Algorithms for Radio Networks

Network Coding

University of Freiburg
Technical Faculty
Computer Networks and Telematics
Christian Schindelhauer
Data Flows in Networks

- **Motivation**
  - Optimize data flow from source to target

- **Definition:**
  - (Single-commodity) maximum flow problem
  - Given
    - a graph $G=(V,E)$
    - a capacity function $w:E \rightarrow \mathbb{R}^+$,
    - source set $S$ and target set $T$
  - Find a maximum flow from $S$ to $T$

- **A flow is a function** $f : E \rightarrow \mathbb{R}_0^+$ such that
  - for all $e \in E$: $f(e) \leq w(e)$
  - for all $e \notin E$: $f(e) = 0$
  - for all $u,v \in V$: $f(u,v) \geq 0$

\[ \forall u \in V \setminus (S \cup T) \]
\[ \sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v) \]

- **Maximize flow**
\[ \sum_{u \in S} \sum_{v \in V} f(u,v) \]
Data Flows in Networks
Data Flows in Networks
Data Flows in Networks
Data Flows in Networks
Computation of the Maximum Flow

- Every natural pipe system solves the minimum/maximum flow problem

Algorithms

- Linear Programming
  - for real numbers
  - the flow is described by equations of a linear optimization problem
  - Simplex algorithm (or Ellipsoid method) can solve any linear equation system

- Ford-Fulkerson
  - also for integers
  - as long as open paths exist, increase the flow on these paths
    - open path: path which increases the flow

- Edmonds-Karp
  - special case of Ford-Fulkerson
  - use BFS (breadth first search) to find open paths
Ford-Fulkerson

- Find a path from the source node to the target node
  - where the capacity is not fully utilized
  - or which reduces the existing flow
- Compute the maximum flow on this augmenting path
  - by the minimum of the flow that can be added on all paths
- Add the flow on the path to the existing flow
- Repeat this step until no flow can be added anymore
Edmunds-Karp

- Search path for Ford-Fulkerson algorithm
- Choose the shortest augmenting path
  - Computation by breadth-first-search
- leads to run-time $O(|V| |E|^2)$
  - whereas Ford-Fulkerson could have exponential run-time
Example
Example
Example
Example
Minimum Cut in Networks

- **Motivation**
  - Find bottleneck in networks

- **Definition**
  - Min Cut problem
  - Given
    - graph $G=(V,E)$
    - capacity function $w: E \to \mathbb{R}^+$
    - sources $S$ and targets $T$
  - Find minimum cut between $S$ and $T$

- **A cut $C$ is a set of edges**
  - such that every path from a node of $S$ to a node of $T$, contains an edge of $C$

- **The size of a cut is**
  \[
  \sum_{e \in C} w(e)
  \]
Min-Cut-Max-Flow Theorem

- **Theorem**
  - The minimum cut equals the maximum flow

- **Algorithms for minimum cut**
  - can be obtained from the maximum flow algorithms
Multi-Commodity Flow Problem

- **Motivation**
  - theoretical model for point to point communication

- **Definition**
  - Multi-commodity flow problem
  - given
    - a graph $G=(V,E)$
    - a capacity function $w: E \rightarrow \mathbb{R}^+$
    - commodities $K_1, \ldots, K_k$:
      * $K_i=(s_i,t_i,d_i)$ with
      * $s_i$: source node
      * $t_i$: target node
      * $d_i$: demand

- **Find flows** $f_1, f_2, \ldots, f_k$ for all commodities such that
  - capacities
  - flow property
    $$\forall v \notin \{s_i, t_i\} : \sum_{u \in V} f_i(u,v) = \sum_{u \in V} f_i(v,u)$$
  - demand
    $$\sum_{v \in V} f_i(s_i,v) = \sum_{u \in V} f_i(u,t_i) = d_i$$
Solving the Multi-Commodity Flow Problem

