Wireless Sensor Networks
6. WSN Routing

Christian Schindelhauer
Technische Fakultät
Rechnernetze und Telematik
Albert-Ludwigs-Universität Freiburg

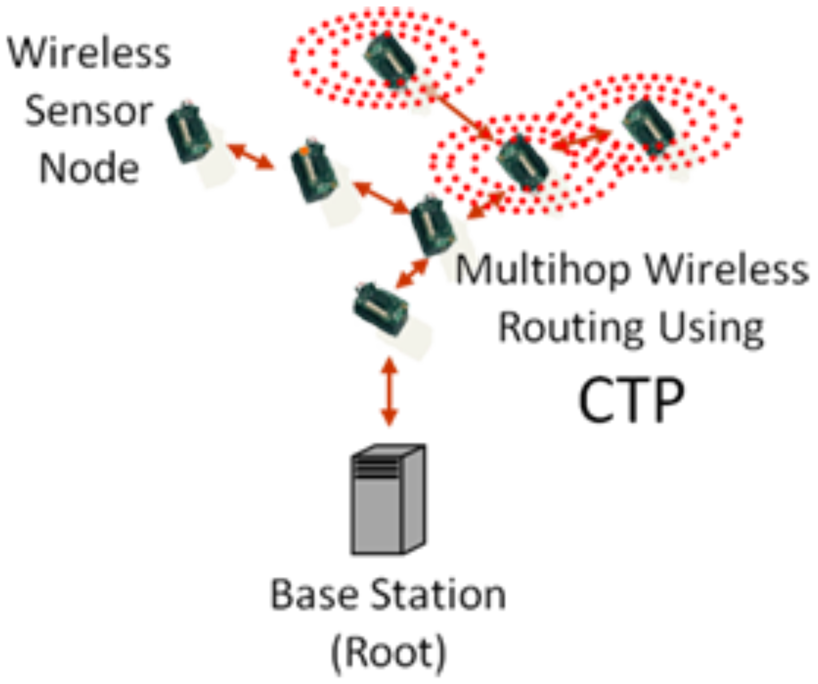
Version 30.05.2016

Collection Tree Protocol

- Literature
  - CTP: An Efficient, Robust, and Reliable Collection Tree Protocol for Wireless Sensor Networks, O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, P. Levis, ACM Transactions on Sensor Networks, Vol. 10, No. 1, Article 16, November 2013.
    - preliminary version appeared at SenSys 09
  - https://sing.stanford.edu/gnawali/ctp/

Collection Tree Protocol (CTP) Overview

- Tree topology based collection
  - Anycast route to the sink(s)
  - To collect data
  - Distance Vector Protocol
- Components
  - Link quality estimation
  - Datapath validation
  - Adaptive beaconing
- CTP become a benchmark protocol
- Many deployments, applications and implementations
- Related to
  - IPv6 Routing Protocol for Low power and Lossy Networks (RPL)
  - RFC 6206 Trickle algorithm



https://sing.stanford.edu/gnawali/ctp/

Collection Tree Protocol (CTP) Goals

- Reliability
  - ≥ 90-99% delivery rate of end-to-end packets
- Robustness
  - Operate without tuning or configuration
  - wide range of network conditions, topologies, workloads, environments
- Efficiency
  - Deliver packets with minimum amount of transmissions
- Hardware independence
  - no assumption of specific radio transceivers

ETX Cost Metric

- Literature
  - A High-Throughput Path Metric for Multi-Hop Wireless Routing, D.S.J. De Couto D. Aguayo, J. Bicket, R. Morris, MobiCom ’03, September 14–19, 2003, San Diego, California, USA.
- Goal
  - Improve throughput of wireless networks by a better metric for routing protocols
- Idea
  - Take link-loss ratios and compute a distance
- ETX: Expected transmission count metric
  - $d_f(e)$: forward delivery ratio of a link e
  - $d_r(e)$: reverse delivery ratio of a link e

$$\mathrm{ETX}(e) = \frac{1}{d_r(e) \cdot d_f(e)}$$

ETX Characteristics

- ETX(P) of a path P= (u_1, u_2, … u_n)

$$\mathrm{ETX}(u_1,\ldots,u_n) = \sum_{i=1}^{n-1} \mathrm{ETX}(u_i,u_{i+1}) = \sum_{i=1}^{n-1} \frac{1}{d_r(u_i,u_{i+1})\cdot d_f(u_i,u_{i+1})}$$

- ETX
  - based on delivery ratios
  - detects asymmetry
  - use link loss ratio measurements
  - penalizes routes with more hops
  - tends to minimum spectrum use

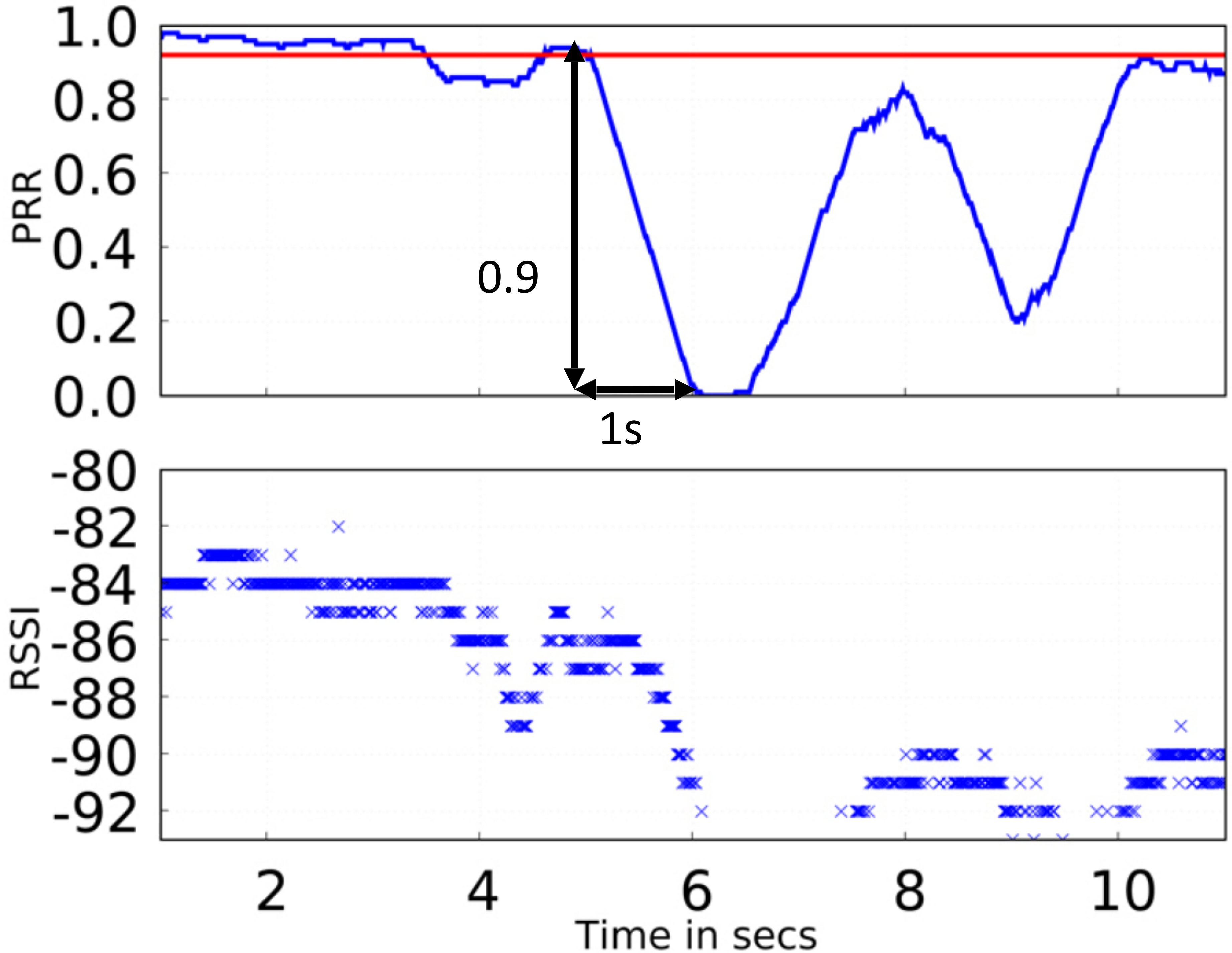
ETX: Computing Delivery Ratios

- Each node broadcasts link probes
  - of fixed size
  - at period τ
- count(t-w,t): number of probes received at window w

$$r(t) = \frac{\text{count}(t-w,t)}{w/\tau}$$

- ETX has been also applied to DSDV, DSR
- ETX is the basis of CTP

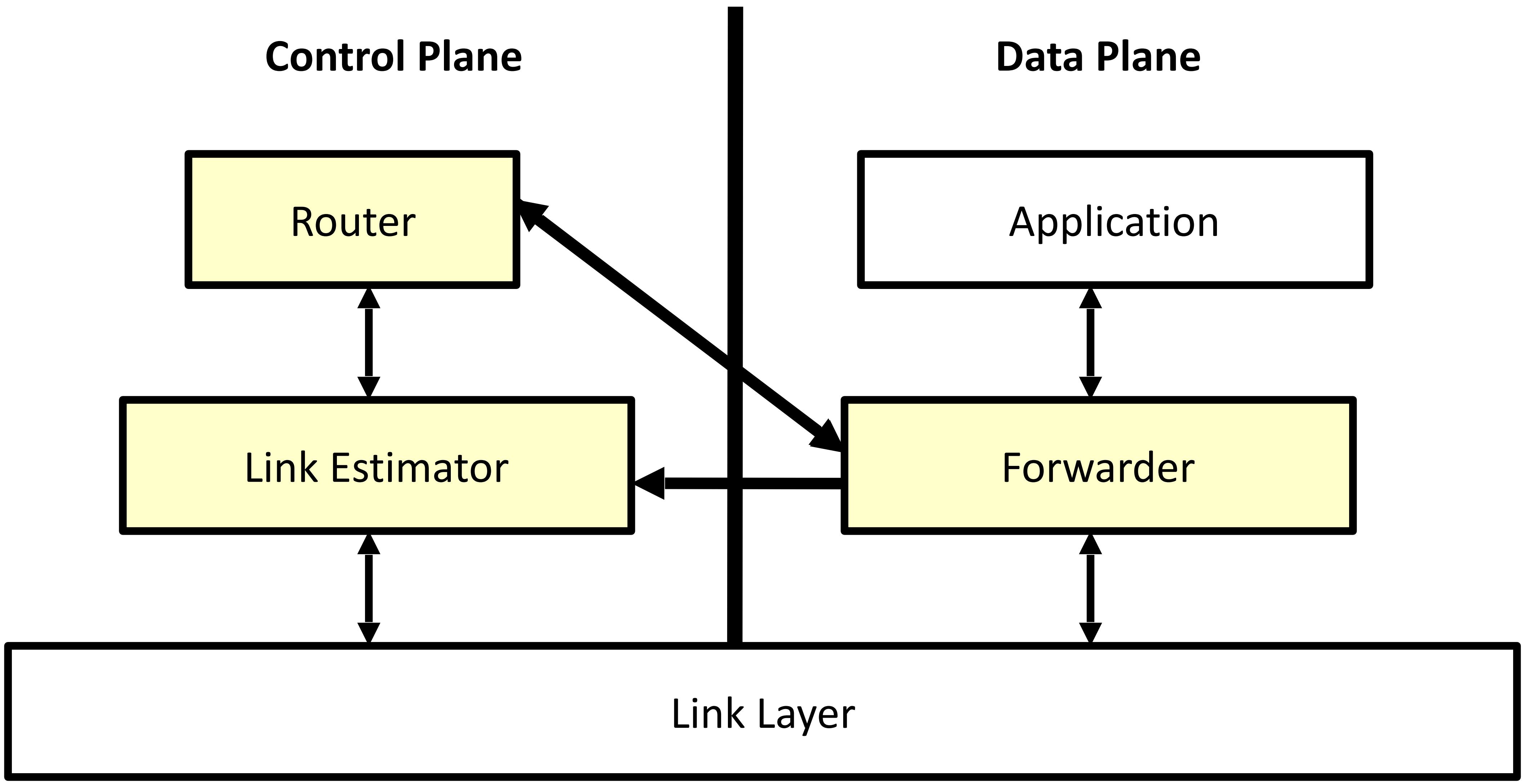
CTP Wireless Link Dynamics



Gnawali, Collection Tree Protocol , SenSys 2009 presentation

CTP: Interplay between Control and Data Plane

- Enable control and data plane interaction

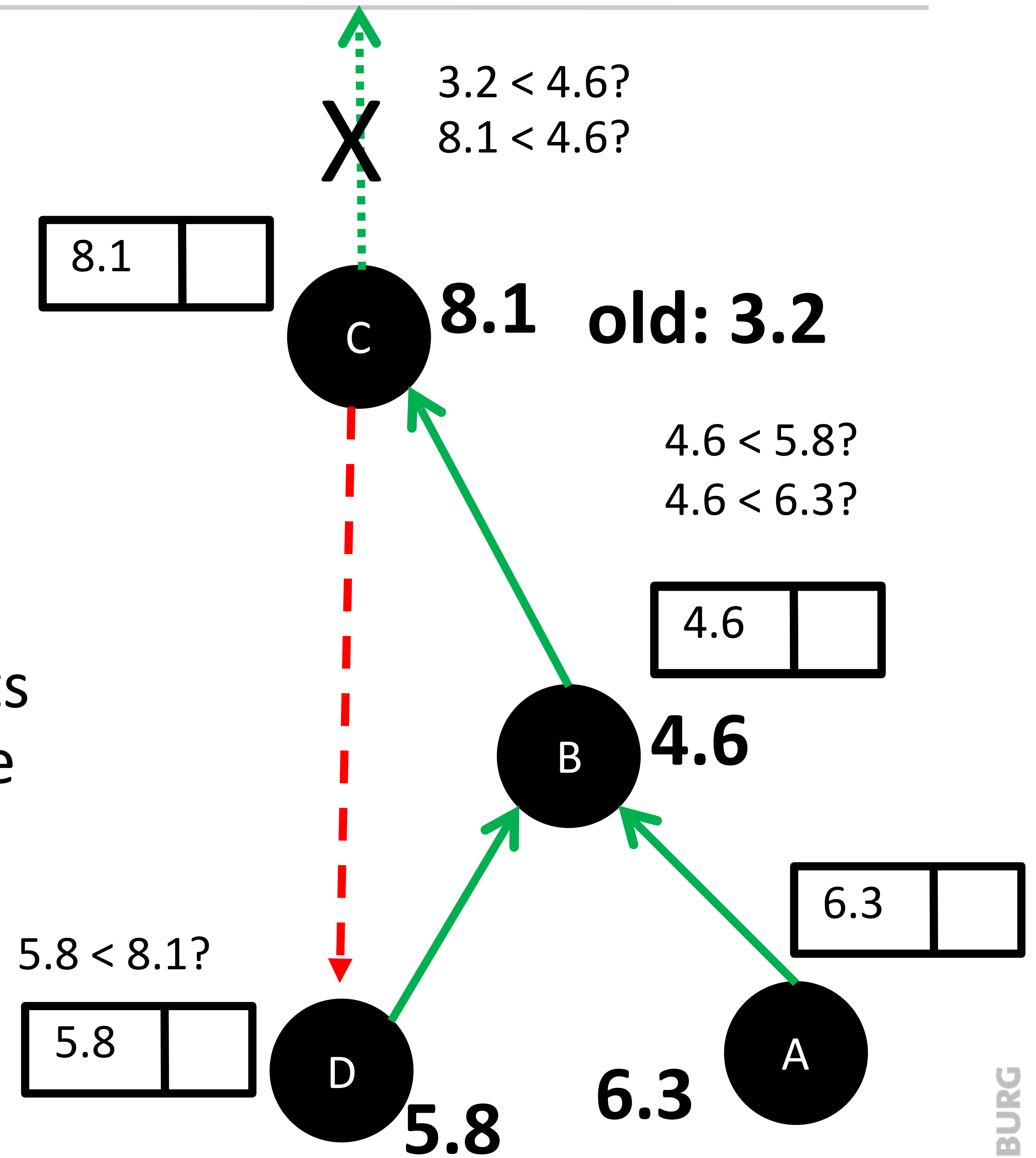


CTP Datapath validation

- Use data packets to validate the topology
  - Inconsistencies
  - Loops
- Receiver checks for consistency on each hop
  - Transmitter's cost is in the header
- Same time-scale as data packets
  - Validate only when necessary

