Design and Development of IoT Applications

Dr. –Ing. Vo Que Son

Email: sonvq@hcmut.edu.vn

Content

- Chapter 5: Routing in WSNs
  - Multi-hop communication
  - Link characteristics
  - Collection Tree Protocol/DCP
  - Trickle algorithm
- Chapter 6: 6LoWPAN and IPv6
  - Challenges in WSNs and IP
  - IPv6 addressing
  - Fragmentation
  - 6LoWPAN Header compression
  - Bootstrapping
  - Border Router

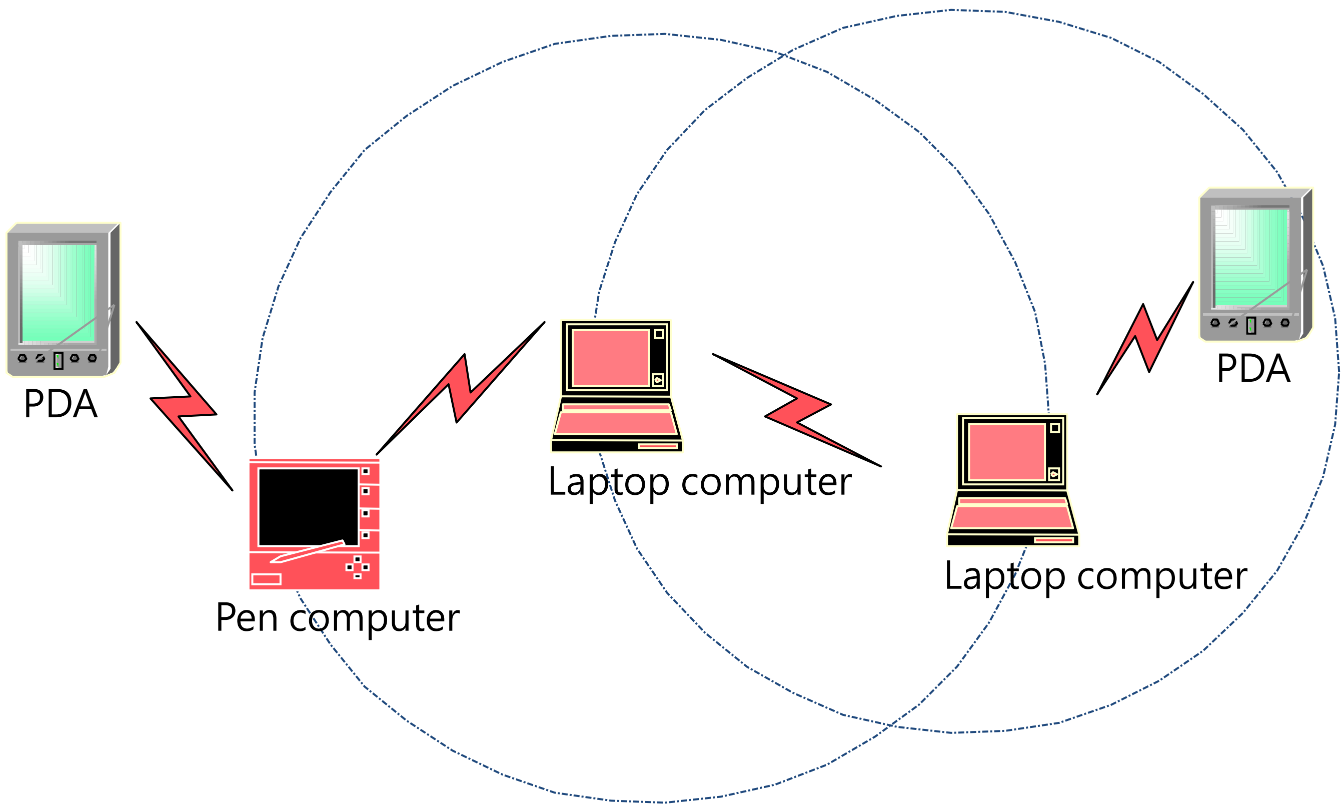
Wireless Ad-hoc Networks

Two types of wireless network:

- Infrastructured (WLAN)
    - the mobile node can move while communicating
    - the base stations are fixed
    - as the node goes out of the range of a base station, it gets into the range of another base station
- Infrastructureless or ad-hoc
    - the mobile node can move while communicating
    - there are no fixed base stations
    - all the nodes in the network need to act as routers
- In Latin “ad-hoc” literally means “for this purpose only”. Then an ad-hoc network can be regarded as “spontaneous network”

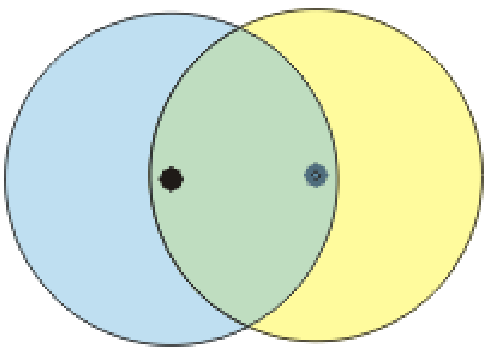
MANET

- Infrastructurless (ad-hoc) network or MANET (Mobile Ad-hoc NETwork)

