Routing in DTNs

ICTP Trieste

Keith Scott

keithlscott@gmail.com

Routing vs. Forwarding

Routing is the process of providing connectivity information to the nodes and deciding to which next hop a bundle should be forwarded, and when

Forwarding is the process of applying the routing information to a bundle and actually transmitting it

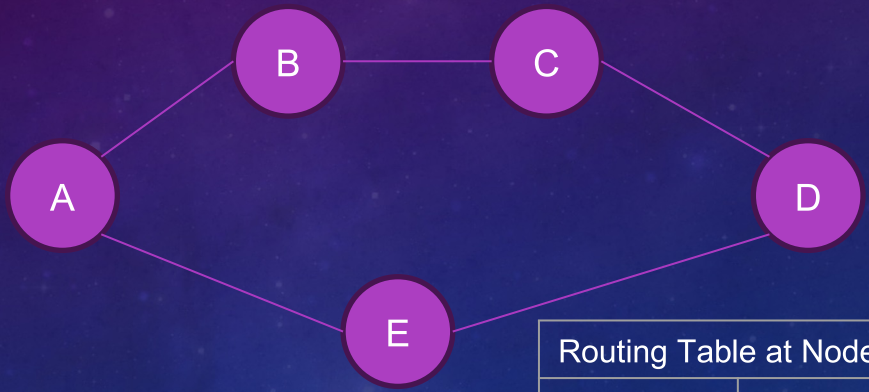
Quick Review: Shortest-Path Routing

- Routing Protocol (on the wire):
  - When links change, the nodes on either end flood that change to all nodes they can reach
  - Result: every node can build a graph representing the entire (sub)network that it can reach
  - This information (the graph information) is the Forwarding Information Base (FIB)
- Computing the Routing Table (Routing Information Base, RIB):
  - Each node, separately, uses the information in its FIB to compute a table that maps destinations to next hops
- Forwarding:
  - To forward a packet, a node just consults the routing table

Shortest Path Routing Example

E.g.: OSPF, IS-IS, EIGRP

Routing Table at Node A	
Dest	Next Hop
B	B
C	B
D	E
E	E



Routing Metric: ‘Shortest’ Path

Routing Table at Node E	
Dest	Next Hop
A	A
B	A
C	D
D	D

What Makes Routing in DTN's Different? Time and planning

Internet routing protocols don't have a notion of time:

- Wired protocols (limit: everything is fixed)
  - Discover the network topology
  - Assume that stays static (at least for a while)
  - Route on the discovered topology
- MANET protocols (limit: everything is random)
  - Proactive: try to look like wired protocols
  - Reactive: send packets and let them thrash around until they get to their destinations
  - Flooding: send LOTS of packets and let them thrash around….

In DTNs, we generally have time-variant topologies and may have foreknowledge of how those topologies will evolve with time

- DTN routing can plan for known, scheduled changes in topology and route accordingly
- DTN routing (almost surely) should be resilient in cases where reality doesn't match the plan

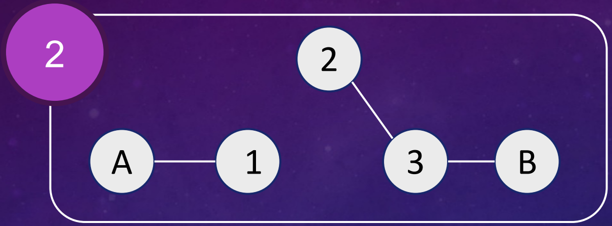
Things that Make DTN Routing Difficult

The network may have intermittent connectivity (no contemporaneous paths).

The network may have long / variable delays.

Various nodes / links that make up the network may have limited storage, power, bandwidth.