CTP Detecting Routing Loops

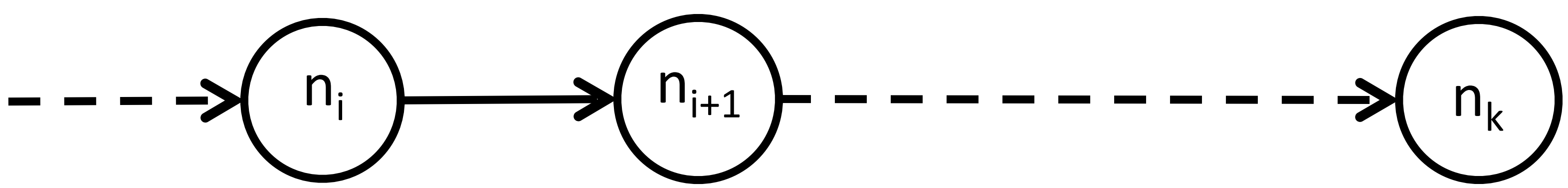
- Datapath validation
  - Cost in the packet
  - Receiver checks
- Inconsistency
  - Larger cost than on the packet
- On Inconsistency
  - Do not drop the packets
  - Signal the control plane



Routing Consistency

- *Next hop* should be closer to the destination
- Maintain this consistency criteria on a path

$$\forall i \in \{0, k-1\},\ ETX(n_i) > ETX(n_{i+1})$$



- Inconsistency due to stale state

CTP: Adaptive Beaconing

- Fixed beacon intervals never fit
  - too many beacons, if no changes appear
  - too few beacons, if drastic changes appear
- Agility-efficiency tradeoff
- Solution: Use Trickle algorithm
- Trickle
  - WSN update mechanism for software updates
  - Code propagation: Version number mismatch
  - Literature
    - Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks, Philip Levis, Neil Patel, David Culler, Scott Shenker, NSDI'04 Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation - Vol. 1, 2-2
    - RFC6206

CoNe Freiburg

Trickle: Idea

- An algorithm for establishing eventual consistency in a wireless network
- Establishes consistency quickly
- low overhead when consistent
- Cost scales logarithmically with density
- Requires very little RAM or code
  - 4-7 bytes of RAM
  - 30-100 lines of code
- Motivation: don't waste messages (energy and channel) if all nodes agrees
- Uses
  - Routing topology
  - Reliable broadcasts
  - Neighbor discovery

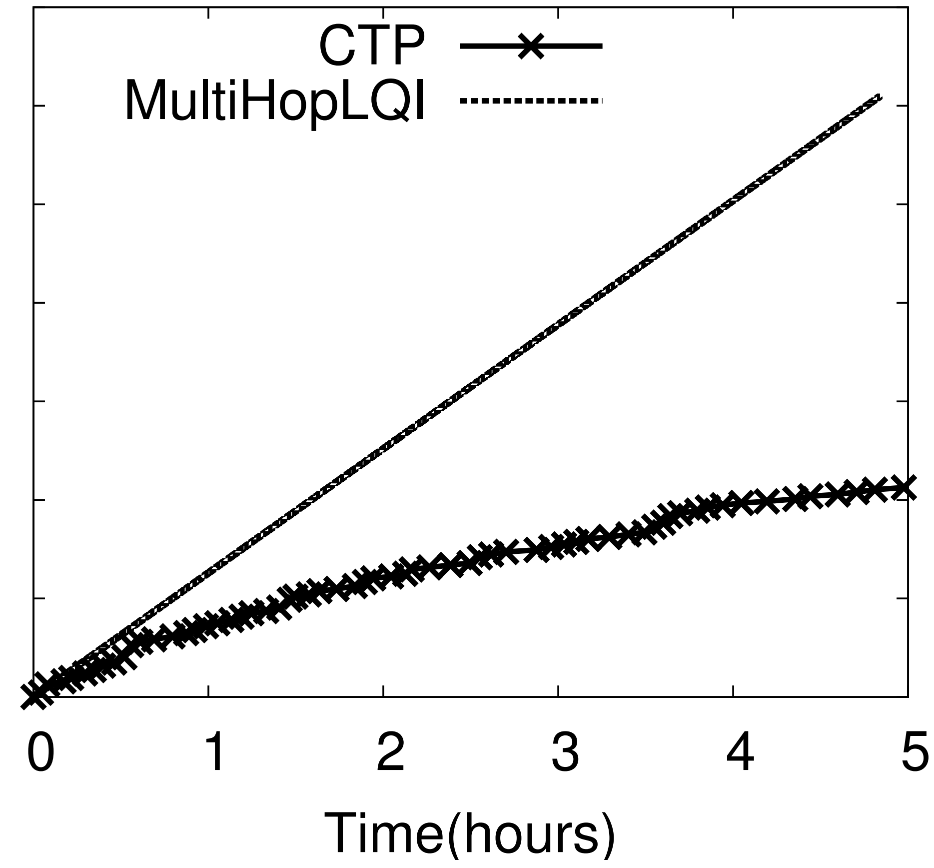
Trickle: Suppression

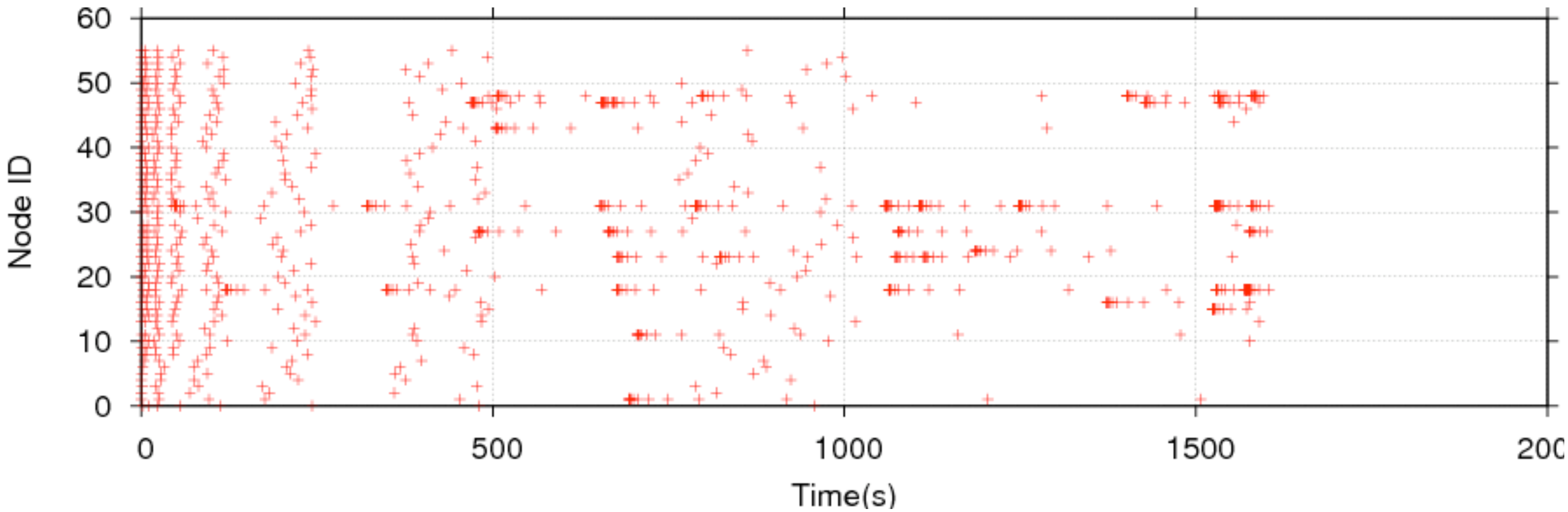
- At beginning of interval of length τ
  - counter c=0
  - On consistent transmission, c++
- Node picks a time t in range [τ/2,τ]
  - At t, transmit if c < k (redundancy constant k=1 or k=2)

Trickle: Variable Interval Length

- Interval varies between
  - τ_l:minimum interval length
  - τ_h: maximum interval length
- Start with intervals of length $\tau = \tau_l$
  - At end of interval τ, double τ up to τ_h
  - On detecting an inconsistency, set τ to τ_l
- Consistency leads to logarithmic number of beacons
- Inconsistency leads to fast updates

Trickle: Beacons



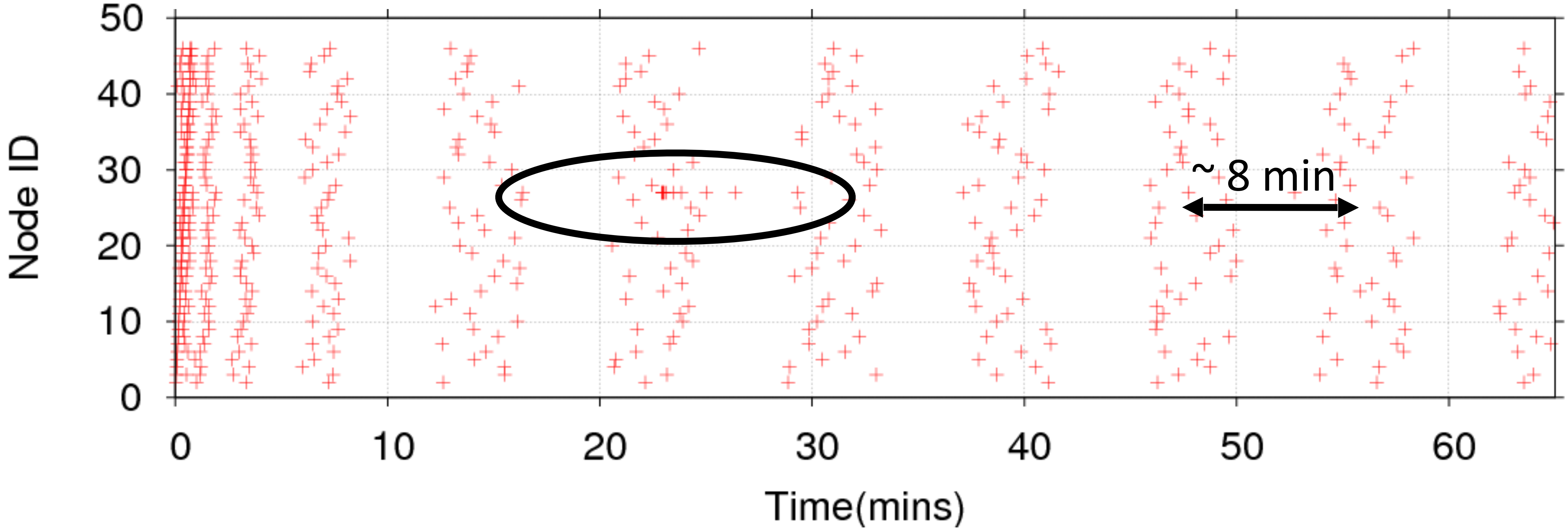


Philip Lewis, The Trickle Algorithm

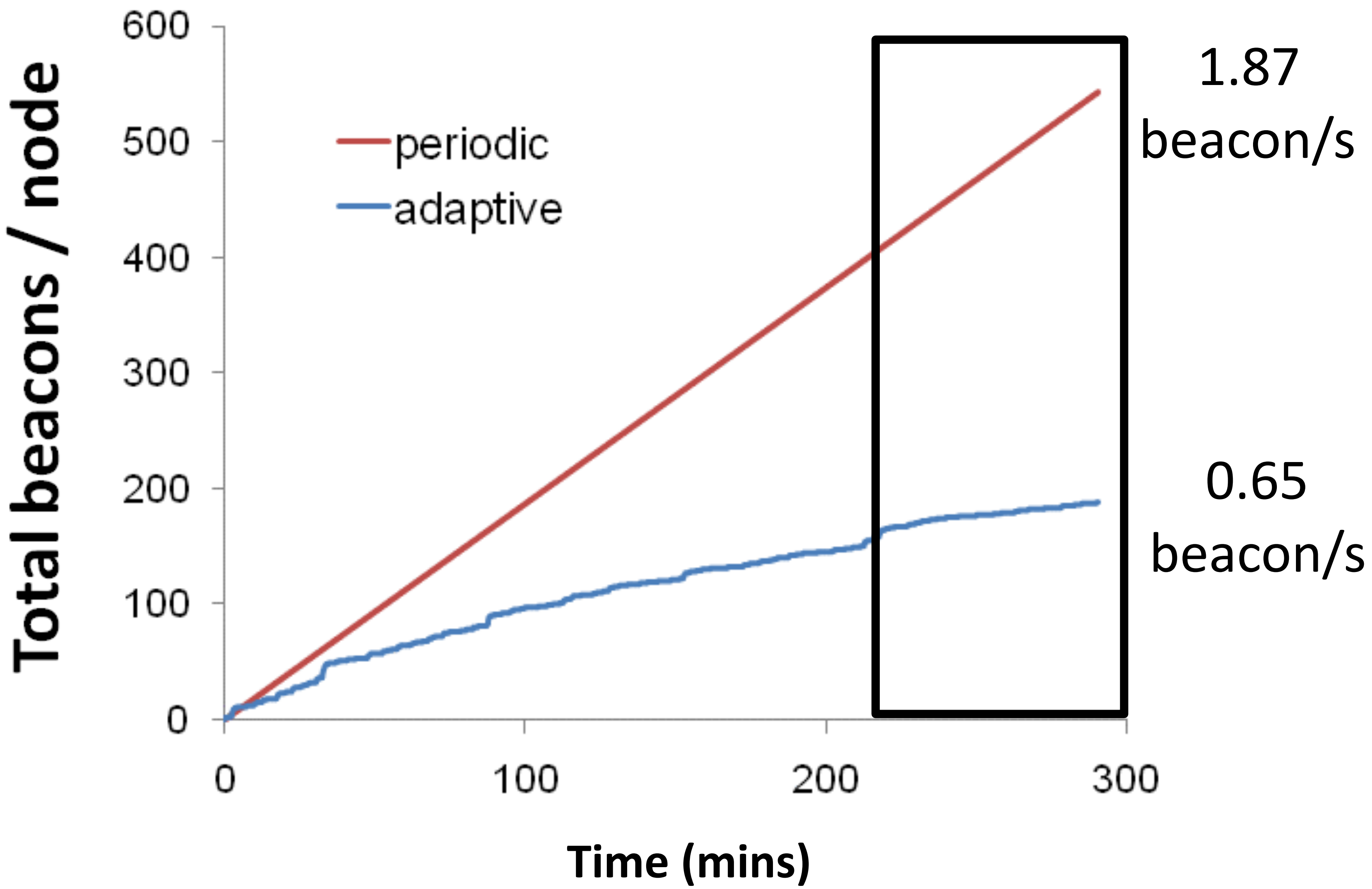
CTP: Control Traffic Timing

- Extend Trickle to time routing beacons
- Reset the interval
  - ETX(receiver) ≥ ETX(sender)
  - Significant decrease in gradient
    - improvement of ≥ 1.5
  - "Pull" bit
    - new node wants to hear beacons from neighbors
  - Optional: automatic reset after some time (e.g. 5 minute
- Beaconing interval between 64ms and 1h

