Robust Communications in CubeSat Clusters using Network Coding

Second Interplanetary CubeSat Workshop, Ithaca, NY
May 28, 29, 2013

By Gokul Bhat, Paul Muri, Obulapathi Challa and Dr. Janise McNair
Outline

• Motivation
• Related Work
• System Architecture
• Definitions
• Transmission scenarios
• Network Coded Transmission
• Experiments
• Conclusion and Future Work
Limitations of CubeSats

• Resource constraints
• Low speed links
• High packet erasures
• No fault tolerance
• Limited computing power
Next-generation communication protocol concepts for future nanosatellite constellations

Patrick Romano, Manuela Unterberger, Otto Koudelka

Institute of Communication Networks and Satellite Communications, Graz University of Technology, Inffeldgasse 12/1, 8010 Graz, Austria
Related Work

• Exploiting link dynamics in LEO-to-Ground station communications
  (J. Palmer, M. Caffrey, 2009)

  Adaptive data rate selection based on RSS

• Increasing downlink capacity with store and forward routing in CubeSat clusters
  (T. Kortiza, J. Bellardo, 2010)

  “Data mule” CubeSat enables faster downlink

• Robust communication link for CubeSats using custom-made space qualified equipments
  (V. Miguel, et. Al, 2011)

  Communication protocol with FEC, Reed Solomon coding & ARQ

Our Contribution: Simple packet combining technique with basic MAC protocol easily developed for CubeSats
Inter-Session Network coding

Wireless transmission without any coding

Wireless transmission with inter-session network coding
Intra-session Network Coding

- Packets belonging to the same stream/session are combined
- Source forms packets by taking random linear combinations of a block of bits in its sending buffer
- Linear combination proven to give max flow from source to each receiving node in a multicast network
- Sink, with a high probability, receives linearly independent coefficients for decoding
- Maximizes throughput, robust to packet losses, robust to link failures
Network Coding Operation

Extract algebraic coefficients, $\alpha_k$ from Finite field dimension 16;

Encoding

Original data packets

At the CubeSat

- P1
- P2

Transmitted packets

- $(\alpha_k 1 P1 + \alpha_k 2 P2)$
- $(\alpha_k 3 P1 + \alpha_k 4 P2)$
- $(\alpha_k 5 P1 + \alpha_k 6 P2)$

Decoding by Gaussian Elimination;
Produce rank $\geq 2$ matrix

Received packets

At the Ground Station

- $(\alpha_k 1 P1 + \alpha_k 2 P2)$
- $(\alpha_k 3 P1 + \alpha_k 4 P2)$
- $(\alpha_k 5 P1 + \alpha_k 6 P2)$

Decoded packets

- P1
- P2
Architecture
Definitions

• Master and Slave Nodes
• Ground Stations
• Central Server
• High-speed & Low-speed links
Initial packet chunk loss scenario

TIME ELAPSED: 00
Initial packet chunk loss scenario

TIME ELAPSED: 02
Initial packet chunk loss scenario
Initial packet chunk loss scenario

TIME ELAPSED: 06
Initial packet chunk loss scenario

TIME ELAPSED: 08
Tail-end packet chunk loss scenario

TIME ELAPSED: 00
Tail-end packet chunk loss scenario

TIME ELAPSED: 02
Tail-end packet chunk loss scenario

TIME ELAPSED: 04
Tail-end packet chunk loss scenario

TIME ELAPSED: 06
Tail-end packet chunk loss scenario

TIME ELAPSED: 08

NACK_2
Simple Retransmission

TIME ELAPSED: 10
Redundant Transmission Scenario

TIME ELAPSED: 06
Network Coded Transmission

TIME ELAPSED: 06
Experimental setup & Conditions

- Intel Pentium Processor
- Raspberry Pi processor
- Simulate network coding for varying cluster sizes
- Packet erasure rate: 10%
Raspberry Pi

- Useful modules
- Simple, cheap and easily available
- Light-weight and open source Linux operating system
- Quick implementation of new modules possible
- ARM-based processor that models CubeSats accurately
- Minimal resources consumed
### Comparison

<table>
<thead>
<tr>
<th>Features</th>
<th>Beagle board</th>
<th>Raspberry Pi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>TI Cortex A8 @ 1 GHz</td>
<td>Broadcom ARM11 @ 700 MHz</td>
</tr>
<tr>
<td>RAM</td>
<td>512 MB DDR</td>
<td>512 MB SDRAM</td>
</tr>
<tr>
<td>Storage</td>
<td>2 GB eMMC w/ microSD slot</td>
<td>SD slot</td>
</tr>
<tr>
<td>Video</td>
<td>Micro HDMI</td>
<td>HDMI (1080p)</td>
</tr>
<tr>
<td>Audio</td>
<td>Via HDMI</td>
<td>Via HDMI Audio jack</td>
</tr>
<tr>
<td>Peripherals</td>
<td>SPI, I2C, SPI, CAN, Timers, LCD, MMC, Analog, 65 GPIO</td>
<td>12 GPIO, USART, SPI, I2C (P1 and P5), CSI (camera serial interface) and DSI (display serial interface)</td>
</tr>
<tr>
<td>Assistance</td>
<td>Full documentation available for both</td>
<td></td>
</tr>
</tbody>
</table>

**BBB is faster due to Cortex Arch., DDR**

**BBB has lower video resolution;**

**Summary:** R-Pi will be useful where missions involve video and audio recording while most other applications can work with BBB.
Results

Block Size vs Encoding Time

Comparison of downlink time with and without network coding
Current Drawn

Average Current Draw for Idle and Encoding States (mA)

<table>
<thead>
<tr>
<th>State of CPU</th>
<th>Current Draw (mA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle</td>
<td>335</td>
</tr>
<tr>
<td>While Running Encoding</td>
<td>390</td>
</tr>
</tbody>
</table>

State of CPU
Conclusion

• First evaluation of transmission scenarios for CubeSat cluster system
• Network coding for tail-end packet chunks greatly improves latency of the transmission
• Encoding time for network coding is insignificant compared to the total transmission time
• Negligible increase in current drawn while encoding
• Future work includes using network coding for real-time applications
Acknowledgments

This work was partially supported by a NASA Office of the Chief Technologist’s Space Technology Research Fellowship (NSTRF) grant #NNX11AM73H.