Routing and Forwarding in Bundle Protocol

Scott Burleigh, IPNSIG

Routing vs Forwarding in BP

- ***Forwarding*** is the act undertaken by a BP node to copy a bundle to another node (a *neighboring* – topologically adjacent – node in the network) in order to increase the likelihood that the bundle's destination endpoint will receive a copy of that bundle.
- A ***Route*** is any sequence of forwarding acts that is expected to result in a non-zero likelihood that a given bundle's destination endpoint will receive a copy of that bundle.
- ***Routing*** is the computation, at a given node, of the route that will maximize the likelihood that the destination endpoint of a given bundle – currently resident at that node – will receive a copy of that bundle.

Routing Isn’t Always Possible (1 of 2)

- In order to compute a route, it’s necessary to have knowledge of the topology of the network – that is, knowledge of all available forwarding acts. The route will be composed as available forwarding acts are selected and sequenced in a way that will result in delivery of the data.
- DTN is designed to support data reception, storage, and forwarding at any time, whether it is anticipated or not. Delay-tolerant communication is possible even if no forwarding acts are known to be available, either now or in the future.
- But if there is no knowledge of all available forwarding acts, computation of a route is not possible. All the DTN node can do is wait until a random forwarding opportunity is spontaneously discovered, then decide whether or not forwarding a given bundle during that opportunity will increase the likelihood that the bundle’s destination node will receive a copy of that bundle.

Routing Isn’t Always Possible (2 of 2)

- A great deal of thought has gone into opportunistic DTN, the devising of procedures that maximize bundle delivery probability when there is no knowledge of the topology of the network. Some well-known examples include:
  - Epidemic forwarding
  - Spray and Wait
  - MaxProp
  - RAPID
  - BubbleRap
- When we consider DTN for environmental monitoring, however, it seems likely that the nodes acting as sensors and relays will not be randomly distributed over time and space: some will be in fixed locations operating on known duty cycles, while others will be navigating predefined sampling circuits at known intervals. Awareness of these predefined deployment choices will constitute topological knowledge upon which route computation can be based.

An analogy: routing is like travel planning

- Suppose you need to travel from San Diego to Pasadena:
  - (Absent Google Maps) You consult a road atlas for California.
  - You find that it's easy:
    - You drive [that is, you *forward* yourself, in your car] north from San Diego on I-15.
    - When you reach the intersection with I-210 you drive west on I-210 and exit at Pasadena.
  - This is easy, in part, because the I-15 and I-210 freeways operate continuously. You can always drive on them, at any time.

Continuing the analogy

- Suppose instead you need to travel from San Diego to Jakarta.
  - You can't drive; there's an ocean. You have to fly (that is, *forward* yourself in an airplane). No problem: both San Diego and Jakarta have airports.
  - But the air corridor from San Diego to Jakarta does not operate continuously. You have to select one of several scheduled flights.
  - Moreover, there are no direct flights from San Diego to Jakarta: you have to change planes in (let's say) New York and again in Abu Dhabi.
  - In order to plan your travel you have to select a flight from San Diego to New York, a flight from New York to Abu Dhabi that departs after your New York flight arrives, and a flight from Abu Dhabi to Jakarta that departs after your Abu Dhabi flight arrives.

DTN vs IP routing (1 of 2)

- Routing in the Internet is similar to planning travel from San Diego to Pasadena, because the links that constitute the Internet – fiber optic cables, copper wires, WiFi connections – are changed rarely and operate continuously.
- You can compute a route from the known and relatively constant topology formed by those links. Information about that topology is exchanged automatically among Internet routers, by means of “routing protocols”, forming a knowledge base that is analogous to a road atlas.

DTN vs IP routing (2 of 2)

- Routing in a DTN-based network is similar to planning travel from San Diego to Jakarta because the links that constitute the network operate only occasionally, when mission operations teams have commanded spacecraft and ground antennae to point at each other at a certain time and begin radiating information under predefined conditions for a predefined length of time.
- The topology formed by those links is time-varying: it is known and relatively constant, but only in a continuum that has both time and space dimensions. Information about that topology is published by mission operators, much as airlines publish flight schedules.

