representing the knowledge of receiver j at the start of slot t, after incorporating the channel state feedback into $V_j(t-1)$, *i.e.*, $V_j'(t) = V_j(t-1) \oplus span(\tilde{\mathbf{g}}_\mathbf{j}(t-1))$.
- $V_\Delta(t)$:= $\cap_{j=1}^n V_j(t)$
- $V_\Delta'(t)$:= $\cap_{j=1}^n V_j'(t)$

Lemma 4: Let V be a vector space and let $V_\Delta, U_1, U_2, \ldots U_n$ be subspaces of V such that, V_Δ is independent of the span of all the U_j's, i.e., $dim[V_\Delta \cap (U_1 + U_2 + \ldots + U_n)] = 0$. *Then,*

$$V_\Delta \oplus [\cap_{i=1}^n U_i] = \cap_{i=1}^n [V_\Delta \oplus U_i]$$

Proof: For any $z \in V_\Delta \oplus \cap_{i=1}^n U_i$, there is a $x \in V_\Delta$ and $y \in \cap_{i=1}^n U_i$ such that $z = x + y$. Now, for each i, $y \in U_i$. Thus, $z = x + y$ implies that $z \in \cap_{i=1}^n [V_\Delta \oplus U_i]$. Therefore, $V_\Delta \oplus \cap_{i=1}^n U_i \subseteq \cap_{i=1}^n [V_\Delta \oplus U_i]$.

Now, let $w \in \cap_{i=1}^n V_\Delta \oplus U_i$. Then for each i, there is a $x_i \in V_\Delta$ and $y_i \in U_i$ such that $w = x_i + y_i$. But, $w = x_i + y_i = x_j + y_j$ means that $x_i - x_j = y_i - y_j$. Now, $(x_i - x_j) \in V_\Delta$ and $(y_i - y_j) \in (U_1 + U_2 + \ldots + U_n)$. By hypothesis, these two vector spaces have only 0 in common. Thus, $x_i - x_j = y_i - y_j = 0$. All the x_i's are equal to a common $x \in V_\Delta$ and all the y_i's are equal to a common y which belongs to all the U_i's. This means, w can be written as the sum of a vector in V_Δ and a vector in $\cap_{i=1}^n U_i$, thereby proving that $\cap_{i=1}^n [V_\Delta \oplus U_i] \subseteq V_\Delta \oplus \cap_{i=1}^n U_i$. ■

Theorem 1: For all $t \geq 0$,

$$\begin{aligned} V(t) &= V_\Delta(t) \oplus U(t) \\ V_j(t) &= V_\Delta(t) \oplus U_j(t) \quad \forall j = 1, 2, \ldots, n \\ V_\Delta'(t+1) &= V_\Delta(t) \oplus U_\Delta'(t+1) \end{aligned}$$

Proof: The proof is by induction on t.

Basis step:

At $t = 0$, $V(0)$ and $U(0)$ are both initialized to the row space of an identity matrix of size $a(0)$, while all the $V_j(0)$'s and $U_j(0)$'s are initialized to $\{\mathbf{0}\}$. Consequently, $V_\Delta(0)$ is also $\{\mathbf{0}\}$. It is easily seen that these initial values satisfy the equations in the theorem statement.

Induction Hypothesis:

We assume the equations hold at $(t-1)$, *i.e.*,

$$V(t-1) = V_\Delta(t-1) \oplus U(t-1) \tag{6}$$
$$V_j(t-1) = V_\Delta(t-1) \oplus U_j(t-1) \quad \forall j = 1, \ldots n \tag{7}$$
$$V_\Delta'(t) = V_\Delta(t-1) \oplus U_\Delta'(t) \tag{8}$$

Induction Step: Consider $V_\Delta'(t)$ and $U''(t)$. From (6) (induction hypothesis), $V_\Delta(t-1)$ and $U(t-1)$ are independent. Also, from (2), $U(t-1) = U_\Delta'(t) \oplus U''(t)$, and from (8), $V_\Delta'(t) = V_\Delta(t-1) \oplus U_\Delta'(t)$. Using these, we can show that $V_\Delta'(t)$ and $U''(t)$ are independent. Moreover,

$$\begin{aligned} V_\Delta'(t) \oplus U''(t) &= V_\Delta(t-1) \oplus U_\Delta'(t) \oplus U''(t) \\ &= V_\Delta(t-1) \oplus U(t-1) \\ &= V(t-1) \end{aligned}$$

