Network Coding for Speedup in Switches

MinJi Kim, Jay Kumar Sundararajan, and Muriel Médard
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
Email: {minjikim, jaykumar, medard}@mit.edu

***Abstract*— We present a graph theoretic upper bound on speedup needed to achieve 100% throughput in a multicast switch using network coding. By bounding speedup, we show the equivalence between network coding and speedup in multicast switches - *i.e.* network coding, which is usually implemented using software, can in many cases substitute speedup, which is often achieved by adding extra switch fabrics. This bound is based on an approach to network coding problems called the "enhanced conflict graph". We show that the "imperfection ratio" of the enhanced conflict graph gives an upper bound on speedup. In particular, we apply this result to $K \times N$ switches with traffic patterns consisting of unicasts and broadcasts only to obtain an upper bound of $\min(\frac{2K-1}{K}, \frac{2N}{N+1})$.**

I. Introduction

The input-queued crossbar switch has been studied well, especially in the context of unicast traffic. It is known that 100% throughput can be achieved [1], in the sense that as long as no input or output is oversubscribed, traffic can be supported without causing the queues to grow unboundedly. Therefore, to serve any admissible unicast traffic, the input-queued crossbar switch does not need to process packets faster than the input line rate[1], *i.e.* the switch does not need *speedup*.

The extension of the problem to multicast flows, however, is intrinsically more difficult. Marsan *et al.* [5] gave a characterization of the rate region achievable in a multicast switch with fanout splitting[2], and also defined the optimal scheduling policy. Interestingly, this work proved that unlike in the unicast case, 100% throughput cannot be achieved for multicast flows in an input-queued switch. In fact, the minimum speedup needed to achieve 100% throughput grows unboundedly with the switch size.

In this paper, we discuss the same problem as [5], with the following modification. The inputs are allowed to send linear combinations of cells waiting in the queues, *i.e.*, they are allowed to perform linear *network coding* [9] with fanout splitting. The main contributions of this paper are:

1) We show that network coding can in many cases substitute speedup.
2) We provide a simple graph-theoretic upper bound on speedup.
3) We prove an upper bound on speedup of $\min\left(\frac{2K-1}{K}, \frac{2N}{N+1}\right)$ for an arbitrary $K \times N$ switch with traffic pattern restricted to unicasts and broadcasts only.

Our work builds on the work by Sundararajan *et al.* [2], [3], which gave a graph-theoretic formulation of the rate region of a multicast switch with intra-flow coding using *enhanced conflict graphs*. Given a traffic pattern, the enhanced conflict graph $G = (V, E)$ is an undirected graph that contains one vertex for every *subflow*.[3] An edge exists between two vertices if they represent two subflows from the same input or to the same output. Reference [3] shows that the stable set polytope and the fractional stable set polytope of an enhanced conflict graph are the rate region and the admissible region of a network coding switch, respectively. This graph-theoretic formulation helps us transform any given traffic pattern into a conflict graph, and the properties of this graph can be used to derive insight on the speedup required to achieve 100% throughput with coding. A similar graph-theoretic formulation was used by Caramanis *et al.* in [4] in the context of unicast traffic in Banyan networks.

Note that, for the case of fanout splitting without coding, [5] gave a characterization of the rate region as the convex hull of certain modified departure vectors. However, a graph-theoretic formulation of the same is not known. As a result, it is significantly harder to characterize the speedup required to achieve 100% throughput for fanout splitting without coding.

The rest of the paper is organized as follows. Section II states preliminary definitions that will be used throughout this paper. Section III shows that network coding is equivalent to speedup in a multicast switch to some extent. Section IV gives the relationship between speedup and imperfection ratio of a conflict graph, which leads to our main result - an upper bound on the minimum speedup required to achieve 100% throughput in a multicast switch with coding. In Section V, we apply the result from Section IV to a $K \times N$ switch with traffic consisting only of unicasts and broadcasts and give an upper bound on speedup of $\min(\frac{2K-1}{K}, \frac{2N}{N+1})$. Finally, in Section VI, we summarize the contributions of this paper, and present a conjecture on the actual minimum speedup needed to achieve

[1]The line rate of a switch is the rate at which packets arrive or leave the switch at any one port.

