ARCs and EARs
vs. MRT

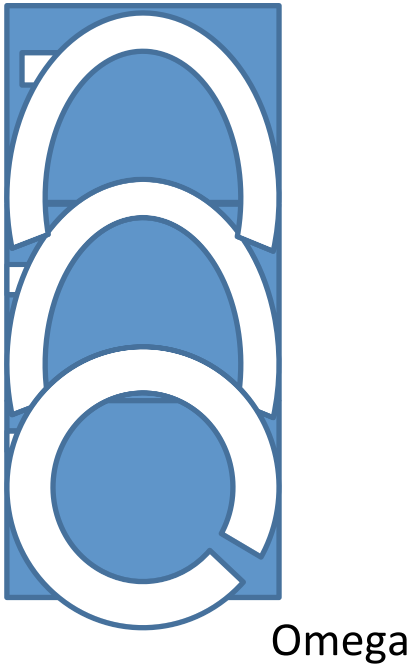
RTG Area WG, Atlanta, 2012

Pascal Thubert, Cisco

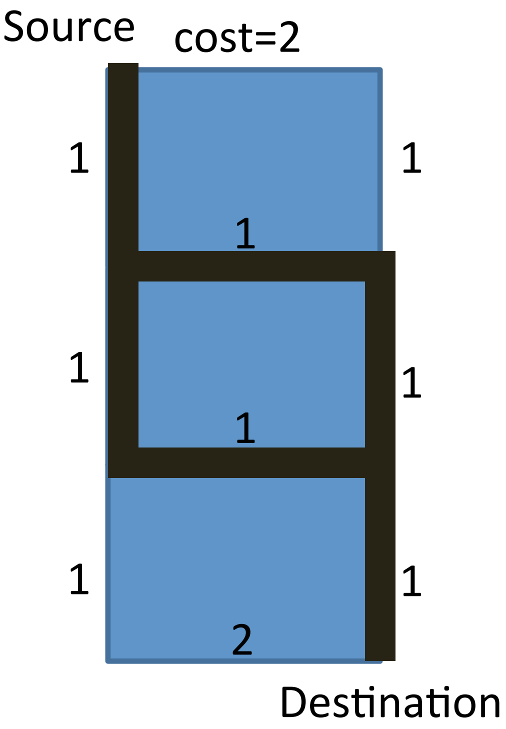
Gábor Sándor Enyedi, Ericsson

Srinivasan Ramasubramanian,
University of Arizona

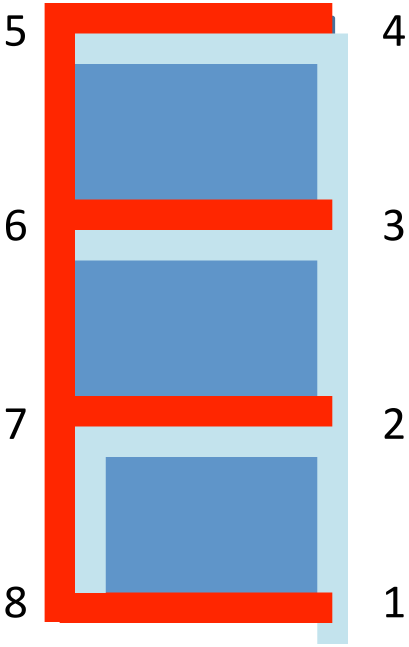
Local recovery domain vs. end to end non-congruence



