Fast Resource Allocation for Network-Coded Traffic
A Coded-Feedback Approach

Chih-Chun Wang and Xiaojun Lin
Center for Wireless Systems and Applications (CWSA)
School of ECE, Purdue University

Content

- Existing results on resource allocation for wireline networks.
- The throughput and delay benefit of network-coded multicast traffic.
- A new primal approach that takes full advantage of network coded traffic.
- ♡ A fast min-cut algorithm based on coded feedback.
- Simulation & comparison to existing works.
  - Comparable performance to the back-pressure algorithms.
  - Many desirable features of a primal approach and for network-coded traffic.
- Conclusion.

The Setting

- Wireline networks:
  $G = (V, E)$, w. edge capacity c_e (packets/sec).
- Multiple multicast sessions: $(s_i, \{d_{i,j}\})$.

The Setting

- Wireline networks:
  $G = (V, E)$, w. edge capacity c_e (packets/sec).
- Multiple multicast sessions: $(s_i, \{d_{i,j}\})$.
- Rate allocation $\mathbf{r}^{(i)} = \{r_e^{(i)}\}$ for multicast session $(s_i, \{d_{i,j}\})$.
- $R(\mathbf{r}^{(i)})$: The $(s_i, \{d_{i,j}\})$ multicast rate supported by the rate allocation $\mathbf{r}^{(i)}$.

The Setting

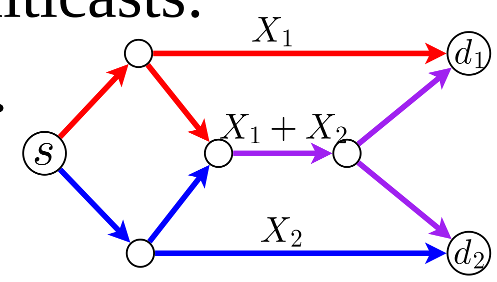
- Wireline networks:
  $G = (V, E)$, w. edge capacity c_e (packets/sec).
- Multiple multicast sessions: $(s_i, \{d_{i,j}\})$.
- Rate allocation $\mathbf{r}^{(i)} = \{r_e^{(i)}\}$ for multicast session $(s_i, \{d_{i,j}\})$.
- $R(\mathbf{r}^{(i)})$: The $(s_i, \{d_{i,j}\})$ multicast rate supported by the rate allocation $\mathbf{r}^{(i)}$.
- Utility maximization by convex optimization:

$$\max_{r_e^{(i)} \geq 0} \quad \sum_i U_i(R(\mathbf{r}^{(i)}))$$

$$\text{subject to} \quad \sum_i r_e^{(i)} \leq c_e, \ \forall e \in E$$

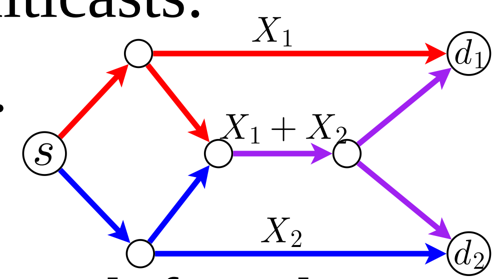
Benefits of (Intrasession) Network Coding

- Strict throughput improvement for multiple multicasts: For the same $\mathbf{r}^{(i)}$, $R_{\text{coding}}(\mathbf{r}^{(i)}) > R_{\text{routing}}(\mathbf{r}^{(i)})$.

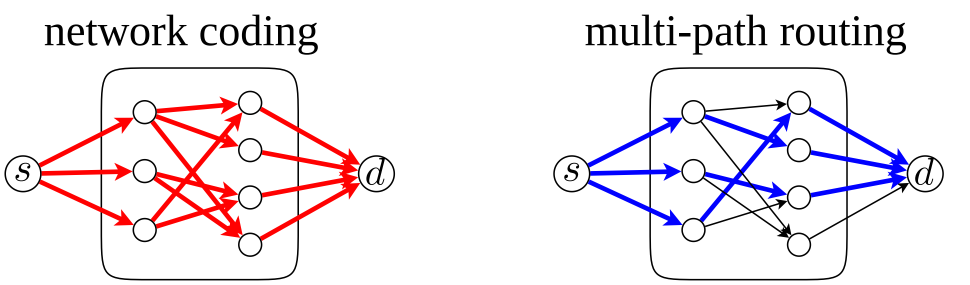


Benefits of (Intrasession) Network Coding

- Strict throughput improvement for multiple multicasts:
  For the same $\mathbf{r}^{(i)}$, $R_{\text{coding}}(\mathbf{r}^{(i)}) > R_{\text{routing}}(\mathbf{r}^{(i)})$.

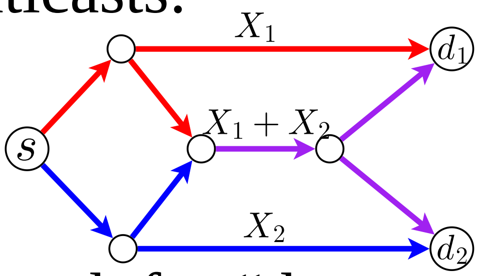


- Even for unicasts:
  Autonomous random mixing $\Longrightarrow$ No need to search for "the edge-disjoint paths" (the max flow)
  - Low-complexity,
  - Instant ON.

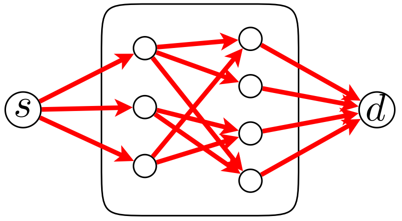


Benefits of (Intrasession) Network Coding

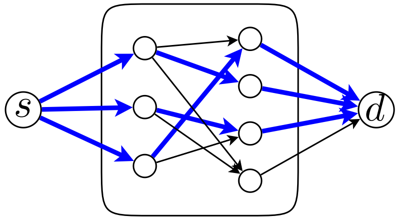
- Strict throughput improvement for multiple multicasts:
  For the same $\mathbf{r}^{(i)}$, $R_{\text{coding}}(\mathbf{r}^{(i)}) > R_{\text{routing}}(\mathbf{r}^{(i)})$.