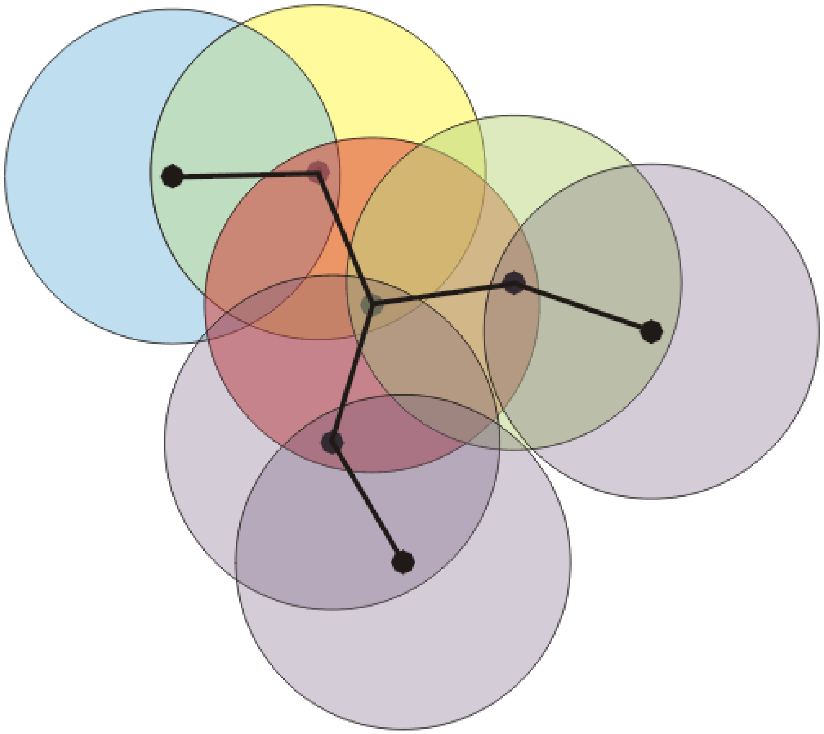


Classification of ad-hoc networks

- **Single hop** – nodes are in their reach area and can communicate directly

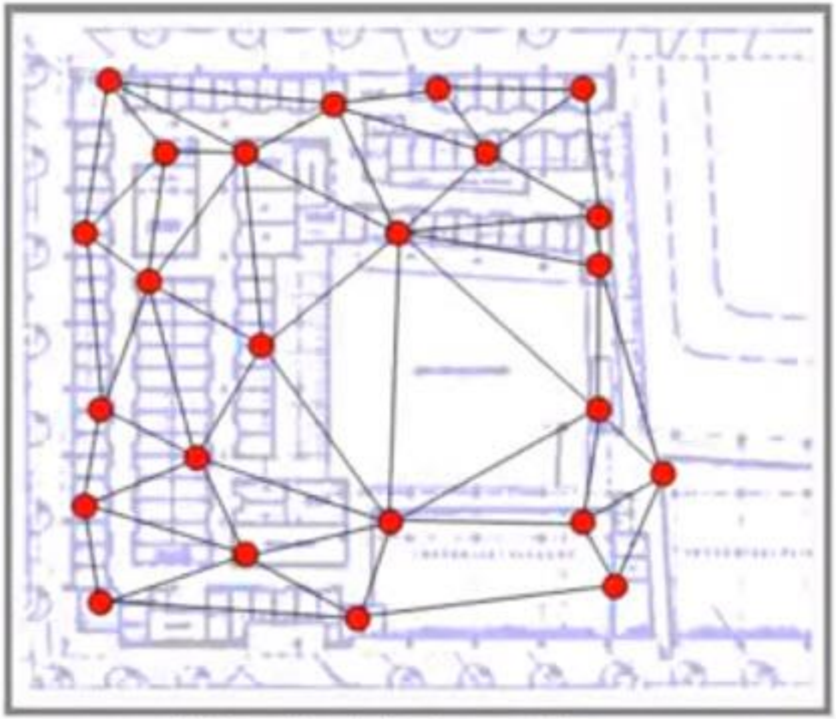


- **Multi hop** – some nodes are far and cannot communicate directly. The traffic has to be forwarded by other intermediate nodes.



Multi-hop network

Star Network
(e.g. 802.11)



Mesh Network
(e.g. ZigBee)

Characteristics of an ad-hoc network

- Collection of mobile nodes forming a temporary network
- Network topology changes frequently and unpredictably
- No centralized administration or standard support services
- Each host is an independent router
- Hosts use wireless RF transceivers as network interface
- Number of nodes 10 to 100 or at most 1000
- Nodes/host are powerful and focus on how to keep the mobile connection; high computation and high power consumption may be acceptable

Classical View of Routing

- Connectivity between nodes defines the network graph.
  - Topology formation
- A Routing algorithm determines the sub-graph that is used for communication between nodes.
  - Route formation, path selection
- Packets are forwarded from source to destination over the routing sub-graph
  - At each node in the path, determine the recipient of the next hop
- The selection at each hop is made based on the information at hand
  - Sender address, current address, destination address, information in the packet, information on the node.
  - Table-driven, source based, algorithmic, …
  - Who knows the route? Do you determine it as you go?

Classification of the Routing Protocols

- Link state
  - Nodes shout (send) and listen (receive) to determine neighbor connectivity.
  - Each floods this information throughout (Link State Advertisement) so every node has a map (Link state data base) of the network.
  - Any node can determine the path or the next hop.
  - management protocol deals with changes in connectivity
  - Classic Example: OSPF
- Distance vector
  - Nodes maintain routing information about “distance” and “direction” to destinations
  - Choose next hop by comparing the cost of routing through neighbors
  - Cost(dest D, neighbor b) = linkCost(b) + pathCost(b,D)
  - Management propagates routing information
  - Sequence numbers, etc.
  - Classic Example: RIP

Classification of the Routing Protocols

- Proactive (table driven)
  - Require each node to maintain one or more tables to store routing information
  - Each node responds to changes in network topology by propagating updates throughout the network in order to maintain a consistent network view
  - DSDV, OLSR (Optimized Link State Protocol)
- Reactive protocols (source initiated)
  - Creates routes only when desired by the source node
  - Once a route has been established, it is maintained by a route maintenance procedure until either the destination becomes inaccessible along every path from the source or until the route is no longer desired
  - DSR, AODV (Ad-hoc On-demand Distance Vector)

Classification of the Routing Protocols

- Various simulation studies have shown that reactive protocols perform better in mobile ad hoc networks than proactive ones.
  - However, no single protocol works well in all environments.
  - Which approach achieves a better trade-off depends on the traffic and mobility patterns.

	Proactive Approach	Reactive Approach
Route Latency	**Lower** ▪A route is kept at all times	**Higher** ▪A route is never kept when not used
Routing Overhead	**Higher** ▪A frequent dissemination of topology information is required	**Lower** ▪Fewer control packets in general

Classification of the Routing Protocols

- Other classification
  - Proactive protocols:
    - DSDV, STAR, WRP, ...
  - Reactive protocols:
    - AODV, DSR, TORA, ...
  - Hierarchical/Clustering protocols:
    - CGSR, ZRP, CBR, FSR, LANMAR, ...
  - Position aware protocols:
    - GPSR, LAR, GRA, ABR, ...

Problems with Routing

- Distance-vector protocols
  - Slow convergence due to “Count to Infinity” Problem
  - Creates loops during node failure, network partition or congestion
- Link state protocols
  - Use flooding technique and create excessive traffic and control overhead
  - Require a lot of processor power and therefore high power consumption
- Limitations of the Wireless Networks
  - packet loss due to transmission errors
  - variable capacity links
  - frequent disconnections/partitions
  - limited communication bandwidth
  - Broadcast nature of the communications
- Limitations Imposed by Mobility
  - dynamically changing topologies/routes
  - lack of mobility awareness by system/applications
- Limitations of the Mobile Computer
  - short battery lifetime
  - limited capacities

Leading Routing Protocols

- Leading protocols chosen by MANET
  - DSR: Dynamic Source Routing
  - AODV: Ad-hoc On-demand Distance Vector Routing

- Both are “on demand” protocols: route information discovered only as needed

MANET vs WSNs

- ❑ WSN nodes have less power, computation and communication compared to MANET nodes
  - ❖ MANET protocols require significant amount of routing data storage and computation
- ❑ MANETs have high degree of mobility, while sensor networks are mostly stationary mostly stationary
  - ❖ Topology changes in WSNs due to nodes dying in the network (due to energy dissipation or due to lossy links)
- ❑ MANET protocols are not being optimized to cater for duty cycles
- ❑ WSNs may be considered a subset of MANET
  - ❖ Routing in WSNs should not necessarily be complex as in MANET

