Collection Tree Protocol

A look into datapath validation and adaptive beaconing.

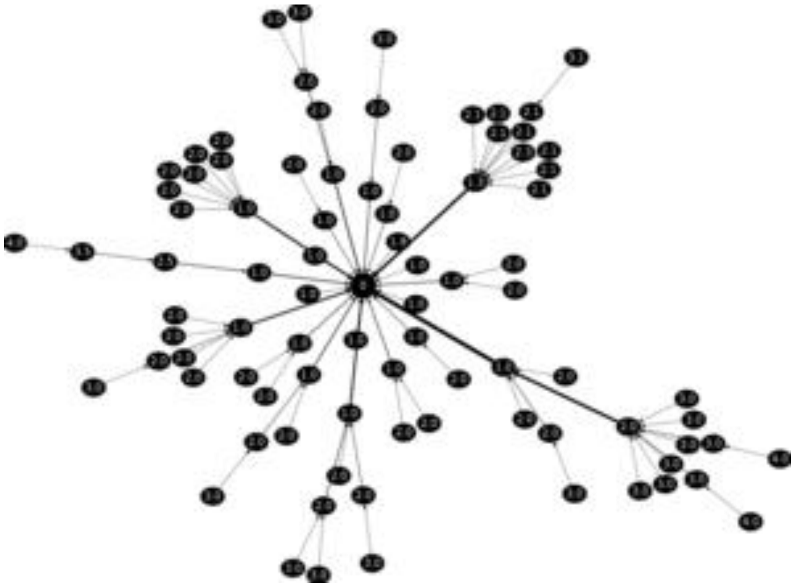
Speaker: Martin Lanter

Collection Protocols

- Why do we need collection protocols?
  - *"Collecting data at a base station is a common requirement of sensor network applications. The general approach used is to build one or more collection trees, each of which is rooted at a base station. When a node has data which needs to be collected, it sends the data up the tree, and it forwards collection data that other nodes send to it."* [TinyOS TEP 119]
- Requirements
  1. Reliability: > 90% of packets
  2. Robustness
  3. Efficiency: Use a minimum of transmissions
  4. Hardware Independence

Collection **T**ree **P**rotocol (CTP)

- Is a protocol that computes routes to one or more sinks
- Builds and maintains minimum cost tree(s) with the sink(s) as root



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

Challenges for CTP

- Link dynamics
  - Wireless links can have coherence as small as 500 ms

http://www.tdwess.de/nepal/nepal.htm

- Routing Inconsistencies
  - Inconsistencies/routing changes might lead to loops