- Multi-Commodity Flow Problem
- Optimize
  - sum of all flows or
  - maximize the worst ratio between commodity and the demand
- Problem can be solved in polynomial time
  - for real numbers
  - using linear programming
Complexity of the Multi Commodity Flow Problem

- Problem is NP-complete
  - for integers
    - e.g. packets
  - even for two commodities
    - Shai, Itai, Even, 1976
- Polynomial solution
  - with respect to the number of paths between sources and targets
- Approximation
  - good central and distributed approximation algorithms exist
    (polylogarithmic approximation factor)
- Weaker forms of the Min-Cut-Max-Flow-Theorems exist
Network Coding

- R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung

Example
- Bits x and y are to be transferred
- Each edge carries only a bit
- If bits are transferred as is
  - then both x and y cannot be received either on the left or right side
Network Coding

- **Example**
  - Bits x and y are to be transferred
  - Each edge carries only a bit
  - If bits are transferred as is
    - then both x and y cannot be received either on the left or right side

\[
\begin{align*}
  & x \\
  & \downarrow \\
  & y \\
  & \downarrow \\
  & y \\
  & \downarrow \\
  & y \\
  & \downarrow \\
  & y
\end{align*}
\]
Network Coding

- Solution
  - Transfer Xor A+B on the middle edge
Network Coding and Flow

- Theorem [Ahlswede et al.]
  - For each graph there exists a network code such that each sink can receive as many bits as the maximum flow allows for each sink.
Linear Codes for Network Coding

› Koetter, Médard
  • Beyond Routing: An Algebraic Approach to Network Coding

› Task
  • Efficiently compute the network code

› Solution
  • Linear codes can always solve network coding

› Practical Network Coding
  • With high probability even random linear combinations suffice
Application Areas

› Satellite Communication
  • Preliminary work was published there

› Peer-to-Peer networks
  • Better information flow better than previous protocols
  • But too inefficient to displace prevalent protocols, e.g. Bittorrent

› WLAN
  • Xor in the Air, COPE
    - Simple network code improves flow

› Ad-Hoc Networks, Wireless Sensor Networks, ...
Coding and Decoding

- Original message: $x_1, x_2, ..., x_m$
- Coding packet: $y_1, y_2, ..., y_m$
- Random variable $r_{ij}$

Then:

$$\begin{pmatrix}
  r_{11} & \ldots & r_{1m} \\
  \vdots & \ddots & \vdots \\
  r_{m1} & \ldots & r_{mm}
\end{pmatrix}
\begin{pmatrix}
x_1 \\
\vdots \\
x_m
\end{pmatrix}
= 
\begin{pmatrix}
y_1 \\
\vdots \\
y_m
\end{pmatrix}$$

- If the matrix $(r_{ij})$ is invertible

$$\begin{pmatrix}
x_1 \\
\vdots \\
x_m
\end{pmatrix}
= 
\begin{pmatrix}
r_{11} & \ldots & r_{1m} \\
\vdots & \ddots & \vdots \\
r_{m1} & \ldots & r_{mm}
\end{pmatrix}^{-1}
\begin{pmatrix}
y_1 \\
\vdots \\
y_m
\end{pmatrix}$$
Inverse of a Random Matrix

- **Theorem**
  - If the numbers of an $m \times m$ Matrix are chosen randomly from a finite field with $b$ elements, then the matrix is invertable with probability of at least

  $$1 - \sum_{i=1}^{m} \frac{1}{b^i}$$

- **Idea: Consider Galois-Field GF[2^k]**
  - Computation is efficient
  - Binary representation of data straight-forward
Galois Field