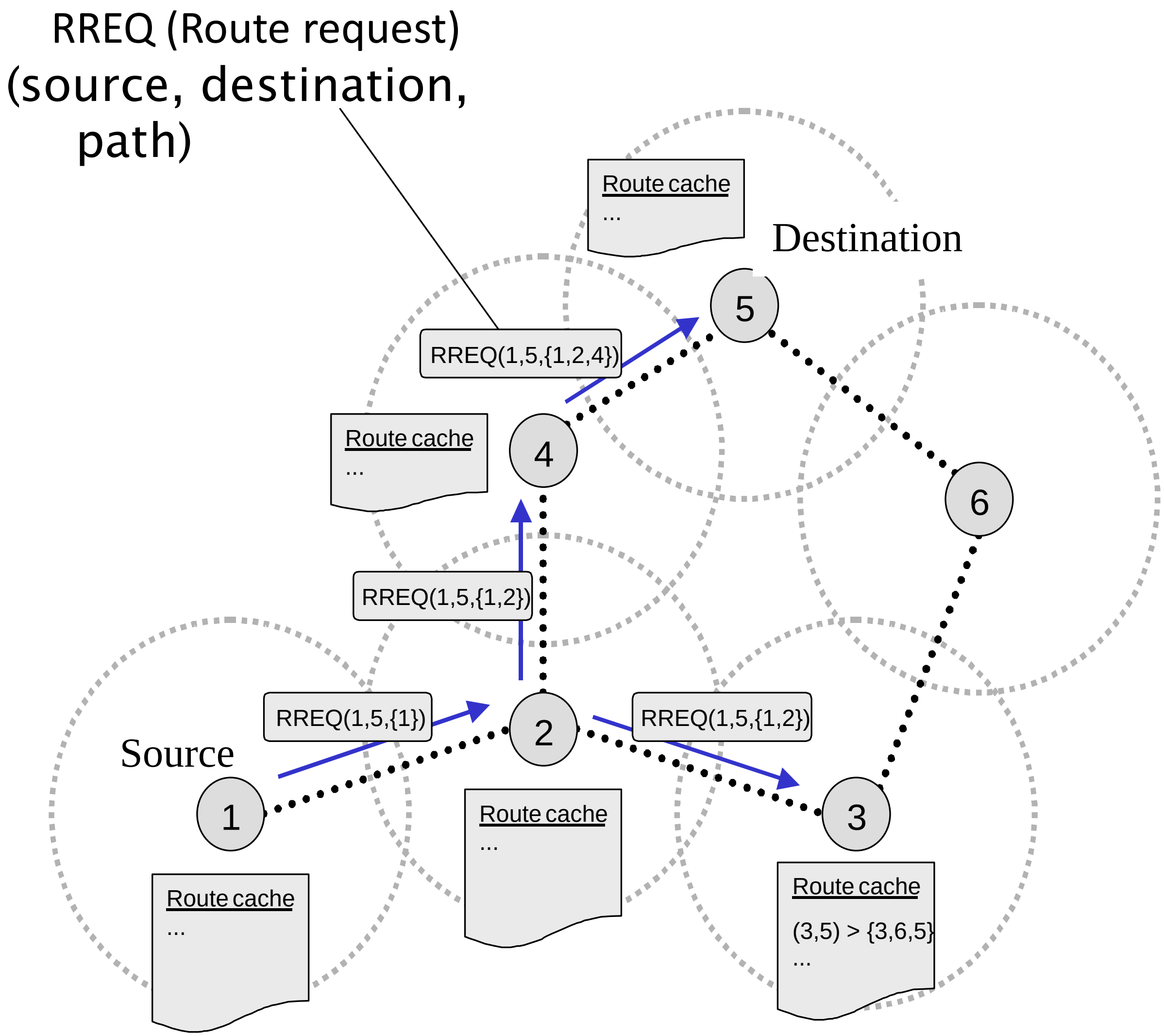
Dynamic Source Routing

- DSR (Dynamic Source Routing)
  - Similar to the source routing in traditional networks
  - A node maintains route cache containing the routes it knows
  - Includes route discovery on request and route maintenance when needed
  - Reactive routing

DSR

- Route discovery
  - The source sends a broadcast packet which contains source address, destination address, request id and path.
  - If the host receiving this packet, saw this packet before, discards it.
  - Otherwise, it looks up its route caches to look for a route to destination. If a route is not found, it appends its address into the packet and rebroadcasts it.
  - If the route is found, it sends a reply packet to the source node.
  - The route will be eventually found when the request packet reaches the destination

DSR


RREQ (Route request)
(source, destination,
path)
Route cache
...
Destination
5
RREQ(1,5,{1,2,4})
Route cache
...
4
6
RREQ(1,5,{1,2})
RREQ(1,5,{1})
2
RREQ(1,5,{1,2})
Source
1
3
Route cache
...
Route cache
...
Route cache
(3,5) > {3,6,5}
...

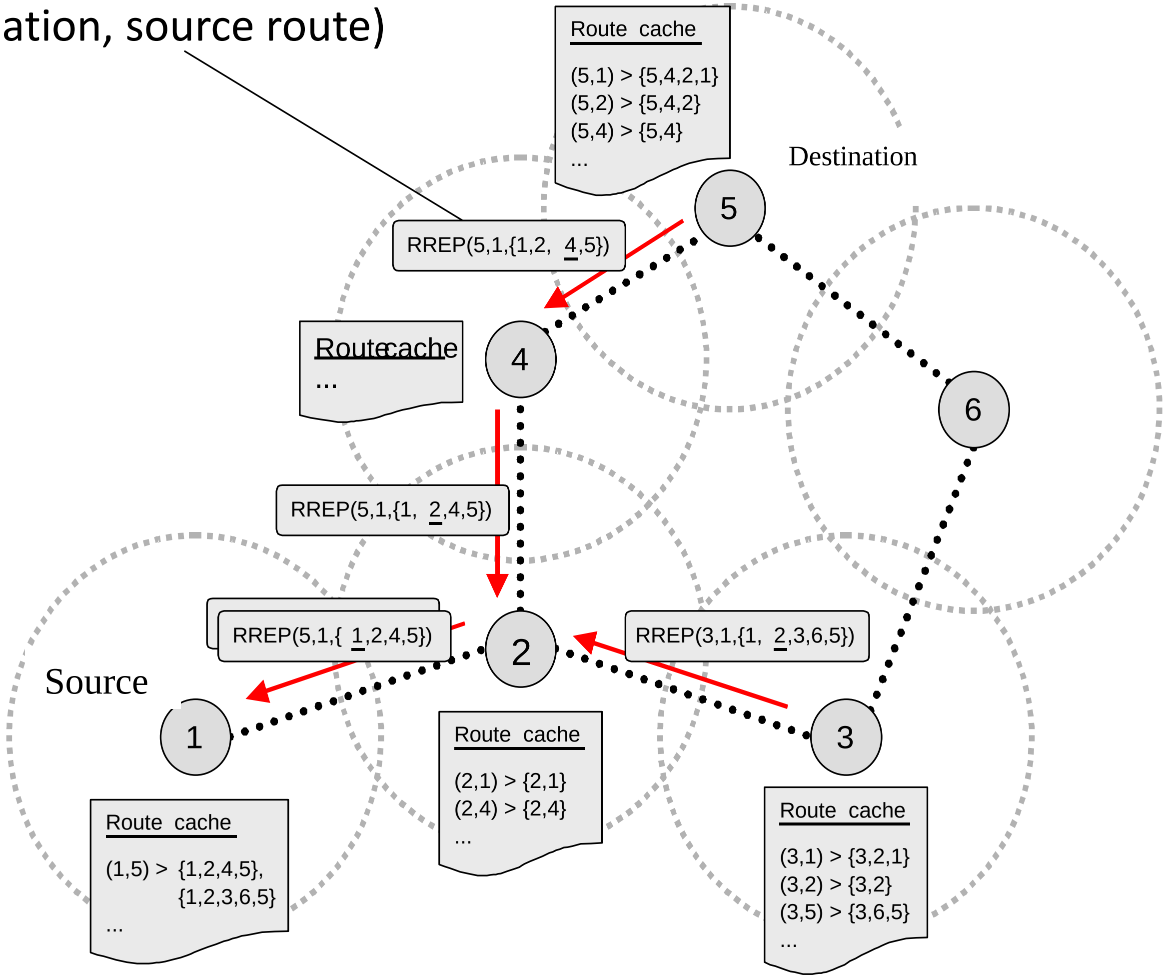
DSR

- How to send a reply packet?
  - If the destination has a route to the source in its cache, use it
  - Else if symmetric links are supported, use the reverse of the route record
  - Else, if symmetric links are not supported, the destination initiate route discovery to source

DSR

RRREP (Route reply)

Source, destination, source route)