[2]Fanout splitting is the ability to serve partially a multicast cell to only a subset of its destined outputs, and complete the service in subsequent time slots.

[3]A flow is a stream of packets that have common source and destination set. It is represented by a 2-tuple (i, J) consisting of the input i and a subset J of outputs corresponding to the destination set of the multicast stream. A subflow of flow (i, J) is a part of a flow from input i that goes to a particular output in J. Therefore, a subflow is a 3-tuple (i, J, j) consisting of an input i, a subset of outputs J and one output $j \in J$.

100% throughput in a $2 \times N$ multicast switch with unicasts and broadcasts only.

II. Notation and Definitions

Let $G = (V, E)$ be an undirected graph with vertex set V and edge set E. A graph $G_1 = (V_1, E_1)$ is a subgraph of G if $V_1 \subseteq V$ and $E_1 \subseteq E$. A graph $G_2 = (V_2, E_2)$ is an *induced subgraph* of G if $V_2 \subseteq V$ and $(v_1, v_2) \in E_2$ if and only if $(v_1, v_2) \in E$. In addition, G_2 is often denoted as $G(V_2)$ and is said to be induced by V_2. The *complement* of graph G is a graph $\overline{G}$ on the same vertex set V such that two vertices of $\overline{G}$ are adjacent if and only if they are not adjacent in G. The *chromatic number* of a graph G is the smallest number of colors $\chi(G)$ needed to color the vertices of G so that no two adjacent vertices share the same color.

G is a *complete graph* if for every pair of vertices in V there exists an edge connecting the two, and V is called a *clique*. If for every pair of vertices in V there is no edge connecting the two, then V is said to be a *stable set*. G is a *hole* if it is a chordless cycle; G is called an *odd hole* if it is a hole of odd length at least 5. G is an *anti-hole* if its complement is a hole; G is an *odd anti-hole* if its complement is an odd hole. G is said to be *perfect* if for every induced subgraph of G, the size of the largest clique equals the chromatic number.

A. Stable Set Polytope

The *stable set polytope* $STAB(G)$ of a graph G is the convex hull of the incidence vectors of the stable sets of the graph G. In this section, we discuss how the stable set polytope of a conflict graph can translate to the rate region of a switch.

Let $\mathbf{r} \in \mathbb{R}^f$ be the *rate vector* of a traffic pattern that has f flows. Suppose that the total number of subflows in the pattern is m. Then, the *enhanced rate vector* $e(\mathbf{r}) \in \mathbb{R}^m$ corresponding to $\mathbf{r}$ is defined as:

$$e_{(i,J,j)}(\mathbf{r}) = \mathbf{r}_{(i,J)}, \text{ for all } j \in J.$$

We use the enhanced rate vector as *weights* for vertices of the enhanced conflict graph.

A traffic pattern $\mathbf{r}$ is said to be *achievable* if there exists a switch schedule that can serve it; it is called *admissible* if no input or output is oversubscribed. We also call the collection of all achievable and admissible vectors as the *achievable rate region* $\mathbf{R} \subseteq \mathbb{R}^f$ and *admissible rate region* $\mathbf{A} \subseteq \mathbb{R}^f$ respectively. For $\mathbf{r} \in \mathbf{R}$, we can construct a switch schedule, which can be viewed as a time sharing between valid switch configurations. In a conflict graph, a valid switch configuration corresponds to a stable set, and a switch schedule corresponds to a convex combination of stable sets of the conflict graph G. Therefore, if a rate vector $\mathbf{r} \in \mathbf{R}$, then $e(\mathbf{r}) \in STAB(G) \subseteq \mathbb{R}^m$.