ARC

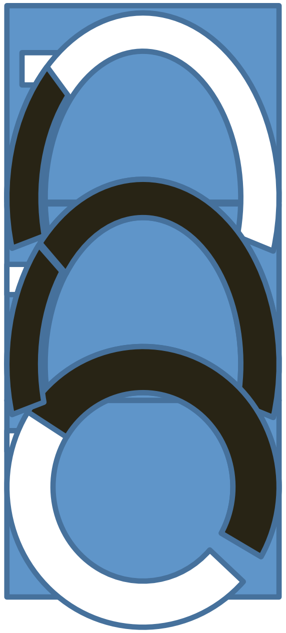


SPF

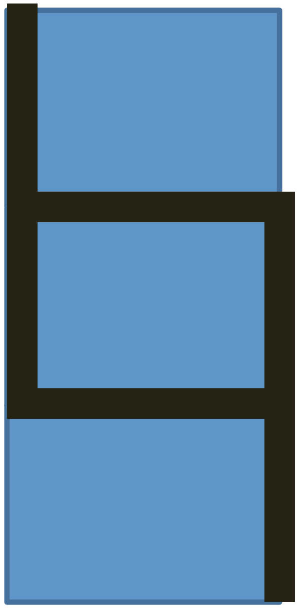


MRT

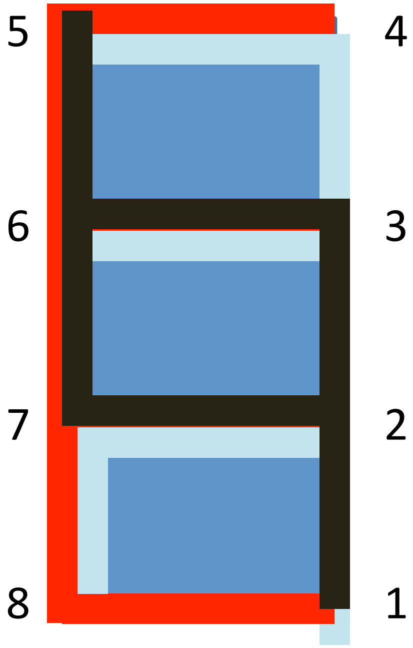
No breakage: Same route



ARC

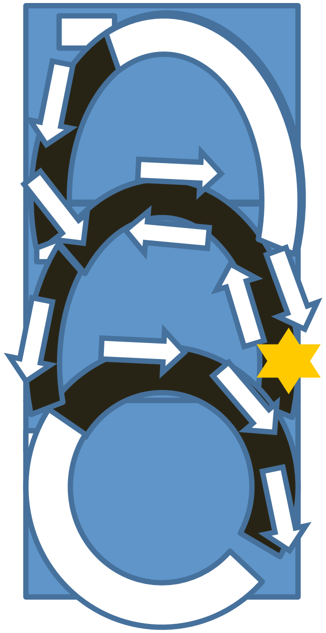


SPF

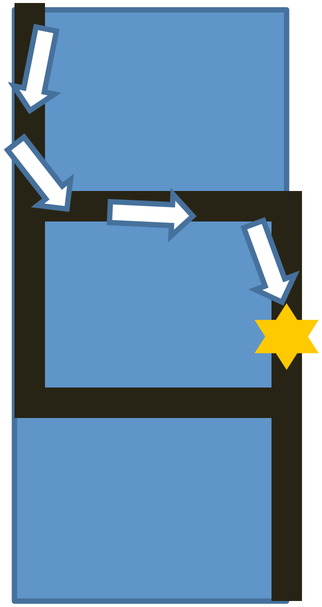


MRT

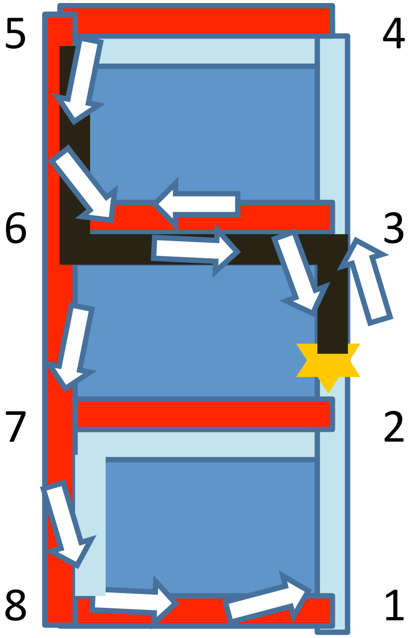
One breakage: ARCs closer to Shortest



ARC

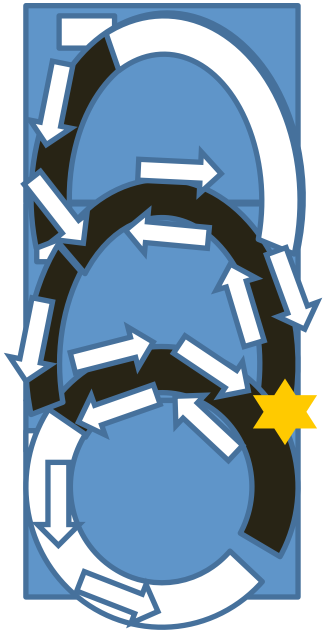


SPF

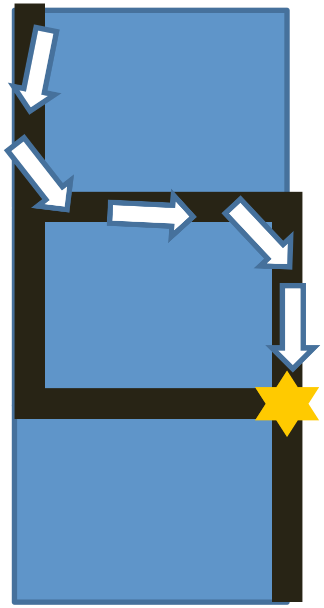


MRT

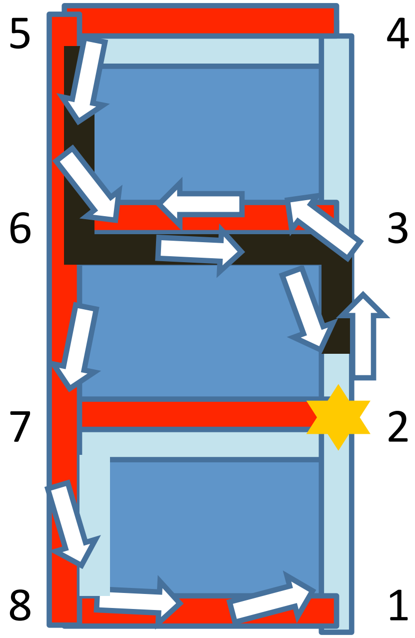
One breakage: ARCs explore twice



ARC

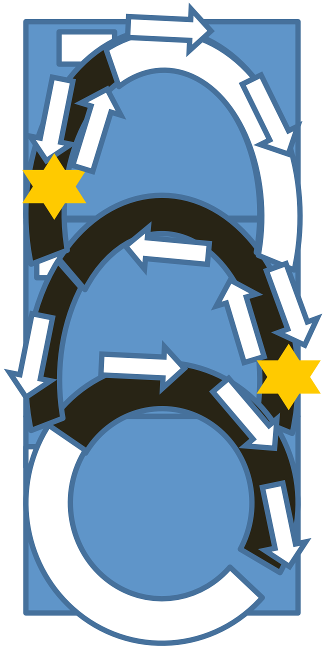


