Adaptive Forwarding with Stateful Data Plane in NDN

Beichuan Zhang
The University of Arizona

Delivering data packets

- The most fundamental function of a network.
  - The ideal case is what Paul Baran called "*Perfect Switching*"
- Today's reality is far from perfect switching, e.g.,
  - Link failure (slow convergence)
  - Congestion (single path)
  - Prefix hijack (packet blackhole)
- This talk: why today's architecture doesn't work well, and how Named Data Networking does it.

Packet delivery in IP networks

- Control plan computes the routing table
  - Maintain routing states, adapt to failures.
  - Stateful and smart
- Data plane forwards packets
  - No states resulted from packet forwarding, simply follow control plane’s order.
  - Stateless and dumb.
- Thus routing is responsible for making packet delivery robust and efficient, and yet it is not part of the forwarding process nor takes input from it.

Routing has a tough job

- Detecting problems
  - use keep-alive messages to maintain routing sessions, not enough to detect delivery problems.
    - Prefix hijack: cannot see it.
    - Congestion: cannot see it or misinterpret it.
    - Link failure: slow to see it.
- Resolving problems
  - Re-compute paths, require network-wide convergence due to the possibility of loops.
  - It takes time. The result mostly is single best paths.

Past efforts

- Improve routing
  - Make routing converge faster, become more secure, adapt to traffic load, etc.
  - It’s hard to solve data-plane problems out-of-band.

- More recently, make forwarding smarter
  - Routing pre-computes multiple paths; Forwarding or end hosts pick paths to use.
  - Limited choice, still depend on routing.