DSR

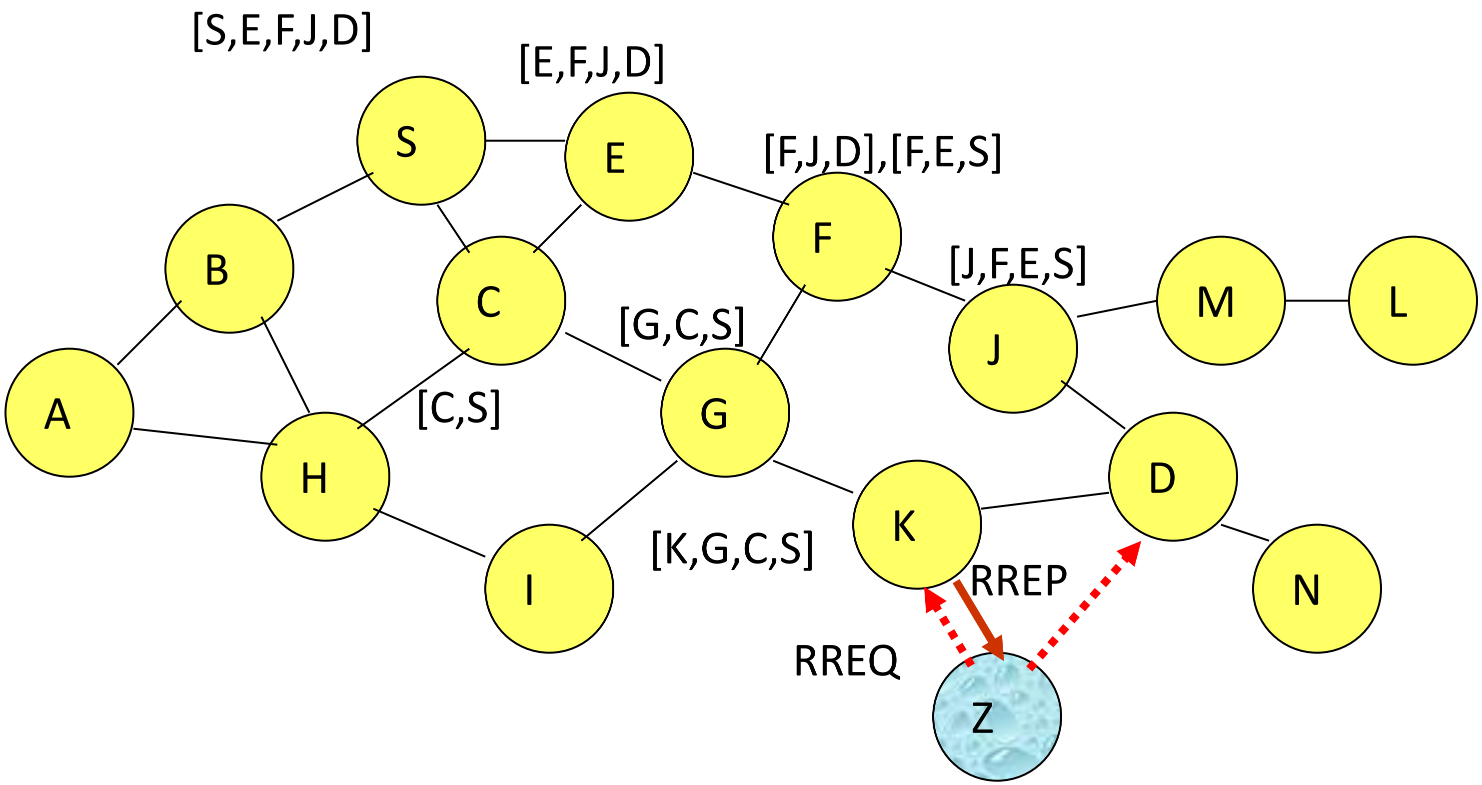
❑ Route maintenance

- ❖ Whenever a node transmits a data packet, a route reply or a route error, it must verify that the next hop correctly receives the packet.
- ❖ If not, the node must send a route error to the node responsible for generating this route header.
- ❖ The source restarts the route discovery

❑ Route caching

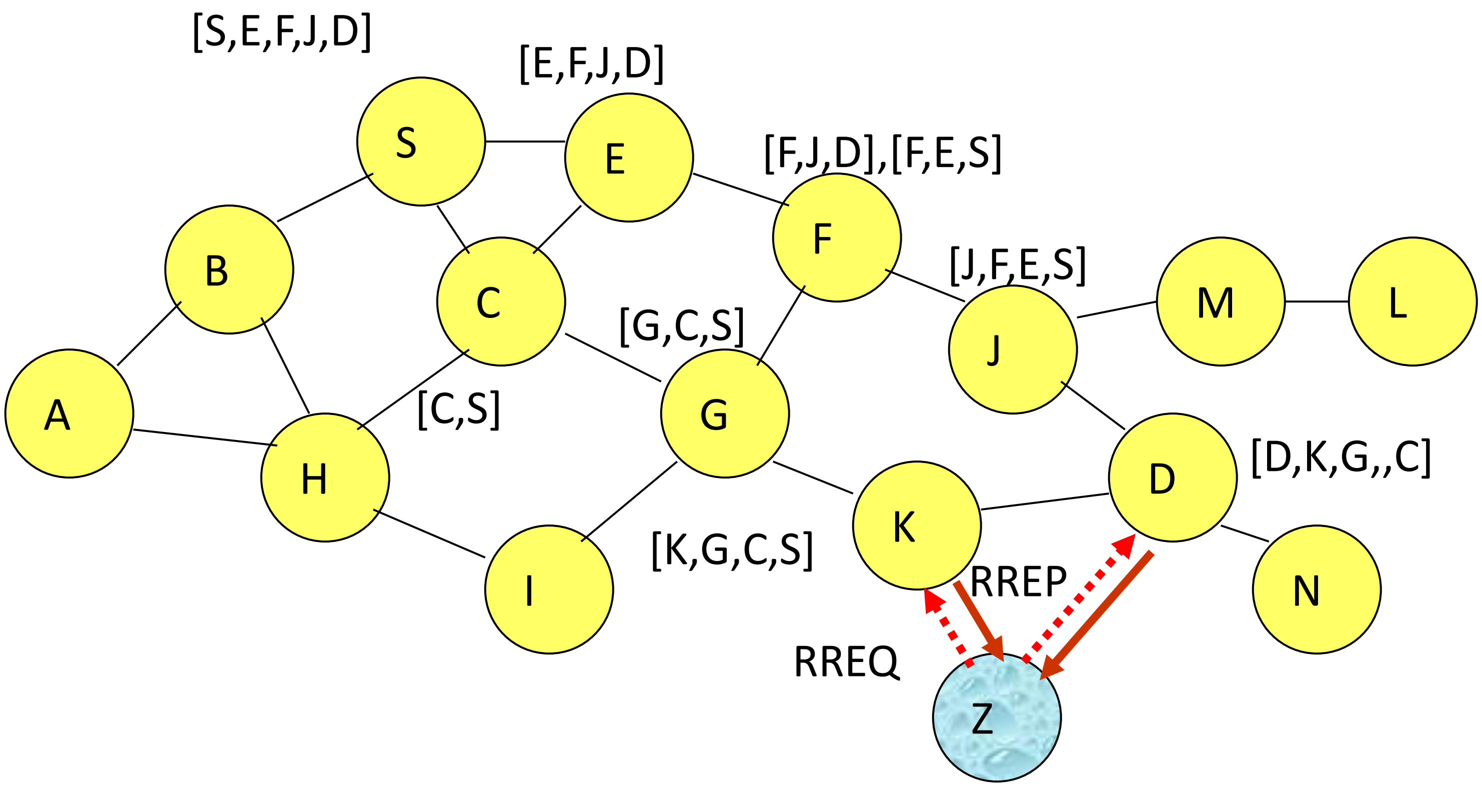
- ❖ When S finds route [S,E,F,J,D] to D, S also learns route [S,E,F] to F
- ❖ When K receives Route Request [S,C,G] destined for some node D, K learns route [K,G,C,S] to S if links are bi-directional
- ❖ F forwards Route Reply RREP [S,E,F,J,D], F learns route [F,J,D] to D
- ❖ When E forwards Data [S,E,F,J,D] it learns route [E,F,J,D] to D