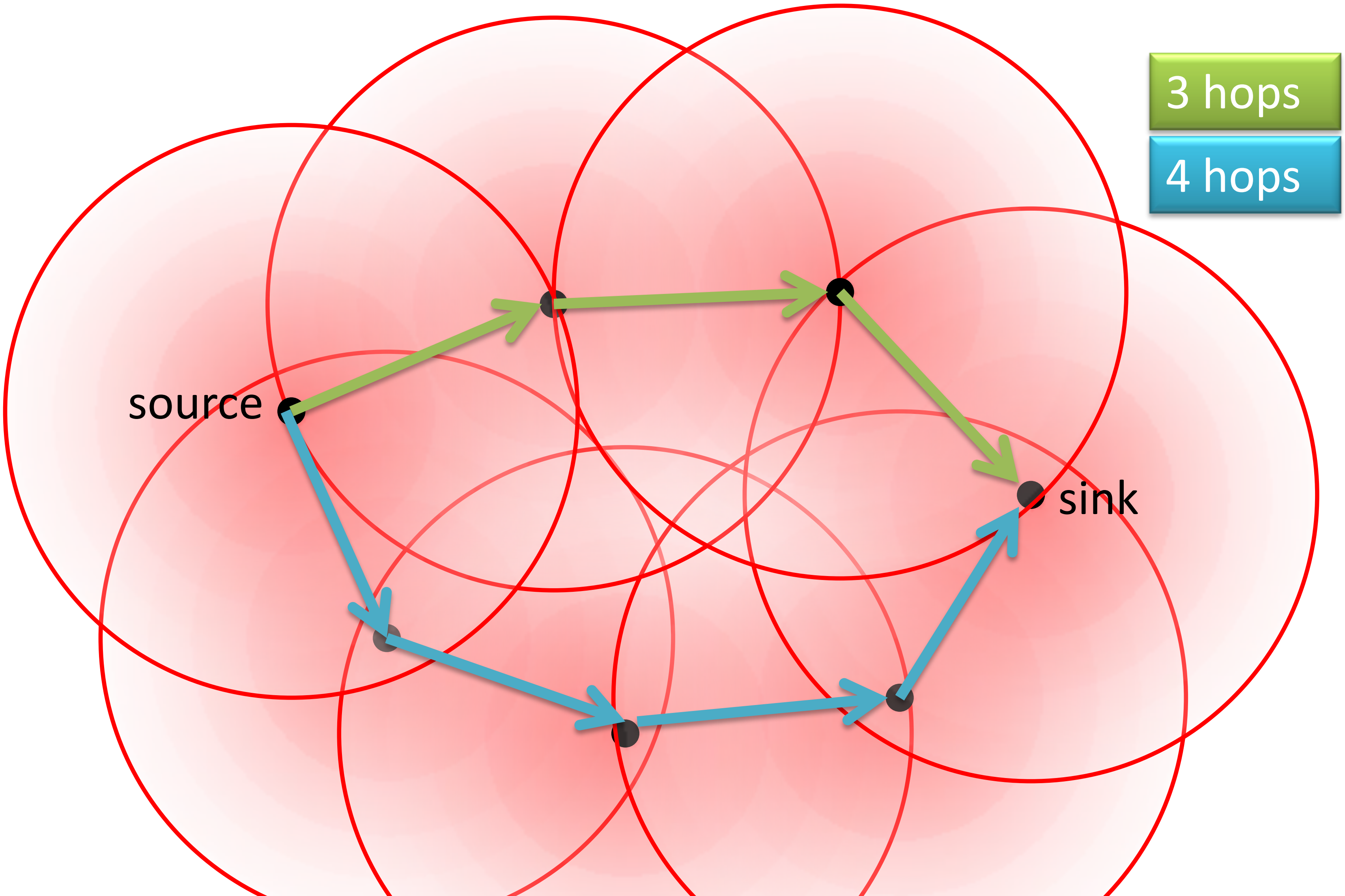
http://tinyurl.com/6x9dh4r

How to find a route

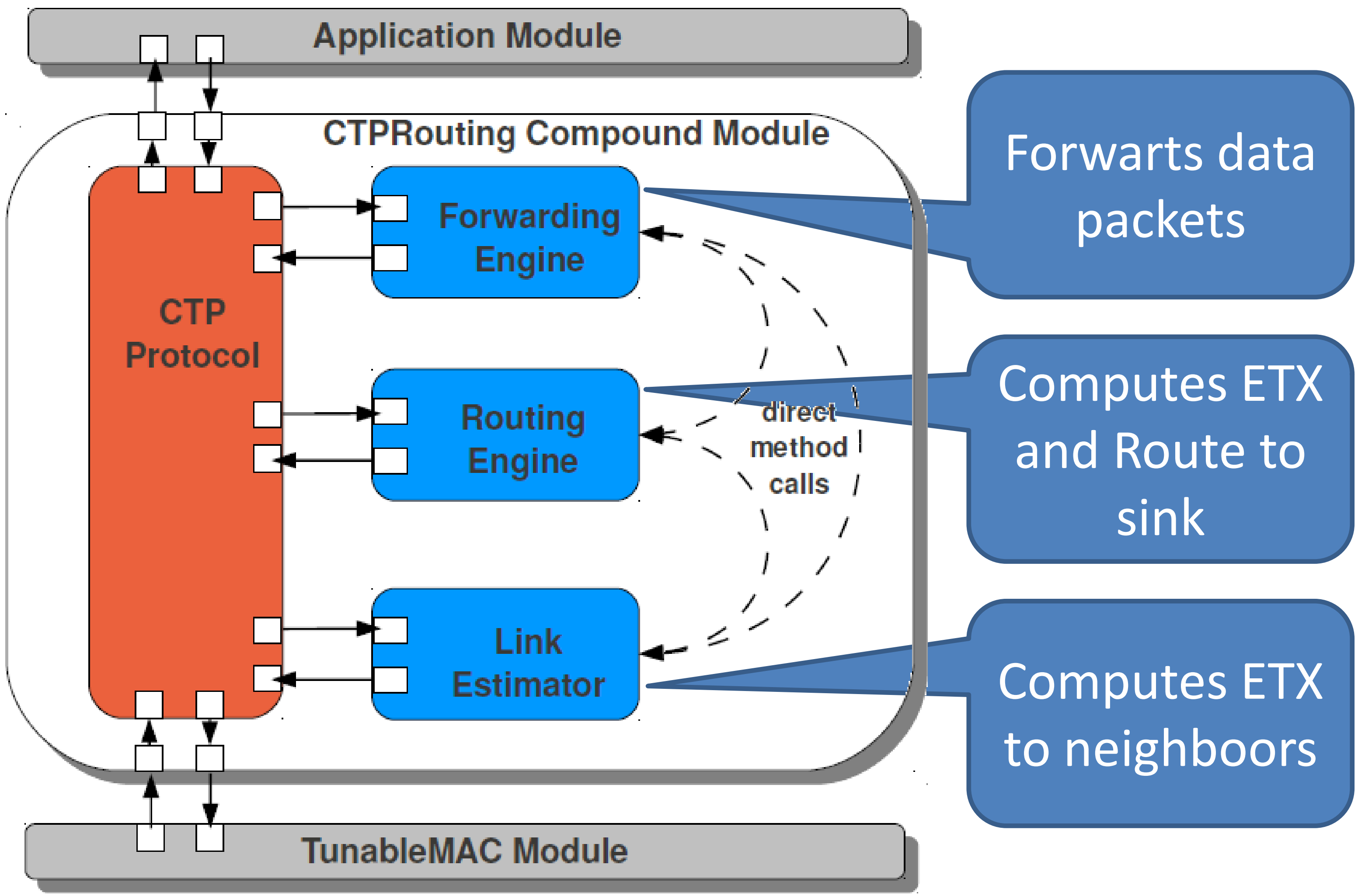
- Minimize transmission costs
  - ETX = Expected number of transmissions

- Every node maintains an estimate of the cost of a route to a collection point

Routing in CTP

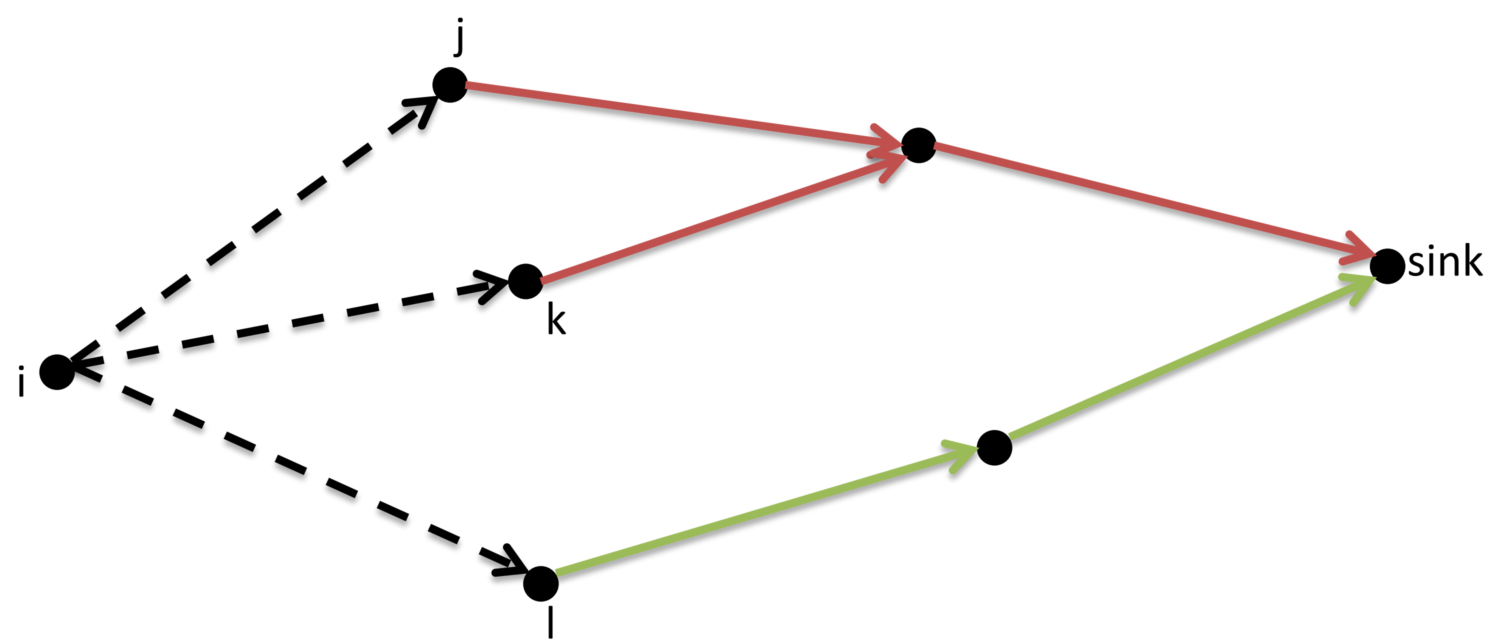


CTP Architecture



[Colesanti and Santini, 2010]