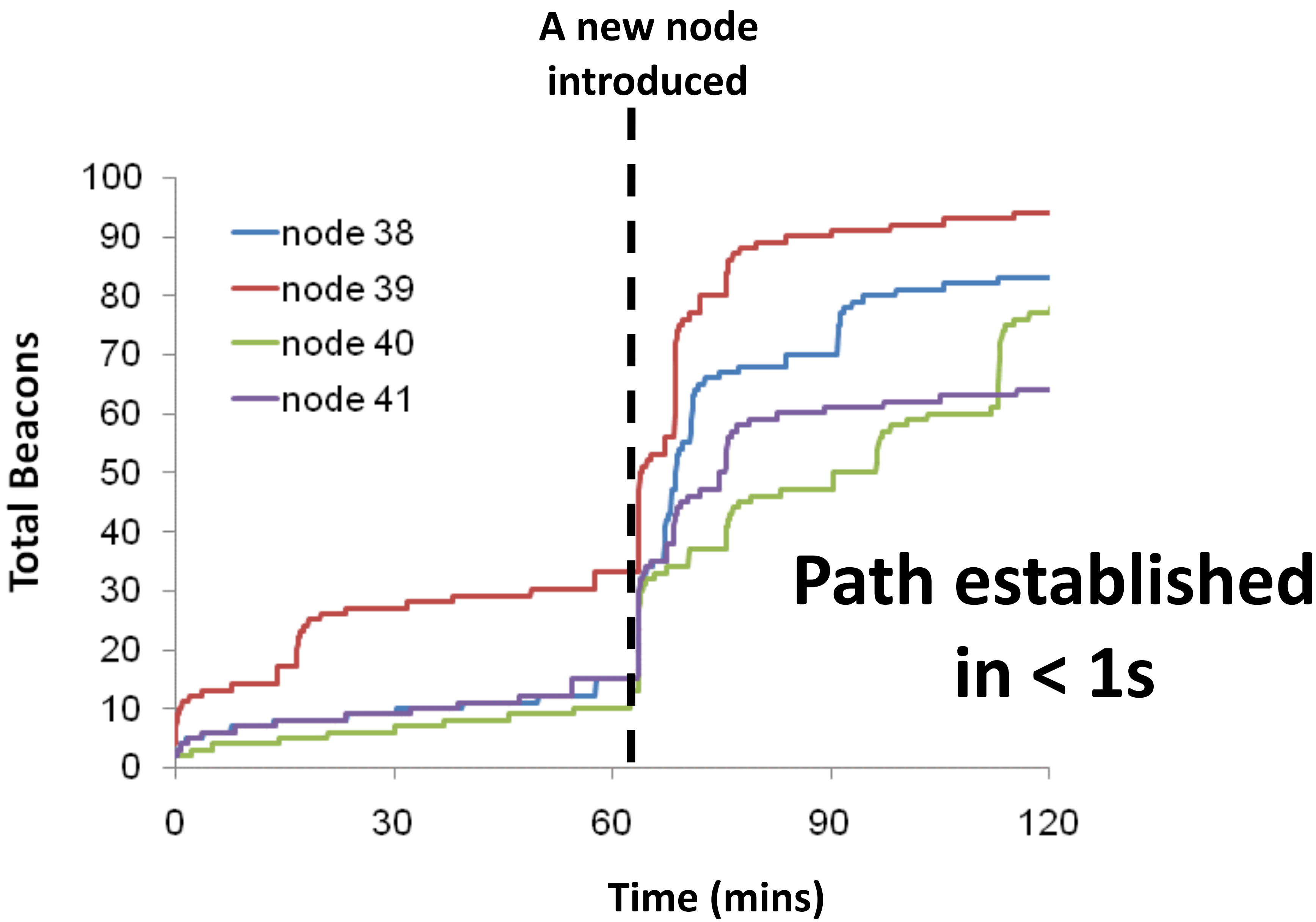
CTP Adaptive Beacon Timing (without automatic reset)



Adaptive vs Periodic Beacons



Node Discovery

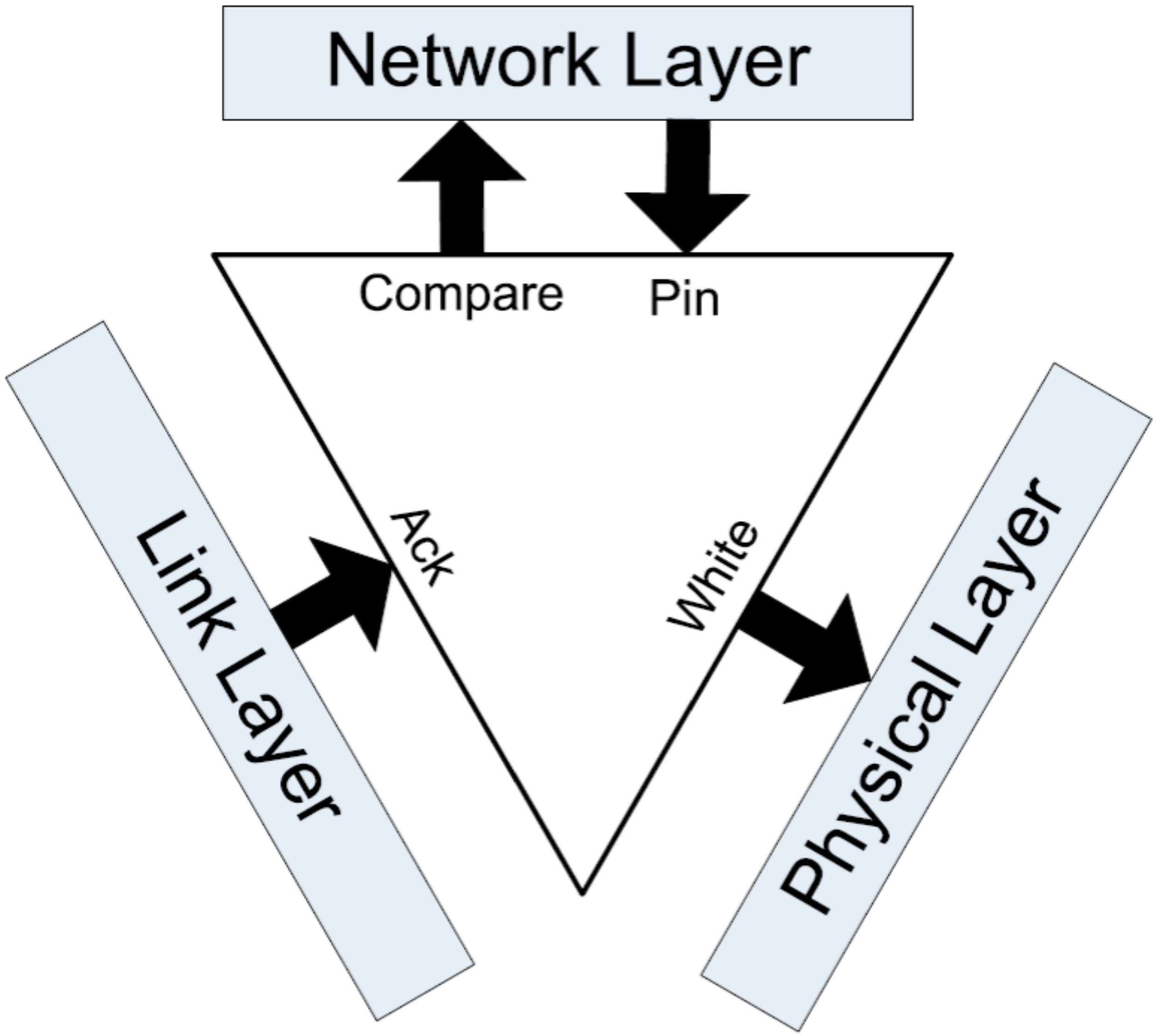


Tutornet

Gnawali, Collection Tree Problem – SenSys 2009 presentation

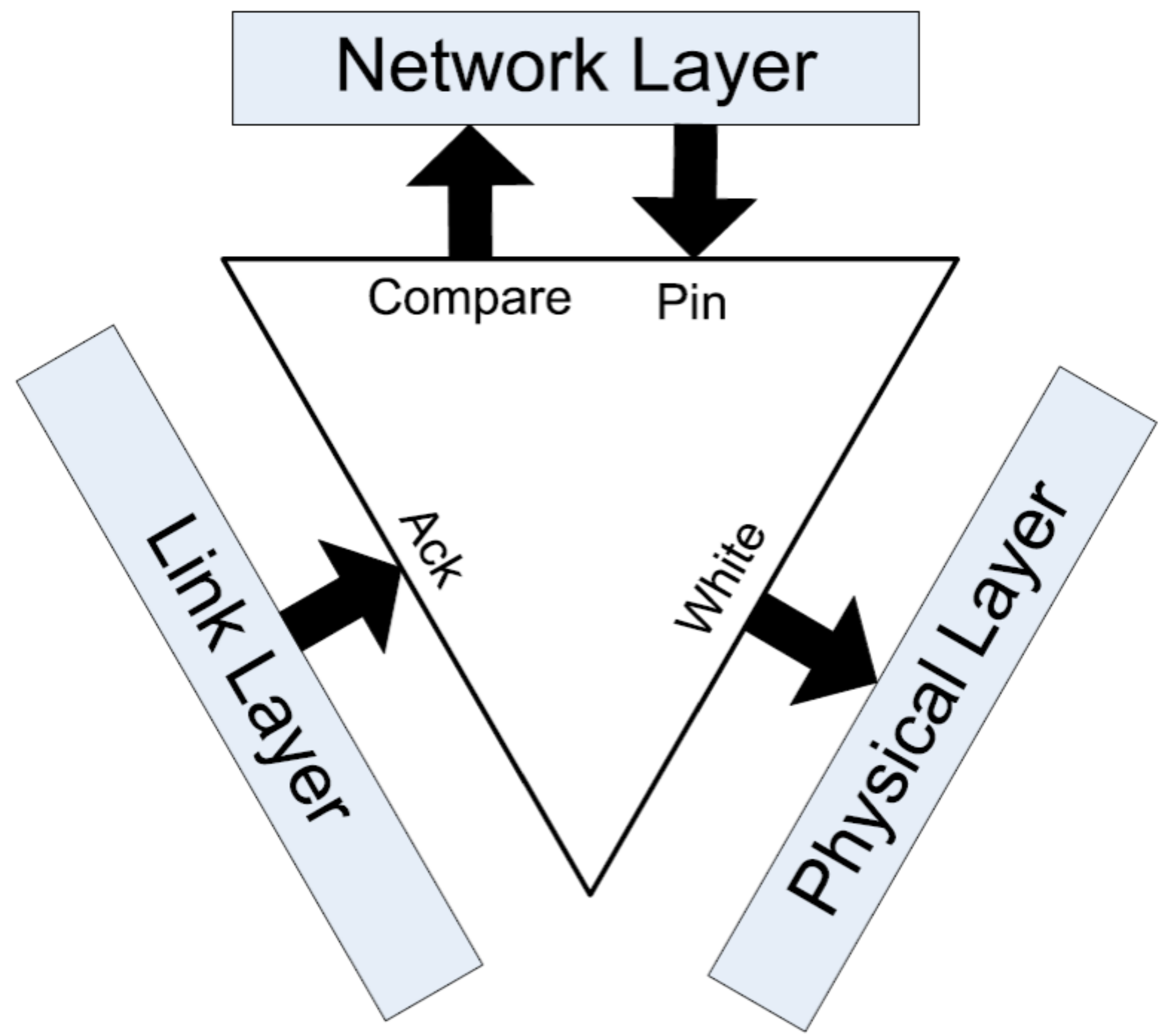
CTP: Link Estimation Layer Information

- Physical Layer
  - LQI: estimate of how easily a received signal can be demodulated by accumulating the magnitude of the error between ideal constellations and the received signal
  - RSSI: Received Signal Strength Indicator (not used)
  - PRR: Packet Reception Ratio (not used)
- Link Layer
  - Number of received Acknowledgements
  - Periodic beaconing (for ETX)
- Network Layer
  - Link on the shortest hop route to sink
  - Geometric information



Link Estimation by Four Bits

- COMPARE
  - Is this a useful link?
- PIN
  - Network layer wants to keep this link in the table
- ACK=1
  - A packet transmission on this link was acknowledged
- WHITE=1:
  - each symbol in the packet has a very low probability of decoding error



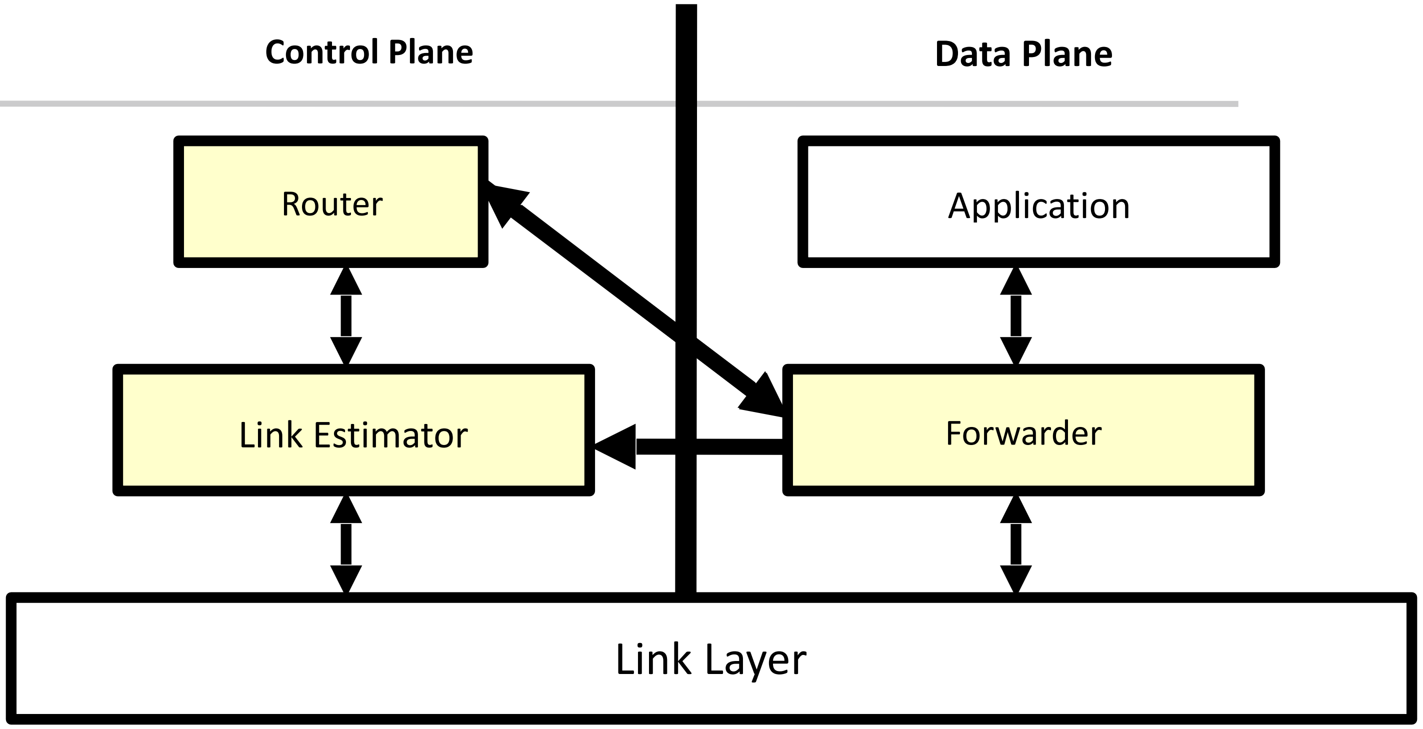
Link Estimation Details

- Network layer
  - receives packet from new link
- Estimator checks
  - white bit is set?
  - asks network layer whether link improves routing -> set compare bit
  - If both bits are set
    - remove an unpinned entry from routing table and replace it with packet
- Use ack bit to compute ETX
  - separately compute ETX for unicast and broadcast value every k_u or k_b (~5) packets by k_u/a
    - a: number of acknowledgements
  - Average by windowed exponentially weighted moving average over reception probabilities (EWMA)
  - ETX = 1/average
  - Combine unicast and broadcast ETX by a second EWMA