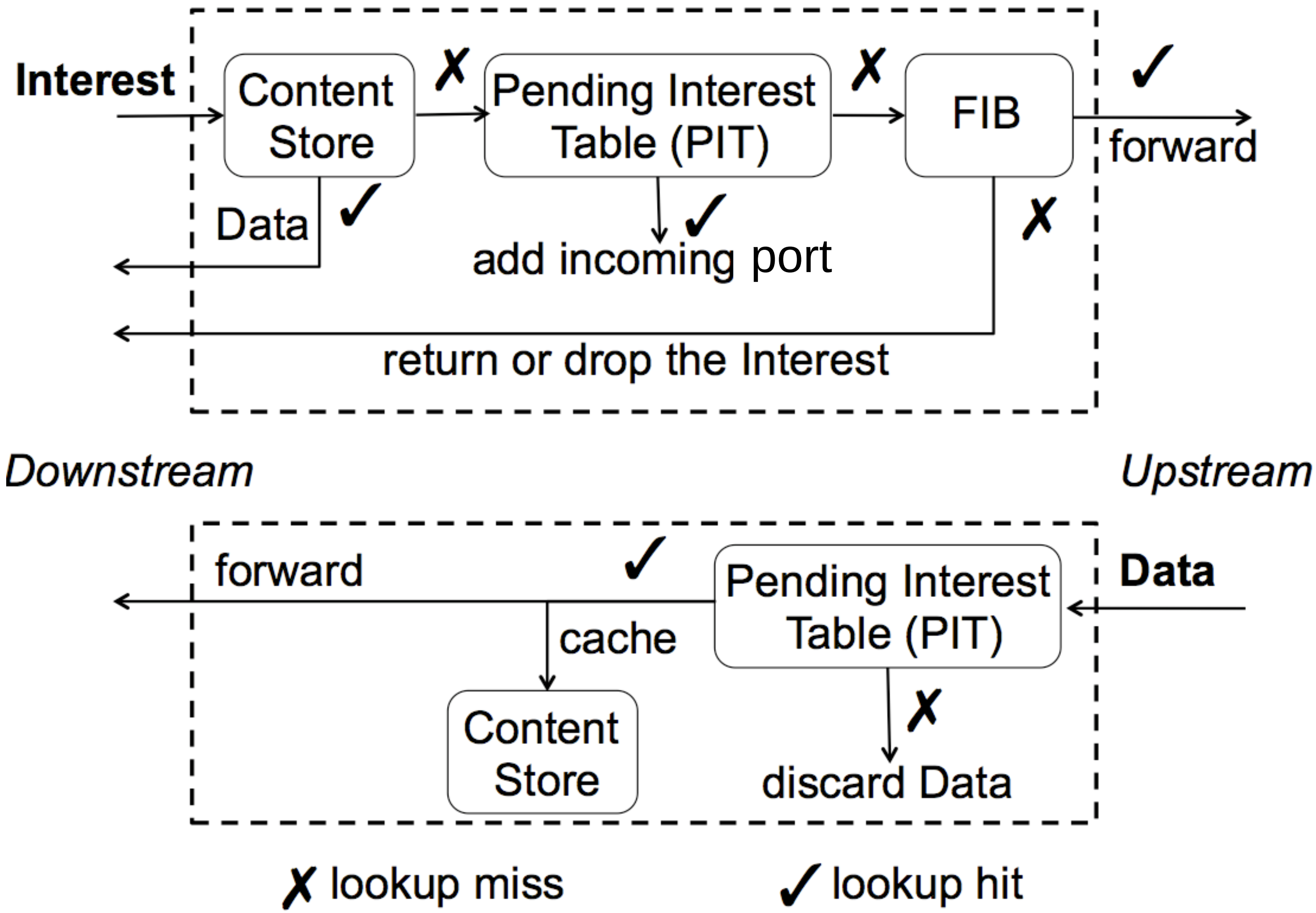
Adaptive forwarding with states

- Can we solve data-plane problems at the data plane?
  - Observe what’s going on in the data traffic
  - Detect problems directly and quickly
  - Explore alternate (multiple) paths without loops.
- These require routers to
  - Be able to identify packets passing by
  - Keep states at the data plane
- and NDN has both.

NDN primer

- A new network architecture focusing on *what* rather than *where*.
  - Packets carry names, not addresses.
  - Two types of packets: Interest and Data.
  - Interests are routed towards data sources based on names.
  - Routers remember pending Interests and their incoming and outgoing ports in the *Pending Interest Table (PIT)*.
  - Data come back on the same path, consuming PIT entries along the way.
  - Data are signed, and can be cached in the Content Store.

NDN's data plane



How do names and states help?

- Detecting problems
  - PIT entry timeout signals a problem at the data plane.
  - Link failure, congestion, prefix hijack, …
- Resolving problems
  - Add a random nonce to each Interest to detect loops.
  - Can forward Interests to many different paths, lots of choices.

Interest NACK

- When a node cannot forward or satisfy an Interest, it returns the Interest to the downstream node.
  - Downstream learns about problems quickly.
  - And explicitly from an error code in the Interest NACK.
  - Useful in scenarios like link failure, congestion, etc.

- PIT timeout is the fallback to detect packet loss.
  - E.g., prefix hijack.