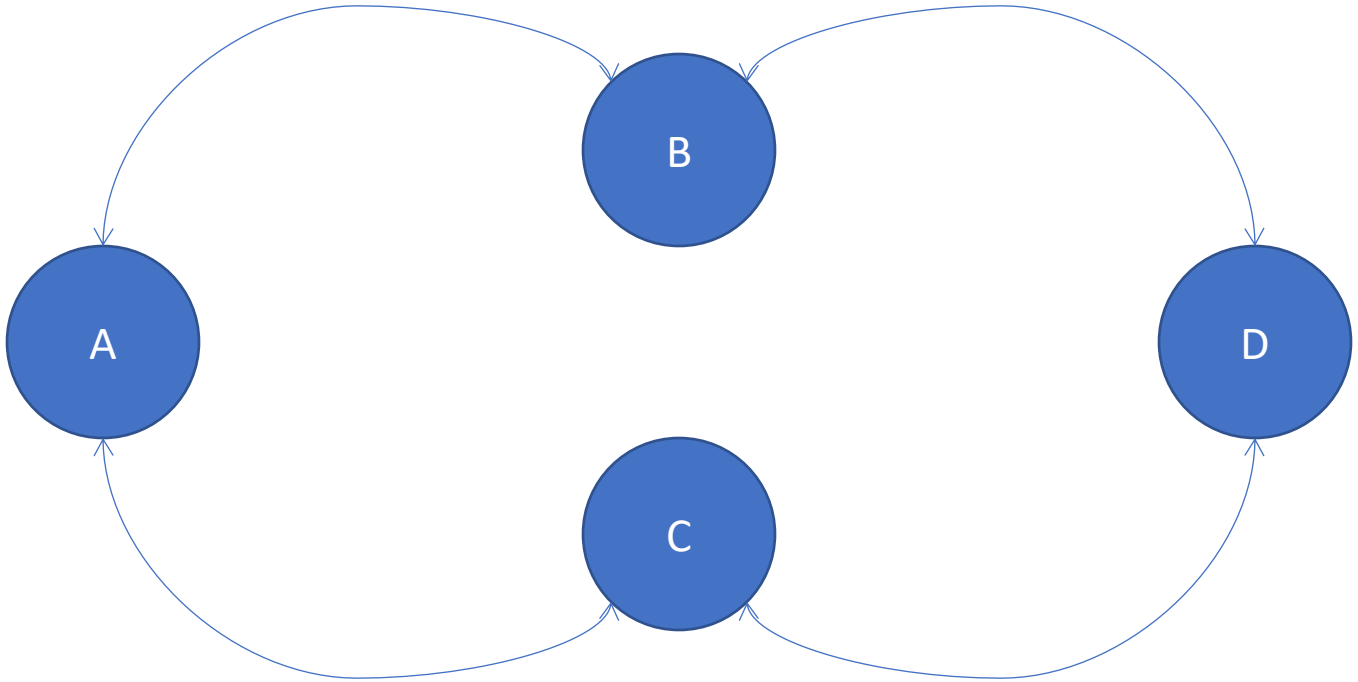
Contact plans

- We can call those planned episodes of mutual radiation ***contacts***, and the published information that enables a BP node to understand the time-dimensioned topology of the network can be termed a ***contact plan***.

Using a contact plan for DTN routing

- The topological information provided by a contact plan can be used in a variety of ways to compute routes through the DTN-based network.
- The oldest of these methods is Contact Graph Routing (CGR), first described in 2007.
  - CGR has been implemented in the Interplanetary Overlay Network (ION) implementation of the DTN architecture since its inception in 2006; it is currently operative on the International Space Station.
  - CGR is the core technology standardized by CCSDS in the Schedule-Aware Bundle Routing (SABR) standard 734.3-B-1, published in July of 2019.