CTP: Routing

- Prevent fast route changes
  - by hysteresis in path selection
  - switch only routes if other route is significantly better
    - i.e. ETX is at least 1.5 lower
- Looping packets
  - are not dropped
  - but paused
  - recognized by the Transmit Cache
  - and resent

CTP: Data Plane



- Concepts
  - THL: time has lived field (instead of TTL)
  - Aggressive retransmission strategy
    - 32 tries per packet
- Per-Client Queueing
  - client = application or service
  - one outstanding packet per client
- Hybrid Send Queue
  - lower level FIFO queue of route-through and generated packets
  - size = #clients + forward-buffer-size
- Transmit Timer
  - Prevent self-interference by waiting on the expectation two packet times between transmissions
  - i.e. choose random time in waits in the range of (1.5*p*,2.5*p*)
    - where p is the packet time
- Transmit Cache
  - False (negative/positive) acknowledgments
  - Distinguish duplicate packets from loop packets (using THL)
  - Looping packets are forward to repair routing tables
  - Remembering is important to identify duplicates (size: 4 packets)

CTP: Experiments at Stanford

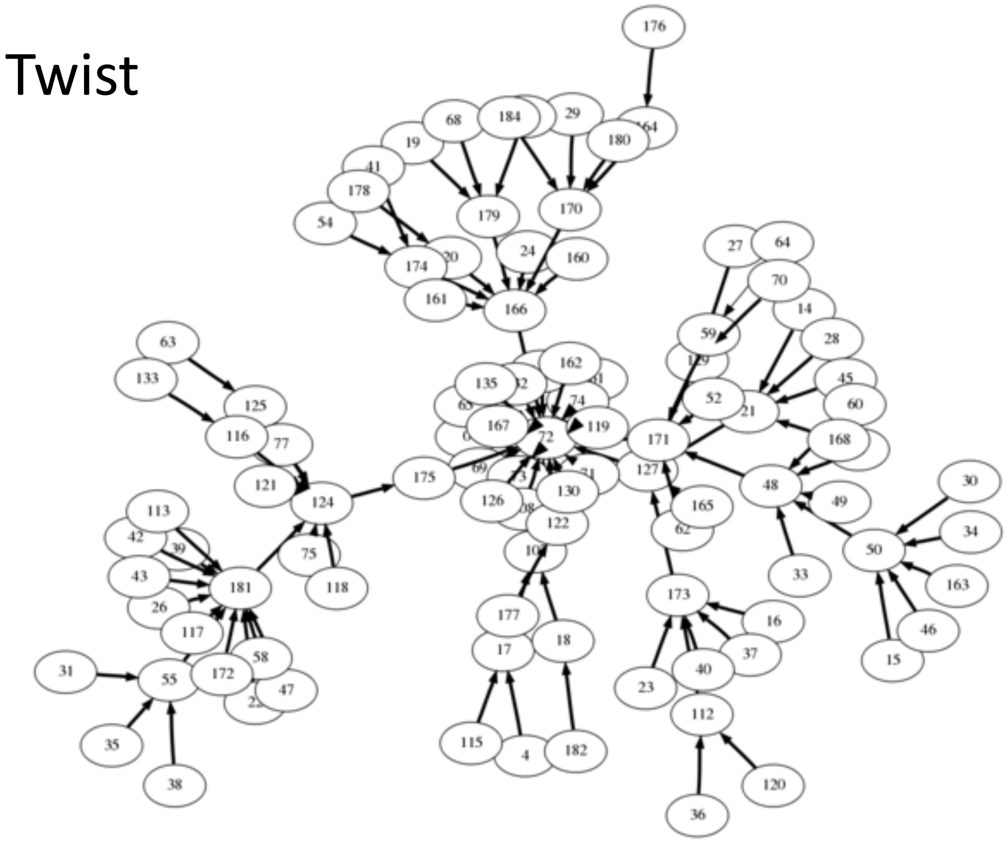
Testbed	Platform	Nodes	Size m^2 or m^3	Degree Min	Degree Max	PL	Cost	Cost / PL	Churn / node·hr
Tutornet (16)	Tmote	91	50×25×10	10	60	3.12	5.91	1.90	31.37
Wymanpark	Tmote	47	80×10	4	30	3.23	4.62	1.43	8.47
Motelab	Tmote	131	40×20×15	9	63	3.05	5.53	1.81	4.24
Kansei[a]	TelosB	310	40×20	214	305	1.45	–	–	4.34
Mirage	Mica2dot	35	50×20	9	32	2.92	3.83	1.31	2.05
NetEye	Tmote	125	6×4	114	120	1.34	1.40	1.04	1.94
Mirage	MicaZ	86	50×20	20	65	1.70	1.85	1.09	1.92
Quanto (15)	Quanto	49	35×30	8	47	2.93	3.35	1.14	1.11
Twist	Tmote	100	30×13×17	38	81	1.69	2.01	1.19	1.01
Twist	eyesIFXv2	102	30×13×17	22	100	2.58	2.64	1.02	0.69
Vinelab	Tmote	48	60×30	6	23	2.79	3.49	1.25	0.63
Indriya	TelosB	126	66×37×10	1	36	2.82	3.12	1.11	0.05
Tutornet	Tmote	91	50×25×10	14	72	2.02	2.07	1.02	0.04
Blaze[b]	Blaze	20	30×30	9	19	1.30	–	–	–

[a] Packet cost logging failed on ten nodes.
[b] Blaze instrumentation does not provide cost and churn information.
Note: Cost is transmissions per delivery and PL is path length, the average number of hops a data packet takes. Cost/PL is the average transmissions per link. All experiments are on 802.15.4 channel 26 except for the Quanto testbed (channel 15) and one of the Tutornet experiments (channel 16).

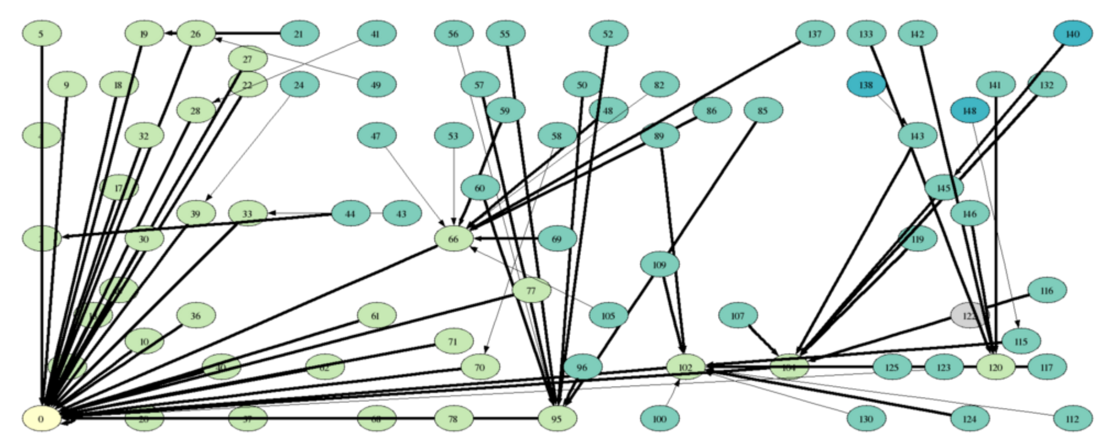
CTP: High end-to-end delivery ratio

Testbed	Delivery Ratio
Wymanpark	0.9999
Vinelab	0.9999
Tutornet	0.9999
NetEye	0.9999
Kansei	0.9998
Mirage-MicaZ	0.9998
Quanto	0.9995
Blaze	0.9990
Twist-Tmote	0.9929
Mirage-Mica2dot	0.9895
Twist-eyesIFXv2	0.9836
Motelab	0.9607

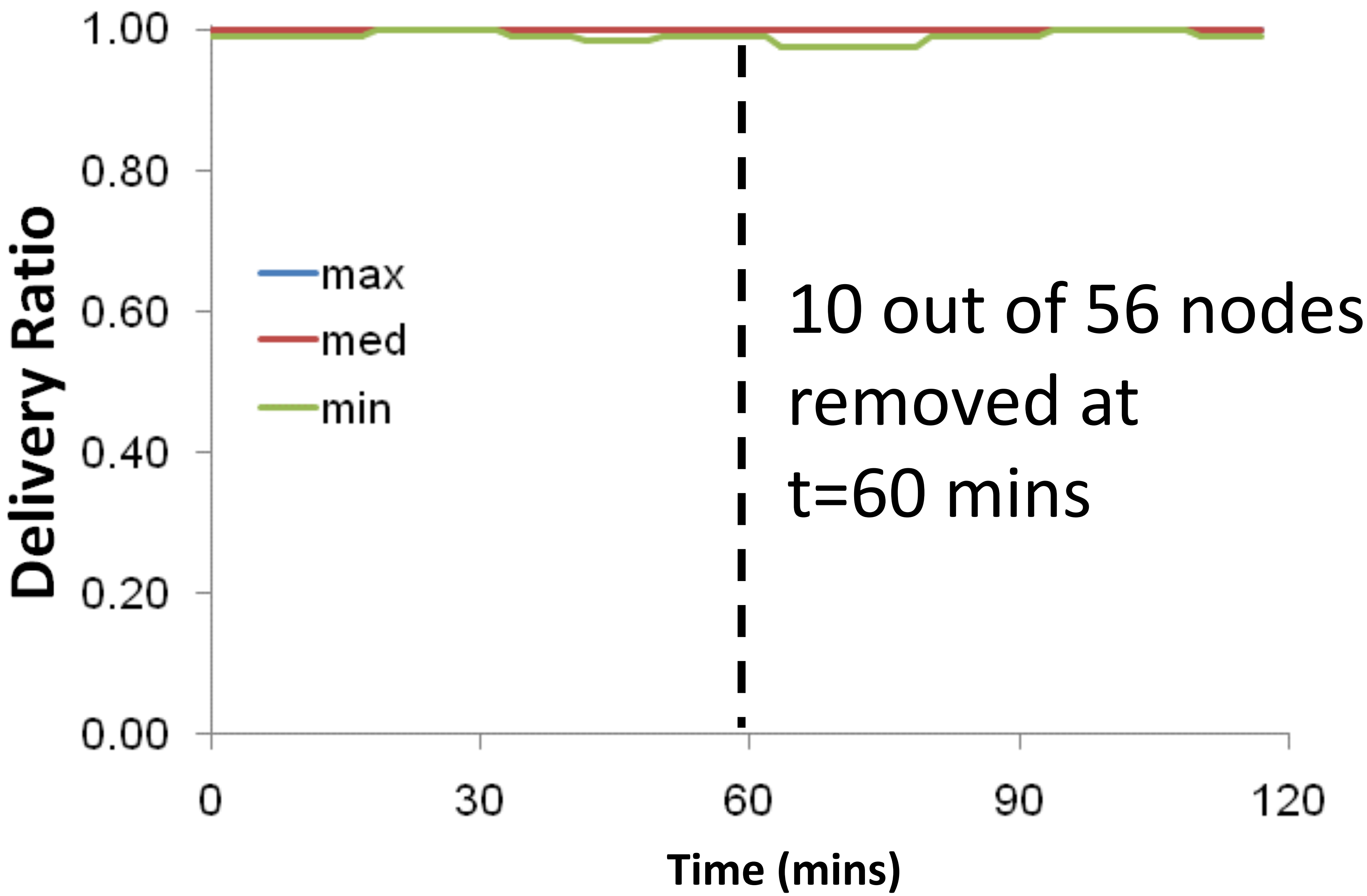
Twist



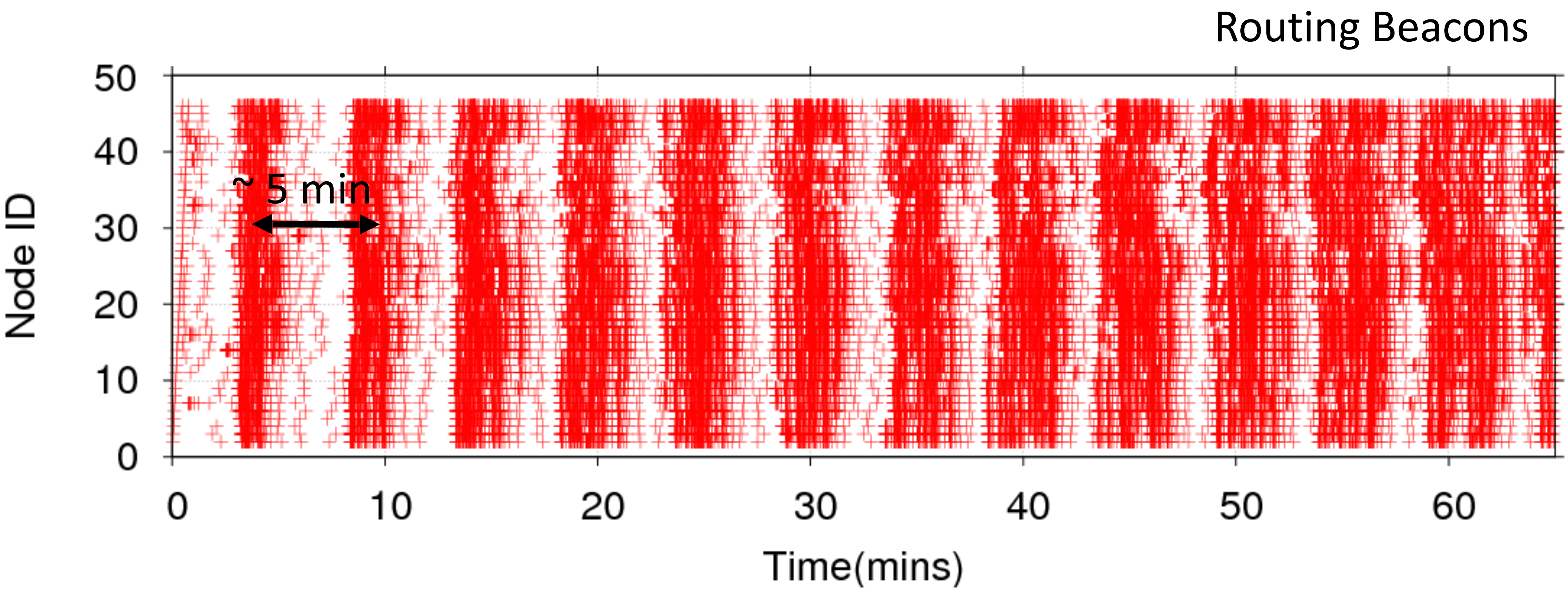
Mirage



CTP: No disruption in packet delivery



CTP: Nodes reboot every 5 mins



CTP Performance

- Reliability
  - Delivery ratio > 90% in all cases
- Efficiency
  - Low cost and 5% duty cycle
- Robustness
  - Functional despite network disruptions