Now, we incorporate the arrivals in slot t. This converts $V(t-1)$ to $V(t)$, $V_\Delta'(t)$ to $V_\Delta(t)$ and $U''(t)$ to $U(t)$, by the following operations:

$$\begin{aligned} \text{Basis of } V(t) &= \begin{bmatrix} \text{Basis of } V(t-1) & 0 \\ 0 & I_{a(t)} \end{bmatrix} \\ \text{Basis of } U(t) &= \begin{bmatrix} \text{Basis of } U''(t) & 0 \\ 0 & I_{a(t)} \end{bmatrix} \\ \text{Basis of } V_\Delta(t) &= \begin{bmatrix} \text{Basis of } V_\Delta'(t) & 0 \end{bmatrix} \end{aligned}$$

Clearly, these modifications do not affect the above relation, and we get:

$$V(t) = V_\Delta(t) \oplus U(t)$$

Using similar arguments as above, we can show that $V_\Delta'(t)$ and $U_j''(t)$ are independent. Moreover,

$$\begin{aligned} &V_\Delta'(t) \oplus U_j''(t) \\ &= V_\Delta(t-1) \oplus U_\Delta'(t) \oplus U_j''(t) && \text{(from (8))} \\ &= V_\Delta(t-1) \oplus U_j'(t) && \text{(from (3))} \\ &= V_\Delta(t-1) \oplus U_j(t-1) \oplus span(\tilde{\mathbf{g}}_\mathbf{j}(t-1)) && \text{(from (1))} \\ &= V_j(t-1) \oplus span(\tilde{\mathbf{g}}_\mathbf{j}(t-1)) && \text{(from (6))} \\ &= V_j'(t) \end{aligned}$$

Once again, as shown above, the incorporation of the new arrivals into the subspaces does not affect their relation to each other, and hence we get:

$$V_j(t) = V_\Delta(t) \oplus U_j(t), \quad \forall j = 1, 2, \ldots n$$

And finally,

$$\begin{aligned} V_\Delta'(t+1) &= \cap_{j=1}^n V_j'(t+1) \\ &= \cap_{j=1}^n [V_j(t) \oplus span(\tilde{\mathbf{g}}_\mathbf{j}(t))] \\ &= \cap_{j=1}^n [V_\Delta(t) \oplus U_j(t) \oplus span(\tilde{\mathbf{g}}_\mathbf{j}(t))] \\ &\overset{(a)}{=} V_\Delta(t) \oplus \cap_{j=1}^n [U_j(t) \oplus span(\tilde{\mathbf{g}}_\mathbf{j}(t))] \\ &= V_\Delta(t) \oplus U_\Delta'(t+1) \end{aligned}$$

Step (a) is justified as follows. Using equation (5) and the fact that $\tilde{\mathbf{g}}_\mathbf{j}(t)$ was chosen to be inside $U(t)$, we can show that the span of all the $[U_j(t) \oplus span(\tilde{\mathbf{g}}_\mathbf{j}(t))]$'s is inside $U(t)$. Now, from the induction step above, $V_\Delta(t)$ is independent of $U(t)$. Therefore, $V_\Delta(t)$ is independent of the span of all the $[U_j(t) \oplus span(\tilde{\mathbf{g}}_\mathbf{j}(t))]$'s. We can therefore apply Lemma 4. ■

Theorem 2: Let $K(t)$ denote the size of the queue after the arrivals in slot t have been appended to the queue.

$$K(t) = dim\ V(t) - dim\ V_\Delta(t)$$

Proof:

$$\begin{aligned} K(t) = U(t) &= dim\ U''(t) + a(t) \\ &= dim\ U(t-1) - dim\ U'_\Delta(t) + a(t) \\ &\qquad \text{(since } U(t-1) = U'_\Delta(t) \oplus U''(t)\text{)} \\ &= dim\ V(t-1) - dim\ V_\Delta(t-1) - dim\ U'_\Delta(t) + a(t) \\ &\qquad \text{(from Theorem 1)} \\ &= dim\ V(t-1) - dim\ V'_\Delta(t) + a(t) \\ &\qquad \text{(from Theorem 1)} \\ &= dim\ V(t) - dim\ V_\Delta(t) \end{aligned}$$

■

Lemma 5: Let $V_1, V_2, \ldots, V_k$ be subspaces of a vector space V. Then, for $k \geq 1$,

$$dim(V_1 \cap V_2 \cap \ldots \cap V_k) \geq \sum_{i=1}^{k} dim(V_i) - (k-1)dim(V)$$

Proof: For any two subspaces X and Y of V,

$$dim(X \cap Y) + dim(X + Y) = dim(X) + dim(Y)$$

where $X + Y$ denotes the span of subspaces X and Y.

