# Methods for Efficient Network Coding

**Petar Maymounkov**, Nick Harvey and Desmond Lun

**MIT**

44th Annual Allerton Conference · Sep, 2006

# Roadmap

| | 2-Party Erasure Channel<br>$s \bullet\!\!\longrightarrow\!\!\bullet\, t$ | Network Erasure Channel<br>$s \bullet\!\!\longrightarrow\!\!\bullet\!\!\longrightarrow\!\!\bullet\!\!\longrightarrow\!\!\bullet\, t$ |
|---|---|---|
| **Classical (Shannon) Codes**<br>**Quadratic Time**<br>$\epsilon = \exp(-k)$ | **Dense Linear Coding**<br>**[Elias'65]** | **Dense Linear Codes**<br>**[Lun, et al.'05, Pakzad'05]** |
| **Practical (Fountain) Codes**<br>**Linear Time,** $\epsilon = \mathrm{poly}(1/k)$<br>**Rateless/Adaptive** | **LP, Raptor, Online**<br>**[Luby'02, Shokrollahi'03, Maymounkov'02]** | |

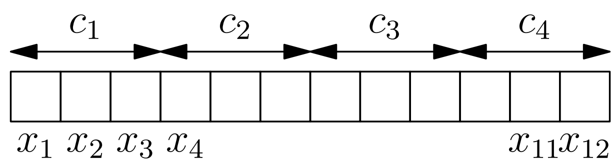
All codes on picture are capacity-approaching.

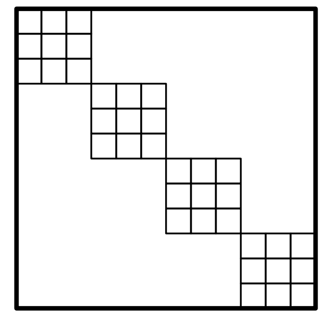
# Roadmap

| | 2-Party Erasure Channel $s \rightarrow t$ | Network Erasure Channel $s \rightarrow \cdot \rightarrow \cdot \rightarrow t$ |
|---|---|---|
| **Classical (Shannon) Codes**<br>**Quadratic Time**<br>$\epsilon = \exp(-k)$ | **Dense Linear Coding**<br>**[Elias'65]** | **Dense Linear Codes**<br>**[Lun, et al.'05, Pakzad'05]**<br>**Tight asymptotics**<br>**For many networks** |
| **Practical (Fountain) Codes**<br>**Linear Time,** $\epsilon = \text{poly}(1/k)$<br>**Rateless/Adaptive** | **LP, Raptor, Online**<br>**[Luby'02, Shokrollahi'03,**<br>**Maymounkov'02]** | **Chunked Codes!** |

All codes on picture are capacity-approaching.

# Chunked Codes



- Logically partition input message into disjoint "chunks" of contiguous symbols
- **Encode at source node**
  - i. Randomly and uniformly choose a chunk
  - ii. Output symbol = dense linear combination of input symbols from this chunk
- **Encode at intermediate node**
  - i. Randomly and uniformly choose a chunk
  - ii. Output symbol = dense linear combination of so-far received output symbols pertaining to this chunk
- **Decode**
  - i. Solve a block diagonal matrix

## Network Model = Schedule of Successful Transmissions

- The **outcome** of most network models can be described as a simple schedule
- Schedule is a set of these:

  *"One packet was sent at time $t_0$ from node $v$ and was received at time $t_1 \geq t_0$ at node $w$."*

  (Schedule does not mention lost packets!)
- Formally, a schedule is specified by:

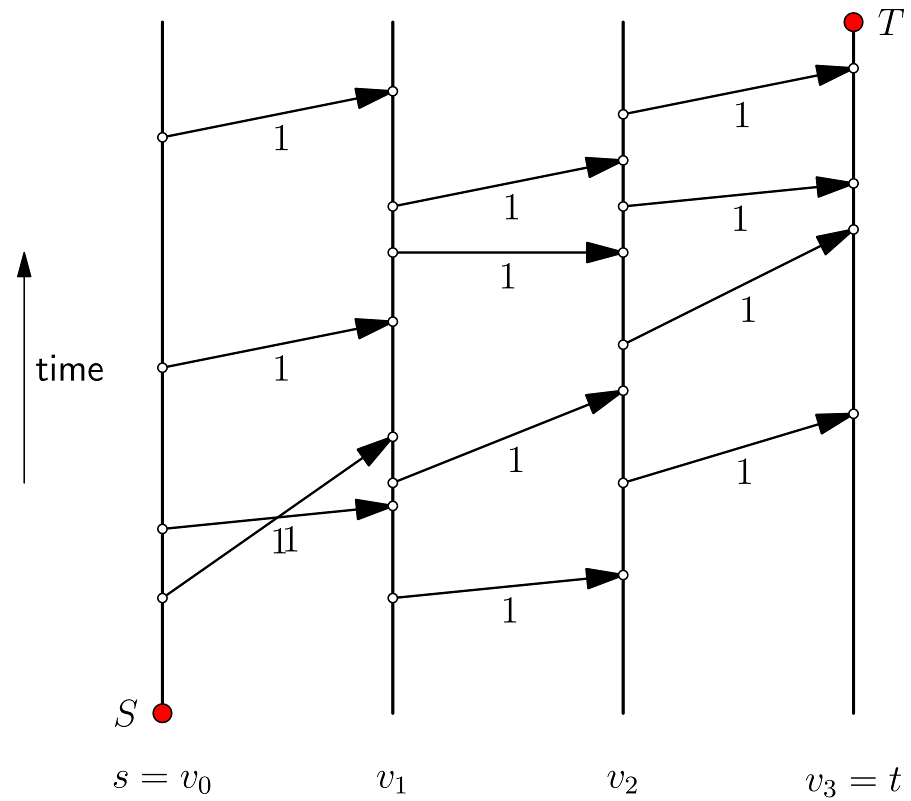
  i. Set of participating nodes $V$,

  ii. Source $s \in V$, destination $t \in V$, and

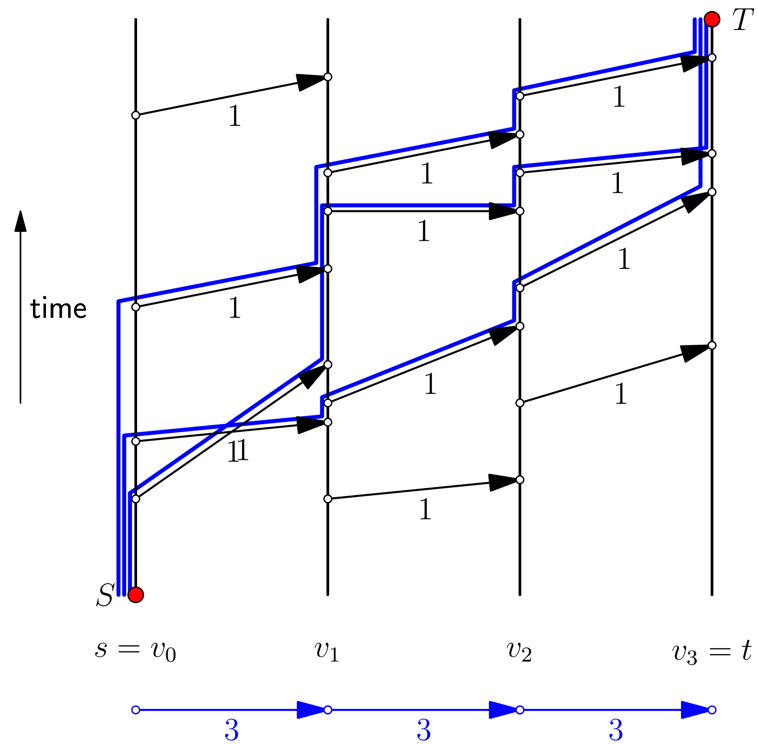
  iii. Set of successful transmissions:

$$\{(v, t_0, w, t_1) | v, w \in V \wedge t_1 \geq t_0\}$$

# Schedule Illustrated

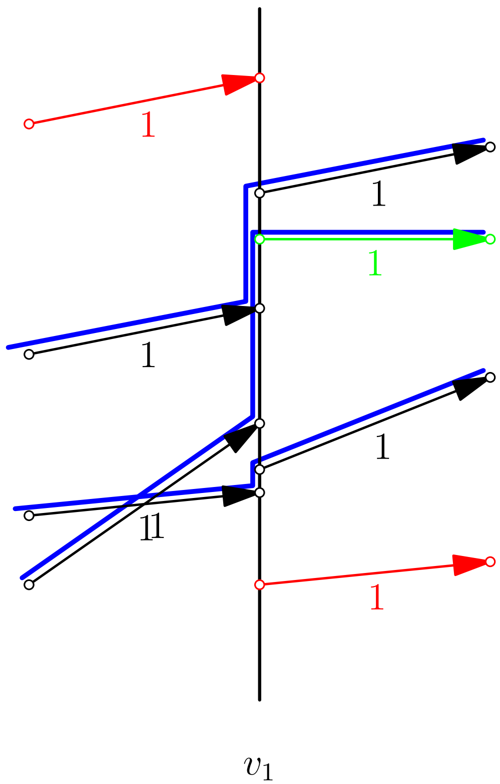


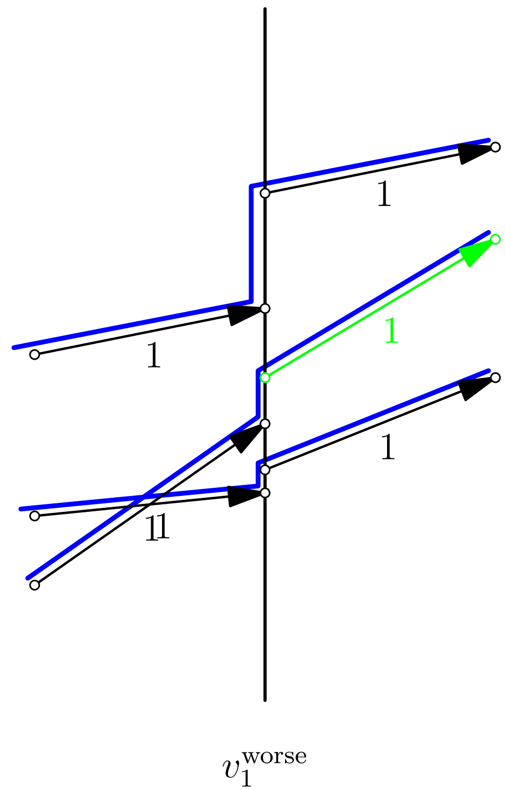
## Adversarial Schedule



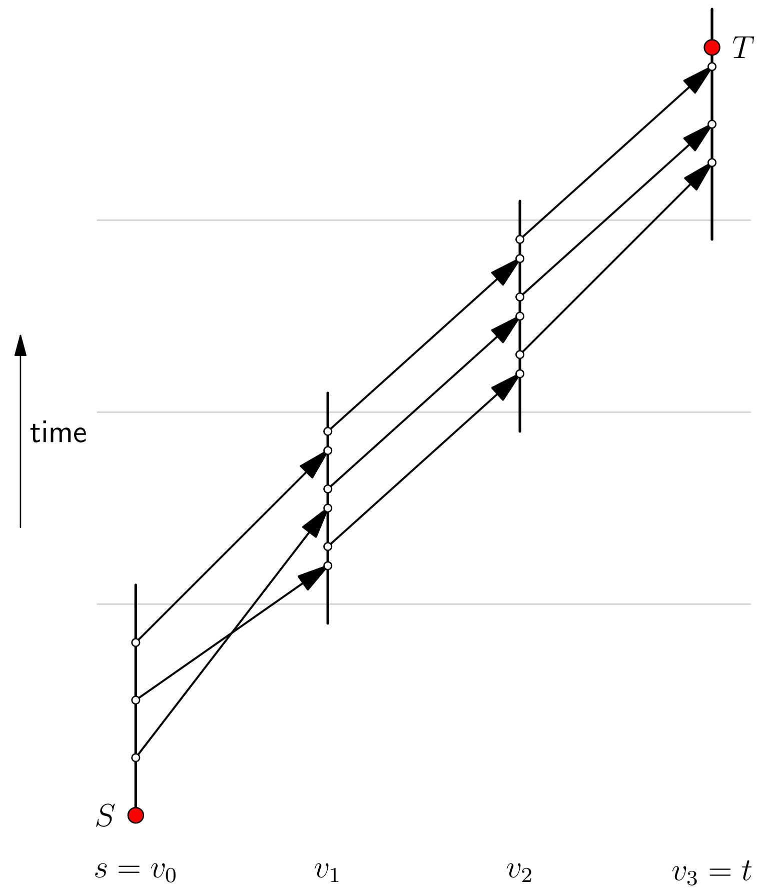

- **Observation** – Maximum flow is upper bound on capacity
- **Idea**
  - Analyze equivalence class (called Adversarial Schedule) of all schedules with same maximum flow
  - Sufficient to analyze (asymptotically) worst-case schedule in equivalence class