Routing Protocol for Low power and Lossy Networks (RPL)

- Literature
  - IETF RFC 6550, RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks, Winter, Thubert, Brandt, Hui, Kelsey, Levis, Pister, Struik, Vasseur, Alexander, March 2012
- Designed for Low-power and Lossy Networks (LLN)
  - limited processing power, memory, energy
  - interconnected by lossy links, low data rates
  - traffic patterns
    - Multipoint to point (convergecast)
    - Point to multipoint (multicast)
    - point to point (unicast)

- Design Principles
  - Routing Metric is variable
  - bidirectional links required
  - uses Trickle for data dissemination
  - uses DAG as basic topology

CoNe Freiburg

RPL: Terms

- DAG: directed acyclic graph
  - routed towards root nodes
- DAG root = sink of a DAG = LBR (LLN Border Router)
- DODAG: destination-oriented DAG
  - DAG with single root
- Rank:
  - partial order in corresponding with the DODAG
- Grounded DODAG
  - DODAG where RPL can find the root
- Floating DODAG
  - A DODAG where there is no path to the root because wrong pointers

RPL: Ideas

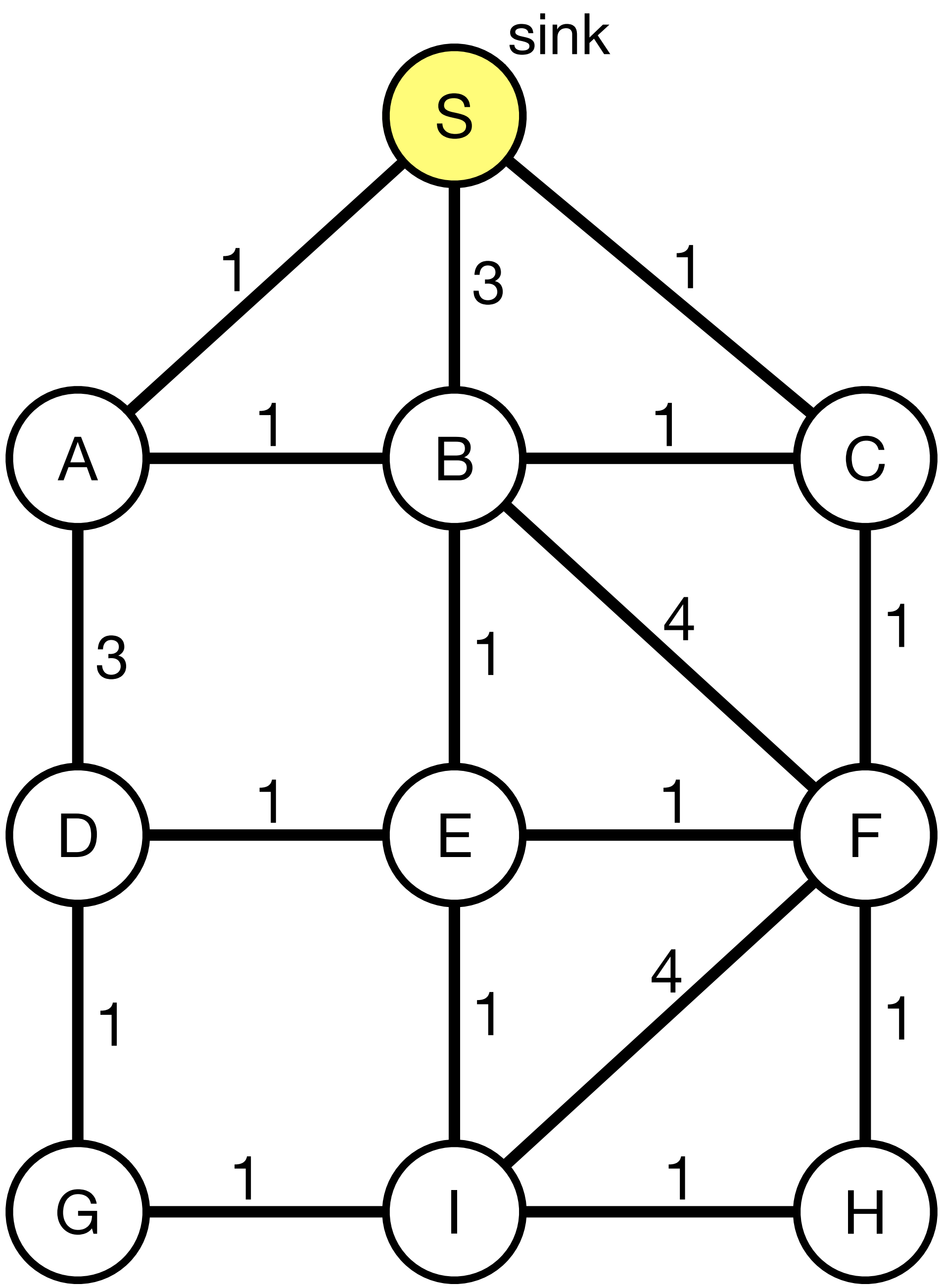
- Convergecast (MP2P)
  - DAG with multiple successors if possible
  - DAG defined by specific metrics (e.g. ETX, latency, DAG rank/hop count)
  - Least expensive paths
- Multicast
  - DAG also used for P2MP flows
- MP2P and P2MP for P2P (unicast)
- DAG
  - Depth (aka. rank), i.e. cost towards the sink (root)
  - Rank defines position in the DAG

RPL: MP2P Forwarding

- Forward to nodes of lesser rank
  - avoids loops
  - loops may occur when the metric has changed or nodes leave due to rank inconsistency
  - use redundancy
- Forward to nodes of equal rank
  - not using DAG links
  - if forwarding to lesser rank (DAG-link) fails
- Do not forward to nodes of higher rank
  - causes loops

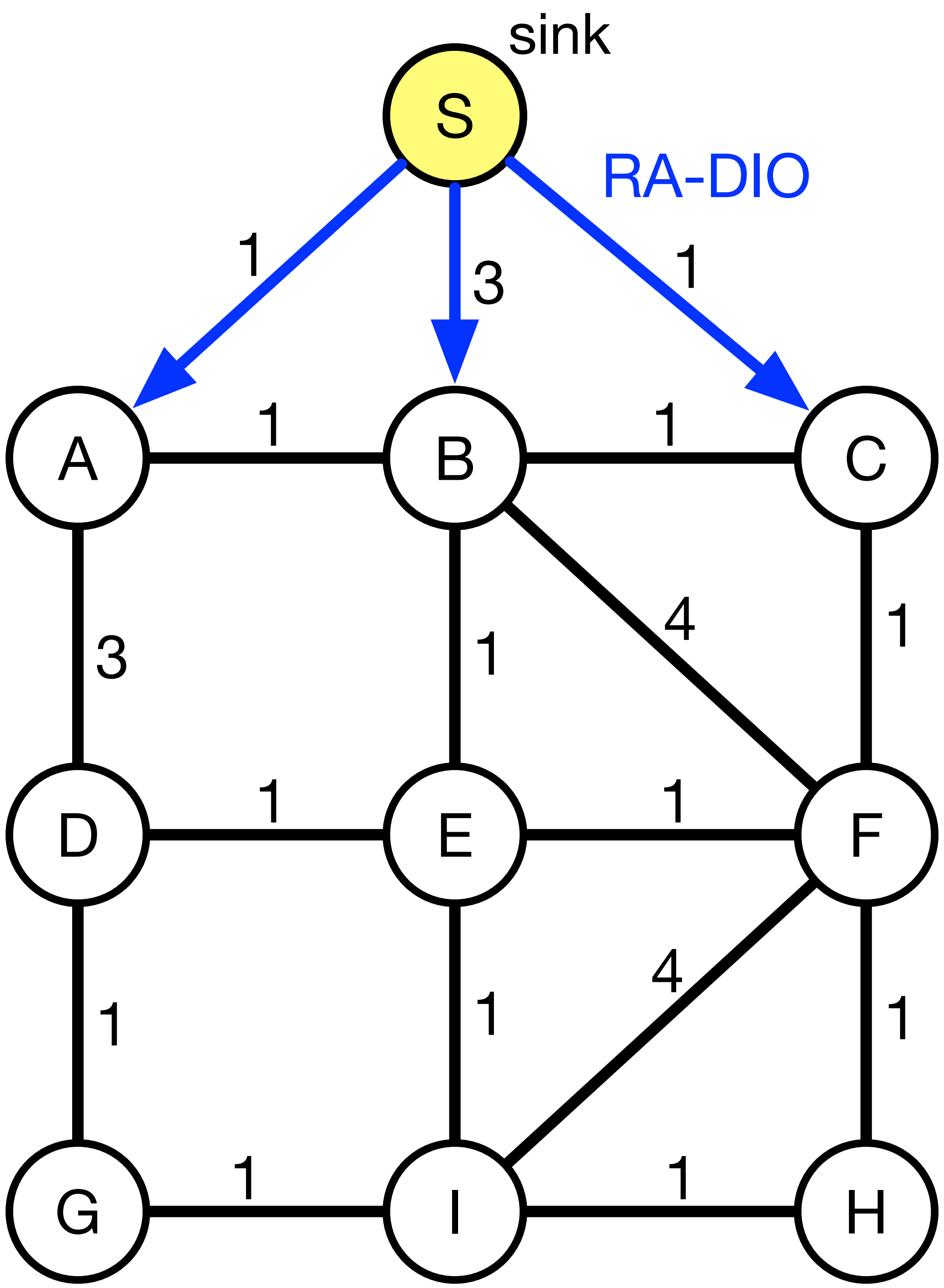
RPL: DAG Construction

- Given LLN with ETX
  - ETX should be stable enough for route computation
  - Nodes are bidirectional and ETX is known at both ends
  - Or use any other comparable metric, e.g. hop distance
- Minimize ETX



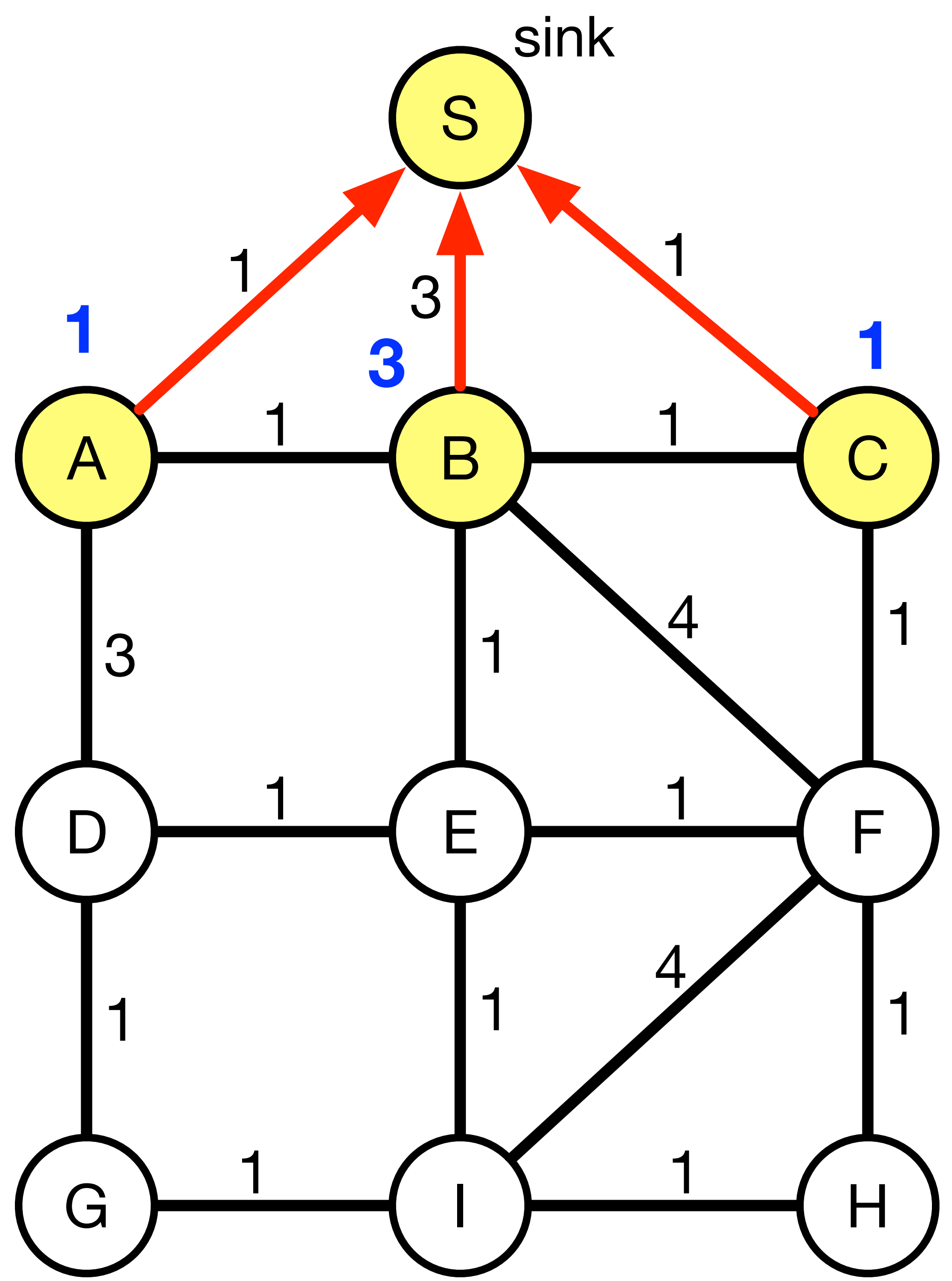
RPL: DAG Construction

- Sink broadasts RA-DIO
  - Router Advertisement (RA)
  - DODAG Information Object (DIO)
- Nodes A, B, C
  - receive RA-DIO
  - join DAG rooted to sink (LBR)