Parent Selection



Link estimator:

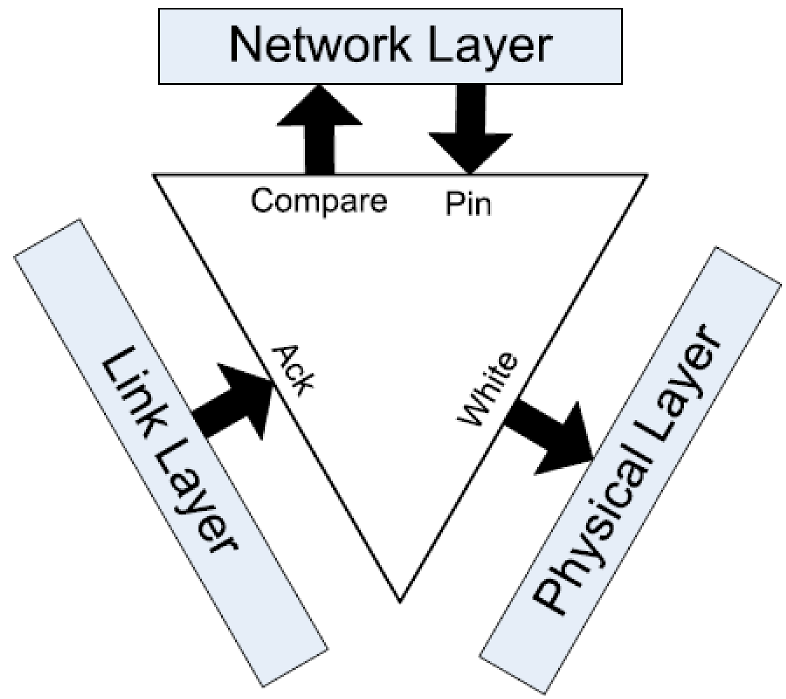
- $ETX_{1hop}(i,j)$
- $ETX_{1hop}(i,k)$
- $ETX_{1hop}(i,l)$

Routing Engine:

- $ETX_{multihop}(i,j) = ETX_{1hop}(i,j) + ETX_{multihop}(j)$
- $ETX_{multihop}(i,k) = ETX_{1hop}(i,k) + ETX_{multihop}(k)$
- $ETX_{multihop}(i,l) = ETX_{1hop}(i,l) + ETX_{multihop}(l)$

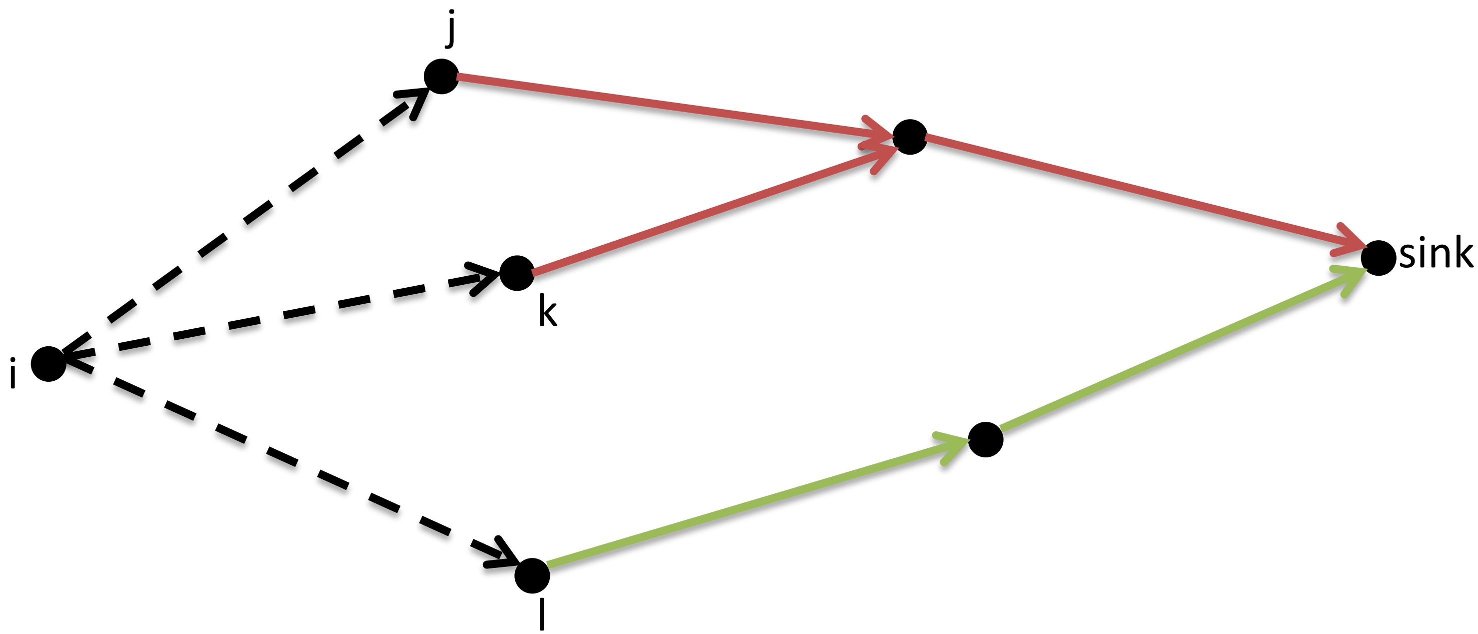
Link estimator

- Link estimator:
- - $ETX_{1hop}(i,j)$
- - $ETX_{1hop}(i,k)$
- - $ETX_{1hop}(i,l)$



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

Parent Selection



Link estimator:

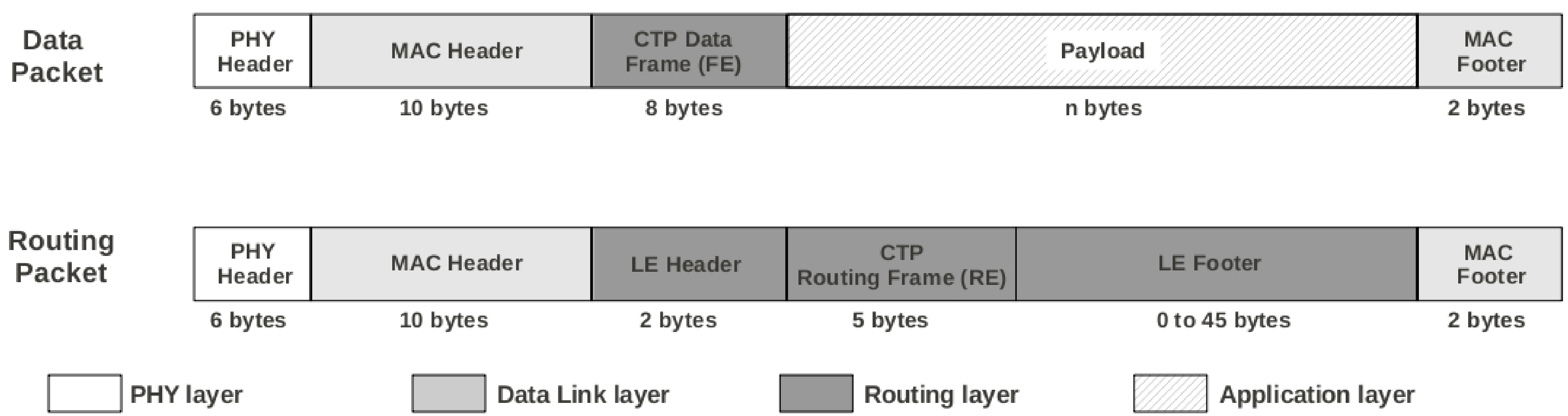
- $ETX_{1hop}(i,j)$
- $ETX_{1hop}(i,k)$
- $ETX_{1hop}(i,l)$

Routing Engine:

- $ETX_{multihop}(i,j) = ETX_{1hop}(i,j) + ETX_{multihop}(j)$
- $ETX_{multihop}(i,k) = ETX_{1hop}(i,k) + ETX_{multihop}(k)$
- $ETX_{multihop}(i,l) = ETX_{1hop}(i,l) + ETX_{multihop}(l)$

Data vs. Control Traffic

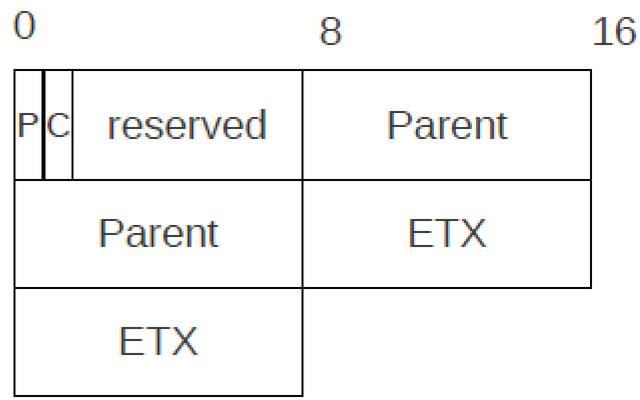
- Data packets
  - Unicast
- Control Beacons
  - Broadcast



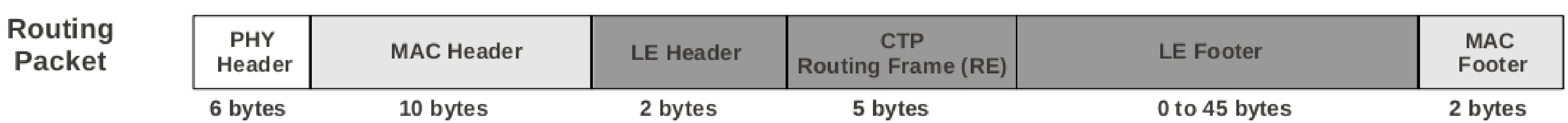
[Colesanti and Santini, 2010]

Control Beacon

- Control beacon has
  - Two fields: Parent and cost
  - Two control bits:
    - Pull bit (P)
    - Congestion bit (C)



[Colesanti and Santini, 2010]



[Colesanti and Santini, 2010]

Link Dynamics

- Other protocols typically use periodic beacons to update network topology and link estimates
  - Faster rates lead to higher cost
  - Slower rates lead to misinterpretations
- But CTP uses adaptive beaconing!

Adaptive Beaconing

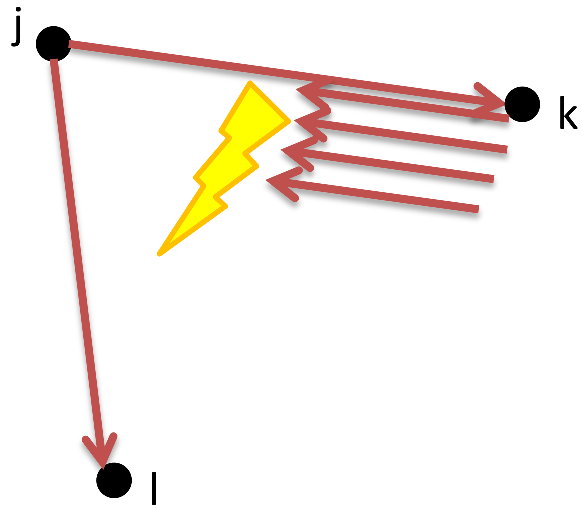
- CTP uses the Trickle Algorithm [Levis, 2004]
- In CTP:
  - Start with lowest interval of 64 ms
  - When interval expires double it up to 1 hour

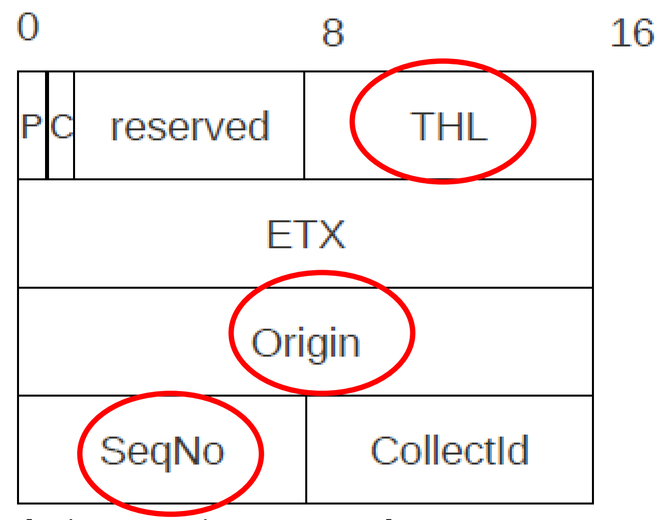
- Node resets the interval if
  - It is asked to forward a packet from a node whose ETX is lower or equal to its own
  - Is routing cost degrees significantly
  - It receives a packet with the P bit set