- Even for unicasts :
  Autonomous random mixing $\Longrightarrow$ No need to search for "the edge-disjoint paths" (the max flow)
  - Low-complexity,
  - Instant ON.



network coding

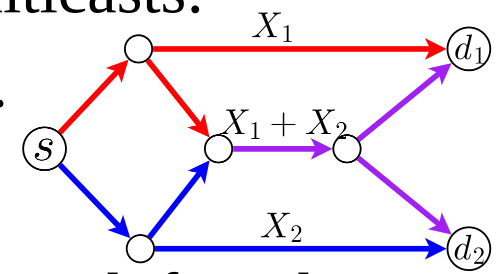


multi-path routing

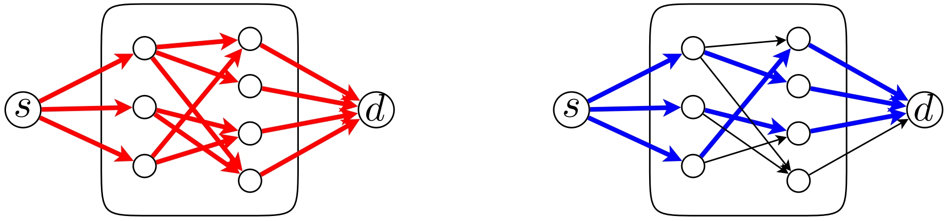
- Autonomous random mixing may easily waste bandwidth.

Benefits of (Intrasession) Network Coding

- Strict throughput improvement for multiple multicasts:
  For the same $\mathbf{r}^{(i)}$, $R_{\text{coding}}(\mathbf{r}^{(i)}) > R_{\text{routing}}(\mathbf{r}^{(i)})$.



- Even for unicasts :
  Autonomous random mixing $\Longrightarrow$ No need to search for “the edge-disjoint paths" (the max flow)
  - Low-complexity,
  - Instant ON.



- Autonomous random mixing may easily waste bandwidth.
- Rate allocation is critical.

Existing Rate Control Results

- The dual approaches (solving the dual problem): [Lun *et al.* 06], [Wu *et al.* 06], [Chen *et al.* 07], [Khreishah *et al.* 08], and many more.
  - Disconnection between the dual (price) and the primal (rate) variables.
  - Convergence of the dual $\not\Rightarrow$ convergence of the primal
  - The utility may not be monotonically increasing.
  - Rate assignment > capacity. Queue build-up. Delay?!
- The primal approaches (directly solving the $\mathbf{r}^{(i)}$): [Xi *et al.* 05], [Wu *et al.* 06].
  - The utility is monotonically increasing.
  - Rate assignment $\leq$ capacity. No queue build-up.

A Subgradient Approach

- The min-cut/max-flow theorem [Ahlswede *et al.* 00]: $R(\mathbf{r}^{(i)})$ is the min-cut/max-flow value $\mathrm{mcMF}(\mathbf{r}^{(i)})$.

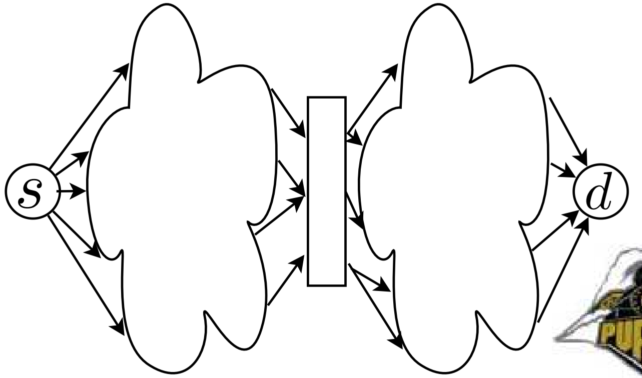
A Subgradient Approach

- The min-cut/max-flow theorem [Ahlswede *et al.* 00]: $R(\mathbf{r}^{(i)})$ is the min-cut/max-flow value $\mathrm{mcMF}(\mathbf{r}^{(i)})$.
- $$\begin{aligned} \max_{r_e^{(i)} \geq 0} \quad & F(\mathbf{r}) = \sum_i U_i(\mathrm{mcMF}(\mathbf{r}^{(i)})) \\ \text{subject to} \quad & \sum_i r_e^{(i)} \leq c_e, \; \forall e \in E \end{aligned}$$

A Subgradient Approach

- The min-cut/max-flow theorem [Ahlswede *et al.* 00]: $R(\mathbf{r}^{(i)})$ is the min-cut/max-flow value $\mathrm{mcMF}(\mathbf{r}^{(i)})$.
- $$\max_{r_e^{(i)} \geq 0} \quad F(\mathbf{r}) = \sum_i U_i(\mathrm{mcMF}(\mathbf{r}^{(i)}))$$
  $$\text{subject to} \quad \sum_i r_e^{(i)} \leq c_e, \ \forall e \in E$$
- A subgradient approach [Wu *et al.* 06]:
  - Denote the session min-cut by $C_{\min}^{(i)}$ (based on $\mathbf{r}^{(i)}$).
  - The subgradient of the $F(\mathbf{r})$ is

$$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\mathrm{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$



[Wu *et al.* 06] Approach

- The subgradient: $$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\text{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$
- Step 1:
  -
  -
- Step 2:
  -
  -
- Step 3: Subgradient update for the primal variables.

$$\{r_e^{(i)} : i\} \triangleq \mathbf{r}_e^{(\cdot)} \leftarrow \left[\mathbf{r}_e^{(\cdot)} + \alpha \frac{\partial F(\mathbf{r})}{\partial \mathbf{r}_e^{(\cdot)}}\right]_{c_e}$$

[Wu *et al.* 06] Approach

- The subgradient: $$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\mathrm{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$
- Step 1: Finding the $\mathrm{mcMF}(\mathbf{r}^{(i)})$ value.
  -
  -
- Step 2:
  -
  -
- Step 3: Subgradient update for the primal variables.

$$\{r_e^{(i)} : i\} \triangleq \mathbf{r}_e^{(\cdot)} \leftarrow \left[\mathbf{r}_e^{(\cdot)} + \alpha \frac{\partial F(\mathbf{r})}{\partial \mathbf{r}_e^{(\cdot)}}\right]_{c_e}$$

[Wu *et al.* 06] Approach

- The subgradient: $$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\mathrm{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$
- Step 1: Finding the $\mathrm{mcMF}(\mathbf{r}^{(i)})$ value.
  -
  -
- Step 2: Finding the session min-cut $C_{\min}^{(i)}$.
  -
  -
- Step 3: Subgradient update for the primal variables.

$$\{r_e^{(i)} : i\} \triangleq \mathbf{r}_e^{(\cdot)} \leftarrow \left[\mathbf{r}_e^{(\cdot)} + \alpha \frac{\partial F(\mathbf{r})}{\partial \mathbf{r}_e^{(\cdot)}}\right]_{c_e}$$