Can Speed up Route Discovery



- When node Z sends a route request for node C, node K sends back a route reply [Z,K,G,C] to node Z using a locally cached route

Reduce Propagation of Route Requests



- Route Replies (RREP) from node K and D ***limit flooding*** of RREQ

DSR

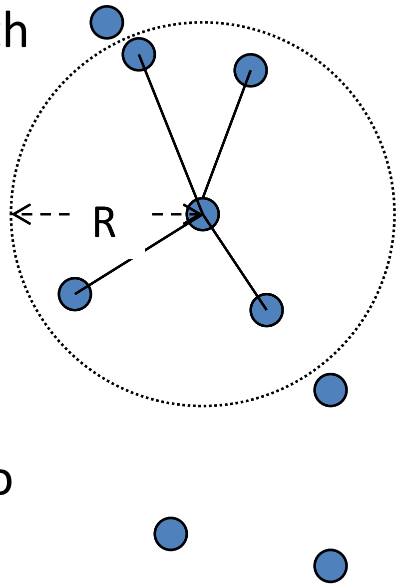
- Advantages
  - Routes are discovered only they are needed: Reduces overhead of route maintenance
  - Route caching reduce the cost of route discovery
  - A single route discovery may yield many routes to the destination, due to intermediate nodes may reply route request from local caches
  - Does not require symmetric links
- Disadvantages
  - Packet header size grows with route length due to source routing: Inefficiency
  - Route request packet may potentially reach all nodes in the network: RREQ flooding
  - Route requests may collide at the targeted node: Pay so much but get nothing
  - Every node needs to turn on its receiver all the time: No energy saving
  - Increased contention if too many route replies come back: Route Reply Storm
  - An intermediate node may send Route Reply using a stale cached route, thus polluting other nodes' caches: Mess up routing and forwarding

What's different in WSN?

- There is no a priori network graph
  - It is discovered by sending packets and seeing who receives them.
  - The link relationship is not binary.
  - pairs of nodes communicate with some probability that is determined by many of factors.
  - It is not static.
- The embedding of the "network" in space is important.
  - Need to get information to travel between particular physical places.
  - But the "communication range" is not a simple function of distance.
- addressing & naming
  - Flat EUID? Hierarchical IP? Topologically meaningful? Spatially meaningful?

Topology Formation

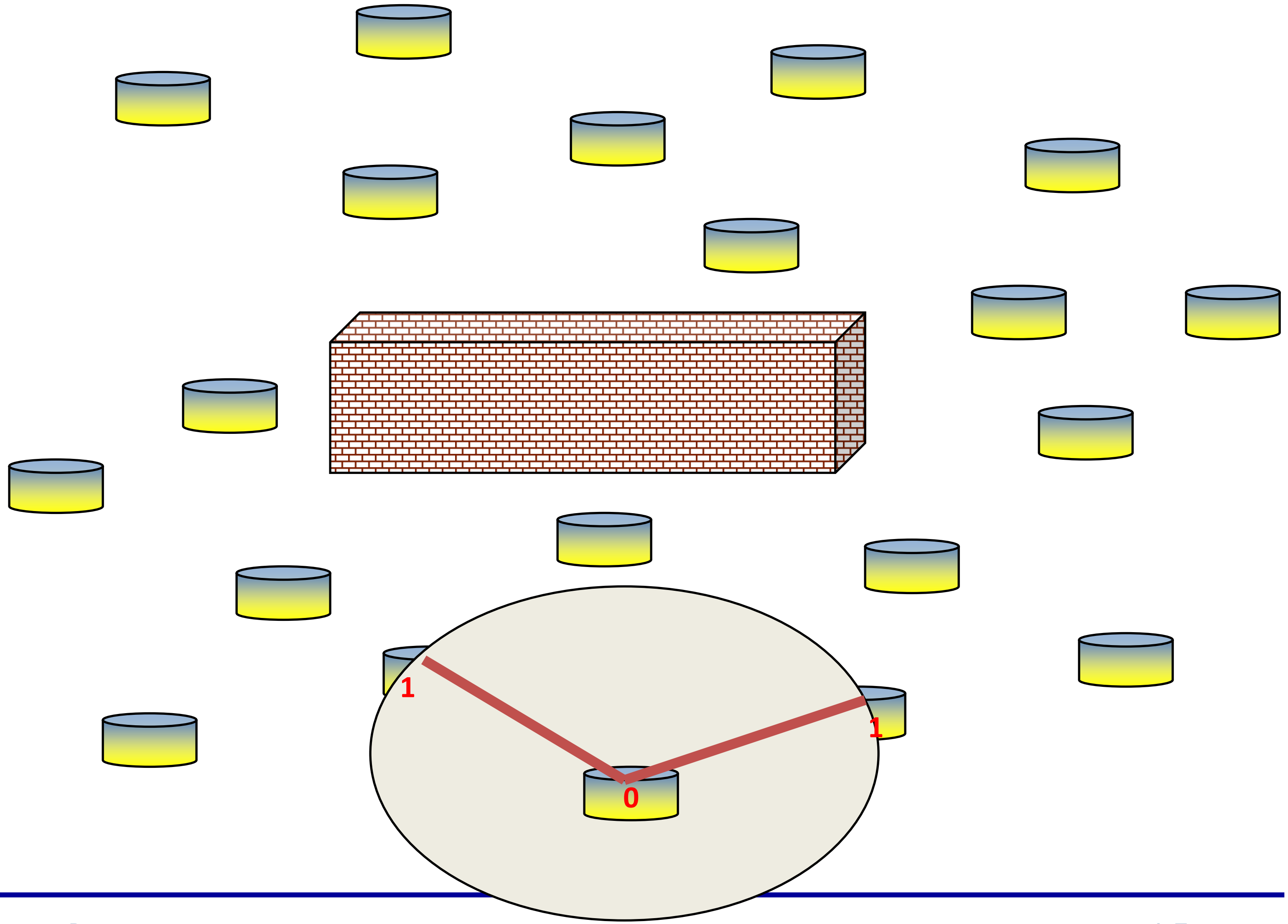
- Much of the "paper protocols" define connectivity graph with unit disk model
  - Link(A,B) if dist(A,B) ≤ R
- OK for rough calculations, but not for protocol design
  - Nearby nodes may not be able to communicate.
  - Far away nodes may be able to communicate.
  - Nodes that communicated in the past may not be able to communicate in the future.
  - Nodes may have intermittent communication depending on external factors.
- Connectivity is determined by communication
  - If B receives packet reasonably reliably from A, then A → B
  - If A receives packet reasonably reliably from B, then A ← B
  - And if both are true, A ←→ B



Wireless Routing Protocols

- Many wireless protocols in the IP context have been development in the IETF MANET (Mobile Ad Hoc Networking) working group in the context of 802.11 links carrying traditional TCP/IP point-to-point traffic.
  - AODV – ad hoc on-demand distance vector
  - OLSR – Optimized link state Routing
  - DSDV - Destination Sequenced Distance Vector
  - DSR – Dynamic Source Routing
  - TDRPF - Topology Dissemination Based on Reverse-Path Forwarding
- Assume a fairly “classic” view of connectivity
  - Naïve radio
- Routing protocols for MANET require high computation, powerful MCU, which are not satisfied in sensor nodes