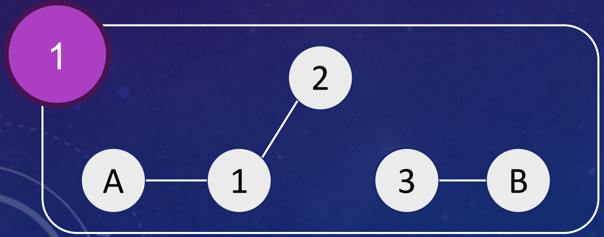
Consider pinging from A to B Where the Topology Cycles as Follows

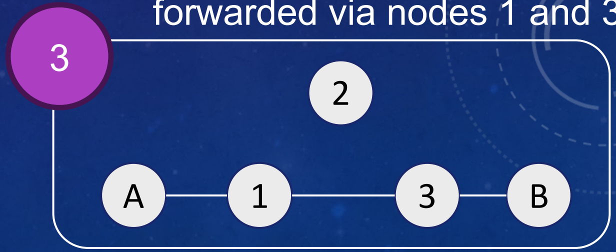


Packets at 2 delivered to B; replies are routed to node 3 to wait; packets from A to B are queued at node 1

Packets from A to B are routed to node 2 to wait.

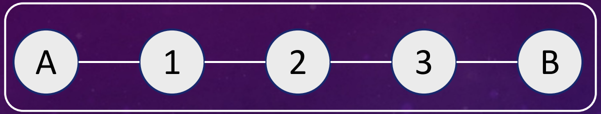
Packets from 3 are delivered; packets between A and B are forwarded via nodes 1 and 3.





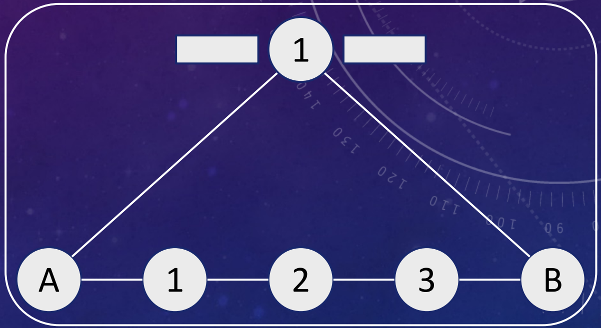
Routing is (generally) a bi-directional problem (lab)

Two Paths, Unequal Over Time



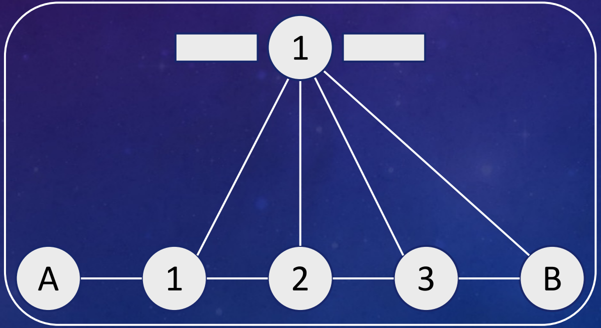
Terrestrial Wireless Links
(low-rate, intermittent, …)

Dedicated Satellite Link
(High-rate, reserved bandwidth, intermittent, scheduled)



Terrestrial Wireless Links
(low-rate, intermittent, …)

Shared Satellite Link
(High-rate, shared bandwidth, intermittent, scheduled)



Terrestrial Wireless Links
(low-rate, intermittent, …)