[Wu *et al.* 06] Approach

- The subgradient: $$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\mathrm{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$
- Step 1: Finding the $\mathrm{mcMF}(\mathbf{r}^{(i)})$ value.
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  -
- Step 2: Finding the session min-cut $C_{\min}^{(i)}$.
  -
  -
- Step 3: Subgradient update for the primal variables.

$$\{r_e^{(i)} : i\} \triangleq \mathbf{r}_e^{(\cdot)} \leftarrow \left[\mathbf{r}_e^{(\cdot)} + \alpha \frac{\partial F(\mathbf{r})}{\partial \mathbf{r}_e^{(\cdot)}}\right]_{c_e}$$

[Wu *et al.* 06] Approach

- The subgradient: $$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\mathrm{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$
- Step 1: Finding the $\mathrm{mcMF}(\mathbf{r}^{(i)})$ value. Commun. Complexity
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  -
- Step 2: Finding the session min-cut $C_{\min}^{(i)}$.
  -
  -
- Step 3: Subgradient update for the primal variables.

$$\{r_e^{(i)} : i\} \triangleq \mathbf{r}_e^{(\cdot)} \leftarrow \left[\mathbf{r}_e^{(\cdot)} + \alpha \frac{\partial F(\mathbf{r})}{\partial \mathbf{r}_e^{(\cdot)}}\right]_{c_e}$$

[Wu *et al.* 06] Approach

- The subgradient: $$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\mathrm{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$
- Step 1: Finding the $\mathrm{mcMF}(\mathbf{r}^{(i)})$ value. Commun. Complexity
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  -
- Step 2: Finding the session min-cut $C_{\min}^{(i)}$.
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  -
- Step 3: Subgradient update for the primal variables.

$$\{r_e^{(i)} : i\} \triangleq \mathbf{r}_e^{(\cdot)} \leftarrow \left[\mathbf{r}_e^{(\cdot)} + \alpha \frac{\partial F(\mathbf{r})}{\partial \mathbf{r}_e^{(\cdot)}}\right]_{c_e}$$

[Wu *et al.* 06] Approach

- The subgradient: $$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\mathrm{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$
- Step 1: Finding the $\mathrm{mcMF}(\mathbf{r}^{(i)})$ value. Commun. Complexity
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  - Method 2: $\mathcal{O}(|V|)$ Random network coding + rank checking
- Step 2: Finding the session min-cut $C_{\min}^{(i)}$.
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  -
- Step 3: Subgradient update for the primal variables.

$$\{r_e^{(i)} : i\} \triangleq \mathbf{r}_e^{(\cdot)} \leftarrow \left[\mathbf{r}_e^{(\cdot)} + \alpha \frac{\partial F(\mathbf{r})}{\partial \mathbf{r}_e^{(\cdot)}}\right]_{c_e}$$

[Wu *et al.* 06] Approach

- The subgradient: $$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\text{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$
- Step 1: Finding the $\text{mcMF}(\mathbf{r}^{(i)})$ value. Commun. Complexity
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  - Method 2: $\mathcal{O}(|V|)$ Random network coding + rank checking
- Step 2: Finding the session min-cut $C_{\min}^{(i)}$.
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  - ??
- Step 3: Subgradient update for the primal variables.

$$\{r_e^{(i)} : i\} \triangleq \mathbf{r}_e^{(\cdot)} \leftarrow \left[\mathbf{r}_e^{(\cdot)} + \alpha \frac{\partial F(\mathbf{r})}{\partial \mathbf{r}_e^{(\cdot)}}\right]_{c_e}$$

[Wu *et al.* 06] Approach

- The subgradient: $$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\text{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$
- Step 1: Finding the mcMF$(\mathbf{r}^{(i)})$ value. Commun. Complexity
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  - Method 2: $\mathcal{O}(|V|)$ Random network coding + rank checking
- Step 2: Finding the session min-cut $C_{\min}^{(i)}$. Bottleneck
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  - ??
- Step 3: Subgradient update for the primal variables.

$$\{r_e^{(i)} : i\} \triangleq \mathbf{r}_e^{(\cdot)} \leftarrow \left[\mathbf{r}_e^{(\cdot)} + \alpha \frac{\partial F(\mathbf{r})}{\partial \mathbf{r}_e^{(\cdot)}}\right]_{c_e}$$

[Wu *et al.* 06] Approach

- The subgradient: $$\frac{\partial F(\mathbf{r})}{\partial r_e^{(i)}} = U_i'(\mathrm{mcMF}(\mathbf{r}^{(i)}))1_{\{e \in C_{\min}^{(i)}\}}$$
- Step 1: Finding the $\mathrm{mcMF}(\mathbf{r}^{(i)})$ value. Commun. Complexity
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  - Method 2: $\mathcal{O}(|V|)$ Random network coding + rank checking
- Step 2: Finding the session min-cut $C_{\min}^{(i)}$. Bottleneck
  - Method 1: $\mathcal{O}(|V|^2)$ Push-&-Relabel [Goldberg *et al.* 88].
  - ?? Take advantage of network coding as in Method 2?
- Step 3: Subgradient update for the primal variables.

$$\{r_e^{(i)} : i\} \triangleq \mathbf{r}_e^{(\cdot)} \leftarrow \left[\mathbf{r}_e^{(\cdot)} + \alpha \frac{\partial F(\mathbf{r})}{\partial \mathbf{r}_e^{(\cdot)}}\right]_{c_e}$$

Goal: Design a new, ultra-fast min-cut algorithm that takes full advantage of network coding.

Goal: Design a new, ultra-fast min-cut algorithm that takes full advantage of network coding.

- Practical Network Coding [Chou *et al.* 03]: Coding coefficients are embedded in the header of each packet, which record the coding operations experienced by each packet.

Goal: Design a new, ultra-fast min-cut algorithm that takes full advantage of network coding.

- Practical Network Coding [Chou *et al.* 03]: Coding coefficients are embedded in the header of each packet, which record the coding operations experienced by each packet.
- Using the coding vectors as the probing signals [Ho *et al.* 05], [Fragouli *et al.* 05, 06], [Gjoka *et al.* 07].
  - Finding the mcMF value: $\mathcal{O}(|V|)$ Random network coding + rank checking

Goal: Design a new, ultra-fast min-cut algorithm that takes full advantage of network coding.