SPF

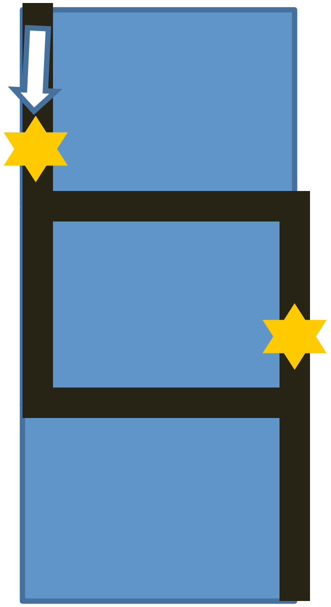


MRT

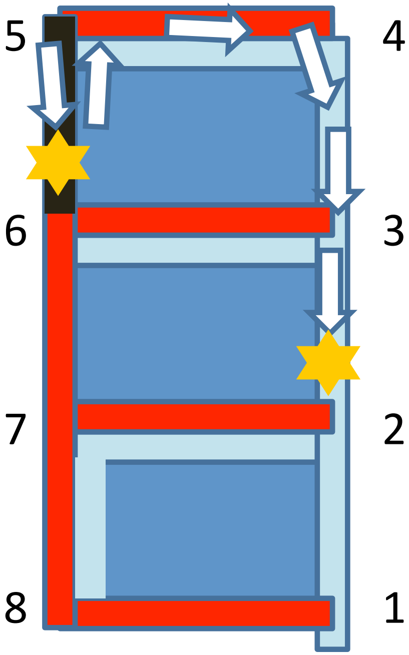
Second breakage: ARCs find a way



ARC

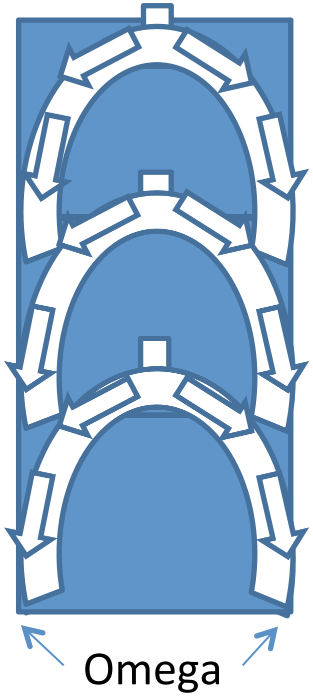


SPF

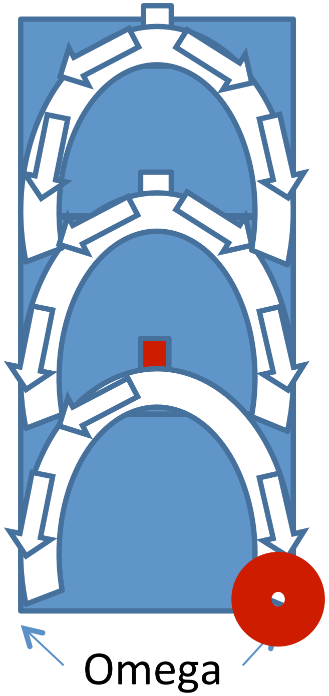


MRT

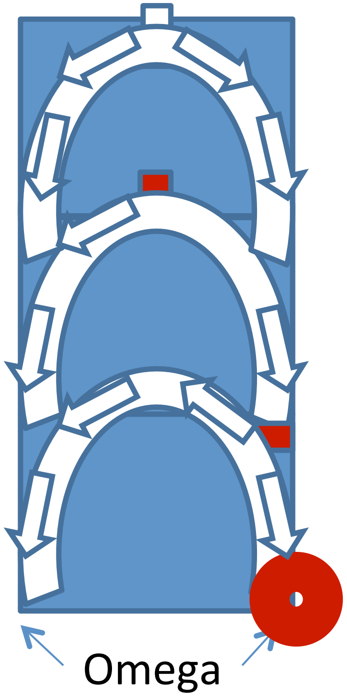
Complex Destination and Load Balancing



Normal

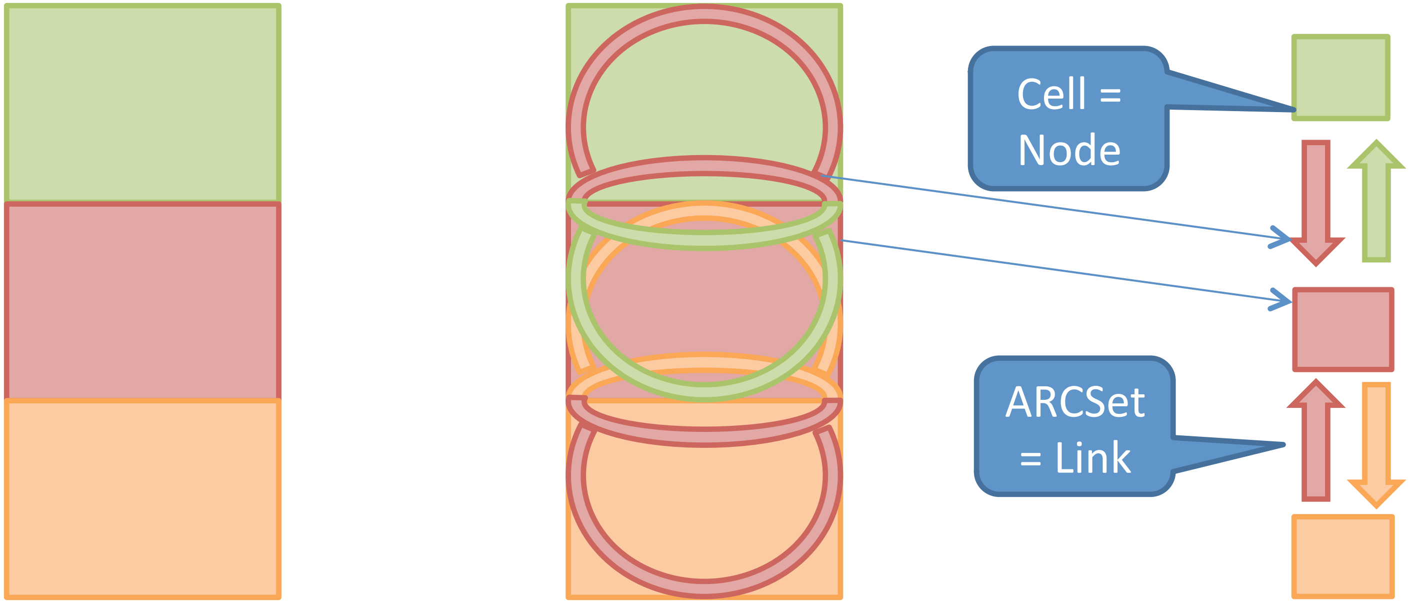


Load Balance
At Cursor



Migrate Cursor
And Back Pressure

Hierarchical Routing



Cell = Node

ARCSet = Link

Isolate cells

Build ARCSet To
Neighbor Cells

Route over
Resilient Network