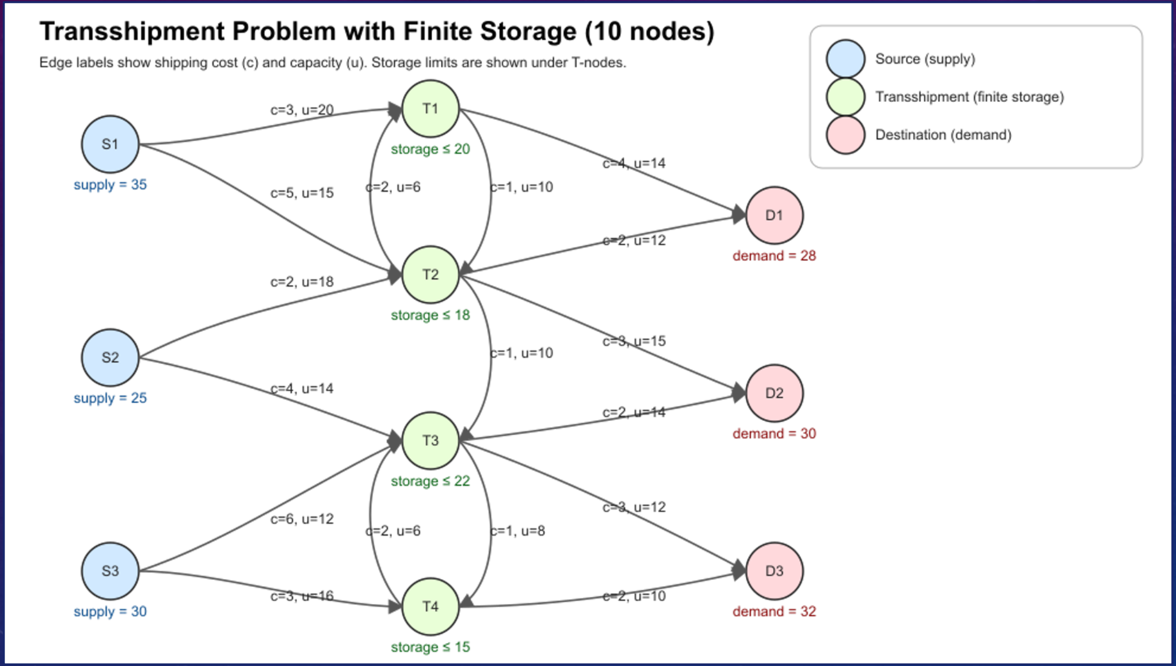
DTN Routing Mechanisms

More is Known ←→ Less is Known

- Linear Programming Formulation
- Static Routing
- Contact Graph Routing (CGR) / Schedule-Aware Bundle Routing (SABR)
- Delay-Tolerant Link State Routing (dtlsr)
- Probabilistic Routing Protocol using History of Encounters and Transitivity (PRoPHET)
- Various ‘spray’ protocols
- Flooding

ION Supports These (Static Routing; Contact Graph Routing (CGR) / Schedule-Aware Bundle Routing (SABR))

DTN Routing as a Transshipment Problem



To turn this into a time-variant routing problem you need to introduce the times when topology / conditions change and ensure that all constraints hold during the appropriate time intervals. Yes, your state space will explode.

Simple Routing Protocols

Static Routing

Flooding

- Epidemic routing

Contact Graph Routing (CGR) / Schedule-Aware Bundle Routing (SABR)

Routing 'protocol':

- Provide the link schedules to all nodes ahead of time

Forwarding:

- On receiving a bundle, a node computes, using the link schedules and its own knowledge of its own network loading, the 'best' path to the destination and sends the bundle to the next hop in that path
  - 'Best': earliest arrival

Schedule-Aware Bundle Routing (Contact Plan Routing) Example

Contact	Sender	Recvr	From	Until	Rate
1	A	B	1000	1100	1000
2	B	A	1000	1100	1000
3	B	D	1100	1200	1000
4	D	B	1100	1200	1000
5	A	C	1100	1200	1000
6	C	A	1100	1200	1000
7	A	B	1300	1400	1000
8	B	A	1300	1400	1000
9	B	D	1400	1500	1000
10	D	B	1400	1500	1000
11	C	D	1500	1600	1000
12	D	D	1500	1600	1000

Figure 3-2: Contact Plan Example: Contacts

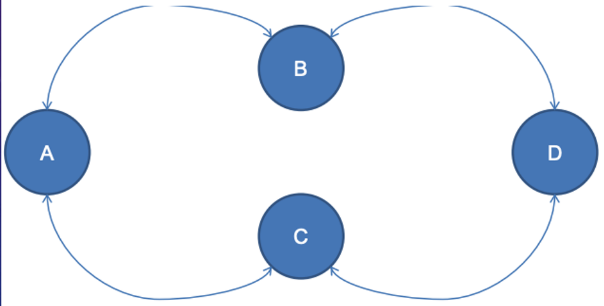
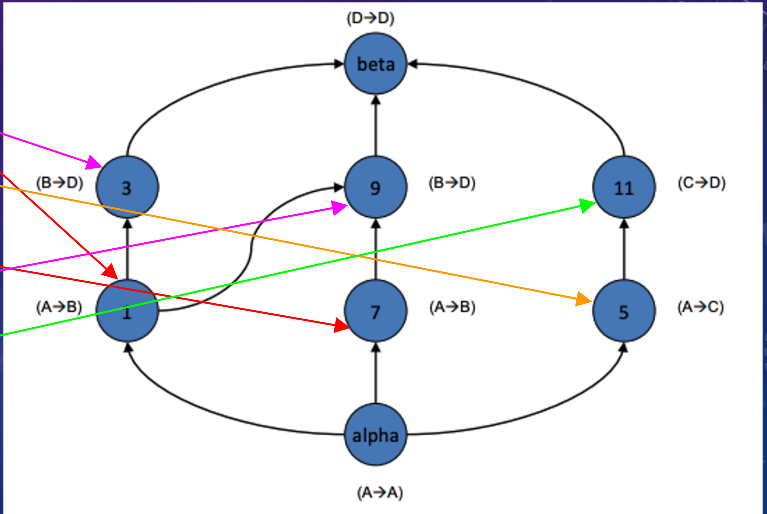