- Practical Network Coding [Chou *et al.* 03]: Coding coefficients are embedded in the header of each packet, which record the coding operations experienced by each packet.
- Using the coding vectors as the probing signals [Ho *et al.* 05], [Fragouli *et al.* 05, 06], [Gjoka *et al.* 07].
  - Finding the mcMF value: $\mathcal{O}(|V|)$ Random network coding + rank checking
- Use coded feedback. Two-way messages: source $\leftrightarrow$ dest.
  - The information is interpreted at the intermediate nodes.

Goal: Design a new, ultra-fast min-cut algorithm that takes full advantage of network coding.

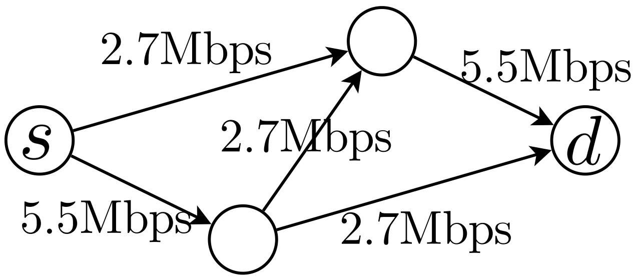
- Practical Network Coding [Chou *et al.* 03]: Coding coefficients are embedded in the header of each packet, which record the coding operations experienced by each packet.
- Using the coding vectors as the probing signals [Ho *et al.* 05], [Fragouli *et al.* 05, 06], [Gjoka *et al.* 07].
  - Finding the mcMF value: $\mathcal{O}(|V|)$ Random network coding + rank checking
- Use coded feedback. Two-way messages: source $\leftrightarrow$ dest.
  - The information is interpreted at the intermediate nodes.
  - The min-cut can be obtained very efficiently with low cost.

A Coding-Based Perspective

- $G' = (V', E')$ [Chou *et al.* 03] [Li *et al.* 03].
- Unit edge-capacity (1 packet per use). Allow parallel edges.
- Each session has a *generation* size $n = 32$–100.
- Minimize the generation flushing time D to maximize transmission rate.
- A link e of rate $r_e^{(i)} \Rightarrow \lfloor r_e^{(i)} D \rfloor$ parallel edges.
- In the integral rate graph G', the min-cut/max-flow value for session i is roughly n.

A Coding-Based Perspective

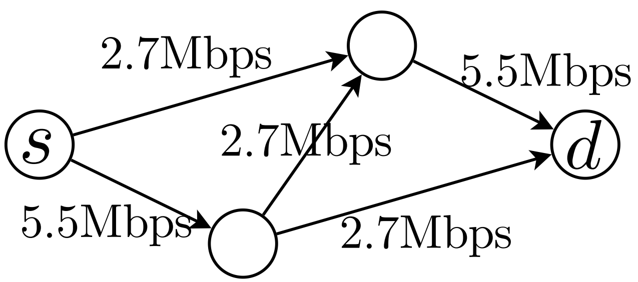
- $G' = (V', E')$ [Chou *et al.* 03] [Li *et al.* 03].
- Unit edge-capacity (1 packet per use). Allow parallel edges.
- Each session has a *generation* size $n = 32$–100.
- Minimize the generation flushing time D to maximize transmission rate.
- A link e of rate $r_e^{(i)} \Rightarrow \lfloor r_e^{(i)} D \rfloor$ parallel edges.
- In the integral rate graph G', the min-cut/max-flow value for session i is roughly n.

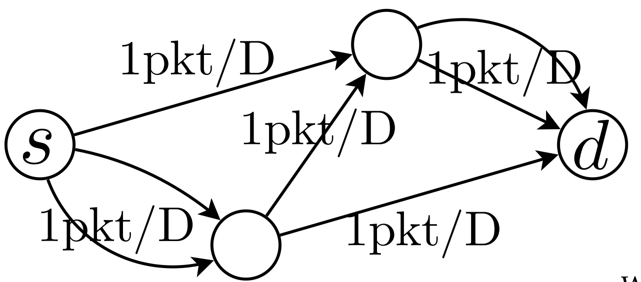


A Coding-Based Perspective

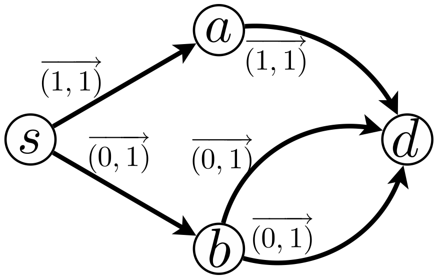
- $G' = (V', E')$ [Chou *et al.* 03] [Li *et al.* 03].
- Unit edge-capacity (1 packet per use). Allow parallel edges.
- Each session has a *generation* size $n = 32$–100.
- Minimize the generation flushing time D to maximize transmission rate.
- A link e of rate $r_e^{(i)} \Rightarrow \lfloor r_e^{(i)} D \rfloor$ parallel edges.
- In the integral rate graph G', the min-cut/max-flow value for session i is roughly n.

1 packet = 1kB, D ≈2.97ms





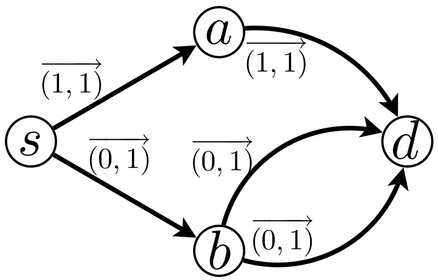
A Fast Min-Cut Algorithm



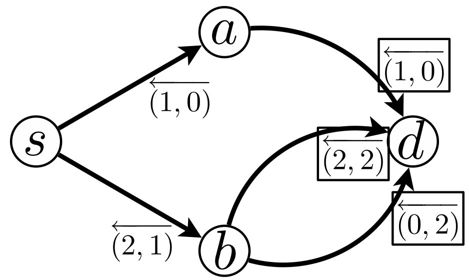
in GF(3)

- Step 0: Run (random) network coding.