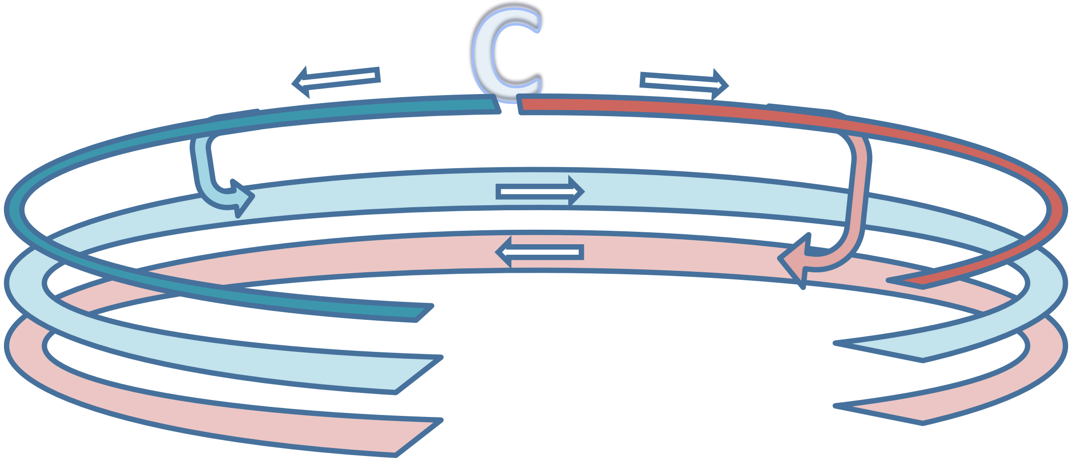
Comparison

MRT	ARCs
Limited complexity - can be even O(e)	Complexity inherited from SPF
Detour, unrelated to Shortest Path	Short detour then Shortest Path again
Small chance to avoid unrelated failures	Higher chance to avoid unrelated failures -> may address SRLG cases
Single failure: reroute at most once	Single failure may incur double reroute
Source-centric computation -> easier to distribute	Destination-centric computation -> allows for complex destinations
No load balancing	NeCM Load Balancing capabilities
Non-Congruent bicasting	Shorter Path bicasting with collision avoidance

Backup

Labels

- MRT: 3



- ARCs: 3 to 4
  - 1 from cursor to edge
  - + 2*1 from edge to edge for recovery

  +1 for load balancing

Tags

- MRT: reroute + color
- ARCs: reroute (reset when leaving ARC)
  - For more complex combs, capability to index edges

Similarity between MRT and ARC

Both approaches provide two forwarding edges for every destination at a node. Consequently, for a given destination, if one views only the red or blue forwarding edges, we get two directed trees (red and blue) towards the destination.