# Worst-case Arrival/Departure Reduction

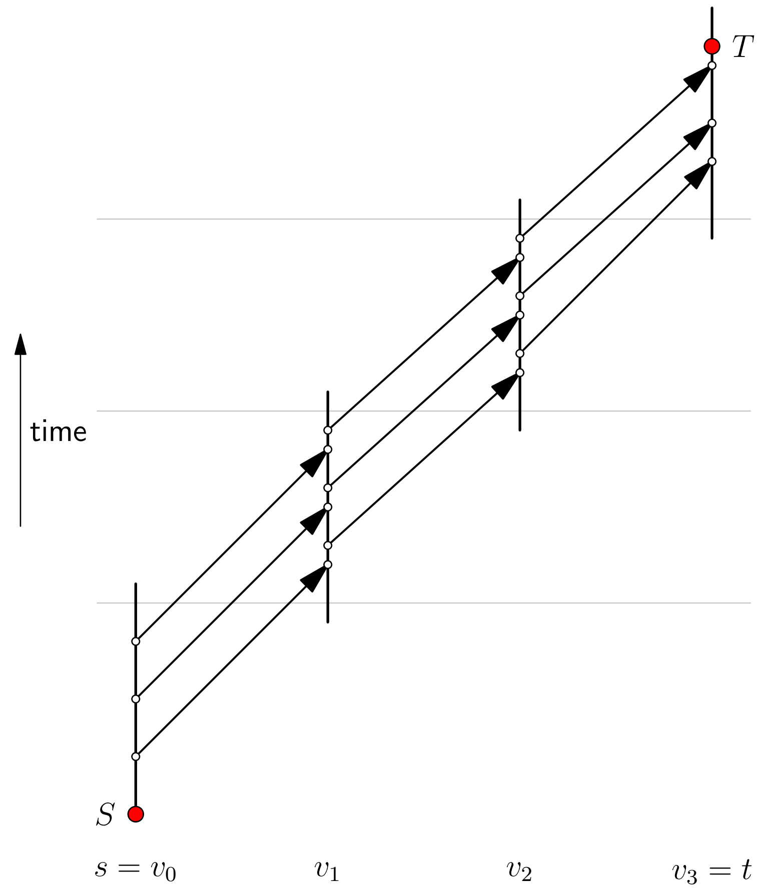


# Schedule with Worst-case Arrivals/Departures

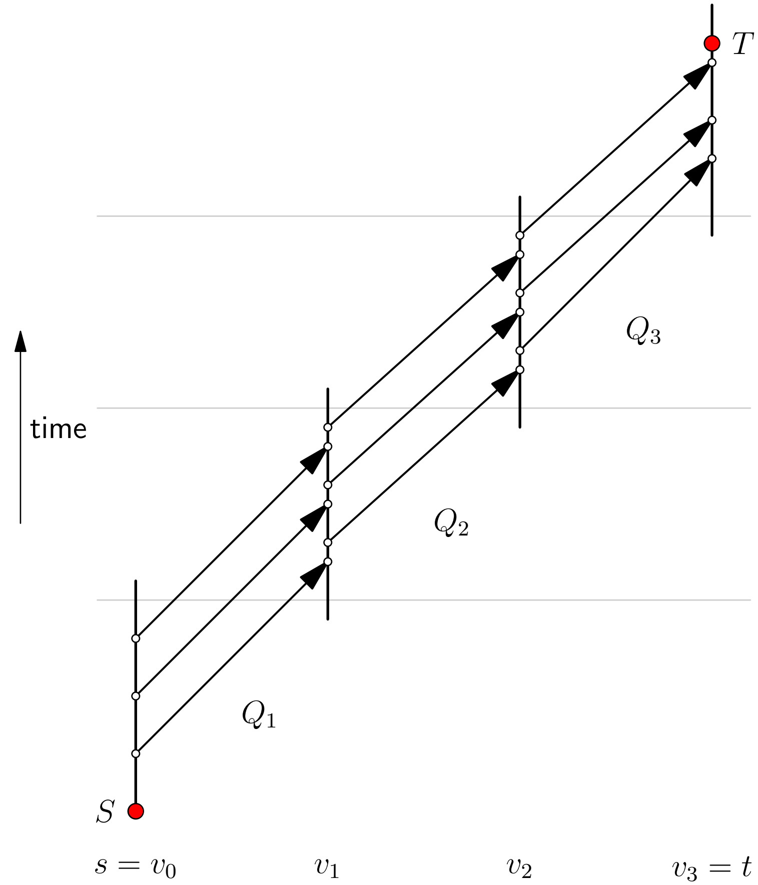


# Canonical Worst-case Schedule



## Information Representation

- Number of input symbols is $k$, maximum flow is $n \geq k$
- Each transmitted symbol is represented by a $k$-dimensional payload vector over $\mathbb{F}_2$
- Payload of an entire network edge is $Q_i \in M_{k,n}(\mathbb{F}_2)$