Network topology



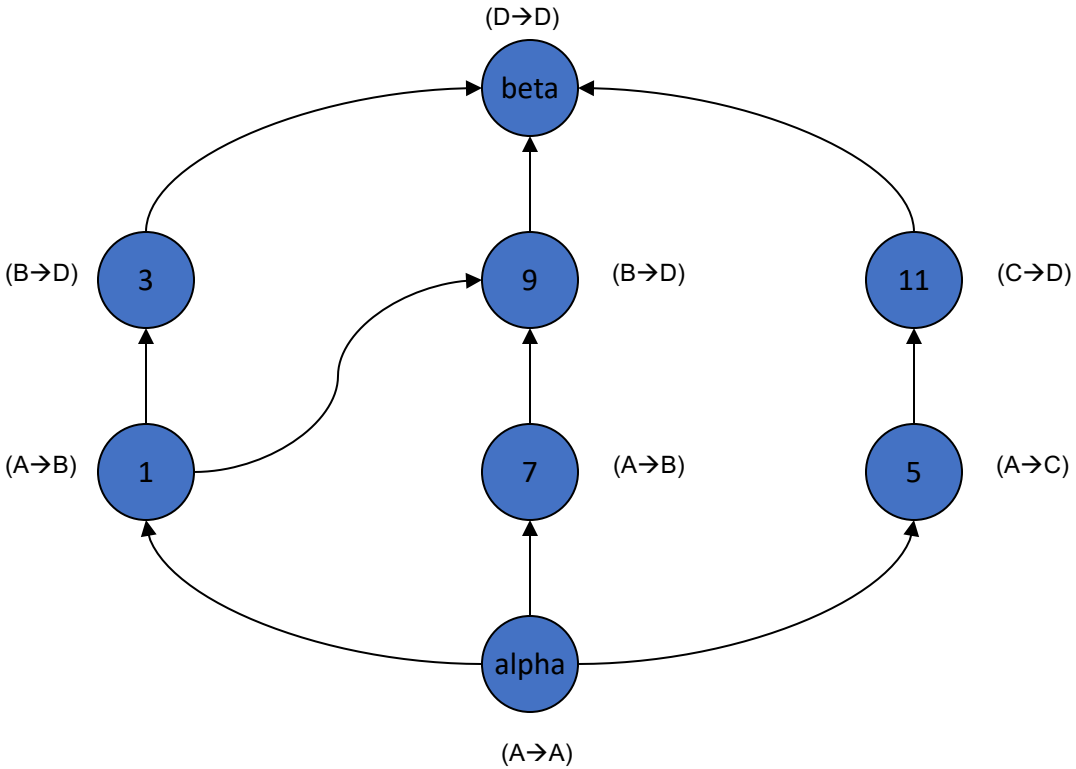
Example contact plan

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

Rules for drawing a contact graph

- Root vertex, at top of diagram, is the “delivery” vertex, labeled Beta. It’s the identity contact from the destination node to itself.
- Common terminal vertex, at bottom of diagram, is the “transmission” vertex, labeled Alpha. It’s the identity contact from the local node (at which the route is being computed) to itself.
- All contacts for which Recvr is the Sender of Beta are drawn as children of Beta and labeled with their contact numbers. Likewise, all contacts for which Recvr is the Sender of some contact are drawn as children of that contact’s vertex (and labeled with their contact numbers) except:
  - No contact whose Sender is Beta is drawn.
  - No contact whose Recvr is Alpha is drawn.
  - No contact X is drawn as the child of a contact Y that ends before contact X begins. (That is, the upward direction of each edge always signifies forward progression in time.)
- Alpha is drawn as the child of all contacts whose Sender is Alpha.

Contact graph for node D, given contact plan shown above



Notes

- Lookup data structures can be implemented as red-black trees for quick reference.
- Routes are computed by using Dijkstra's Algorithm to search through the contact graph.
- Routes are computed only when the contact plan (and thus the contact graph) changes. Normally they are cached for re-use.
- Route *selection* (not computation) is performed individually per bundle.

Routing Procedure

- Given a bundle with stated destination, size, and TTL:
  - If no routing table exists for the destination node, create it (see next slide).
  - For each route in the routing table for the destination node (in increasing final delivery time):
    - If final delivery time is after TTL expiration, exit loop.
    - If aggregate capacity < bundle size, skip this route.
    - Else add route's proximate node to list of candidate proximate nodes.
  - Queue bundle for transmission to best candidate.