For a general graph G, a complete characterization of $STAB(G)$ in terms of linear inequalities is unknown. However, several families of necessary conditions are known. One example is the clique inequalities[4]. The polytope described by these conditions along with non-negativity constraints[5] is the *fractional stable set polytope* $QSTAB(G)$. In terms of the switch, [3] shows that the clique inequalities of the enhanced conflict graph correspond to the *admissibility conditions*. Therefore, if a rate vector $\mathbf{r} \in \mathbf{A}$, then $e(\mathbf{r}) \in QSTAB(G) \subseteq \mathbb{R}^m$.

Note that, for most graphs, $STAB(G) \subsetneq QSTAB(G)$, since the clique inequalities are necessary but not sufficient conditions for stable set polytope. Thus, the admissible region is often a strict superset of the achievable rate region, which implies that it is not possible to achieve 100% throughput even with fanout splitting and coding - we need speedup.

B. Perfect Graph

In this section, we focus on the properties of perfect graphs. We first start by stating three well-known facts that characterize perfect graphs.

Theorem 2.1: (Weak Perfect Graph Theorem [7]) A graph G is perfect if and only if its complement is perfect.

Theorem 2.2: (Strong Perfect Graph Theorem [8]) A graph G is perfect if and only if it contains no odd hole and no odd anti-hole.

Lemma 2.3: (Replication Lemma [7]) Let $G = (V, E)$ be a perfect graph and $v \in V$. Create a new vertex v' and join it to v and to all the neighbors of v. Then, the resulting graph G' is perfect.

From Section II-A, we have that $STAB(G) \subseteq QSTAB(G)$ for any graph with equality for perfect graphs only. This implies that the admissible region $\mathbf{A}$ and the achievable rate region $\mathbf{R}$ are the same if the enhanced conflict graph is perfect. Thus, as given in *Corollary 1* from [3], if an enhanced conflict graph is perfect, then speedup is not required to achieve 100% throughput.

From this, we can observe that there is an intrinsic connection between speedup and the "perfectness" of the enhanced conflict graph. As a result, to compute the minimum speedup, it is helpful to measure how perfect an enhanced conflict graph is. In this paper, we use the *imperfection ratio* introduced by Gerke and McDiarmid [6] as such a measure.

C. Imperfection ratio

In [6], the imperfection ratio $\mathsf{imp}(G)$ of graph G is defined as $\mathsf{imp}(G) = \min\{t : QSTAB(G) \subseteq t\ STAB(G)\}$. As we noted in Section II-A, in terms of a switch, the admissible region $\mathbf{A}$ and the achievable region $\mathbf{R}$ are projections of $QSTAB(G)$ and $STAB(G)$ respectively. Therefore, given the imperfection ratio $\mathsf{imp}(G)$ of an enhanced conflict graph G, we have $\mathbf{A} \subseteq \mathsf{imp}(G)\mathbf{R}$.

A useful bound on the imperfection ratio is presented in [6], which we reproduce below.

Proposition 2.4: (Gerke and McDiarmid [6]) For a graph G, if each vertex in G can be covered p times by a family of q induced perfect subgraphs, then $\mathsf{imp}(G) \leq \frac{q}{p}$.

[4]Clique inequalities of a graph say that the total weight on the vertices of maximal cliques must not exceed 1. In an enhanced conflict graph, the clique inequalities imply that no input nor any output may be overloaded.

[5]Non-negativity constraints of a graph say that the weight on each vertex is non-negative.

D. Speedup

A switch is said to have a *speedup* s if the switching fabric can transfer packets at a rate s times the incoming and outgoing line rate of the switch. If we define a time slot to be the reciprocal of the line rate, then this means the switching fabric can go through s configurations within one time slot. With this definition, it is easy to see that a rate vector $\mathbf{r}$ is achievable with speedup s if and only if it is admissible and $\frac{1}{s}\mathbf{r}$ is within the rate region.

Note that the admissible and achievable rates correspond to $\mathbf{A}$ and $\mathbf{R}$ respectively. Then, $s_{\min} = \min\{s \mid \mathbf{A} \subseteq s\ \mathbf{R}\}$ is the *minimum speedup* required for the switch to achieve all admissible rates, *i.e.* it is the minimum of all s such that $\frac{1}{s}\mathbf{r}$ is within the rate region for all admissible rate vectors $\mathbf{r}$.