Figure 3-1: Network Topology Example

Routing Metric: Earliest Arrival Time

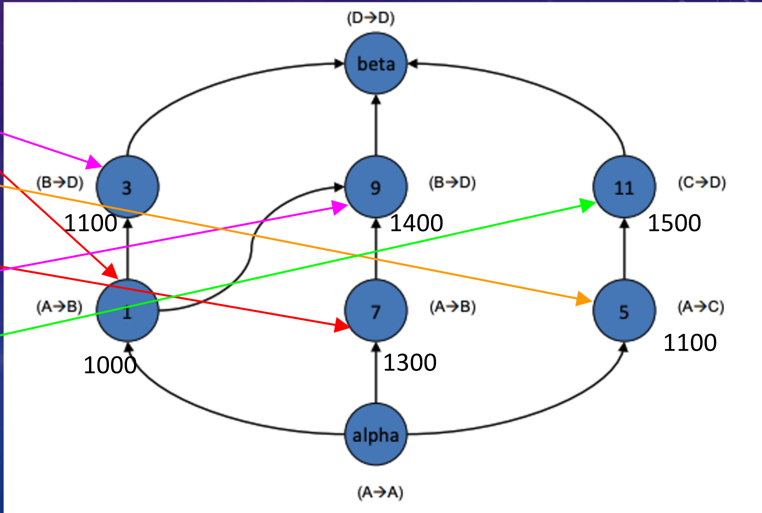
Contact	Sender	Recvr	From	Until	Rate
1	A	B	1000	1100	1000
~~2~~	~~B~~	~~A~~	~~1000~~	~~1100~~	~~1000~~
3	B	D	1100	1200	1000
~~4~~	~~D~~	~~B~~	~~1100~~	~~1200~~	~~1000~~
5	A	C	1100	1200	1000
~~6~~	~~C~~	~~A~~	~~1100~~	~~1200~~	~~1000~~
7	A	B	1300	1400	1000
~~8~~	~~B~~	~~A~~	~~1300~~	~~1400~~	~~1000~~
9	B	D	1400	1500	1000
~~10~~	~~D~~	~~B~~	~~1400~~	~~1500~~	~~1000~~
11	C	D	1500	1600	1000
~~12~~	~~D~~	~~D~~	~~1500~~	~~1600~~	~~1000~~

Figure 3-2: Contact Plan Example: Contacts



Contact	Sender	Recvr	From	Until	Rate
1	A	B	1000	1100	1000
~~2~~	~~B~~	~~A~~	~~1000~~	~~1100~~	~~1000~~
3	B	D	1100	1200	1000
~~4~~	~~D~~	~~B~~	~~1100~~	~~1200~~	~~1000~~
5	A	C	1100	1200	1000
~~6~~	~~C~~	~~A~~	~~1100~~	~~1200~~	~~1000~~
7	A	B	1300	1400	1000
~~8~~	~~B~~	~~A~~	~~1300~~	~~1400~~	~~1000~~
9	B	D	1400	1500	1000
~~10~~	~~D~~	~~B~~	~~1400~~	~~1500~~	~~1000~~
11	C	D	1500	1600	1000
~~12~~	~~D~~	~~D~~	~~1500~~	~~1600~~	~~1000~~

Figure 3-2: Contact Plan Example: Contacts



Probabilistic Routing Protocol using History of Encounters and Transitivity (PRoPHET) Routing (RFC6693)

Routing Protocol

- When two nodes form a link, they exchange information about the other nodes that each has had access to (either directly or over multiple hops)