Routing Table Creation Procedure

- While the number of contacts from the local node to any neighbor that (a) can potentially be on routes to the destination node but (b) are not in the routing table exceeds 0:
  - Use Dijkstra's Algorithm to find the route to the destination node that has the earliest delivery time, excluding all neighbor contacts that have already been added to the routing table.
  - Add to the routing table the neighbor contact that is the first hop on that route, and note with that contact the contacts for all subsequent hops on the route.

Routing cases

- In all cases, a bundle is being sent from node A to node D.
  - The bundle's length is 6000.
  - The bundle's creation time is 900.

Case 1

- Bundle's TTL is 1000, so expiration at 1900.
- Backlog on A→B queue is 50,000.
- Backlog on A→C queue is 80,000.
- Graph traversal is (beta), 3, 1, (alpha), 9, 1 again, (alpha), 7, (alpha), 11, 5, alpha.
  - All contacts are usable.
- Bundle is queued for transmission to B on contact 1, increasing backlog to 56,000, because contact 1 has an earlier end time than contact 7 or contact 5. Path is A→B→D.

Case 2

- Bundle's TTL is 550, so expiration at 1450.
- Backlog on A→B queue is 50,000.
- Backlog on A→C queue is 80,000.
- Graph traversal is (beta), 3, 1, (alpha), 9, 1 again, (alpha), 7, (alpha).
  - Contact 11 starts too late, so no descent into its subgraph (i.e., contact 5).
- Bundle is queued for transmission to B on contact 1, increasing backlog to 56,000, because contact 1 has an earlier end time than contact 7. Path is A→B→D.

Case 3

- Bundle's TTL is 1000, so expiration at 1900.
- Backlog on A→B queue is 130,000.
- Backlog on A→C queue is 80,000.
- Graph traversal is (beta), 3, 1 [found to be fully subscribed], (alpha), 9, 1 again [again, fully subscribed], (alpha), 7 [has capacity 70,000 so okay], (alpha), 11, 5, alpha.
  - Contact 1 is fully subscribed. Contacts 7 and 5 are usable.
- Bundle is queued for transmission to B on contact 7, increasing backlog to 136,000, because contact 7 has an earlier end time than contact 5. Path is A→B→D.

Case 4

- Bundle's TTL is 1000, so expiration at 1900.
- Backlog on A→B queue is 230,000.
- Backlog on A→C queue is 80,000.
- Graph traversal is (beta), 3, 1 [found to be fully subscribed], (alpha), 9, 1 again [again, fully subscribed], (alpha), 7 [also fully subscribed], (alpha), 11, 5, alpha.
  - Contacts 1 and 7 are fully subscribed. Only contact 5 is usable.
- Bundle is queued for transmission to B on contact 5, increasing backlog to 86,000. Path is A→C→D.

Case 5

- Bundle's TTL is 550, so expiration at 1450.
- Backlog on A→B queue is 130,000.
- Backlog on A→C queue is 80,000.
- Graph traversal is (beta), 3, 1 [found to be fully subscribed], (alpha), 9, 1 again [again, fully subscribed], (alpha), 7, (alpha).
  - Contact 11 starts too late, so no descent into its subgraph (i.e., contact 5). Contact 1 is fully subscribed. Only contact 7 is usable.
- Bundle is queued for transmission to B on contact 7, increasing backlog to 136,000. Path is A→B→D.