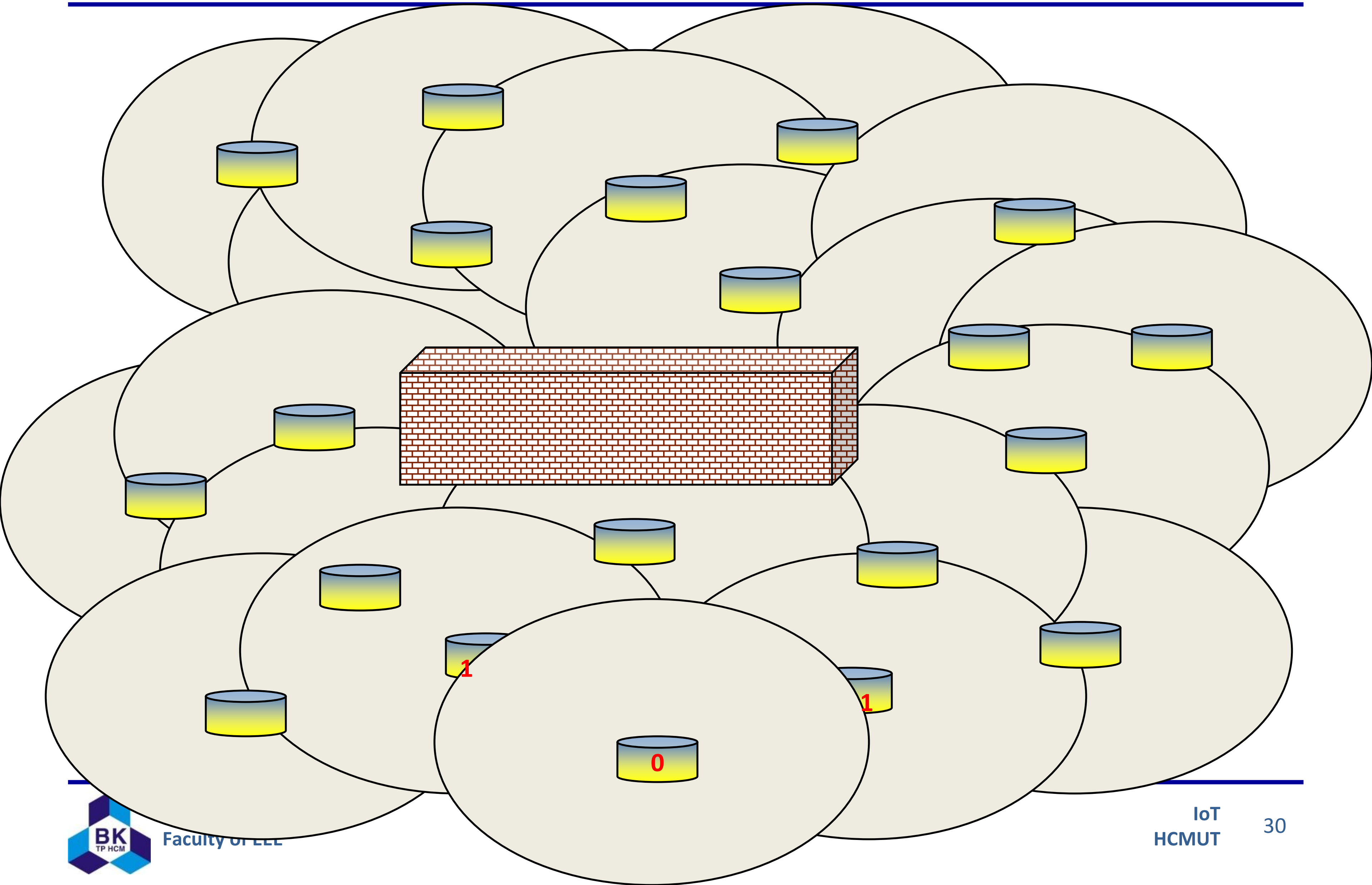
Neighbor Communication



Fundamental Primitive

- Transmit to whatever receivers happen to hear it
- This is the fundamental primitive that is buried underneath complex protocols like Bluetooth, but not made available.
- It is what make it possible to build higher level protocols on the link, especially IP.
- To determine connectivity,
  - Local broadcast
  - Respond
  - on-going protocol to estimate quality of the link
    - Packet reliability (sequence numbers, acks)
    - Note 802.15.4 acks only from a specific destination
    - RSSI, LQI, …

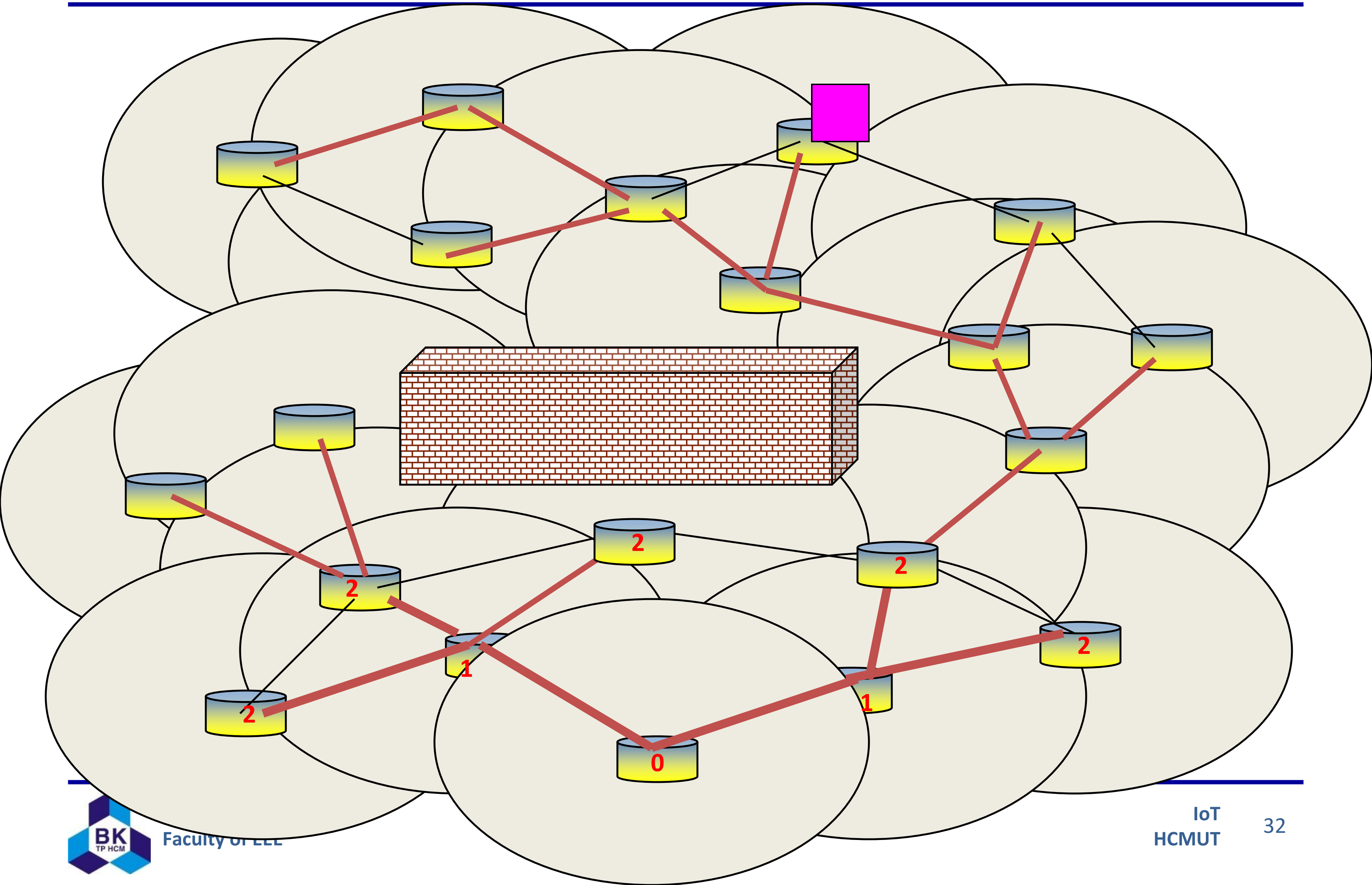
Route-Free “Flood”