- $\text{GF}(2^w) = \text{finite field with } 2^w \text{ elements}$
  - elements are binary strings of length $w$
  - $0 = 0^w$ neutral element of addition
  - $1 = 0^{w-1}1$ neutral element of multiplikation
- $u + v = \text{bit-wise Xor of strings}$
  - z.B. $0101 + 1100 = 1001$
- $a \cdot b = \text{product of polynomials modulo a given irreducible polynomial and modulo 2}$
  - i.e. $(a_{w-1} \ldots a_1 a_0) (b_{w-1} \ldots b_1 b_0) =$
    
    $$((a_0 + a_1 x + \ldots + a_{w-1} x^{w-1}) (b_0 + b_1 x + \ldots + b_{w-1} x^{w-1}) \mod q(x)) \mod 2)$$
Example: $\mathbb{GF}(2^2)$

$q(x) = x^2 + x + 1$

<table>
<thead>
<tr>
<th>Generator of GF(4)</th>
<th>Polynomial in GF(4)</th>
<th>Binary Representation in GF(4)</th>
<th>Decimal Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>$x^0$</td>
<td>1</td>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td>$x^1$</td>
<td>$x$</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>$x^2$</td>
<td>$x+1$</td>
<td>11</td>
<td>3</td>
</tr>
</tbody>
</table>
Example: $\text{GF}(2^2)$

<table>
<thead>
<tr>
<th>$+$</th>
<th>0 = 00</th>
<th>1 = 01</th>
<th>2 = 10</th>
<th>3 = 11</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 = 00</td>
<td>00</td>
<td>01</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>1 = 01</td>
<td>01</td>
<td>00</td>
<td>11</td>
<td>10</td>
</tr>
<tr>
<td>2 = 10</td>
<td>10</td>
<td>11</td>
<td>00</td>
<td>01</td>
</tr>
<tr>
<td>3 = 11</td>
<td>11</td>
<td>10</td>
<td>01</td>
<td>00</td>
</tr>
</tbody>
</table>
Example: GF($2^2$)

$q(x) = x^2 + x + 1$

<table>
<thead>
<tr>
<th>*</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 = 0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1 = 1</td>
<td>0</td>
<td>1</td>
<td>x</td>
<td>x^2</td>
</tr>
<tr>
<td>2 = x</td>
<td>0</td>
<td>x</td>
<td>x^2</td>
<td>1</td>
</tr>
<tr>
<td>3 = x^2</td>
<td>0</td>
<td>x^2</td>
<td>1</td>
<td>x</td>
</tr>
</tbody>
</table>
Irreducible Polynomial

- Irreducible polynomial cannot be factorized
  - Irreducible polynomial $x^2+1 = (x+1)^2 \mod 2$
- Irreducible polynomials
  - $w=2$: $x^2+x+1$
  - $w=4$: $x^4+x+1$
  - $w=8$: $x^8+x^4+x^3+x^2+1$
  - $w=16$: $x^{16}+x^{12}+x^3+x+1$
  - $w=32$: $x^{32}+x^{22}+x^2+x+1$
  - $w=64$: $x^{64}+x^4+x^3+x+1$
Fast Multiplication

- **Power law**
  - Consider \{2^0, 2^1, 2^2, ...\}
  - = \{x^0, x^1, x^2, x^3, ...\}
  - = \exp(0), \exp(1), ...
  - \exp(x+y) = \exp(x) \exp(y)
- **Inverse function:** \log(\exp(x)) = x
  - \log(x \cdot y) = \log(x) + \log(y)
  - \( x \cdot y = \exp(\log(x) + \log(y)) \)
  - Caution: in the exponent standard addition
- **Tables store exponential function and logarithm**
Example: GF(16)

\[ q(x) = x^4 + x + 1 \]