III. Network Coding for Speedup

In this section, we show the equivalence between network coding and speedup in multicast switches - *i.e.* network coding, which is usually implemented using software, can in many cases substitute speedup, which is often achieved by adding extra switch fabrics.

In Figure 1, we show a special traffic pattern in a $2 \times N$ switch, which demonstrates the benefit of intra-flow coding. At input 1, there is one broadcast flow with rate $1-\frac{1}{N}$; at input 2, there is one unicast to each output with rate $\frac{1}{N}$. Reference [3] shows that this traffic is achievable if network coding with fanout splitting is allowed; however, a speedup of $1.5 - \frac{1}{N}$ is needed if only fanout splitting is allowed. This example shows that network coding is equivalent to a speedup of at least $1.5 - \frac{1}{N}$.

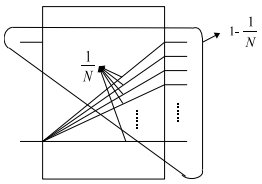

Fig. 1. A traffic pattern which demonstrates the benefit of coding

However, it is important to note that network coding cannot completely replace speedup. As noted above in Figure 1, there are situations where network coding reduces speedup; however, there are situations where speedup needed remains the same for with and without network coding. For instance, in Figure 2, we show a traffic pattern that requires speedup of 1.25 with or without network coding. At input 1, there is a broadcast flow and a unicast to output 1 with rate $\frac{1}{2}$ each; at input 2, there is one unicast flow to each output 2 and 3 with rate $\frac{1}{2}$. In Figure 2, we show that the enhanced conflict graph for this traffic, where u_{ij} represents the unicast flow from input i to output j, and the vertex b_{ij} represents the broadcast subflow from input i to output j. The enhanced conflict graph contains an odd hole; therefore, it is not perfect.

Note that the traffic pattern in Figure 2 gives a lower bound on the speedup needed to achieve 100% throughput in a multicast switch using network coding. Therefore, $s_{\min} \geq 1.25$.

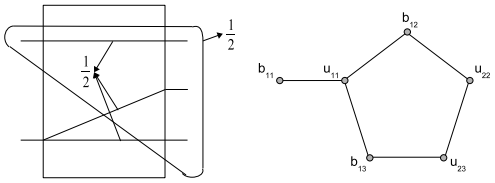

Fig. 2. A traffic pattern which requires speedup in a 2×3 switch and its enhanced conflict graph

IV. Imperfection Ratio Bounds Speedup

This section develops our main result, which relates speedup with imperfection ratio. Note that, the definition of imperfection ratio in Section II-C is very similar to that of minimum speedup in Section II-D. As a result, Corollary 4.1 follows from Proposition 2.4.

Corollary 4.1: Given a traffic pattern, let G be its enhanced conflict graph and $s_{\min}$ be the minimum speedup required to achieve all admissible rates. Then, $s_{\min} \leq \mathsf{imp}(G)$.

Note that the converse of Corollary 4.1 is not true. This is because $\mathbf{A}$ and $\mathbf{R}$ are projections of $QSTAB(G)$ and $STAB(G)$ such that the subflows corresponding to the same multicast flow have the same weight. As a result, $QSTAB(G) \subseteq \mathsf{imp}(G)STAB(G)$ implies the $\mathbf{A} \subseteq \mathsf{imp}(G)\mathbf{R}$, but $\mathbf{A} \subseteq s_{\min}\mathbf{R}$ may not imply $QSTAB(G) \subseteq s_{\min}STAB(G)$.

V. Bounds on Speedup for $K \times N$ switch with unicasts and broadcasts

In this section, we apply Corollary 4.1 to $K \times N$ switches using intra-flow coding with traffic patterns consisting of unicasts and broadcasts only. We show that the minimum speedup needed for 100% throughput in this case is bounded by $\min(\frac{2K-1}{K}, \frac{2N}{N+1})$. In this section, coding implies intra-flow coding, since enhanced conflict graphs handle intra-flow, not inter-flow, coding. The rest of this section is organized as follows. First, we give a description of the enhanced conflict graph for a $K \times N$ switch. In Section V-B and V-C, we show the two bounds on speedup of $\frac{2K-1}{K}$ and $\frac{2N}{N+1}$ respectively.