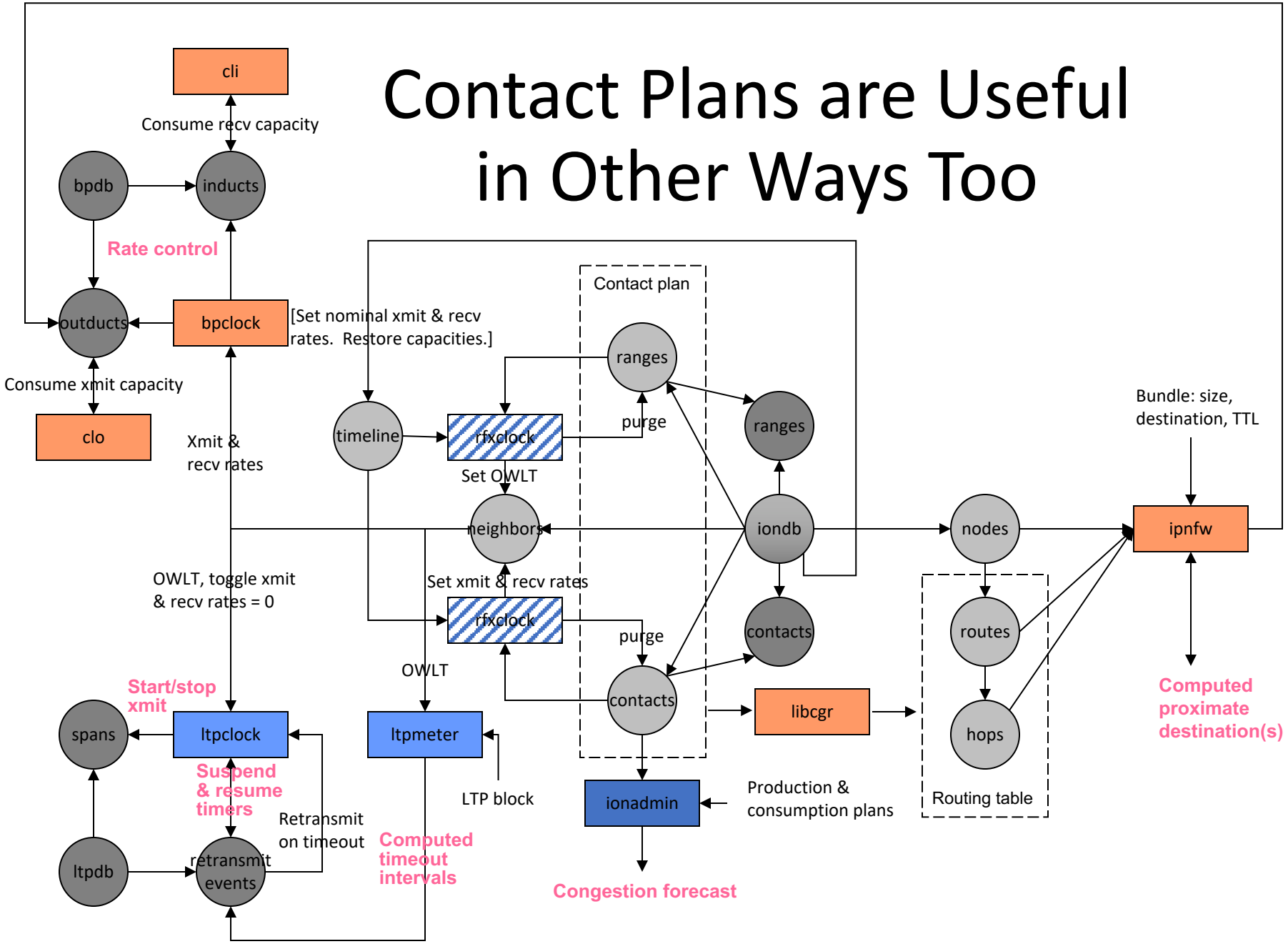
Exception Handling (1): Contact Failure

- We queue a bundle for transmission to a node in the expectation that it will be transmitted during an identified contact.
- If that contact happens to be canceled or truncated, or data rate is less than expected, then that transmission may not occur.
- So we set a timer for the end of the identified contact for each queued bundle. If the timer expires before the bundle has been transmitted, we re-compute the route for this bundle and re-queue it for transmission, possibly to a different node.
  - Leaving it in place in its current queue could make other bundles in the queue miss their transmission deadlines, with potential for widespread delivery failure.

Exception Handling (2): Overbooking

- Suppose a high-priority bundle is presented for transmission while lower-priority bundles are currently queued for transmission over upcoming contacts to neighboring nodes.
- Lower-priority enqueued bundles must be “bumped” from their current queue positions to make room for the high-priority bundle.
- Being bumped may make it impossible for a low-priority bundle to be transmitted on the contact for which it is currently enqueued.
- In this case, that bundle must be de-queued, a new route must be selected for it (computed if none exists), and the bundle must be re-enqueued on that new route.

Contact Plans are Useful in Other Ways Too



Conclusion

- CGR is a well-established standard and is known to work.
- Other mechanisms for using contact plans in DTN routing are possible.