Forwarding

- When connected to another node, a node will forward those bundles

ProPHET Summary

- **Encounter-based routing**: PROPHET uses the history of node encounters to estimate the probability that a node can deliver a message to a destination.
- **Delivery predictability metric (P)**: Each node maintains a probability value (0–1) for reaching every known destination, updated upon encounters with other nodes.
- Direct encounters increase probability: When two nodes meet, their probability of delivering to each other directly is increased.
- **Transitive property:** If node A frequently meets node B, and node B frequently meets node C, then node A updates its probability of delivering to C via B (transitive delivery predictability).
- **Aging mechanism**: Delivery predictability values decrease over time if nodes do not encounter each other, preventing stale information from dominating.
- **Probabilistic forwarding**: Messages are forwarded to nodes with higher delivery predictability for the destination, improving chances of successful delivery.
  - Controlled replication: Unlike Epidemic Routing, PROPHET does not flood the network; instead, it makes smarter forwarding decisions based on probability values, reducing overhead.
- Lightweight state information: Each node only needs to store and update delivery probabilities, making it scalable in intermittently connected networks.

"OSPF-Like" Routing: dtlsr

Routing Protocol

- Nodes in contact exchange information about all other nodes they can reach
- When a link breaks and information about that break is distributed throughout the network, instead of immediately removing it from the routing graph, nodes 'age' the link, increasing its link cost with time
  - Assumption is that the link could come back

Forwarding

- Forward bundles along the shortest cost path to the destination

SOME Routing Implementations

	Schedule - Aware Bundle Routing	dtlsr	Static Routing	Epidemic Routing	PRoPHET	Flooding	Multicast (imc scheme)	External
ION	X		X				X	
DTN2		X	X		X	X		X
DTNME								X
HDTN	X	Shortest Path						
ibr-dtn			X	X	X	X		
u-D3TN	X*							X

* Maybe not protocol interoperability but core concepts

References

- Juan Alonso and Kevin Fall, “A Linear Programming Formulation of Flows over Time with Piecewise Constant Capacity and Transit Times,” IRB-TR-03-007, June, 2003
- CCSDS 734.3-B-1, “Schedule-Aware Bundle Routing”, July 2019, CCSDS
- S. Ali, J. Qadir and A. Baig, "Routing protocols in Delay Tolerant Networks - a survey," 2010 6th International Conference on Emerging Technologies (ICET), Islamabad, Pakistan, 2010, pp. 70-75, doi: 10.1109/ICET.2010.5638377.
- CCSDS 734.3-B-1, “Schedule-Aware Bundle Routing,” July 2019, Washington, D.C., CCSDS.
- Michael Demmer and Kevin Fall. 2007. DTLSR: delay tolerant routing for developing regions. In Proceedings of the 2007 workshop on Networked systems for developing regions (NSDR '07). Association for Computing Machinery, New York, NY, USA, Article 5, 1–6. https://doi.org/10.1145/1326571.1326579
- A Lindgren et. al., “ Probabilistic Routing Protocol for Intermittently Connected Networks,” RFC 6693, August 2012, Internet Society

Backup

ChatGPT Description of the Transshipment Problem with Finite Storage

Definition: It is an extension of the classical transportation problem where goods are shipped not only directly from sources to destinations but also via intermediate nodes (transshipment points).

Finite storage constraint: Unlike the basic version, intermediate nodes have limited storage capacity, restricting how much flow can pass through them.

Network structure: The system is modeled as a directed graph with sources, transshipment points, and destinations.

Objective: Minimize the total cost of shipping goods while satisfying supply, demand, and storage capacity constraints.

Supply constraints: The total goods shipped out of each source cannot exceed its available supply.

Demand constraints: The total goods arriving at each destination must meet its required demand.

Flow conservation: At each transshipment node, the amount received (inflow) cannot exceed the amount sent out (outflow) plus storage capacity.

Capacity constraints: Each storage node has an upper bound on how much inventory it can hold at any time.

Applications: Found in logistics, supply chains, humanitarian relief distribution, and multi-stage manufacturing systems.

Complexity: The problem is a linear programming model but more challenging than standard transportation because of added storage limits at transshipment nodes.