A. Enhanced conflict graph for $K \times N$ switch

Consider traffic patterns which consist only of unicasts and a broadcast per each input on a $K \times N$ switch. The basic idea behind conflict graph is that vertices representing flows that cannot be served simultaneously are adjacent. In such a case, the enhanced conflict graph $G_{K,N} = (V, E)$ has the following structure.

The vertex set $V = \left(\cup_{i\in[1,K]}U_i\right) \cup \left(\cup_{i\in[1,K]}B_i\right) = \left(\cup_{j\in[1,N]}U_j^o\right) \cup \left(\cup_{j\in[1,N]}B_j^o\right)$ where $U_i = \{u_{ij} \mid j \in [1,N]\}$[6], $B_i = \{b_{ij} \mid j \in [1,N]\}$, $U_j^o = \{u_{ij} \mid i \in [1,K]\}$, and $B_j^o = \{b_{ij} | i \in [1,K]\}$. The vertex u_{ij} represents the unicast flow from input i to output j, and the vertex b_{ij} represents the broadcast subflow from input i to output j. Therefore,

[6] $j \in [1,N]$ means j can be integer from 1 to N.

U_i and U_j^o are collections of the unicast flows from input i and to output j respectively. B_i and B_j^o are collections of the subflows of the broadcast from input i and to output j respectively.

The edge set $E = \left(\cup_{i\in[1,K]} E_i^u\right) \cup \left(\cup_{i\in[1,K]} E_i^b\right) \cup E^o$ where $E_i^u = \{(u_{ij}, u_{ik}) \mid j \neq k, j, k \in [1, N]\}$, $E_i^b = \{(b_{ij}, u_{ik}) \mid j, k \in [1, N]\}$, and $E^o = \cup_{i\in[1,N]} E_i^o$ where $E_i^o = \{(u_{ji}, u_{ki}), (b_{ji}, b_{ki}), (b_{ji}, u_{ki}) \mid j \neq k, j, k \in [1, K]\}$. Each edge set represents a different type of conflict. E_i^u represents conflicts among unicasts at input i; E_i^b represents conflict between any broadcast subflow and any unicast at input i; and E_i^o represents conflicts among all flows and subflows at output i.

It is important to note that each vertex in $G_{K,N}$ represents a subflow in a $K \times N$ switch. For example, u_{11} and u_{21} corresponds to a unicast traffic to output 1 from input 1 and input 2 respectively. The vertex b_{12} represents a partial service of the broadcast from input 1 to output 2. In Figure 3, we show the switch configuration corresponding to u_{11}, u_{21}, and b_{12} in a 2×3 switch.

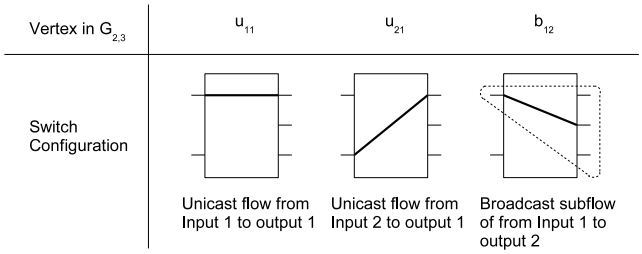

Fig. 3. Switch configuration corresponding to u_{11}, u_{21}, and b_{12} in $G_{2,3}$

The intuition behind a conflict graph is that vertices which represent flows that cannot be served simultaneously are adjacent. As shown in [3], if fanout splitting and network coding are allowed, the switch can simultaneously serve two or more subflows of the same broadcast flow and hence such subflows are not adjacent to each other. For example, in Figure 4, there are edges between u_{11} and b_{12}, since they conflict at input 1, and between u_{11} and u_{21}, since they conflict at output 1; however u_{21} and b_{12} are not adjacent, since they have different input and output. Therefore, from the input perspective, $G_{K,N}$ consists of K induced complete subgraphs $G_{K,N}(U_i)$ for unicasts from each input i, and K induced stable sets $G_{K,N}(B_i)$ for broadcasts from each input i; from the output perspective, $G_{K,N}$ consists of $2N$ induced complete subgraphs $G_{K,N}(U_j^o)$ and $G_{K,N}(B_j^o)$ for unicasts and broadcast subflows to each output j respectively.