"Flooding"

- Route free dissemination is extremely useful in its own right
  - Disseminate information
    - Router advertisements, solicitations, …
  - Network-wide discovery
  - Join, …
- It is also the network primitive that most "ad hoc" protocols used to determine a route
  - Flood from source till destination is reached.
  - Each node records the source of the flood packet
    - This is the parent in the "routing tree"
  - Reverse the links to form the path back

Data Collection in concept



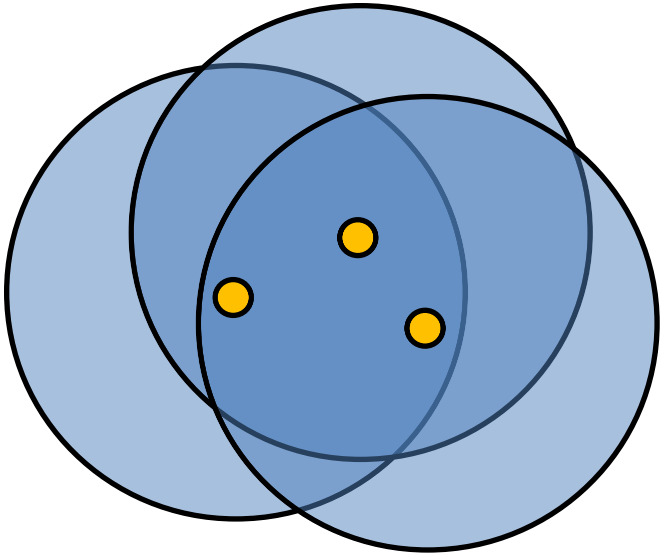
The Problems

- Flood causes tremendous contention
  - Many good links missed because of collisions
  - Huge amount of noise
- Many links are not symmetric

Trickle – better than flood

- Want the communication rate per unit area to be constant, regardless of the density of nodes
  - Lots of nodes, transmit infrequently
  - Few node, transmit more frequently
- Nodes listen before transmitting
- Estimate density based on how many nodes you hear from
  - Arrival during timer wait extends timer
- If new value is disseminated by others, no need for you to transmit it.
- Increase delay over time so ambient rate approaches zero.
- Shorten delay when new epoch appears.

Trickle



- ❑ Concerns
  - ❖ Broadcast is expensive
  - ❖ Wireless channel is a shared, spatial resource
- ❑ Idea
  - ❖ Dynamic adjustment of transmission period
  - ❖ Suppress transmissions that may be redundant

Trickle

- "Every once in a while, broadcast what data you have, unless you've heard some other nodes broadcast the same thing recently."
- Behavior (simulation and deployment):
  - Maintenance: a few sends per hour
  - Propagation: less than a minute
  - Scalability: thousand-fold density changes
- Instead of flooding a network, establish a trickle of packets, just enough to stay up to date.
- As long as each node communicates with others, inconsistencies will be found
- Either reception or transmission is sufficient

Algorithm

- Define a desired detection latency, ***t***
- Choose a redundancy constant k
  - (receptions + transmissions) <= ***k***
  - In an interval of length ***t***
- Trickle keeps the rate as close to ***k/t*** as possible

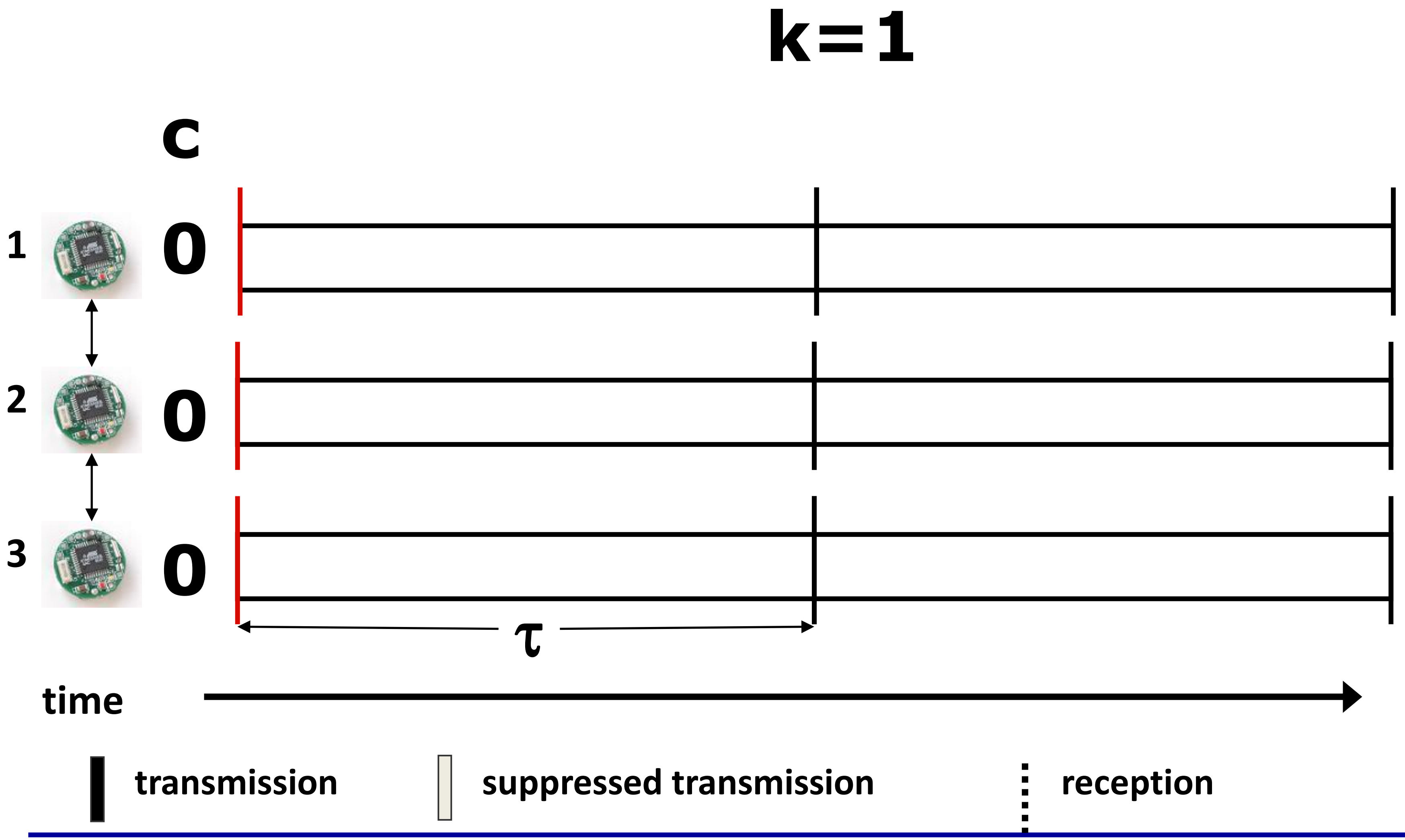
- Choose timer ***t*** random in (t/2, t]
- If inconsistent broadcast is heard before ***t***, reset t to t_{min}.
- If ***c*** < ***k*** consistent broadcasts are heard by ***t***, broadcast
- Otherwise suppress and double t up to t_{max}.