Data Plane Design

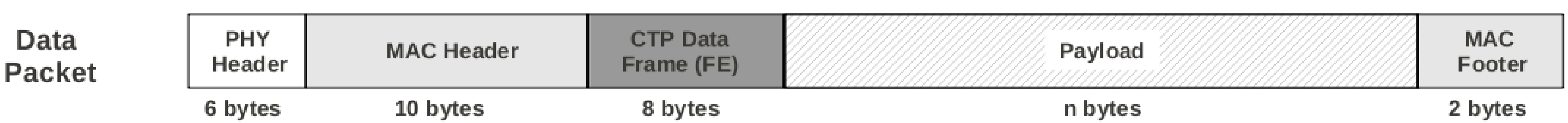
- Per-client Queuing
  - One single outstanding packet per client (process)
- Hybrid Send Queue
  - Route through- and locally-generated traffic buffer
- Transmit Timer
  - Wait two packet times between transmissions
- Transmit Cache
  - Avoid duplicates

Retransmission and Duplicates



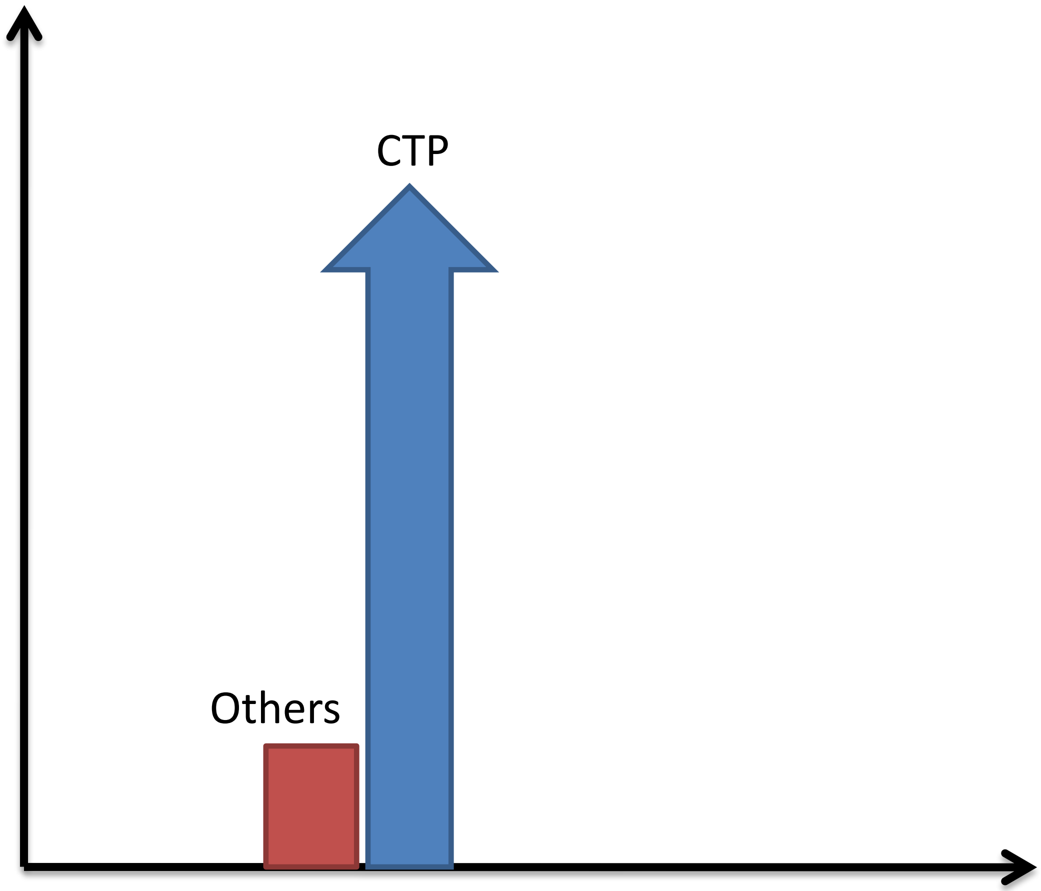


[Colesanti and Santini, 2010]



[Colesanti and Santini, 2010]

Evaluation



Others:
- MultiHopLQI
- Hyper
- RBC
- Dozer

Collection Protocols

- Why do we need collection protocols?
  - *"Collecting data at a base station is a common requirement of sensor network applications. The general approach used is to build one or more collection trees, each of which is rooted at a base station. When a node has data which needs to be collected, it sends the data up the tree, and it forwards collection data that other nodes send to it."* [TinyOS TEP 119]
- Requirements
  1. Reliability: > 90% of packets
  2. Robustness
  3. Efficiency: Use a minimum of transmissions
  4. Hardware Independence

Testbeds

Testbed	Location	Platform	Nodes	Physical size m^2 or m^3	Degree Min	Degree Max	PL	Cost	$\frac{\text{Cost}}{\text{PL}}$	$\frac{\text{Churn}}{\text{node·hr}}$
Tutornet (16)	USC	Tmote	91	50×25×10	10	60	3.12	5.91	1.90	31.37
Wymanpark	Johns Hopkins	Tmote	47	80×10	4	30	3.23	4.62	1.43	8.47
Motelab	Harvard	Tmote	131	40×20×15	9	63	3.05	5.53	1.81	4.24
Kansei[a]	Ohio State	TelosB	310	40×20	214	305	1.45	-	-	4.34
Mirage	Intel Research	Mica2dot	35	50×20	9	32	2.92	3.83	1.31	2.05
NetEye	Wayne State	Tmote	125	6×4	114	120	1.34	1.40	1.04	1.94
Mirage	Intel Research	MicaZ	86	50×20	20	65	1.70	1.85	1.09	1.92
Quanto	UC Berkeley	Epic-Quanto	49	35×30	8	47	2.93	3.35	1.14	1.11
Twist	TU Berlin	Tmote	100	30×13×17	38	81	1.69	2.01	1.19	1.01
Twist	TU Berlin	eyesIFXv2	102	30×13×17	22	100	2.58	2.64	1.02	0.69
Vinelab	UVA	Tmote	48	60×30	6	23	2.79	3.49	1.25	0.63
Tutornet (26)	USC	Tmote	91	50×25×10	14	72	2.02	2.07	1.02	0.04
Blaze[b]	Rincon Research	Blaze	20	30×30	9	19	1.30	-	-	-