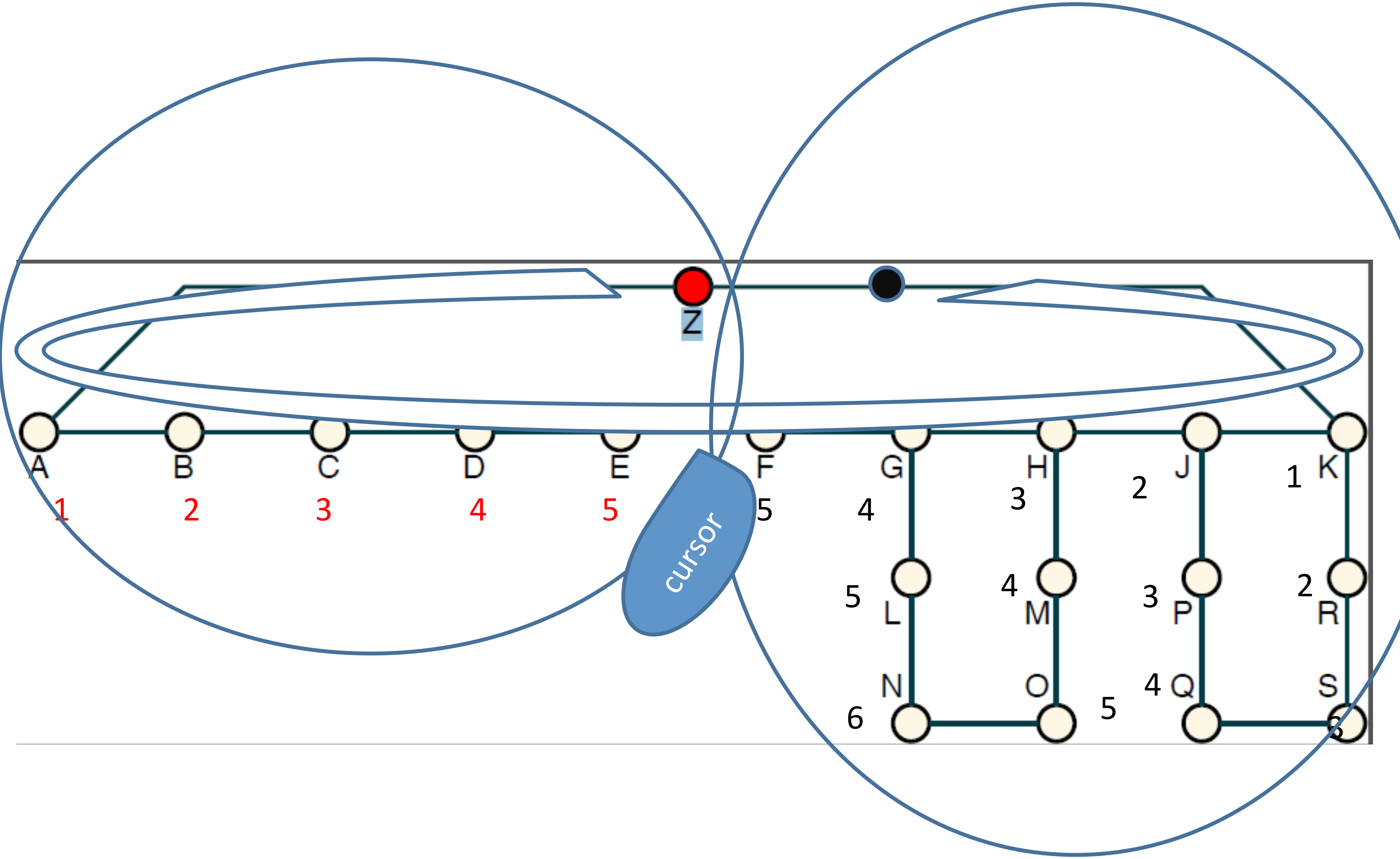
Differences between MRT and ARC

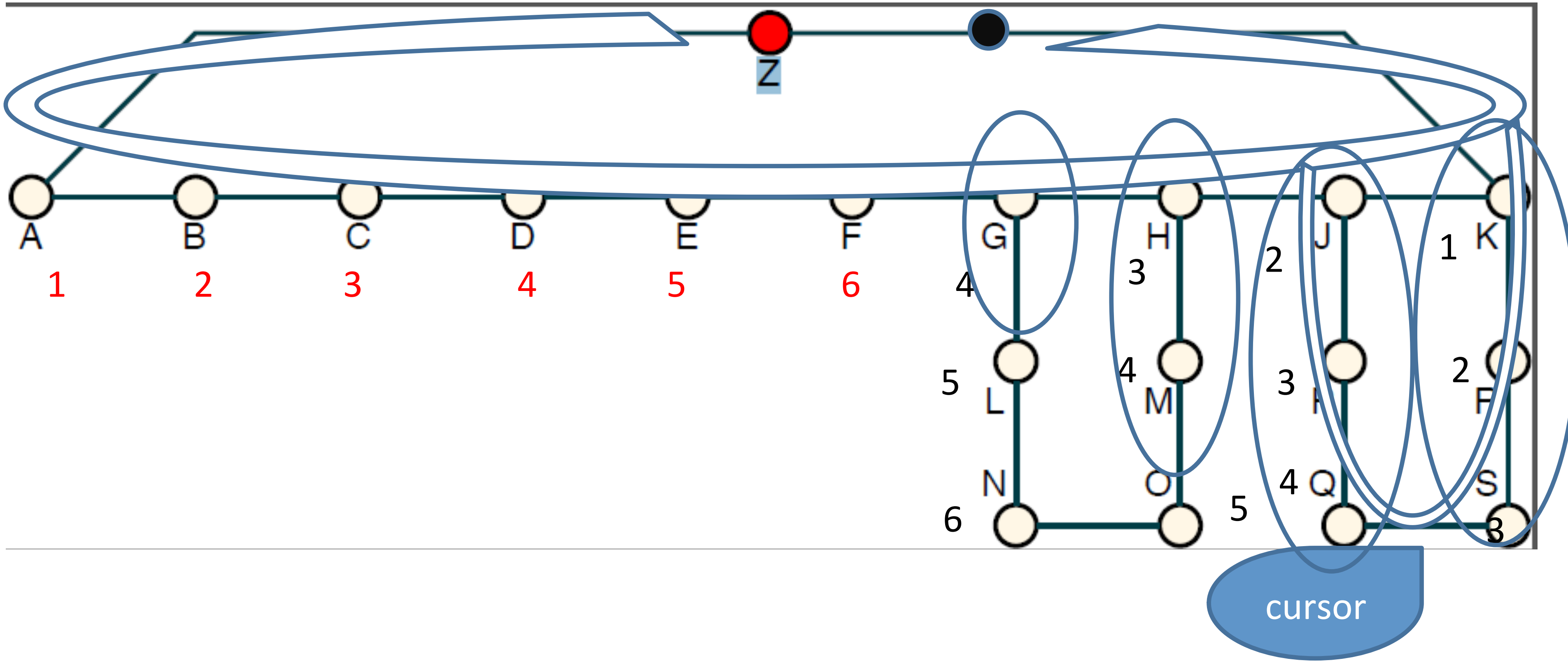
- In MRT, the path from any node to the destination on the red/blue trees is link-disjoint. In ARCs, it is not.