A Fast Min-Cut Algorithm



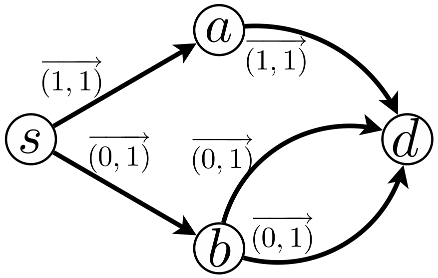
in GF(3)



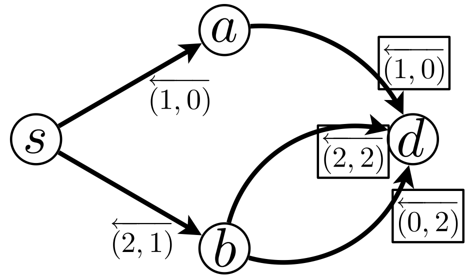
- Step 0: Run (random) network coding.
- Step 1: Send **generalized coded ACK q** that acknowledge the linear independence of the incoming packets **m** and satisfy.

$$\mathbf{q}^{\mathrm{T}}\mathbf{m} = \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} \right) \begin{pmatrix} [1 \quad 1] \\ [0 \quad 1] \\ [0 \quad 1] \end{pmatrix} = \mathbf{I} \triangleq \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

A Fast Min-Cut Algorithm



in GF(3)



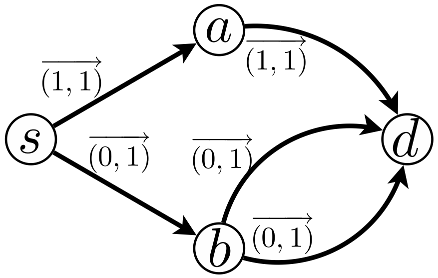
- Step 0: Run (random) network coding.
- Step 1: Send **generalized coded ACK q** that acknowledge the linear independence of the incoming packets **m** and satisfy.

$$\mathbf{q}^{\mathrm{T}}\mathbf{m} = \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} \right) \begin{pmatrix} [1 & 1] \\ [0 & 1] \\ [0 & 1] \end{pmatrix} = \mathbf{I} \triangleq \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

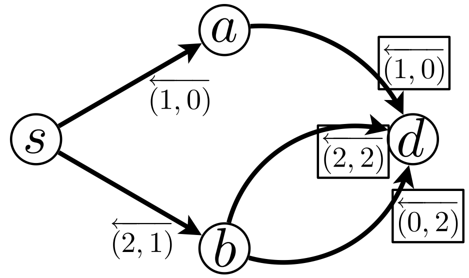
- Step 2: Propagate $\mathbf{q}$ back to s with the same local kernel Γ_v.

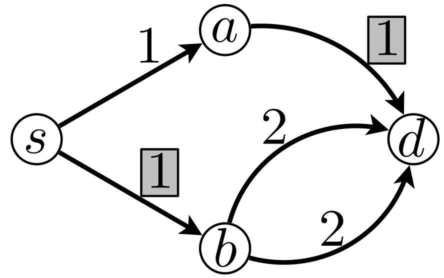
$$\mathbf{m}_{\text{downstream}} = \Gamma_v \mathbf{m}_{\text{upstream}}, \quad \mathbf{q}_{\text{upstream}} = \Gamma_v^{\mathrm{T}} \mathbf{q}_{\text{downstream}}.$$

A Fast Min-Cut Algorithm



in GF(3)





- Step 0: Run (random) network coding.
- Step 1: Send **generalized coded ACK q** that acknowledge the linear independence of the incoming packets **m** and satisfy.

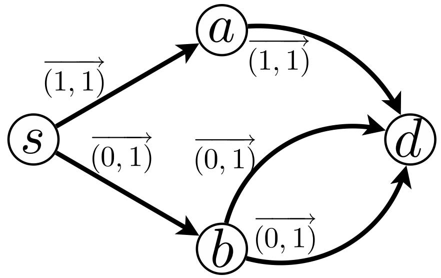
$$\mathbf{q}^{\mathrm{T}}\mathbf{m} = \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} \right) \begin{pmatrix} [1 & 1] \\ [0 & 1] \\ [0 & 1] \end{pmatrix} = \mathbf{I} \triangleq \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

- Step 2: Propagate $\mathbf{q}$ back to s with the same local kernel Γ_v.

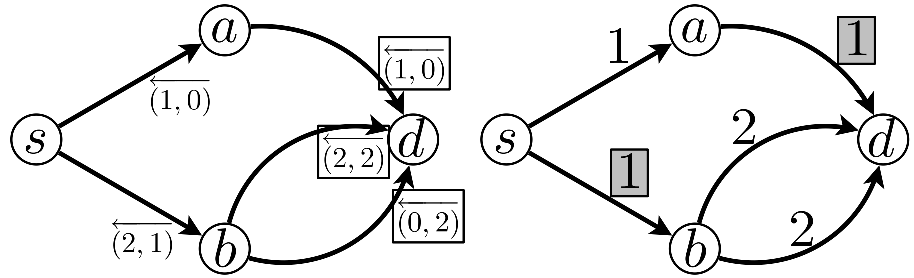
$$\mathbf{m}_{\text{downstream}} = \Gamma_v \mathbf{m}_{\text{upstream}}, \quad \mathbf{q}_{\text{upstream}} = \Gamma_v^{\mathrm{T}} \mathbf{q}_{\text{downstream}}.$$

- Step 3: Check whether $\mathbf{m} \cdot \mathbf{q} = 1$ and return the first such cut.

A Fast Min-Cut Algorithm



in GF(3)



- Step 0: Run (random) network coding.
- Step 1: Send **generalized coded ACK q** that acknowledge the linear independence of the incoming packets $\mathbf{m}$ and satisfy.

$$\left(\begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}\begin{bmatrix}0\\2\end{bmatrix}\right)\begin{pmatrix}[1\quad 1]\\ [0\quad 1]\end{pmatrix} \quad \begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$$

Fully distributed, each edge makes it own decision.

- Step 2: Propagate $\mathbf{q}$ back to s with the same local kernel Γ_v.

$$\mathbf{m}_{\text{downstream}} = \Gamma_v \mathbf{m}_{\text{upstream}}, \qquad \mathbf{q}_{\text{upstream}} = \Gamma_v^T \mathbf{q}_{\text{downstream}}.$$

- Step 3: Check whether $\mathbf{m}\cdot\mathbf{q} = 1$ and return the first such cut.