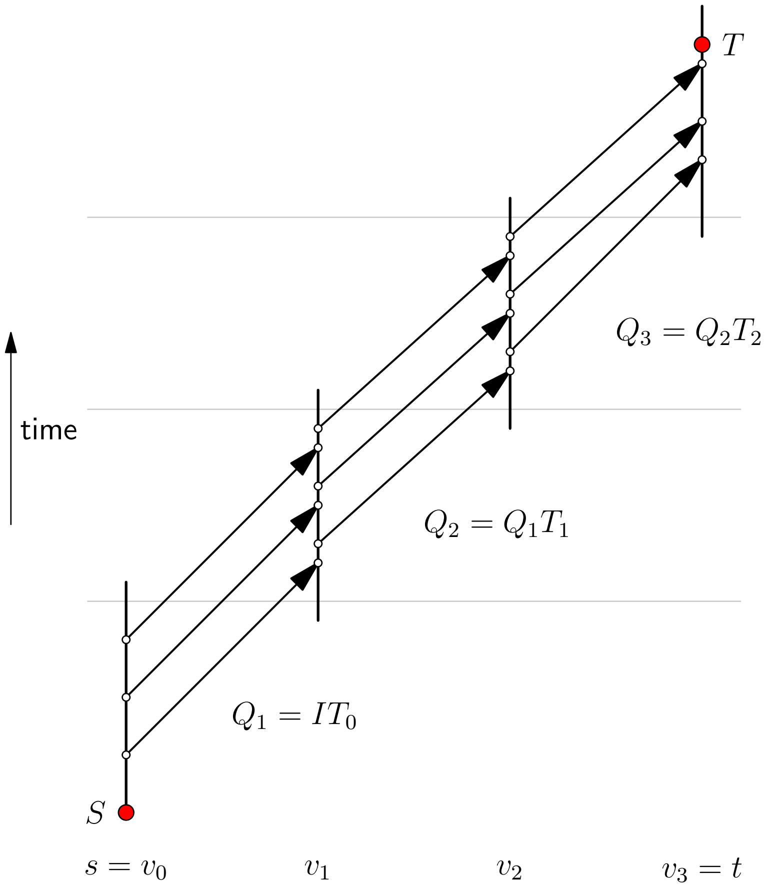


# Information Flow

$$Q_{i+1} = Q_i T_i \quad \text{where} \quad T_i \in M_{n,n}(\mathbb{F}_2)$$

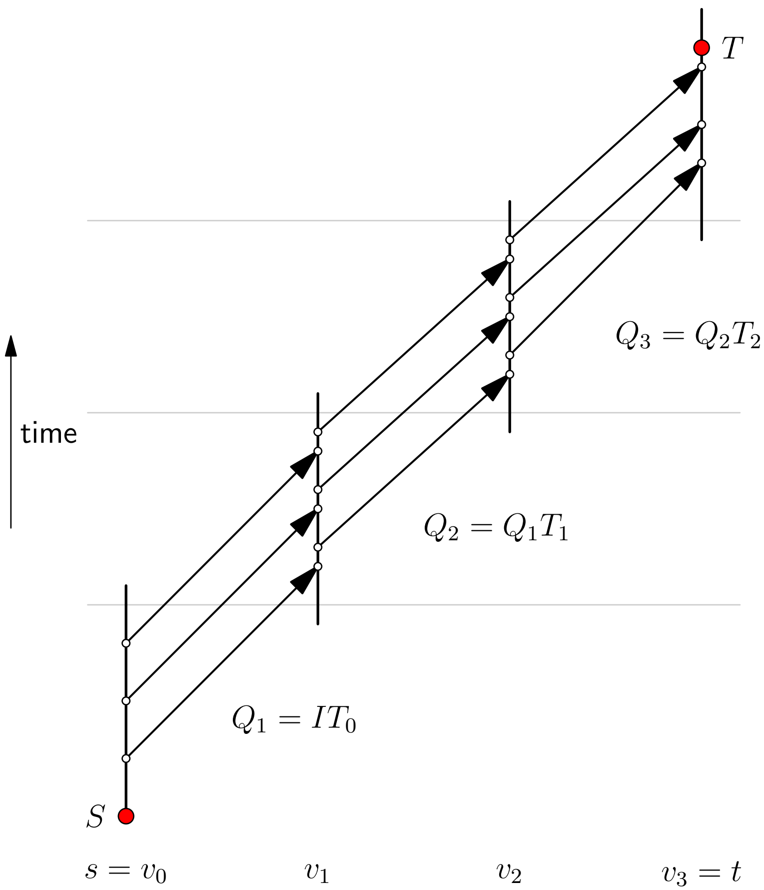
$T_i$ represents the encoding process at node $v_i$

$T_0$ is dense, all other $T_i$ are dense above the diagonal



# Information Quantity

- Rank of last edge $Q_3$ determines quantity of information delivered
- Rank of $Q_3$ directly related (almost $=$) to column randomness of $Q_3$
- **$Q_1$ is fully random. How much randomness leaks on the way to $Q_3$?**



## Analysis Results

- **Dense Coding Theorem:** When

$$n = k + O\Big(l(\log k + \log l + \log 1/\epsilon)\Big),$$

where $l$ is length of line network, the input message can be delivered in full

- **Corollary:** As long as $l = o(\sqrt{k}/\log k)$ the code is capacity-achieving

***

- **To analyze Chunked Codes:**

  i. Calculate the magnitude of subflow that **each** chunk receives
     – Some interesting techniques here

  ii. Apply the result for dense codes

  iii. Add a precode to bring computational costs down to linear

## Summary

- **Chunked Code vs. Raptor Code**
  - Adaptive network coding vs. Adaptive 2-party coding
  - Coding operations per input symbol: $O\left(\frac{\ln 1/\lambda}{\lambda^4}\right)$ vs. $O(\ln 1/\lambda)$, where $\lambda$ is overhead
- **Adversarial Schedules are the right level of generality**
  - Dense and Chunked Codes perform just as well on Adversarial Schedules as they do on the Poission-Arrivals/Departures-with-Random-Erasures Channel
- **Open Problems**
  - Cleverer "chunking" produces codes with 400MBit/sec decoders
  - Open problem: Find precode for these codes