<table>
<thead>
<tr>
<th>x</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>exp(x)</td>
<td>1</td>
<td>x</td>
<td>x^2</td>
<td>x^3</td>
<td>1+x</td>
<td>x+x^2</td>
<td>x^2 + x^3</td>
<td>1+x</td>
<td>x^3</td>
<td>1+x</td>
<td>x+x^3</td>
<td>x</td>
<td>1+x</td>
<td>x^2 + x^3</td>
<td>1+x^2</td>
<td>1+x^3</td>
</tr>
<tr>
<td>exp(x)</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>3</td>
<td>6</td>
<td>12</td>
<td>11</td>
<td>5</td>
<td>10</td>
<td>7</td>
<td>14</td>
<td>15</td>
<td>13</td>
<td>9</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>x</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>log(x)</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>5</td>
<td>10</td>
<td>3</td>
<td>14</td>
<td>9</td>
<td>7</td>
<td>6</td>
<td>13</td>
<td>11</td>
<td>12</td>
</tr>
</tbody>
</table>

- \( 5 \cdot 12 = \exp(\log(5)+\log(12)) = \exp(8+6) = \exp(14) = 9 \)
- \( 7 \cdot 9 = \exp(\log(7)+\log(9)) = \exp(10+14) = \exp(24) = \exp(24-15) = \exp(9) = 10 \)
Special Case GF[2]

- **Network Coding in GF[2]**
  - Boolean Algebra
    - \( x + y = x \text{ XOR } y \)
    - \( x \cdot y = x \text{ AND } y \)

- **Example**
  - Xor in the Air
  - Multicasting in Ad-Hoc Networks

- **Disadvantage**
  - Full potential of network coding is unused

- **Advantage**
  - Transparent, intuitiv and very efficient
Multicasting in Ad Hoc Networks

- Wu, Chou, Sun-Yuan,
  - Minimum-Energy Multicast in Mobile Ad hoc Networks using Network Coding, 2006

- Multicast
  - Distribute message from one node to a given set of nodes

- Cost measure
  - Each one-hop broadcast costs an energy unit

![Diagram of Multicasting in Ad Hoc Networks]

Source

Target A

Target B

$x_1$

$x_2$

$x_1$

$x_2$
Traditionally,
- it costs 5 energy units for a multicast message.
Traditionally,

- it costs 5 energy units for a multicast message
Example

- **Network coding**
  - 9 energy units for 2 messages
  - Average of 4.5

- **Without network coding**
  - 5 units for one multicast message
Multicasting in Ad Hoc Networks

- Solution of the minimal energy multicasting problem without network coding is NP-hard
  - Less than constant factor approximation is NP-hard
  - Requires calculation of the discrete Steiner tree
Condition for Network Coding

- Messages allow flow of the size of the desired number of messages
  - from the sources to each individual sink
- If such flows are guaranteed, network coding can be applied
- Size of the flows describe energy consumption

![Network Coding Diagram](image-url)
Computational Complexity

- **Algorithm**
  - Collect all available link information
  - Formulate as linear program
  - Approximation of the solution

- With the help of network coding, the maximum throughput can be approximated arbitrarily well in polynomial time
Example Demand

Wu, Chou, Sun-Yuan, Minimum-Energy Multicast in Mobile Ad hoc Networks using Network Coding, 2006
Example Multicasting with minimal Energy

Wu, Chou, Sun-Yuan, Minimum-Energy Multicast in Mobile Ad hoc Networks using Network Coding, 2006
Multicasting with Network Coding

Wu, Chou, Sun-Yuan, Minimum-Energy Multicast in Mobile Ad hoc Networks using Network Coding, 2006
Discussion

› Options
  • Energy model can be customized

› Limitations
  • Network coding is not described
  • Central algorithm
  • Any change in the communication requires recalculation
Xors in the Air

- Katti, Hu, Katabi, Médard, Crowcroft
  - XORs in the Air: Practical Wireless Network Coding

- Problem
  - Maximize throughput in ad-hoc network
  - Multihop messages cause interference

- Solution
  - Uses only XORs of multiple messages
  - Local, opportunistic algorithm
Xors in the Air