Correctness & Complexity

Proposition 1 *The new coding-based min-cut algorithm stops after $2|V|$ exchanges of the network coding vectors.*

Correctness & Complexity

Proposition 1 *The new coding-based min-cut algorithm stops after $2|V|$ exchanges of the network coding vectors.*

Proposition 2 *Assume $n \geq \mathrm{mcMF}(s_i, d_i)$ (when generation flushing time D is properly chosen). Using $\mathsf{GF}(2^b)$, we have*

$$\mathsf{Prob}(\textit{a min-cut is found}) \geq (1 + |E'|)(1 - 2^{-b})^{|E'|} - |E'|,$$

Correctness & Complexity

Proposition 1 *The new coding-based min-cut algorithm stops after $2|V|$ exchanges of the network coding vectors.*

Proposition 2 *Assume $n \geq \mathrm{mcMF}(s_i, d_i)$ (when generation flushing time D is properly chosen). Using $\mathsf{GF}(2^b)$, we have*

$$\mathsf{Prob}(\textit{a min-cut is found}) \geq (1 + |E'|)(1 - 2^{-b})^{|E'|} - |E'|,$$

- Calibration with the length of the control messages
  - The proposed scheme: each control message is $\mathcal{O}(n^2)$ bytes. The overall time: $\mathcal{O}\left((n^2 t_{\mathrm{tran}} + t_{\mathrm{prop}})|V|\right)$.
  - Push-&-relabel [Goldberg *et al.* 88]: each control message is $\mathcal{O}(1)$ byte. The overall time: $\mathcal{O}\left((t_{\mathrm{tran}} + t_{\mathrm{prop}})|V|^2\right)$.

Correctness & Complexity

Proposition 1 *The new coding-based min-cut algorithm stops after $2|V|$ exchanges of the network coding vectors.*

Proposition 2 *Assume $n \geq \mathrm{mcMF}(s_i, d_i)$ (when generation flushing time D is properly chosen). Using $\mathsf{GF}(2^b)$, we have*

$$\mathsf{Prob}(\textit{a min-cut is found}) \geq (1 + |E'|)(1 - 2^{-b})^{|E'|} - |E'|,$$

- Calibration with the length of the control messages
  - The proposed scheme: each control message is $\mathcal{O}(n^2)$ bytes. The overall time: $\mathcal{O}\left((n^2 t_{\text{tran}} + t_{\text{prop}})|V|\right)$.
  - Push-&-relabel [Goldberg *et al.* 88]: each control message is $\mathcal{O}(1)$ byte. The overall time: $\mathcal{O}\left((t_{\text{tran}} + t_{\text{prop}})|V|^2\right)$.
  - 10Mbps, $n = 100$, 1000km. $n^2 t_{\text{tran}} = 8\text{ms}$. $t_{\text{prop}} \approx 3.3\text{ms}$. (Ignore the queuing and processing delays)

The Randomized Subgradient Method

- The Prob(a min-cut is found) is large, but not one.
- **Proposition 3** *For any* $\epsilon > 0$, *there exist* $2^b, D > 0$ *such that the combination of the subgradient method and the randomized min-cut algorithm will satisfy:*

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} E(F^* - F(\mathbf{r}(t))) \leq \epsilon.$$

The Randomized Subgradient Method

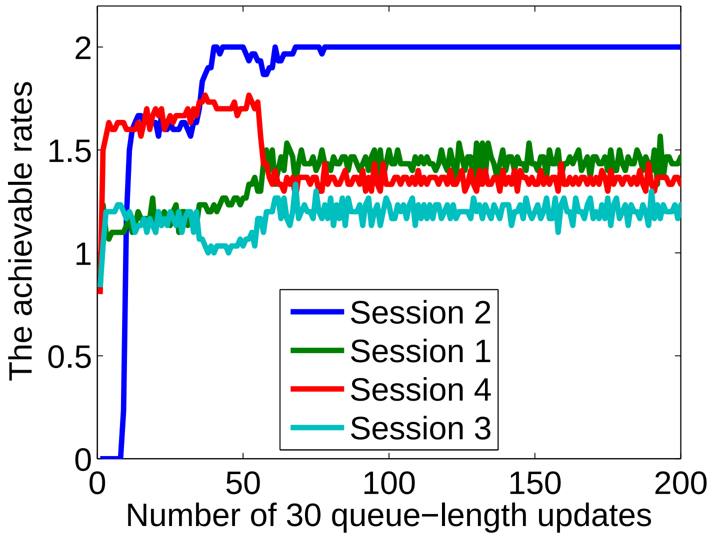
- The $\text{Prob}(\text{a min-cut is found})$ is large, but not one.
- **Proposition 3** *For any $\epsilon > 0$, there exist $2^b, D > 0$ such that the combination of the subgradient method and the randomized min-cut algorithm will satisfy:*

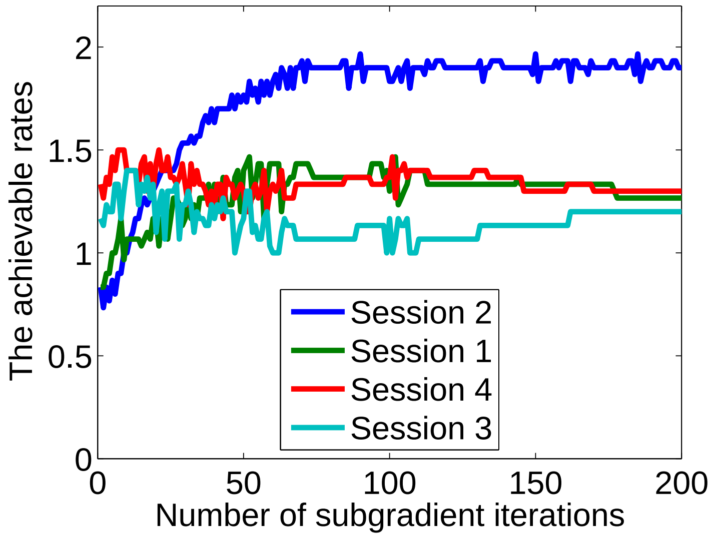
$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} E(F^* - F(\mathbf{r}(t))) \leq \epsilon.$$

- The control messages are part of the traffic: Low overhead and fast reaction to network changes.