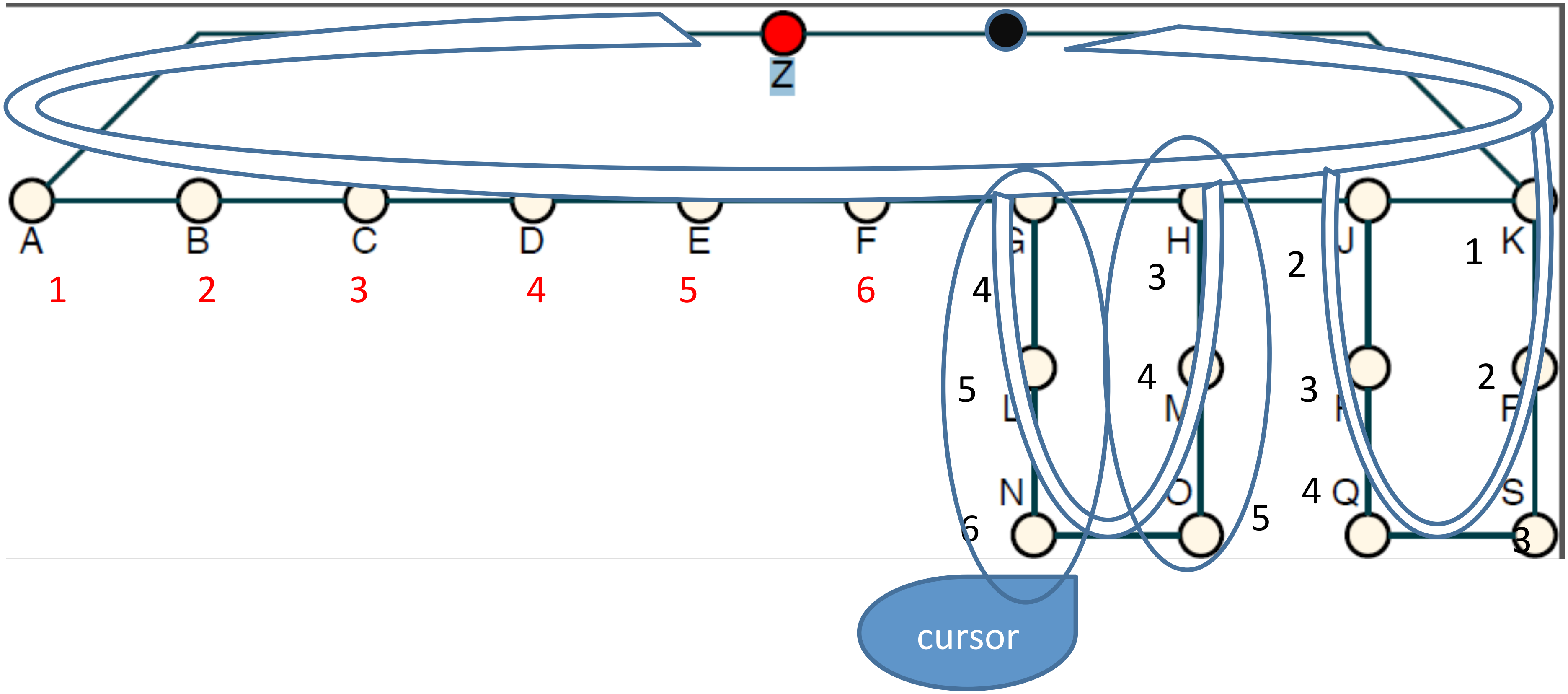
- In MRT, neither the red nor the blue tree is guaranteed to provide shortest path for a node. However, in ARC, packet is forwarded along the shortest path after a short detour.

- As a consequence, when MRT is implemented, one needs to have three FIB entries---one for shortest path forwarding, one for red tree forwarding, and one for blue tree forwarding. However, for ARC, only two trees are required. {At least that's the claim. It's also claimed that every node will have their shortest path on one of the two trees, but I am not sure about this. This has to be proved.}

- In the current version of the MRT draft, the first DAG is constructed by selecting a root node. The paths for all other nodes are computed based on this one DAG. While it is clear to see the recovery domains when the packets are routed towards the root node, it is not clear how the recovery domains would work if the packets are routed to some other node. In the context of ARCs (and MRT when MRTs are constructed for every destination node) that every ear/arc forms the recovery domain. Thus, when a packet moves from one ear/arc to another, the recovery bit can be reset.


Z
A
B
C
D
E
F
G
H
J
K
1
2
3
4
5
5
4
3
2
1
cursor
L
M
P
R
5
4
3
2
N
O
Q
S
6
5
4
6


Z
A
B
C
D
E
F
G
H
J
K
1
2
3
4
5
6
4
3
2
1
5
L
4
M
3
2
P
N
O
Q
S
6
5
4
3
cursor


Z
A
B
C
D
E
F
G
H
J
K
L
M
I
P
N
O
Q
S
R
1
2
3
4
5
6
4
5
6
3
4
5
2
3
4
1
2
3
cursor