Here, we note that conflict graph of a $K \times N$ multicast switch with unicasts and broadcasts traffic can be relaxed to that of unicasts and single multicast per input. This relaxation just removes vertices that represent broadcast subflows, which are not part of the multicast flow, from the conflict graph. This cannot hurt the "perfectness" of the conflict graph. Therefore, any upper bound on the imperfection ratio of the conflict graph for unicasts and broadcasts bounds that of unicasts and single multicast per input.

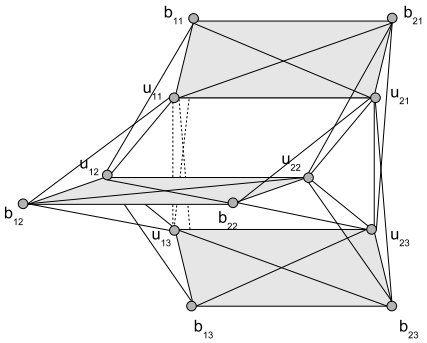

Fig. 4. $G_{2,3}$ for a 2×3 switch with unicasts and broadcasts only

B. Speedup of $\frac{2K-1}{K}$

In this Section, we give an upper bound on speedup for $K \times N$ switches. We present $2K-1$ induced perfect subgraphs of $G_{K,N}$ that cover V K times. Then, with Proposition 2.4, we then have $\frac{2K-1}{K}$ as an upper bound for speedup.

Lemma 5.1: Let $G_u = G_{K,N}(\cup_{i\in[1,K]} U_i)$ be an induced subgraph of $G_{K,N}$. Then G_u is perfect.

Proof: G_u is an enhanced conflict graph for unicast traffic. One may check that G_u is a line graph of a bipartite graph, which is known to be perfect [8]. ■

Lemma 5.1 also follows from the result in [1] which shows that 100% throughput can be achieved in a input-queued crossbar switch in the context of unicast traffic.

Lemma 5.2: Let $G_i = G_{K,N}\left((\cup_{j\in[1,K]} B_j) \cup U_i\right)$ for some $i \in [1, K]$ be an induced subgraph of $G_{K,N}$. Then G_i is perfect.

Proof: Assume that G_i is not perfect. So it must have an odd hole or odd anti-hole as an induced subgraph. Suppose it has an odd hole, say H. In G_i, any broadcast subflow, except the ones from input i, has no conflict on the input side. Suppose such a subflow were part of H, then both its neighbors in H will be due to output side conflicts. But in that case, the two neighbors will themselves conflict at the output, thereby forming a triangle. Since an odd hole cannot contain a triangle, we conclude that H cannot include any b_{jk}, $j \neq i$.

This means H must be an induced subgraph of $G_{K,N}(B_i \cup U_i)$. However, B_i induces a stable set, while U_i induces a clique. Therefore, $G_{K,N}(B_i \cup U_i)$ is a split graph[7] which is known to be perfect [8]. This contradiction shows that G_i cannot contain an odd hole H.

Suppose G_i contains an odd anti-hole, say A. This will happen if and only if $\overline{G_i}$ contains an odd hole H_A. Note that in $\overline{G_i}$, two vertices are connected if the corresponding subflows do not conflict. Now, H_A has to contain at least one unicast, say u_{ij}, since the broadcasts by themselves induce a perfect subgraph in $\overline{G_i}$ (they induce the complement of a disjoint union of complete graphs, which is known to be perfect [8]). Now, u_{ij} in $\overline{G_i}$ is adjacent to any $b_{i'j'}$, where $i \neq i'$ and $j \neq j'$. Let b_{pq} and $b_{p'q'}$ be vertices adjacent to u_{ij} in H_A. Then, using the definition of $\overline{G_i}$, we can infer that $i \neq p \neq p' \neq i$ and

[7] A *split graph* is a graph whose vertex set can be partitioned into a stable set and a clique.