- **Problem**
  - Multihop messages cause interferences

- **Example**
  - Traditional: 4 messages to send
    - a message from A to B
    - and a message from B to A

```
<table>
<thead>
<tr>
<th>Round</th>
<th>A</th>
<th>Relay</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>•</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>•</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>•</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>•</td>
<td>•</td>
<td></td>
</tr>
</tbody>
</table>
```

- Message 1
- Message 2
Xors in the Air

- **Problem**
  - Multihop messages cause interferences

- **Example**
  - Traditional: 4 messages to send
    - a message from A to B
    - and a message from B to A
  - Network Coding
    - 3 messages suffice

<table>
<thead>
<tr>
<th>Round</th>
<th>A</th>
<th>Relay</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$x_1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>$x_2$</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>$x_1 \oplus x_2$</td>
<td>$x_1 \oplus x_2$</td>
<td></td>
</tr>
</tbody>
</table>
Coding Opportunistically

COPE

› Consider of multiple communication paths
  • Opportunistic coding of messages by Xor

› Utilization of the broadcast medium
  • listening to the channel
  • all (even foreign) messages are buffered
  • buffered messages are used for decoding

› Context messages
  • announcement of level of knowledge
  • neighbors can generate code adapted to the receiver‘s knowledge

› Guess the level of knowledge of neighbors
Opportunistic Coding

Known messages:
- A: P_3, P_4
- C: P_1, P_4
- D: P_1, P_3

Messages to be sent:
- B: P_1, P_2, P_3, P_4

Receivers:
- A
- C
- C
- D
Opportunistic Coding

Known messages:
- P1
- P3
- P4

Messages to be sent:
- P1
- P2
- P3
- P4

Receivers:
- A
- C
- C
- D

Known messages:
- A
- C
- C
- D
Opportunistic Coding

- Messages to be send:
P_1, P_2, P_3, P_4

- Receivers:
A, C, C, D

- Known messages:
P_1, P_2, P_3, P_4

- Diagram:

A: P_1, P_3
B: P_1 + P_3
C: P_1, P_4
D: P_1, P_3

- Opportunistic Coding:

- Send P_1 + P_3 to receive P_1 and P_3.

- At receiver A, P_1 + P_3 is received.
- At receiver C, P_1 + P_3 is received.
- At receiver D, P_1 + P_3 is received.

- Example:

P_1 and P_3 are known messages at different receivers.
Opportunistic Coding

messages to be send
receivers

known messages

P_1 + P_3 + P_4

A

B

C

D

known messages

P_1

P_3

P_4

known messages

P_1

P_3

P_4

P_1

P_3

P_4
### Coding Gain

<table>
<thead>
<tr>
<th>Topology</th>
<th>Coding Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-chain</td>
<td>1,333...</td>
</tr>
<tr>
<td>X</td>
<td>1,333...</td>
</tr>
<tr>
<td>Cross</td>
<td>1,666...</td>
</tr>
<tr>
<td>Infinite Chain</td>
<td>2</td>
</tr>
<tr>
<td>Infinite Wheel</td>
<td>2</td>
</tr>
</tbody>
</table>

**Diagram:**
- 3-chain
- Cross
- Infinite Chain
- Infinite Wheel
Summary Network Coding

Figure 12—COPE can provide a several-fold (3-4x) increase in the throughput of wireless Ad hoc networks. Results are for UDP flows with randomly picked source-destination pairs, Poisson arrivals, and heavy-tail size distribution.

Wu, Chou, Sun-Yuan, Minimum-Energy Multicast in Mobile Ad hoc Networks using Network Coding, 2006
Network Coding

- **Benefit**
  - Network throughput can be increased
    - COPE
  - Reduction of energy consumption
  - Higher robustness, small error rate
  - Applications in peer-to-peer networks, wireless sensor networks

- **Problems**
  - complex encoding
  - sometimes high computational cost
  - difficult organization
Algorithms for Radio Networks

Network Coding

University of Freiburg
Technical Faculty
Computer Networks and Telematics
Christian Schindelhauer