Hence,

$$\begin{aligned} dim(X \cap Y) &= dim(X) + dim(Y) - dim(X + Y) \\ &\geq dim(X) + dim(Y) - dim(V) \qquad (9) \\ &\qquad \text{(since } X + Y \text{ is also a subspace of } V\text{)} \end{aligned}$$

Now, we prove the lemma by induction on k.

Basis step:

$k = 1$: LHS = $dim(V_1)$, RHS = $dim(V_1)$

$k = 2$: LHS = $dim(V_1 \cap V_2)$, RHS = $dim(V_1) + dim(V_2) - dim(V)$

The claim follows from inequality (9).

Induction Hypothesis:

For some arbitrary k,

$$dim(\cap_{i=1}^{k-1} V_i) \geq \sum_{i=1}^{k-1} dim(V_i) - (k-2)dim(V)$$

Induction Step:

$$\begin{aligned} &dim(\cap_{i=1}^{k} V_i) = dim(V_k \cap \cap_{i=1}^{k-1} V_i) \\ &\geq dim(V_k) + dim(\cap_{i=1}^{k-1} V_i) - dim(V) \qquad \text{(using (9))} \\ &\geq dim(V_k) + \left[\sum_{i=1}^{k-1} dim(V_i) - (k-2)dim(V)\right] - dim(V) \\ &= \sum_{i=1}^{k} dim(V_i) - (k-1)dim(V) \end{aligned}$$

The above result can be rewritten as:

$$dim(V) - dim(V_1 \cap V_2 \cap \ldots V_k) \leq \sum_{i=1}^{k} [dim(V) - dim(V_i)] \qquad (10)$$

■

If we apply Lemma 5 to the vector spaces $V_j(t), j = 1, 2, \ldots, n$ and $V(t)$, then the left hand side of inequality (10) becomes the sender queue size (using Theorem 2), while the right hand side becomes the sum of the differences in backlog between the sender and the receivers, in terms of the number of degrees of freedom. Thus, we can state the following result.

Theorem 3: The physical queue size at the sender is upper bounded by the sum of the backlog differences between the sender and each receiver in terms of the number of degrees of freedom.

V. Conclusion

In this work we present an online queueing algorithm that allows the physical queue size to track the backlog in degrees of freedom, thereby reducing the amount of storage used up at the transmitter.

From a theoretical point of view, this means that the stability of virtual queues implies the stability of the physical queues. In addition, results from traditional queueing theory about M/G/1 queues or a Jackson network type of result can be extended to the physical queue size in coded networks, as opposed to just the backlog in degrees of freedom. From a practical point of view, if the memory at the transmitter has to be shared among several different flows, then this reduction will prove quite useful in getting statistical multiplexing benefits.

In terms of future work, we wish to explore in what other ways the information about the receivers' state of knowledge can be used. For instance, we could choose the linear combination to be transmitted in such a manner that it helps the receivers decode more original packets quickly. This would help reduce the decoding delay at the receivers.

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. on Information Theory*, vol. 46, pp. 1204–1216, 2000.

[2] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, 2003.

[3] T. Ho and H. Viswanathan, "Dynamic algorithms for multicast with intra-session network coding," in *43rd Allerton Annual Conference on Communication, Control and Computing*, 2005.

[4] A. Eryilmaz and D. S. Lun, "Control for inter-session network coding," in *Proc. of the 2007 Information Theory and Applications Workshop (ITA 2007)*, January–February 2007.

[5] D. S. Lun, "Efficient operation of coded packet networks," PhD Dissertation, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, June 2006.

[6] J. K. Sundararajan, M. Médard, M. Kim, A. Eryilmaz, D. Shah, and R. Koetter, "Network coding in a multicast switch," in *Proc. of IEEE INFOCOM*, 2007.