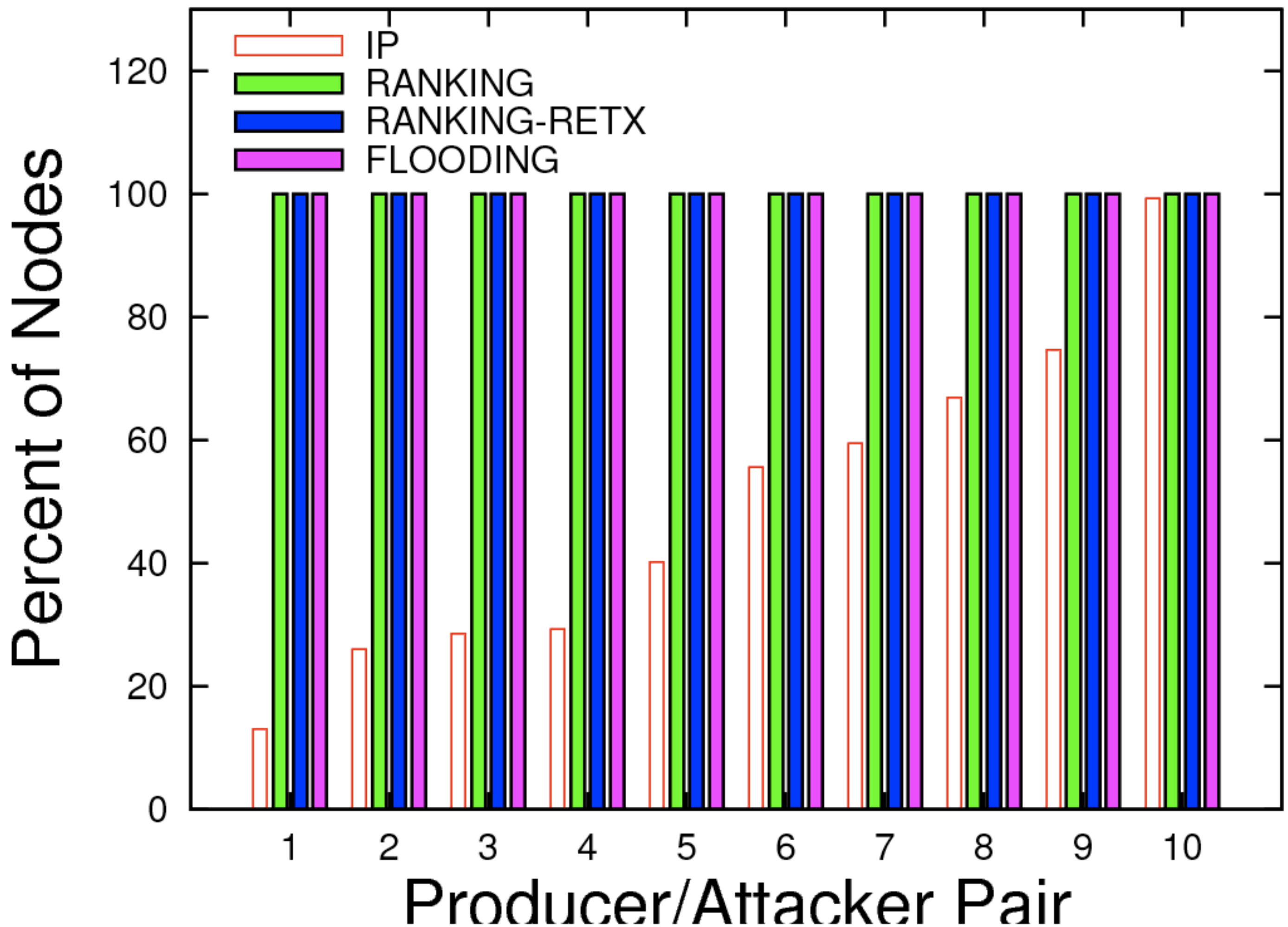
Forwarding Strategy

- To determine which port to forward an Interest.
- Basic process:
  - Outgoing ports are ranked for each prefix.
  - In general try higher ranked ports first
    - If data returns, update RTT
    - Otherwise update the ranking and try other ports
  - Can try multiple ports at the same time.
  - Different strategies may try different ports and update the ranking differently.
  - Round-robin, round-robin with retransmission, etc.