Numerical Experiments

- A 4×4 Network. Back-pressure versus Coding solutions

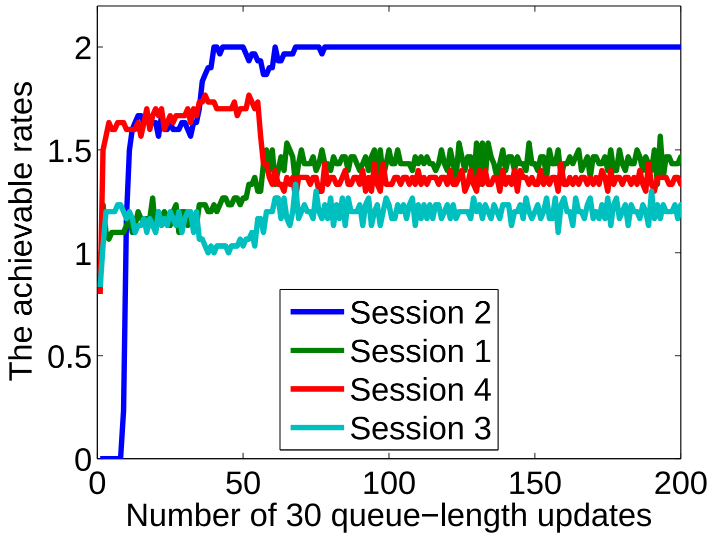


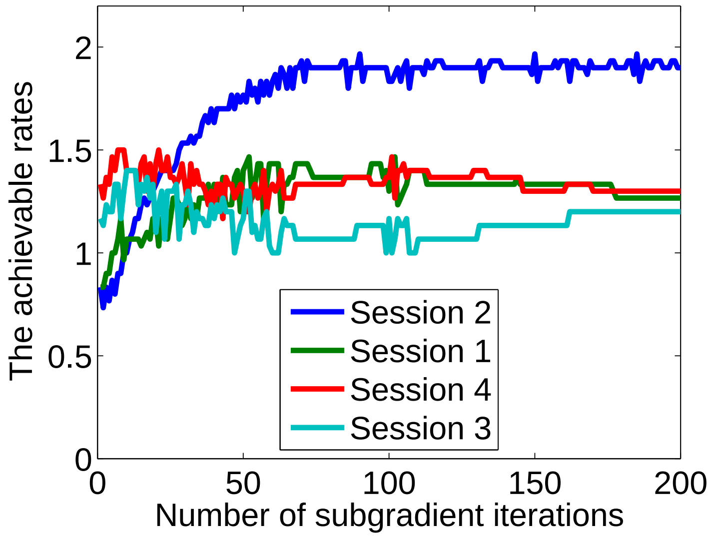


Back Pressure	Coding
Instant feedback after every packet	Batch feedback after every generation

Numerical Experiments

- A 4 $\times$ 4 Network. Back-pressure versus Coding solutions

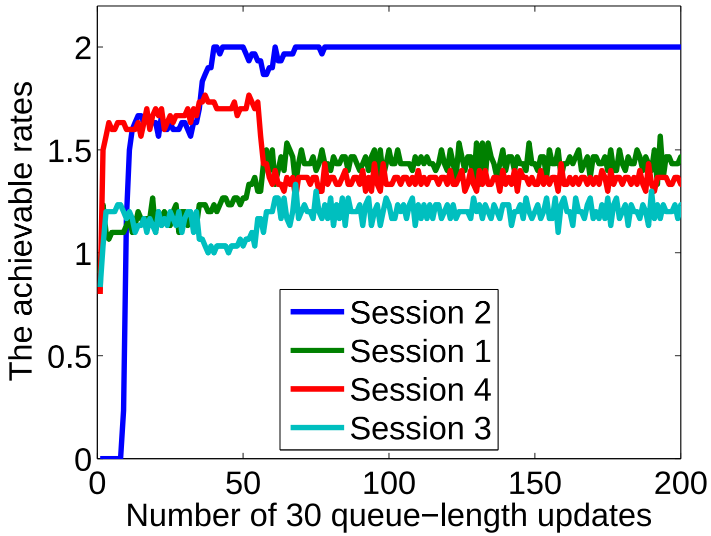


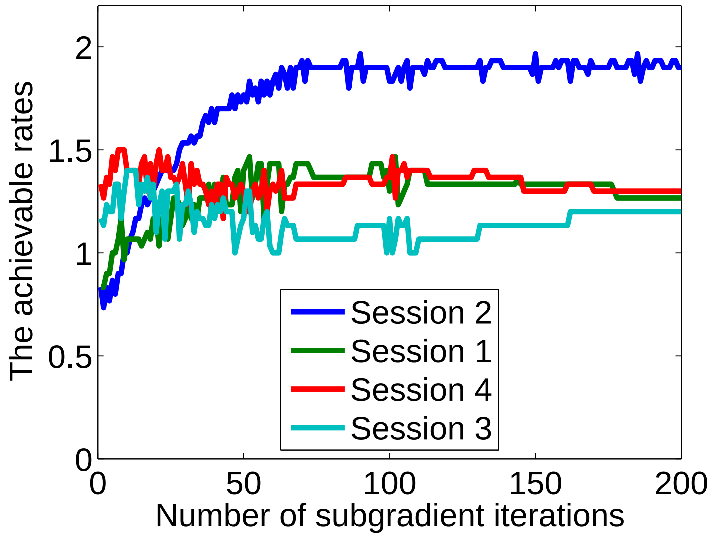


Back Pressure	Coding
Instant feedback after every packet	Batch feedback after every generation
Queue/delay dynamics	No queues. Buffers only packets of the same generation. Fixed delay.