[Gnawali, 2009]

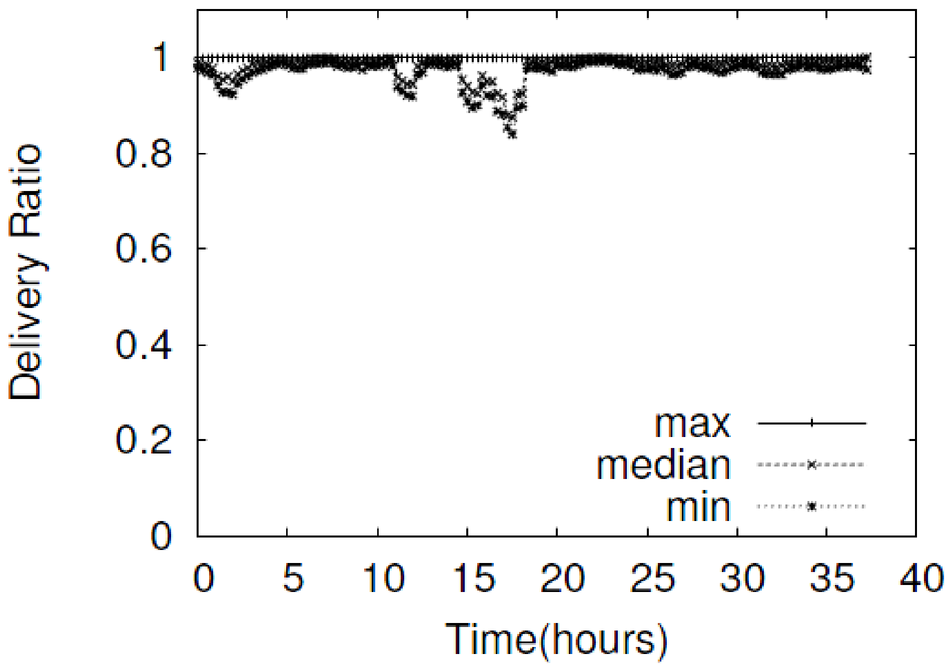
Reliability

Testbed	Frequency	MAC	IPI	Avg Delivery	5th% Delivery	Loss
Motelab	2.48GHz	CSMA	16s	94.7%	44.7%	Retransmit
Motelab	2.48GHz	BoX-50ms	5m	94.4%	26.9%	Retransmit
Motelab	2.48GHz	BoX-500ms	5m	96.6%	82.6%	Retransmit
Motelab	2.48GHz	BoX-1000ms	5m	95.1%	88.5%	Retransmit
Motelab	2.48GHz	LPP-500ms	5m	90.5%	47.8%	Retransmit
Tutornet (26)	2.48GHz	CSMA	16s	99.9%	100.0%	Queue
Tutornet (16)	2.43GHz	CSMA	16s	95.2%	92.9%	Queue
Tutornet (16)	2.43GHz	CSMA	22s	97.9%	95.4%	Queue
Tutornet (16)	2.43GHz	CSMA	30s	99.4%	98.1%	Queue
Wymanpark	2.48GHz	CSMA	16s	99.9%	100.0%	Retransmit
NetEye	2.48GHz	CSMA	16s	99.9%	96.4%	Retransmit
Kansei	2.48GHz	CSMA	16s	99.9%	100.0%	Retransmit
Vinelab	2.48GHz	CSMA	16s	99.9%	99.9%	Retransmit
Quanto	2.425GHz	CSMA	16s	99.9%	100.0%	Retransmit
Twist (Tmote)	2.48GHz	CSMA	16s	99.3%	100.0%	Retransmit
Twist (Tmote)	2.48GHz	BoX-2s	5m	98.3%	92.9%	Retransmit
Mirage (MicaZ)	2.48GHz	CSMA	16s	99.9%	99.8%	Queue
Mirage (Mica2dot)	916.4MHz	B-MAC	16s	98.9%	97.5%	Ack
Twist (eyesIFXv2)	868.3MHz	CSMA	16s	99.9%	99.9%	Retransmit
Twist (eyesIFXv2)	868.3MHz	SpeckMAC-183ms	30s	94.8%	44.7%	Queue
Blaze	315MHz	B-MAC-300ms	4m	99.9%	-	Queue

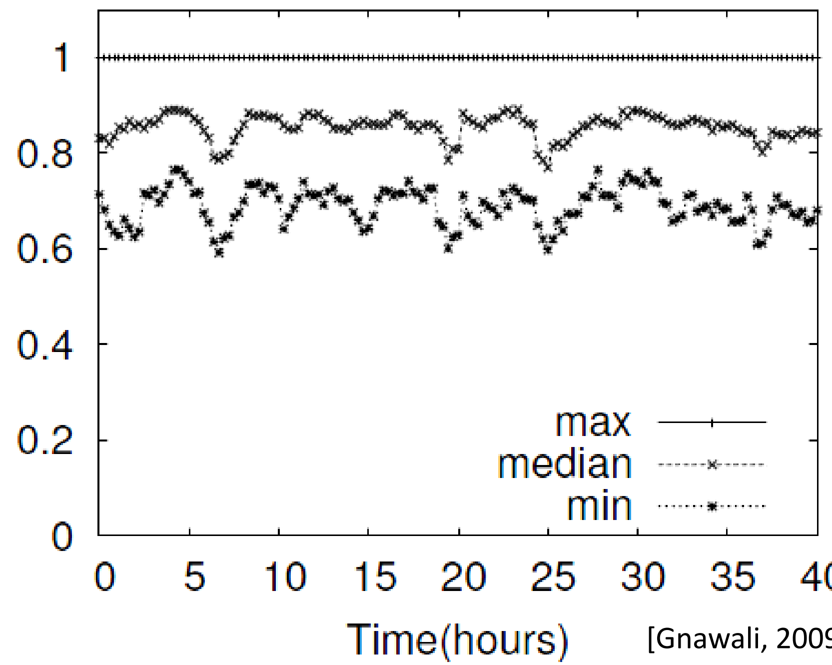
[Gnawali, 2009]