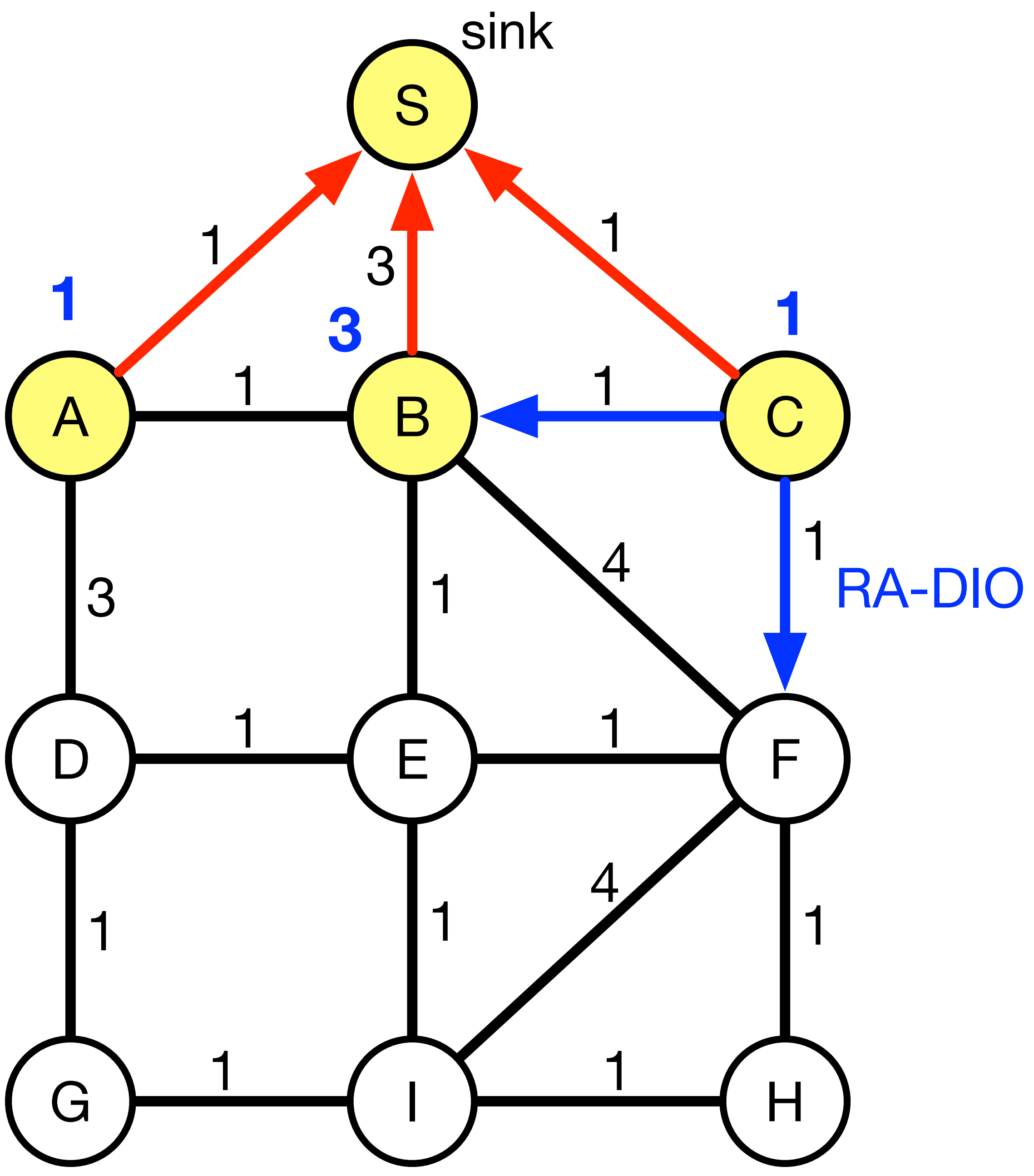
RPL: DAG Construction

- Nodes A, B, C
  - receive RA-DIO
  - join DAG rooted to sink (LBR)
  - compute rank



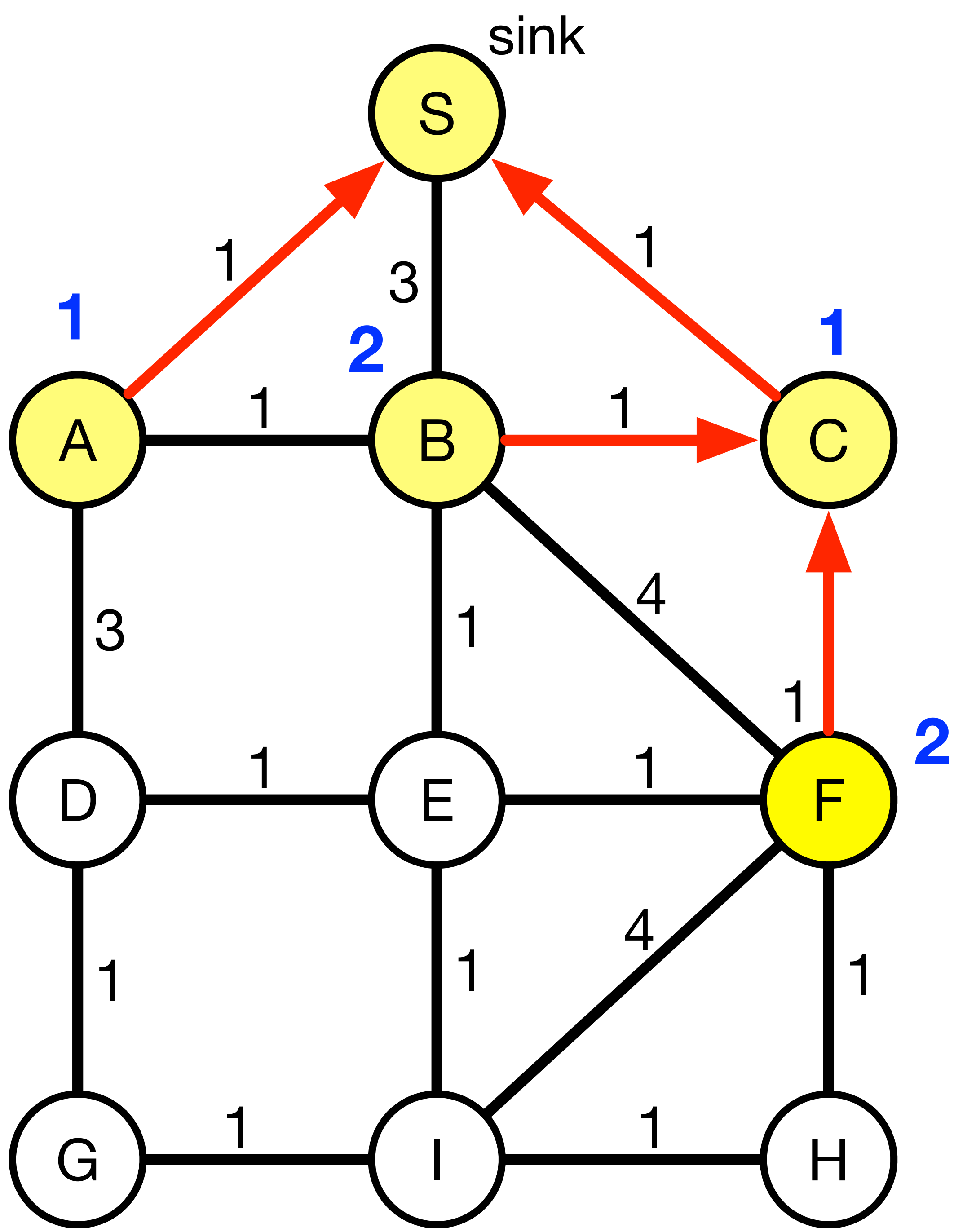
RPL: DAG Construction

- Node C
  - send RA-DIO
- Nodes B,F receive it
  - recompute rank



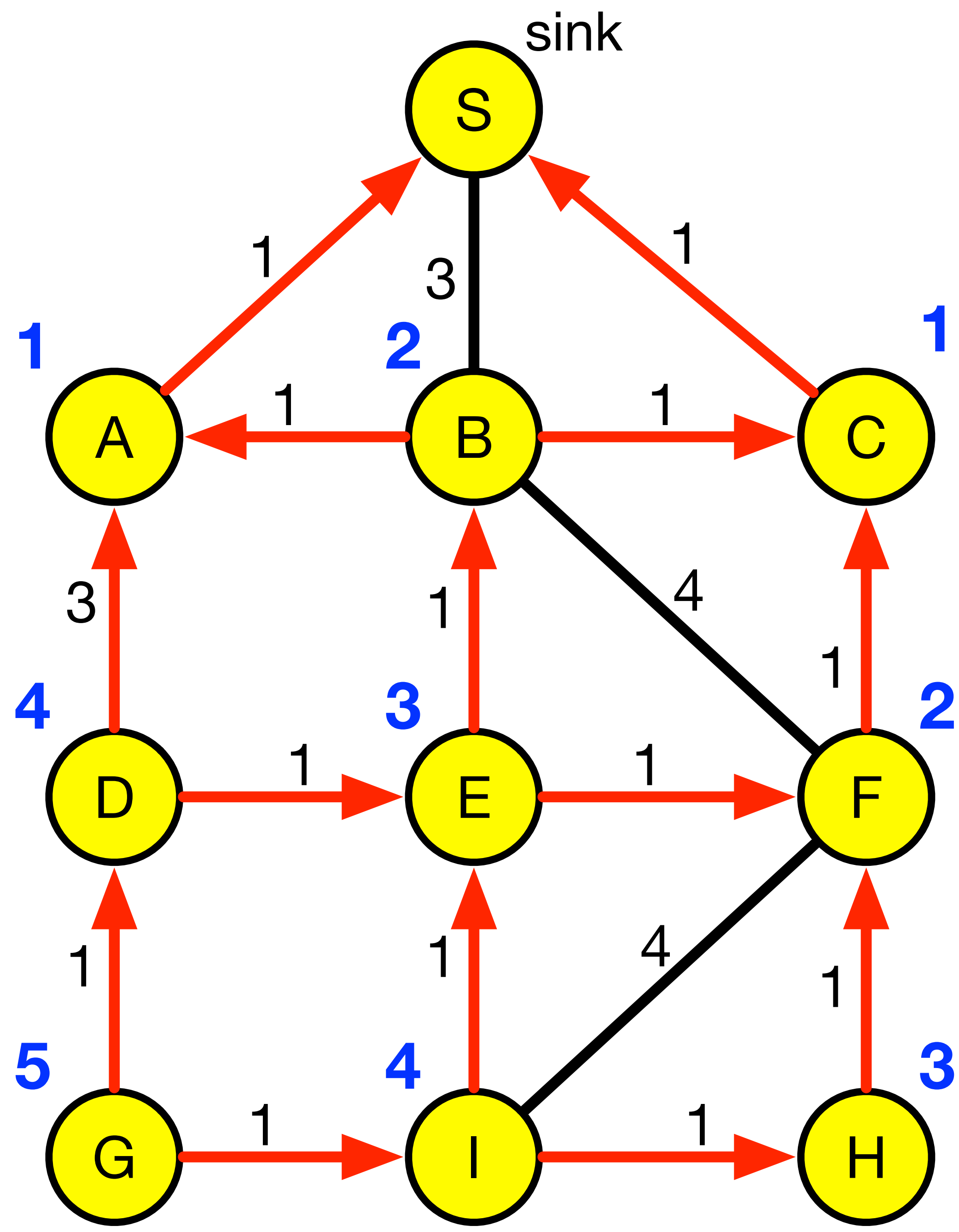
RPL: DAG Construction

- Nodes B and F
  - recompute rank
- Node B
  - redirects to C
- Node F
  - joins the DAG



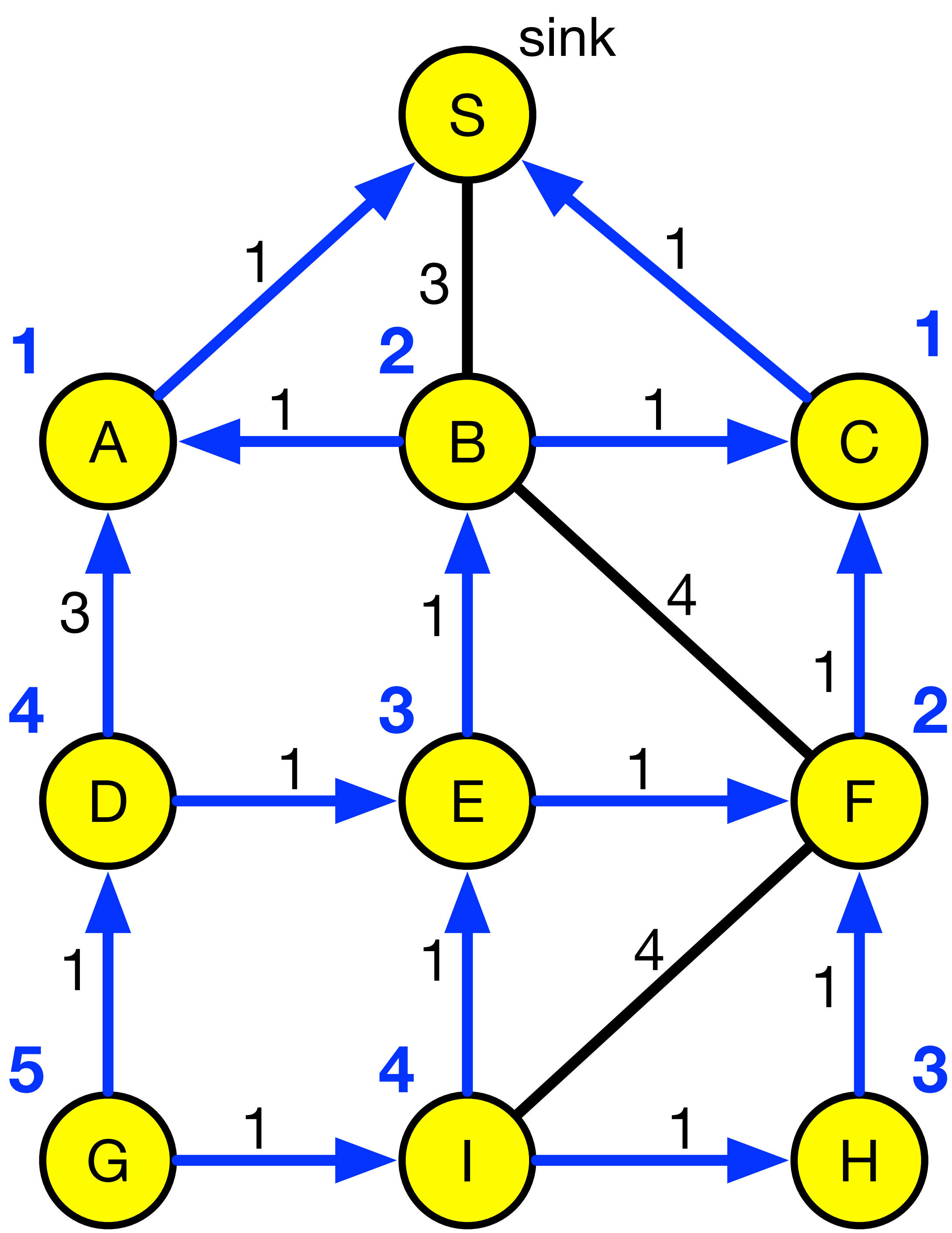
RPL: DAG Construction

- Final network
- Rank is rounded
  - such that multiple paths exist
- Maintenance is continued
  - RA (router announcements) use Trickle algorithm