Numerical Experiments

- A 4×4 Network. Back-pressure versus Coding solutions

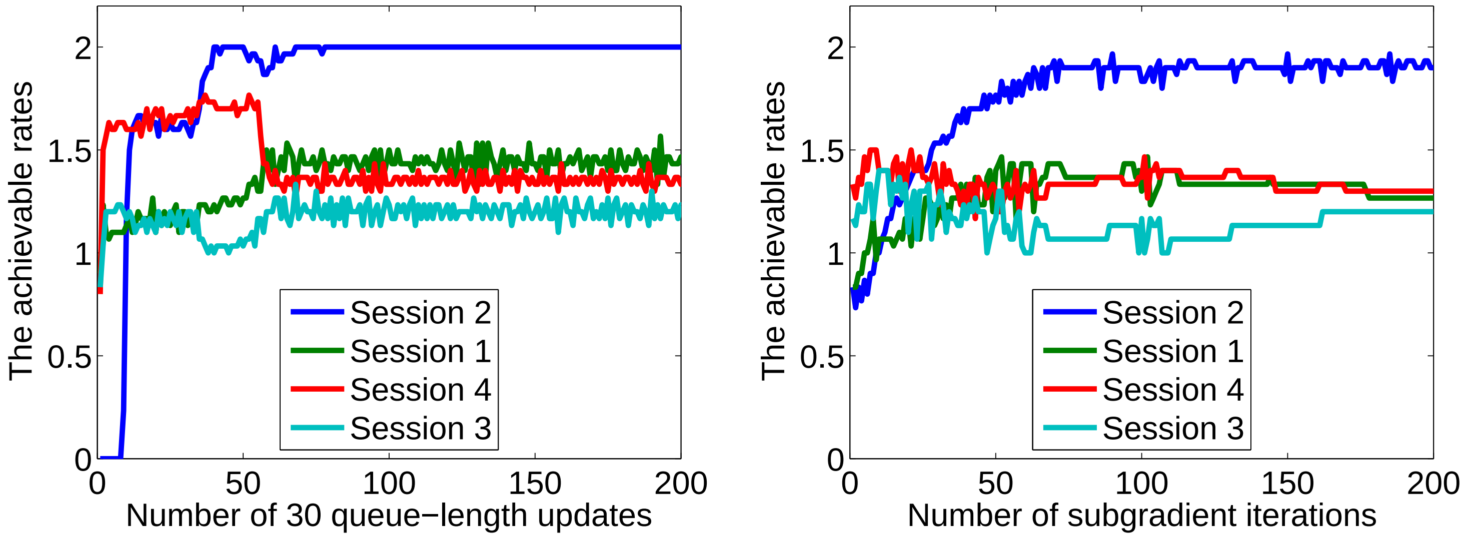




Back Pressure	Coding
Instant feedback after every packet	Batch feedback after every generation
Queue/delay dynamics	No queues. Buffers only packets of the same generation. Fixed delay.
Queue build-up	Quick Start

Numerical Experiments

- A 4×4 Network. Back-pressure versus Coding solutions



Back Pressure (dual)	Coding (primal)
Instant feedback after every packet	Batch feedback after every generation
Queue/delay dynamics	No queues. Buffers only packets of the same generation. Fixed delay.
Queue build-up	Quick Start

Quick Start of Network Coding

- The network coding primal solution.
  - New sessions instantly grab the fair share the edge capacity.

$$\forall e, \quad r_e^{(\text{new})} \leftarrow \frac{1}{\text{no. sessions that use } e} c_e$$

$$r_e^{(\text{existing})} \leftarrow (1 - \frac{1}{\text{no. sessions that use } e}) r_e^{(\text{existing})}$$

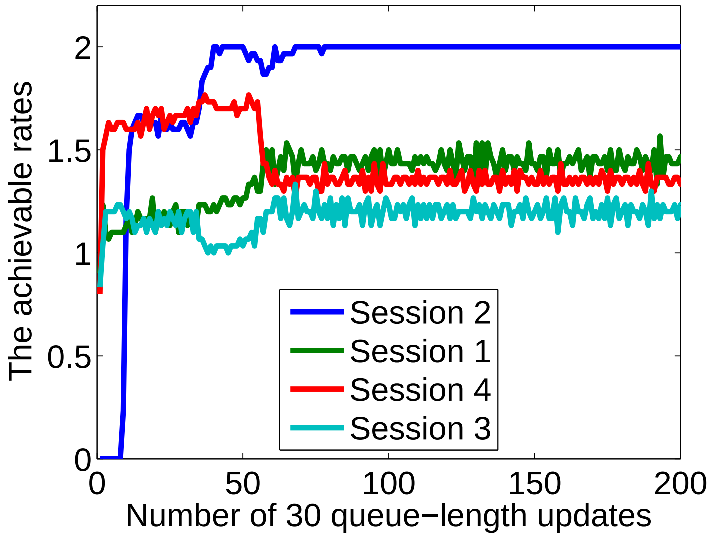
  - There is no need to wait for queue build-up.
  - The rates are optimized thereafter from the new starting point.

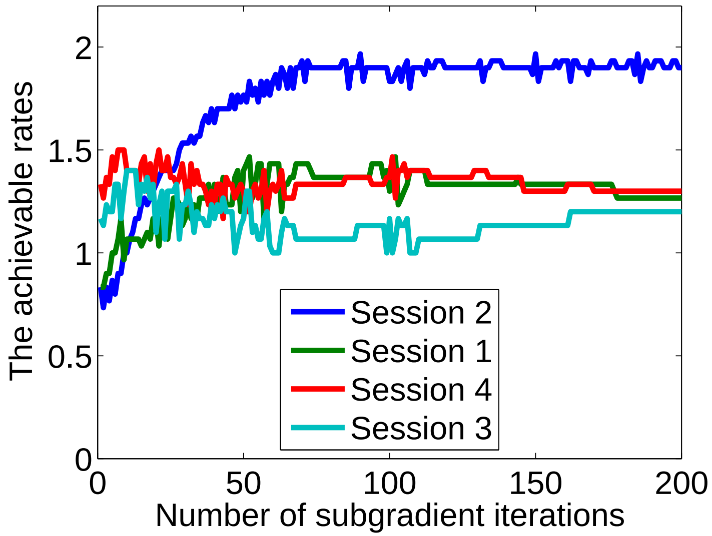
Conclusion

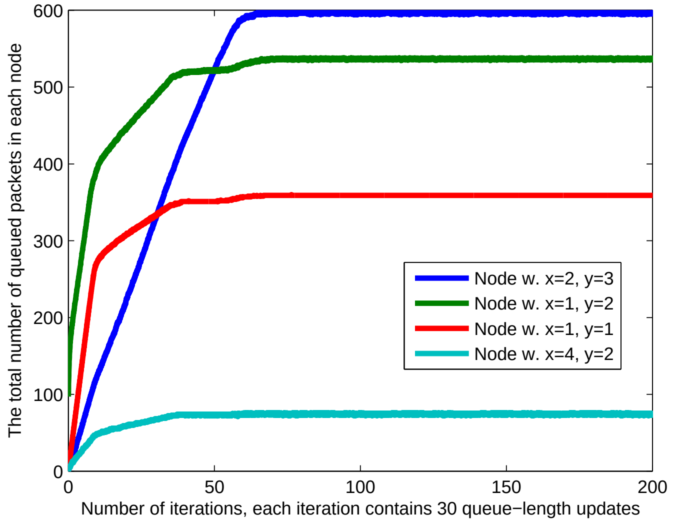
- We present a new primal approach that fully takes advantage of the network coded traffic.
  - The benefits of primal approaches. Not relying on queue lengths.
  - Low computation complexity, low overhead.
- The convergence time outperforms the existing results for many practical scenarios. $\mathcal{O}\left((n^2 t_{\text{tran}} + t_{\text{prop}})|V|\right)$ vs. $\mathcal{O}\left((t_{\text{tran}} + t_{\text{prop}})|V|^2\right)$
- Network coding enables the quick start feature with fairness.
- Future direction: Applications to wireless networks, for which (coding-based) batch feedback is more favorable than (back-pressure) packet-by-packet feedback.

Delay Comparison

- A 4×4 Network. Back-pressure versus Coding solutions







A fixed generation size $n = 60$. Each nodes only needs to buffer at most two consecutive generations. The delay is fixed to D, the time between flushing generations.