Reliability

CTP



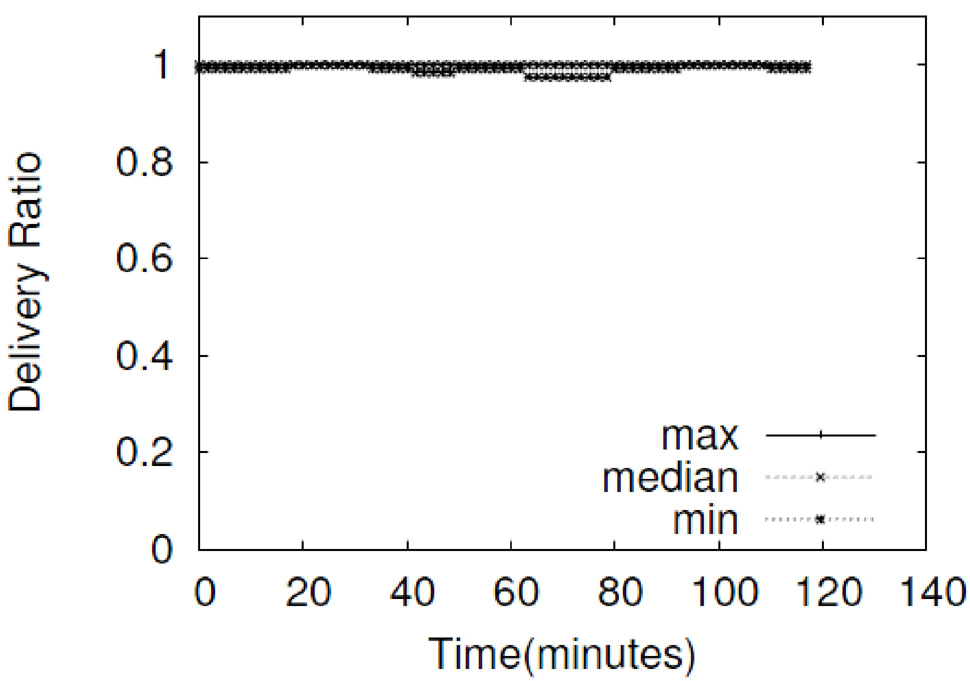
MultiHopLQI



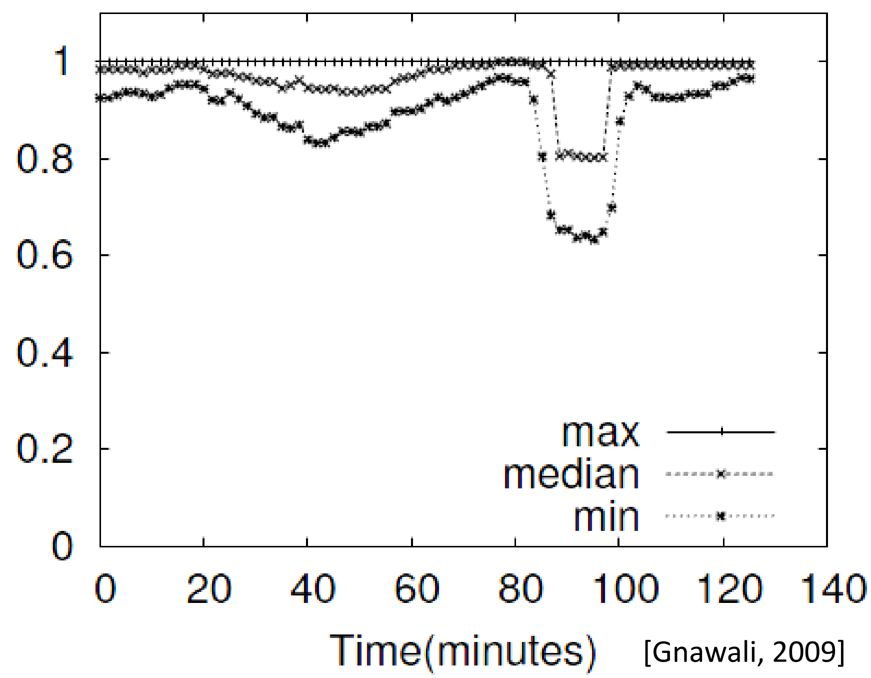
[Gnawali, 2009]