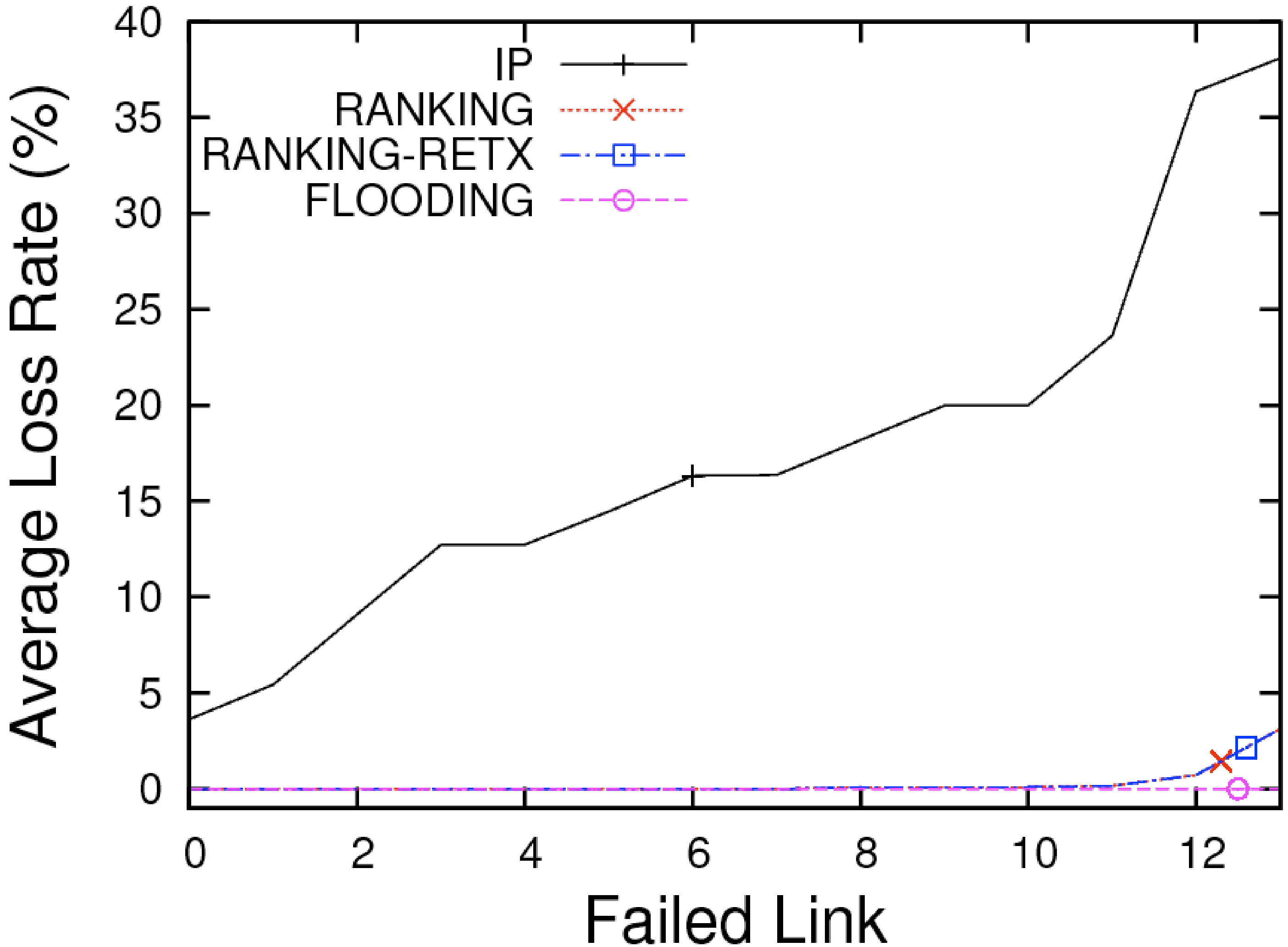
Preliminary evaluation

- Topologies: Abilene and Sprint (Rocketfuel)
- Routing protocol: OSPF
- Scenarios: hijack, link failure, congestion
- Comparison: IP vs. NDN