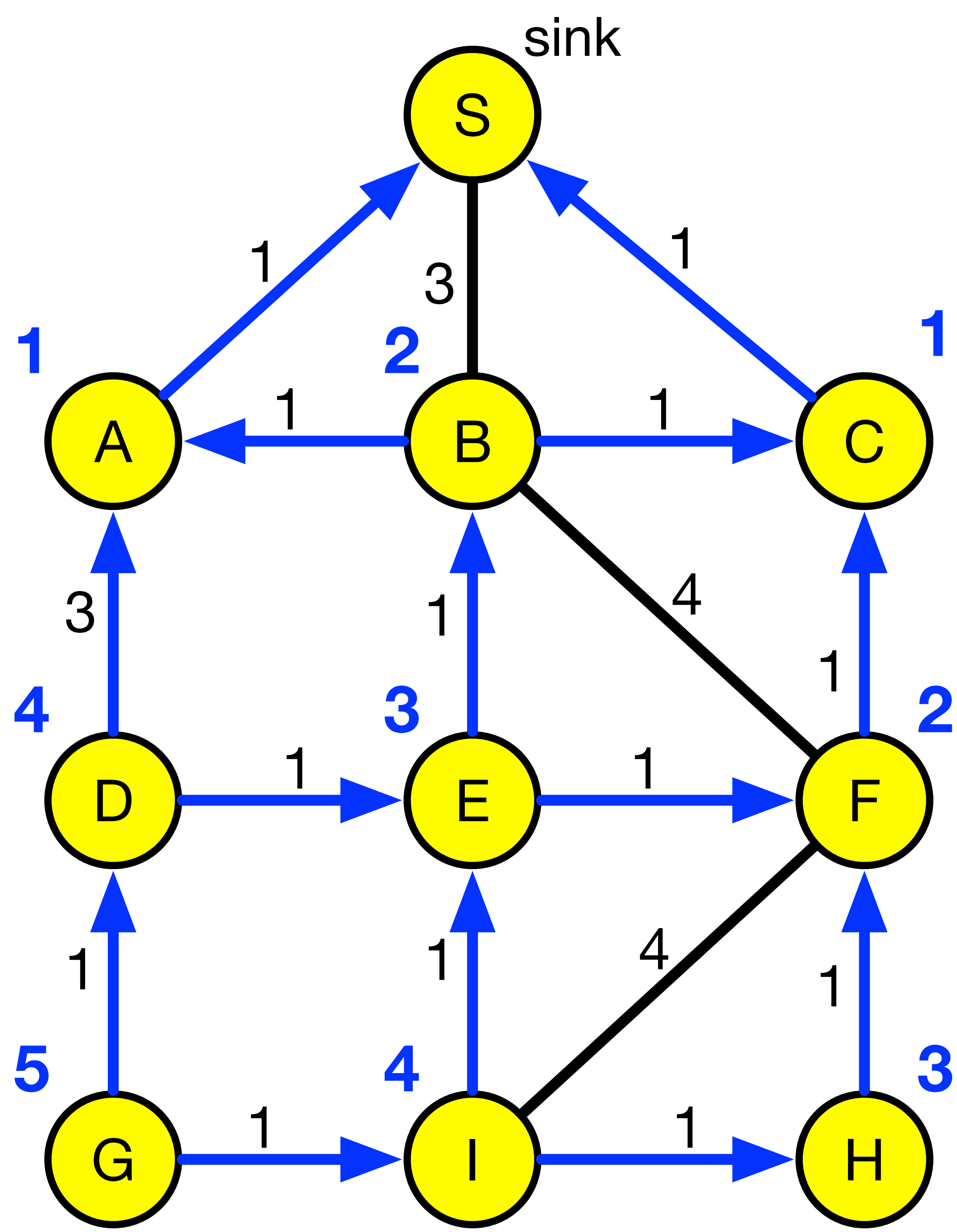
RPL: Convergecast (MP2P)

- MP2P traffic flows along DAG links
  - toward sink/DAG root/LBR



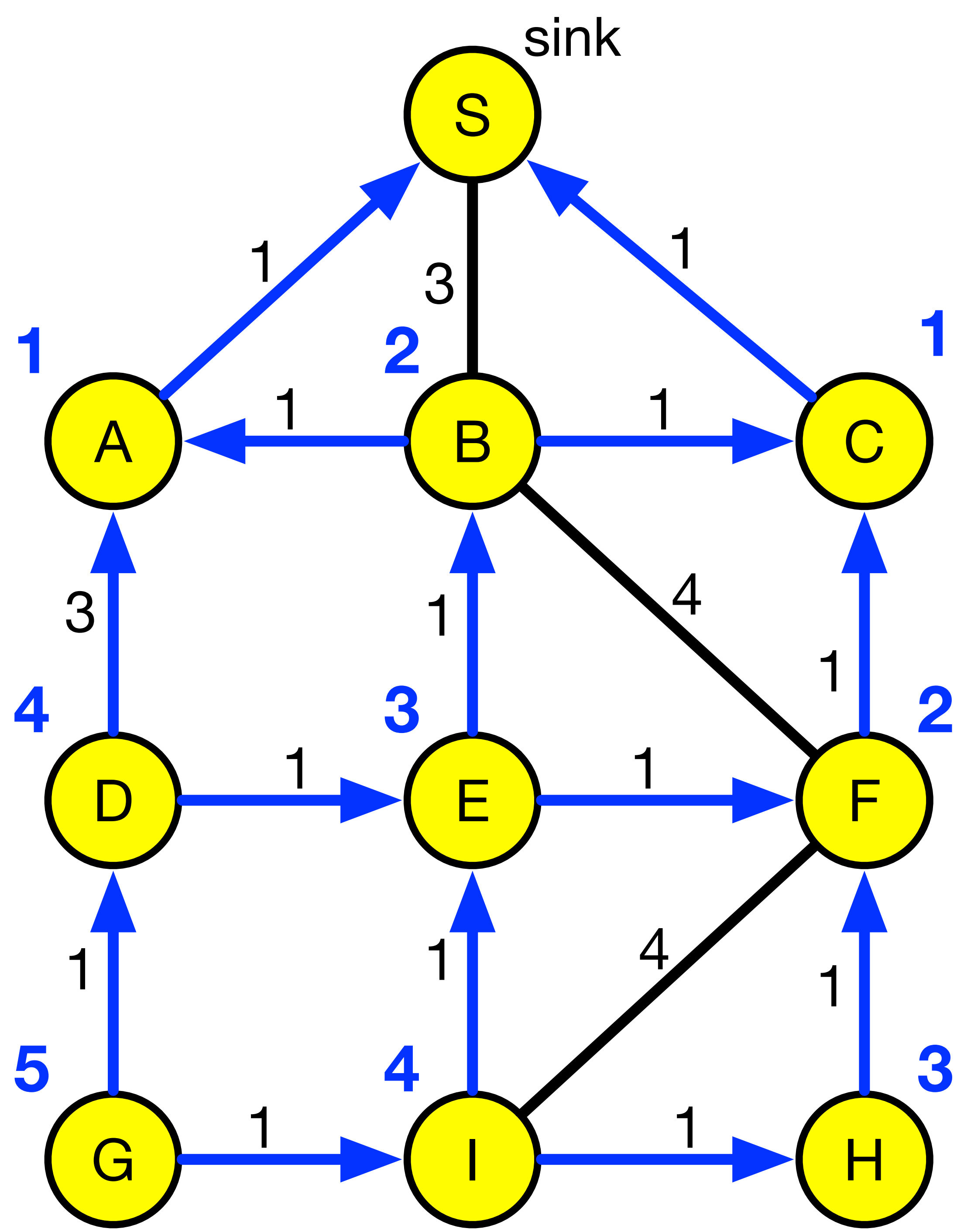
RPL: Multicast (P2MP)

- Destination advertisements object (DAO) message
  - build up routing state outwards from sink
  - toward sink/DAG root/LBR
- Two modes supported
  - Source routing (non storing case)
    - sink gathers information
  - Routing table (storing case)



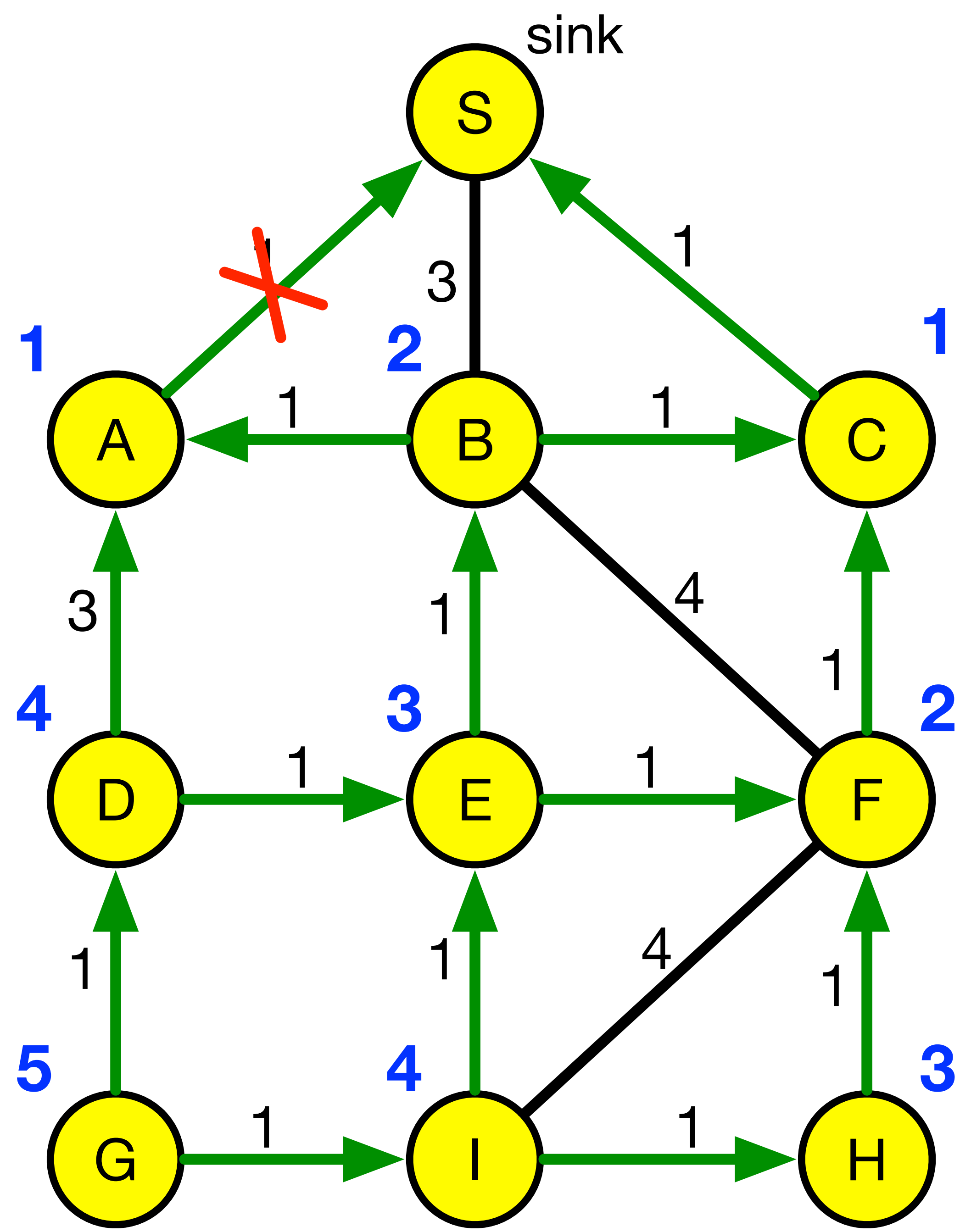
RPL: Unicast (P2P)

- Unicast message
  - travel towards the sink (up)
  - and then towards the target node (down)
- Non-storing case
  - message travels to sink and is sent via source routing
- Storing case
  - message travels up until a node knows the target



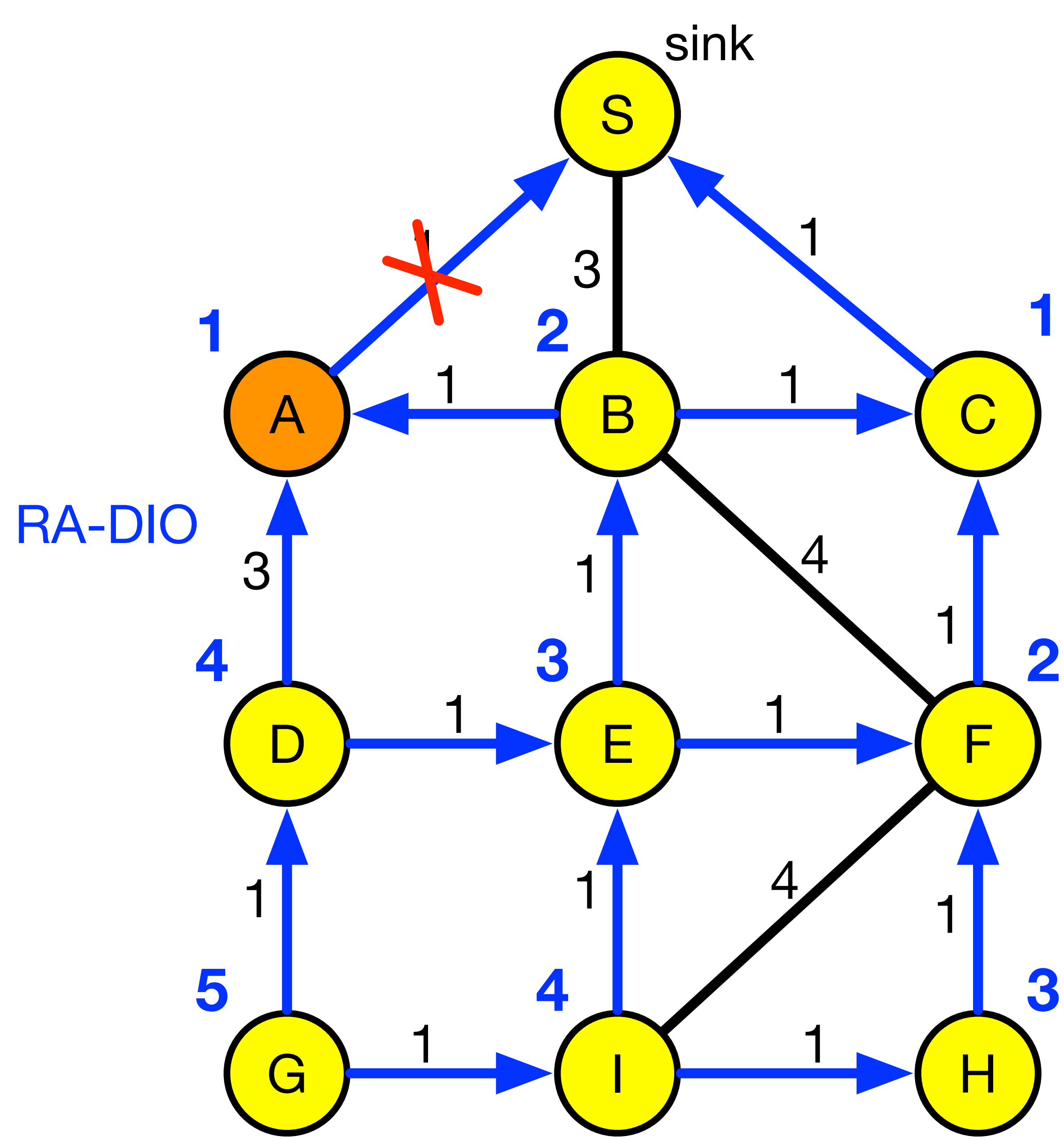
RPL: Loop Avoidance

- Node A looses connection towards sink
  - with no alternatives
- A sends out RA-DIO
  - and becomes root of a **floating DAG**
- Successors of A flood RA-DIO to inform all dependent nodes