Robustness

CTP

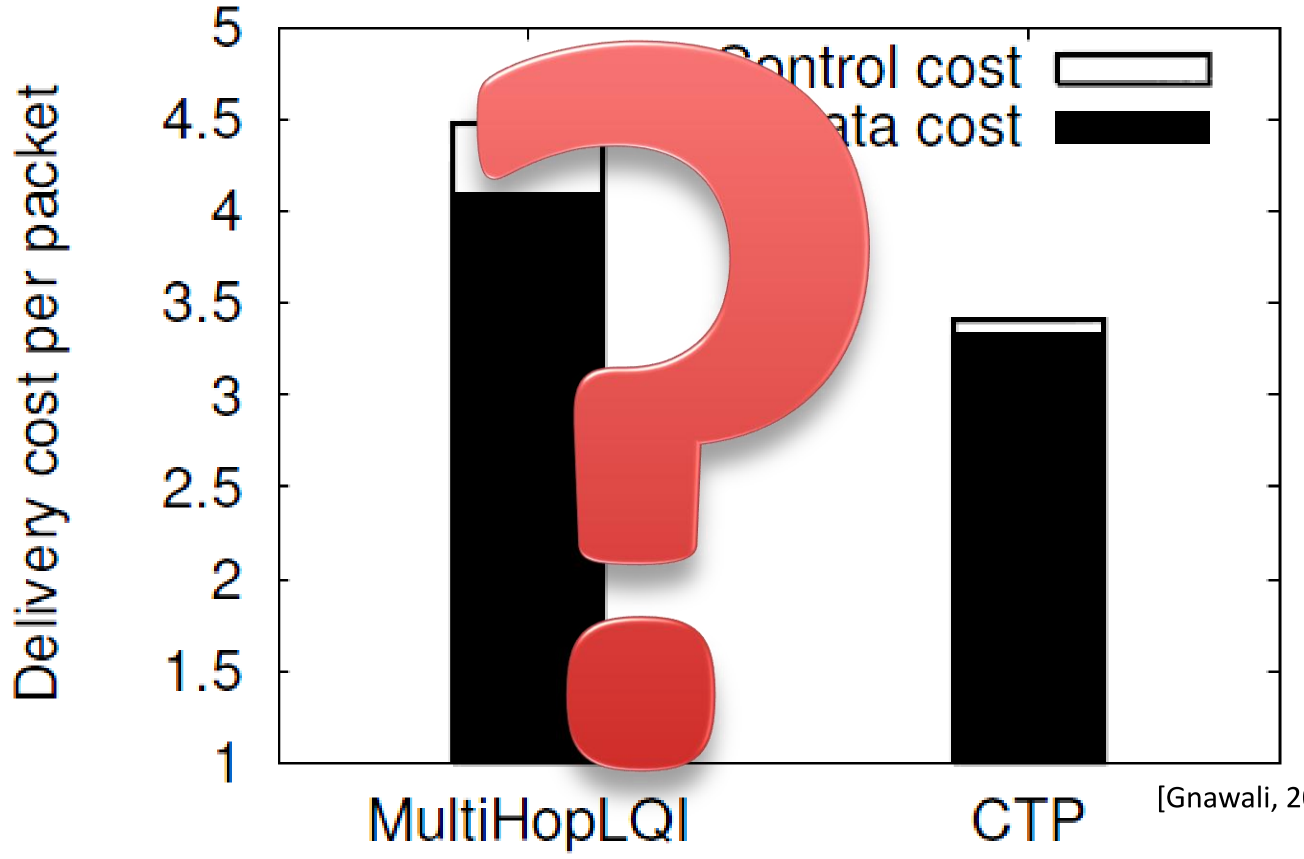


MultiHopLQI



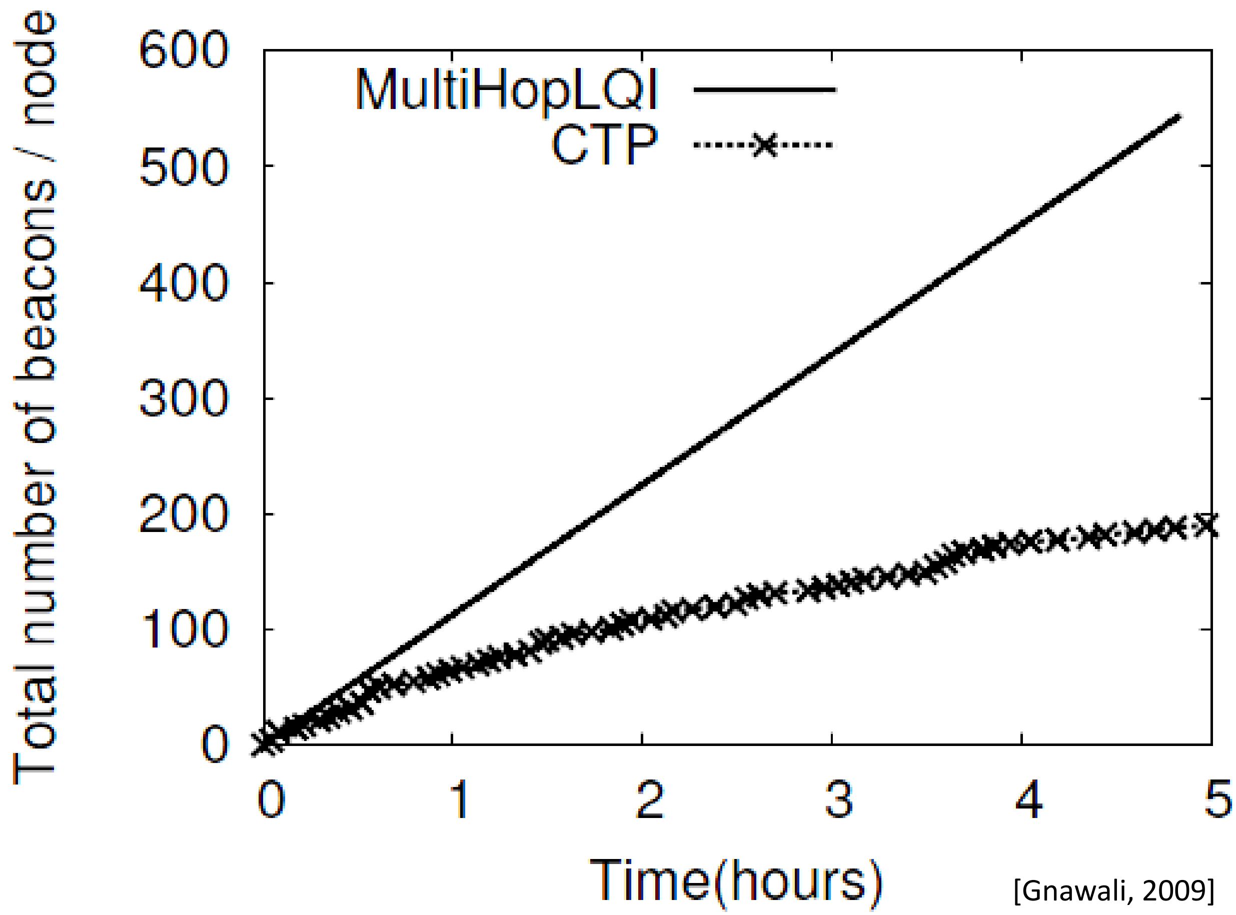
[Gnawali, 2009]

Efficiency



[Gnawali, 2009]

Efficiency



[Gnawali, 2009]

Furthermore

- Agility
  - After a pause of 20 mins and removal of a node the beacon rate decreased to 1 beacon per 8 min
  - Establish a new route withing 325 ms
- Transmit timer
- Transmit cache
- External interference
- Link layers
- Energy profile

Why to use CTP

- CTP delivers >90% of packets (usually 99.9%)
- CTP sends 73% fewer beacons than others
- CTP reduces topology repair latency by 99.8%

In short: CTP is great

Questions?

http://tinyurl.com/67luhsv

References

- [TEP 119] R. Fonseca, O. Gnawali, K. Jamieson, S. Kim, P. Levis, and A. Woo. TEP 119: Collection Protocol, Feb. 2006.
- [TEP 123] R. Fonseca, O. Gnawali, K. Jamieson, S. Kim, P. Levis, and A. Woo. TEP 123: The Collection Tree Protocol, Aug. 2006.
- [Gnawali, 2009] Gnawali et al.: Collection Tree Protocol, 2009.
- [Colesanti and Santini, 2010] U. Colesanti, S. Santini. Tech report: A Performance Evaluation Of The Collection Tree Protocol Based On Its Implementation For The Castalia Wireless Sensor Networks Simulator.
- [Levis, 2004] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A selfregulating algorithm for code maintenance and propagation in wireless sensor networks. In Proc. of the USENIX NSDI Conf., San Francisco, CA, Mar. 2004.
- [Fonseca, 2007] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis. Four Bit Wireless Link Estimation. In Hotnets-VI, Atlanta, GA, Nov. 2007.