$q = q' \neq j$. But this means, any vertex that is adjacent to b_{pq} is also adjacent to $b_{p'q'}$. Hence, H_A cannot be an odd hole.

This proves that G_i is perfect. ■

Using Lemmas 5.1 and 5.2, we derive our first upper bound on speedup in $K \times N$ multicast switches with traffic patterns consisting of unicasts and broadcasts only.

Proposition 5.3: $\mathsf{imp}(G_{K,N}) \leq \frac{2K-1}{K}$.

Proof: Consider the following collection of induced subgraphs: $K-1$ copies of G_u from Lemma 5.1 and G_i from Lemma 5.2 for all $i \in [1, K]$. We know that these subgraphs are all perfect. In addition, these subgraphs cover each vertex in $v \in G_{K,N}$ K times. By Proposition 2.4, the claim follows. ■

C. Speedup of $\frac{2N}{N+1}$

The proof idea in this section is similar to that of Section V-B. We present $2N$ induced perfect subgraphs of $G_{K,N}$ that cover V $N+1$ times, and then appeal to Proposition 2.4. However, unlike Section V-B, here we change our focus from the input to output.

Lemma 5.4: Let $G^o_{1,i} = G_{K,N}(V_i)$ where $V_i = U^o_i \cup \left(\cup_{j\in[1,N]} B^o_j\right)$ be an induced subgraph of $G_{K,N}$. Then $G^o_{1,i}$ is perfect.

Proof: Assume that $G^o_{1,i}$ is not perfect. So it must have an odd hole or odd anti-hole as an induced subgraph. Suppose it has an odd hole, say H. Since $U^o_i \cup B^o_i$ forms a complete graph (known to be perfect), H must contain vertices of B^o_j, $j \neq i$. Suppose $b_{kj} \in B^o_j$ is part of H, then H contains at least two vertices of B^o_j. This is because, in $G^o_{1,i}$, b_{kj} has only one conflict on the input side; thus, neighbors of b_{kj} are u_{ki} (input conflict) and B^o_j (output conflict). However, note that B^o_j itself forms a complete graph, therefore H contains at most two vertices of B^o_j. Thus, b_{kj} and $b_{k'j}$, $k \neq k'$ are in H. Then, u_{ki} and $u_{k'i}$ are in H. However, these four vertices form a cycle, thus $G^o_{1,i}$ cannot contain an odd hole H.

By the same argument as in the proof for Lemma 5.2, we can show that $G^o_{1,i}$ cannot contain an odd anti-hole. ■

Lemma 5.5: Let $G^o_{2,i} = G_{K,N}(V_i)$ where $V_i = B^o_i \cup \left(\cup_{j\in[1,N]} U^o_j\right)$ be an induced subgraph of $G_{K,N}$. Then, $G^o_{2,i}$ is perfect.

Proof: $G^o_{2,i}$ is an enhanced conflict graph for unicast traffic in addition to all broadcast subflows to output i. Consider $b_{1i} \in B^o_i$ and $u_{1i} \in \cup_{i\in[1,K]} U_i$. In a $K \times N$ switch, b_{1i} and u_{1i} represent subflows from input 1 to output i, and thus conflict with the same set of subflows, *i.e.* neighbors of u_{1i} are neighbors of b_{1i}. In addition, b_{1i} and u_{1i} are in conflict. Therefore, by Replication Lemma (Lemma 2.3), we know that $G^o_{2,i}$ is perfect if $G_{K,N}(V_i \setminus \{b_{1i}\})$ is perfect. We can apply this argument repeatedly for each $b_{ji} \in B^o_i$, and deduce that if $G_{K,N}(\cup_{j\in[1,N]} U^o_j)$ perfect then $G^o_{2,i}$ is perfect. Note that from Lemma 5.1, we know that the enhanced conflict graph $G_u = G_{K,N}(\cup_{i\in[1,K]} U_i) = G_{K,N}(\cup_{j\in[1,N]} U^o_j)$ for unicast traffic is perfect. Therefore, $G^o_{2,i}$ is perfect. ■

Now, using Lemmas 5.4 and 5.5, we can derive an upper bound for speedup in $K \times N$ multicast switches with traffic patterns consisting of unicasts and broadcasts only.