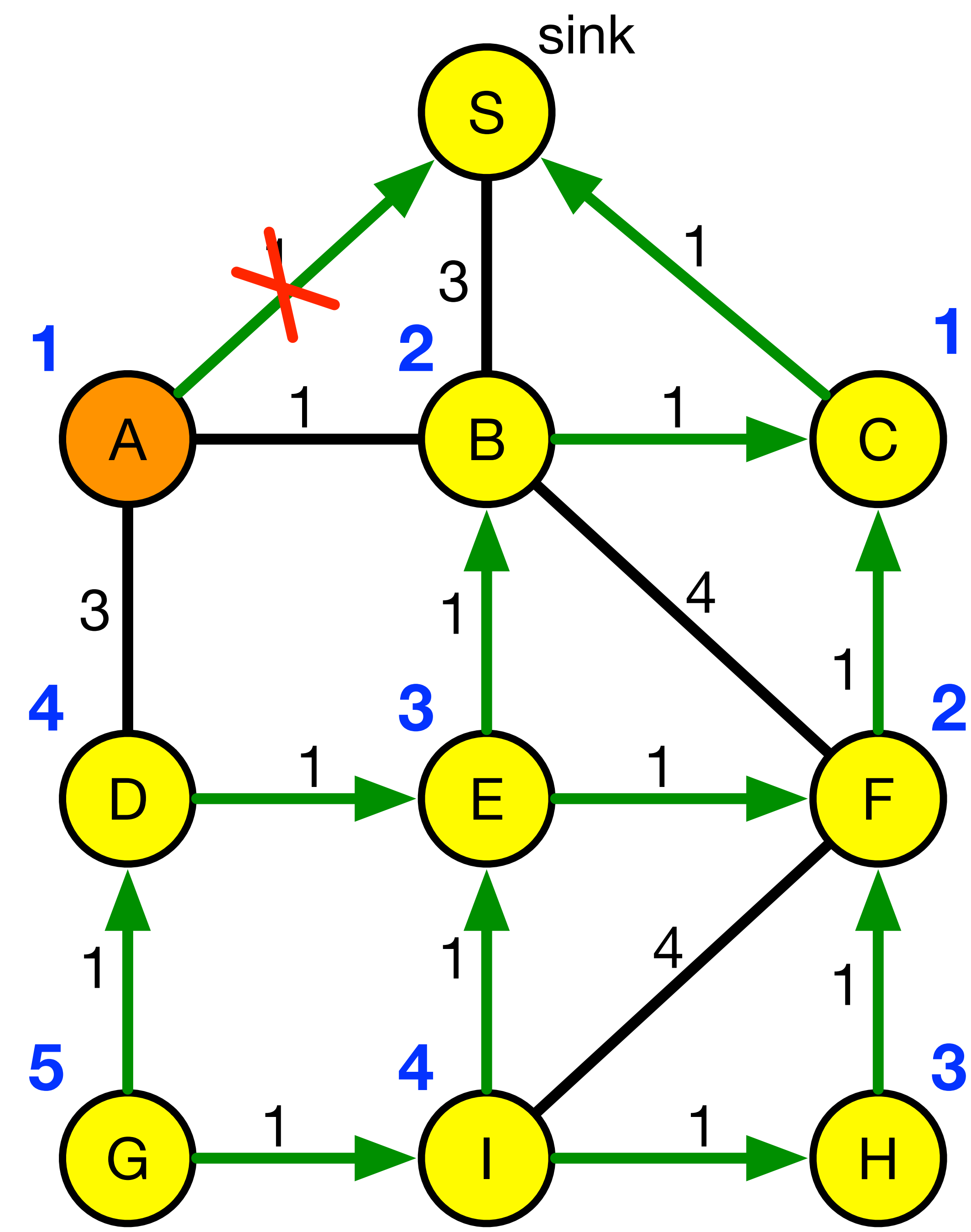
RPL: Floating DAG

- Floating DAG
  - does not need to satisfy the DAG constraint
- Nodes A becomes floating DAG
- Node B and D have alternate parents and remove links towards A



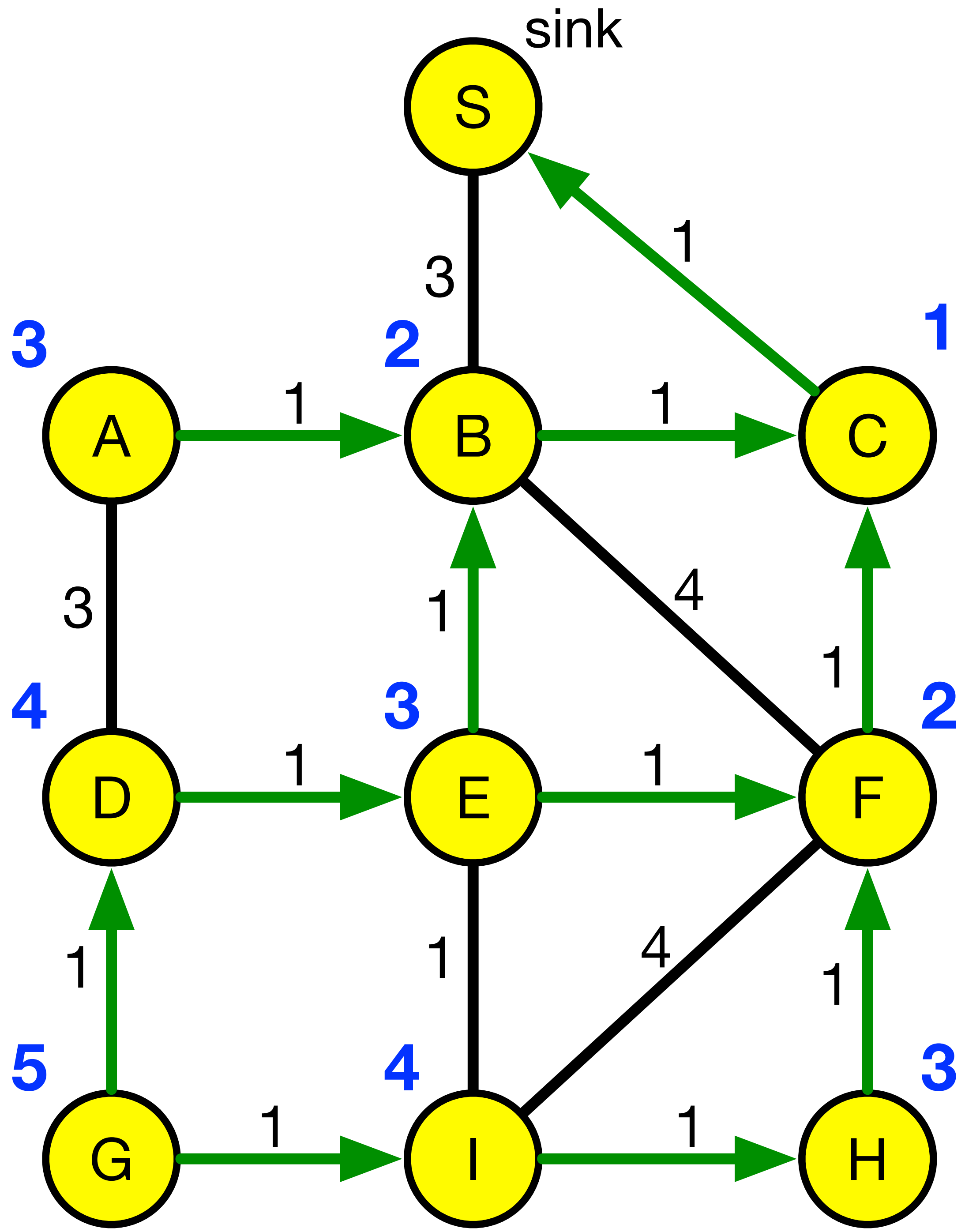
RPL: Floating DAG

- Node B will advertise with RA-DIO
- A joins DAG again



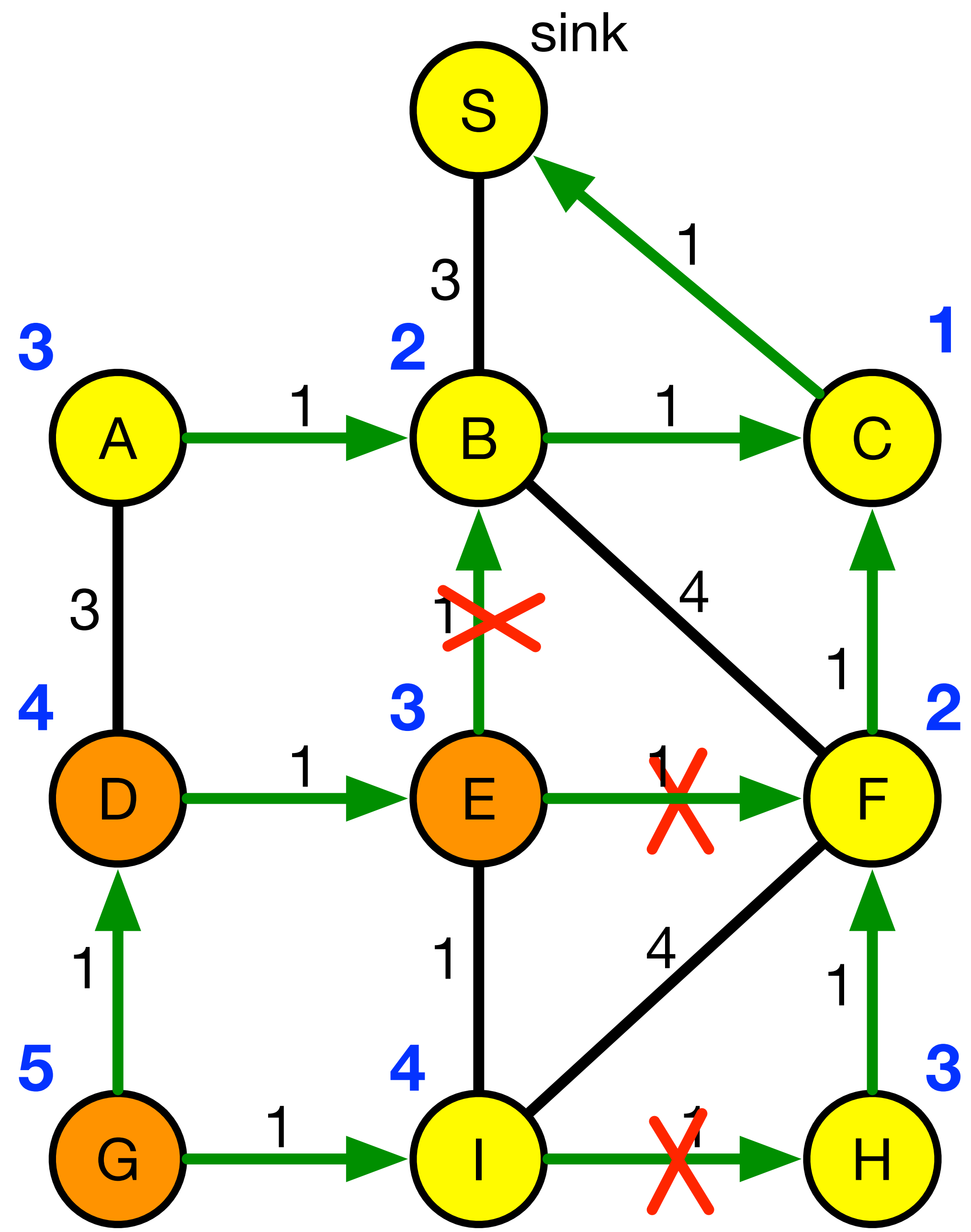
RPL: Floating DAG

- Node B will advertise with RA-DIO
- A joins DAG again



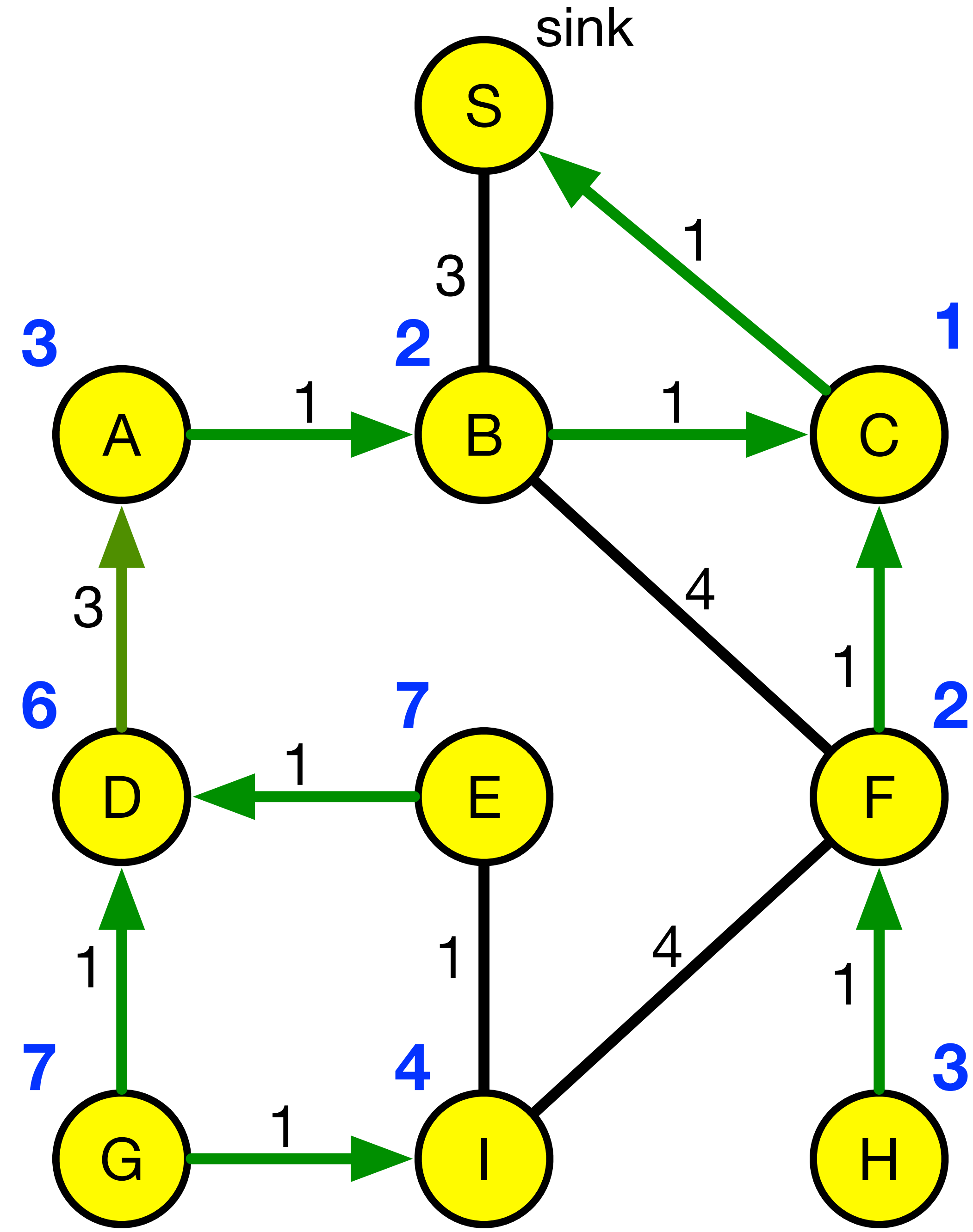
RPL: Floating DAG

- Now link from E to B fails
- Nodes E,D,G become floating DAG
  - informed by E
- Nodes I,F
  - have alternative routes



RPL: Floating DAG

- Assume A advertises link
- D links to A
  - and forwards info to E and G
- Nodes E, G now repair links
- Eventually, again the optimal network will be found



RPL - Conclusion

- specified only for IPv6
- based on Distance Vector
- produces a stable DAG
  - well suited for traffic directions up and down
- problematic for other traffic directions
- Critical evaluation:
  - Clausen, T.; Herberg, U.; Philipp, M.; "A critical evaluation of the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL)", Wireless and Mobile Computing, Networking and Communications (WiMob), 2011 IEEE 7th International Conference on , vol., no., pp. 365-372, 10-12 Oct. 2011
  - assumes bi-directional connections
  - not completely specified
  - Loops are in real experiments a big unresolved problem

Wireless Sensor Networks
6. WSN Routing

Christian Schindelhauer
Technische Fakultät
Rechnernetze und Telematik
Albert-Ludwigs-Universität Freiburg
Version 30.05.2016