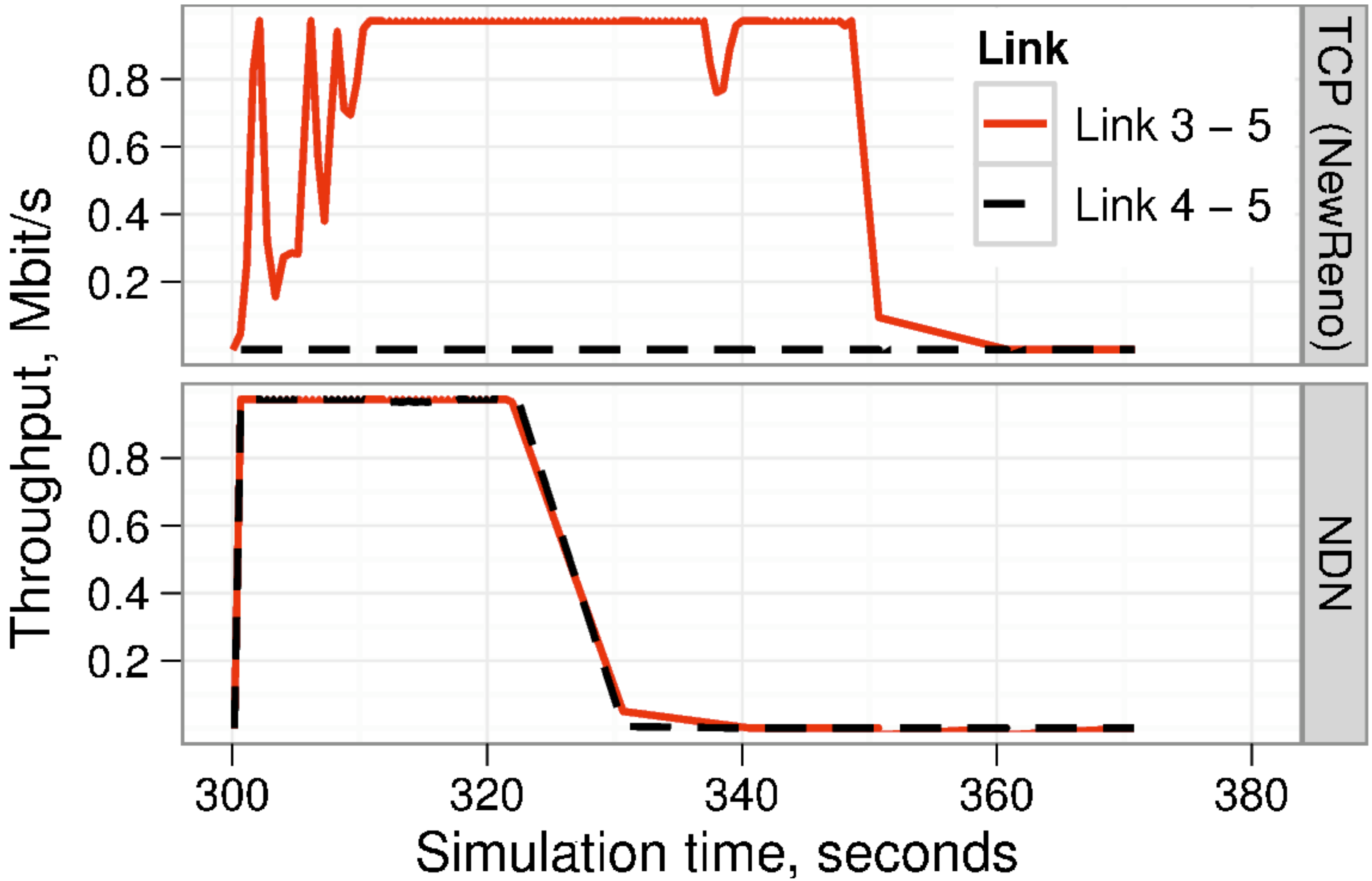
Prefix Hijack (Data Delivery Ratio)



Link Failure (packet loss during convergence)



Congestion (utilization on two paths)



What's the role of routing?

- Routing is now a helper.
- It's still very useful.
  - Maintain topology, propagate prefixes, filter routes based on policy, etc.
  - Help forwarding rank the ports.
- But it doesn't have to be perfect
  - Doesn't have to handle churns.
  - Doesn't have to be very accurate, efficient, secure, ...

Conclusion

- Solve data-plane problems at the data plane.
- Ongoing work
  - Explore different forwarding strategies
  - Explore simpler routing designs
  - Reduce the amount of states
- Credit: Cheng Yi, Alex Afanasyev, Lan Wang, Lixia Zhang, and the NDN team.

Comments and questions?