Proposition 5.6: $\mathsf{imp}(G_{K,N}) \leq \frac{2N}{N+1}$.

Proof: Consider the following collection of induced subgraphs: $G^o_{1,i}$ and $G^o_{2,i}$ for all $i \in [1, N]$. By Lemmas 5.4 and 5.5, we know that these subgraphs are all perfect. In addition, these subgraphs cover each vertex in $v \in G_{K,N}$ $N+1$ times. By Proposition 2.4, the claim follows. ■

VI. Conclusion

In this paper, we introduce a simple graph theoretic bound on speedup needed to achieve 100% throughput in a multicast network coding switch using the concept of conflict graphs. We show that the imperfection ratio of the conflict graph gives an upper bound on speedup. We apply this result to $K \times N$ switches with traffic patterns consisting of unicasts and broadcasts only to obtain an upper bound of $\min(\frac{2K-1}{K}, \frac{2N}{N+1})$. For a $2 \times N$ switch, this gives a bound of 3/2 on speedup; however, we conjecture that the actual speedup required to achieve 100% throughput in a $2\times N$ switch with traffic patterns consisting of unicasts and broadcasts only is 5/4. We have verified this conjecture using a computer for $N = 3, 4$ and 5.

In summary, by allowing network coding in multicast switches, we get not only an insightful characterization of the speedup needed for 100% throughput, but also a gain in speedup. We have shown that network coding, which is usually implemented using software, can substitute speedup, which is often achieved by adding extra switch fabrics.

Acknowledgment

This material is based upon research partly supported by Stanford University under the Complex Network Infrastructures for Communication and Power, Sponsor Award No. PY-1362; University of California under DAWN: Dynamic Adhoc Wireless Networking, Sponsor Award No. S0176938; Air Force Aerospace Research - OSR under the Robust Self-Authenticating Network Coding, Sponsor Award No. FA9550-06-1-0155; and DARPA ITMANET.

References

[1] N. McKeown, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch", in *Proceedings of IEEE INFOCOM*, 1996, pp. 296-302.

[2] J.K. Sundararajan, M. Médard, R. Koetter, and E. Erez, "A systematic appraoch to network coding problems using conflict graphs", in *Proceedings of the UCSD Workshop on Information Theory and its Applications*, San Diego, CA, February 2006.

[3] J.K. Sundararajan, M. Médard, M. Kim, A. Eryilmaz, D. Shah, and R. Koetter, "Network Coding in a Multicast Switch", in *Proceedings of IEEE Infocom*, 2007

[4] C. Caramanis, M. Rosenblum, M. X. Goemans, and V. Tarokh, "Scheduling algorithms for providing flexible, rate-based, quality of service guarantees for packet-switching in Banyan networks", in *Proceedings of the Conference on Information Sciences and Systems*, 2004, pp. 160-166.

[5] M. A. Marsan, A. Bianco, P. Giaccone, E. Leonardi, and F. Neri, "Multicast traffic in input-queued switches: optimal scheduling and maximum throughput", *IEEE/ACM Trans. Netw.*, vol. 11, no. 3, pp. 465-477, 2003.

[6] S. Gerke and C. McDiarmid, "Graph Imperfection I, II", *Journal of Combinatorial Theory*, Series B 83 (2001), 58-78, 79-101.

[7] L. Lovász, "Normal hypergraphs and the perfect graph conjecture", *Discrete Mathematics 2* (1972) 253-267

[8] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, Springer Verlag, 2003.

[9] 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.