- When there is nothing new to say, stay quiet

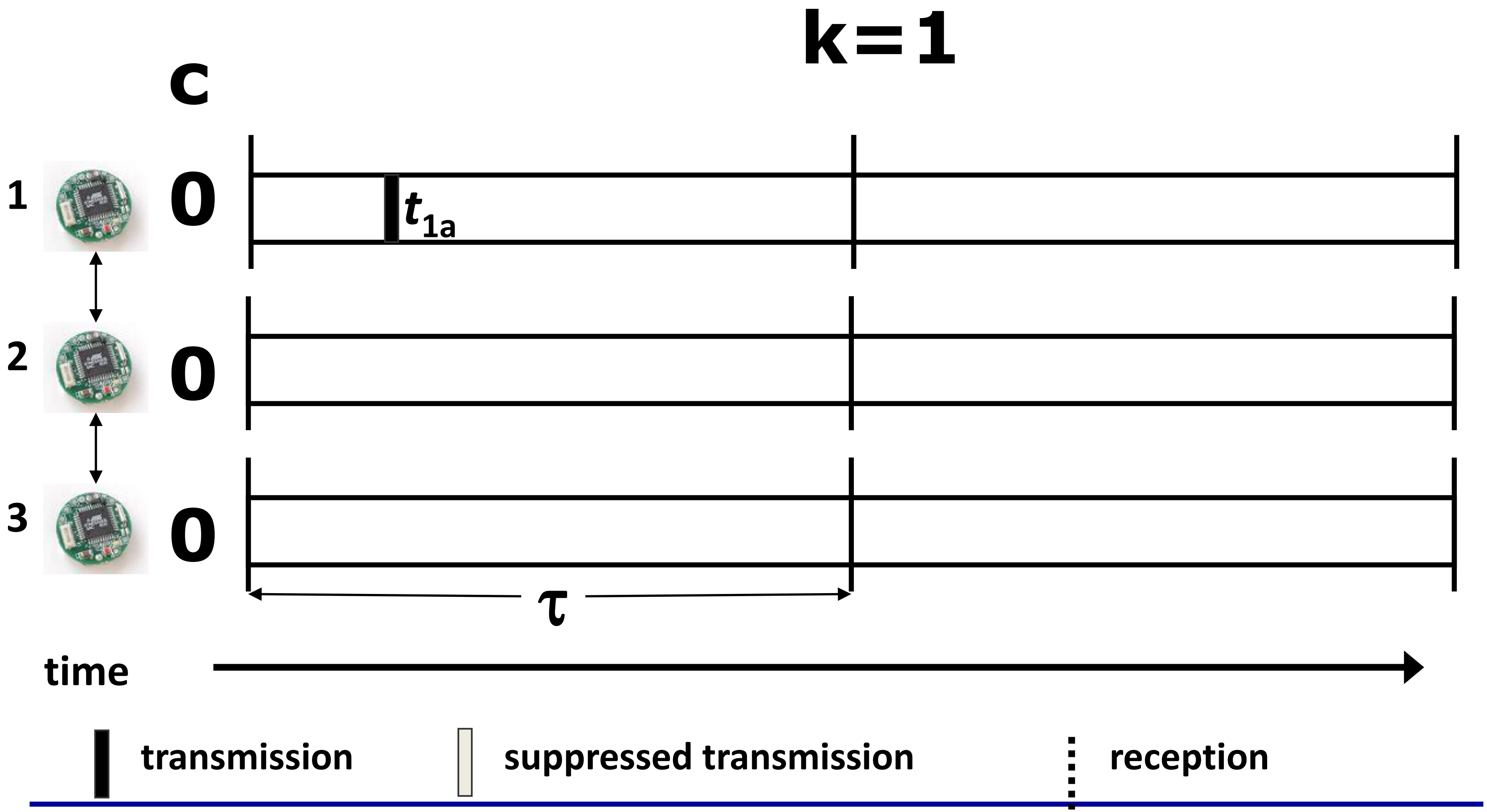
Trickle Algorithm

- Time interval of length τ
- Redundancy constant k (e.g., 1, 2)
- Pick a time t from $[0, \tau]$
- Maintain a counter c, initialized to zero
- At time t, broadcast code metadata if $c < k$
- Increment c when you hear identical metadata to your own
- At end of τ, pick a new t

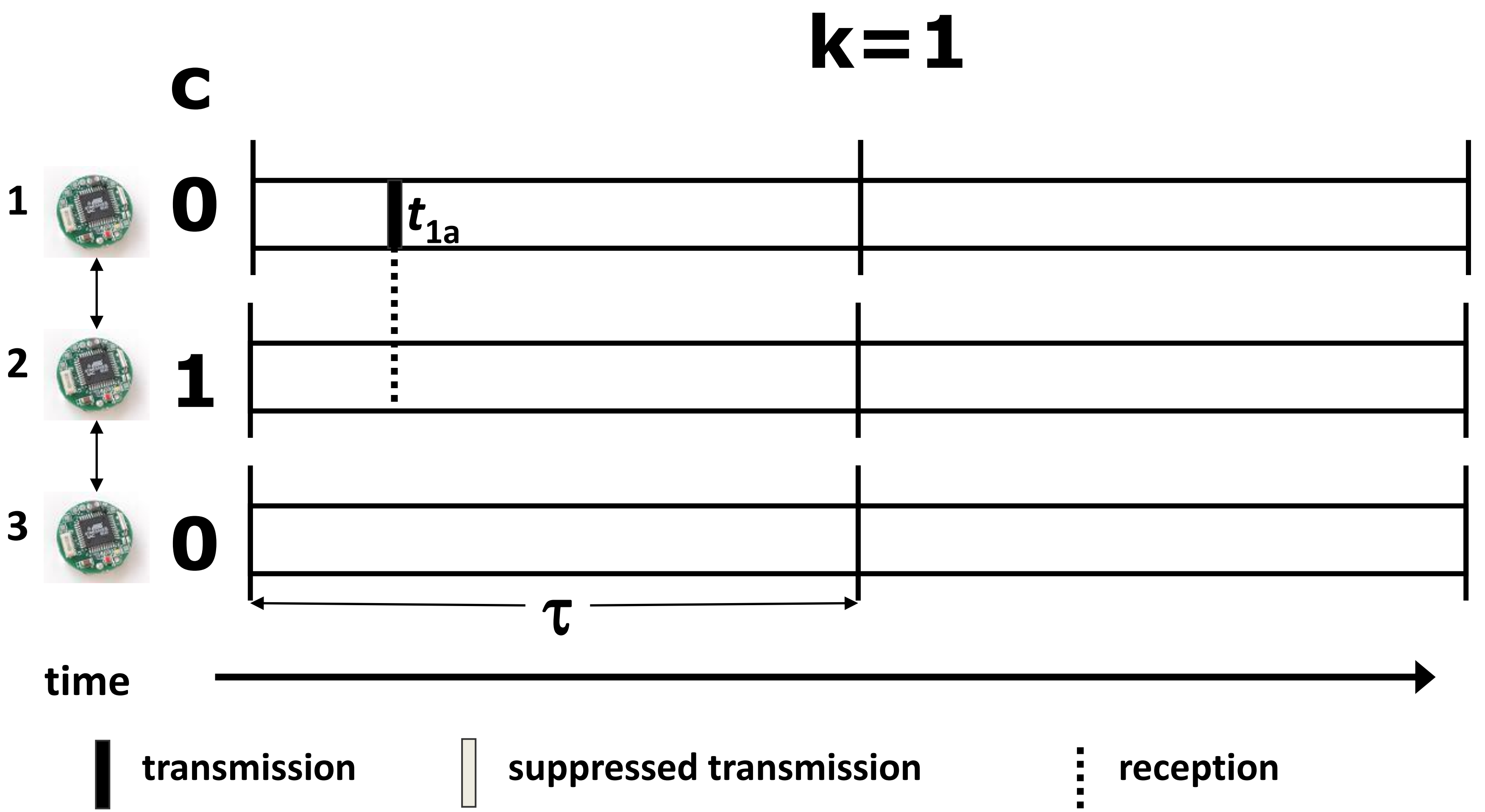
Example Trickle Execution



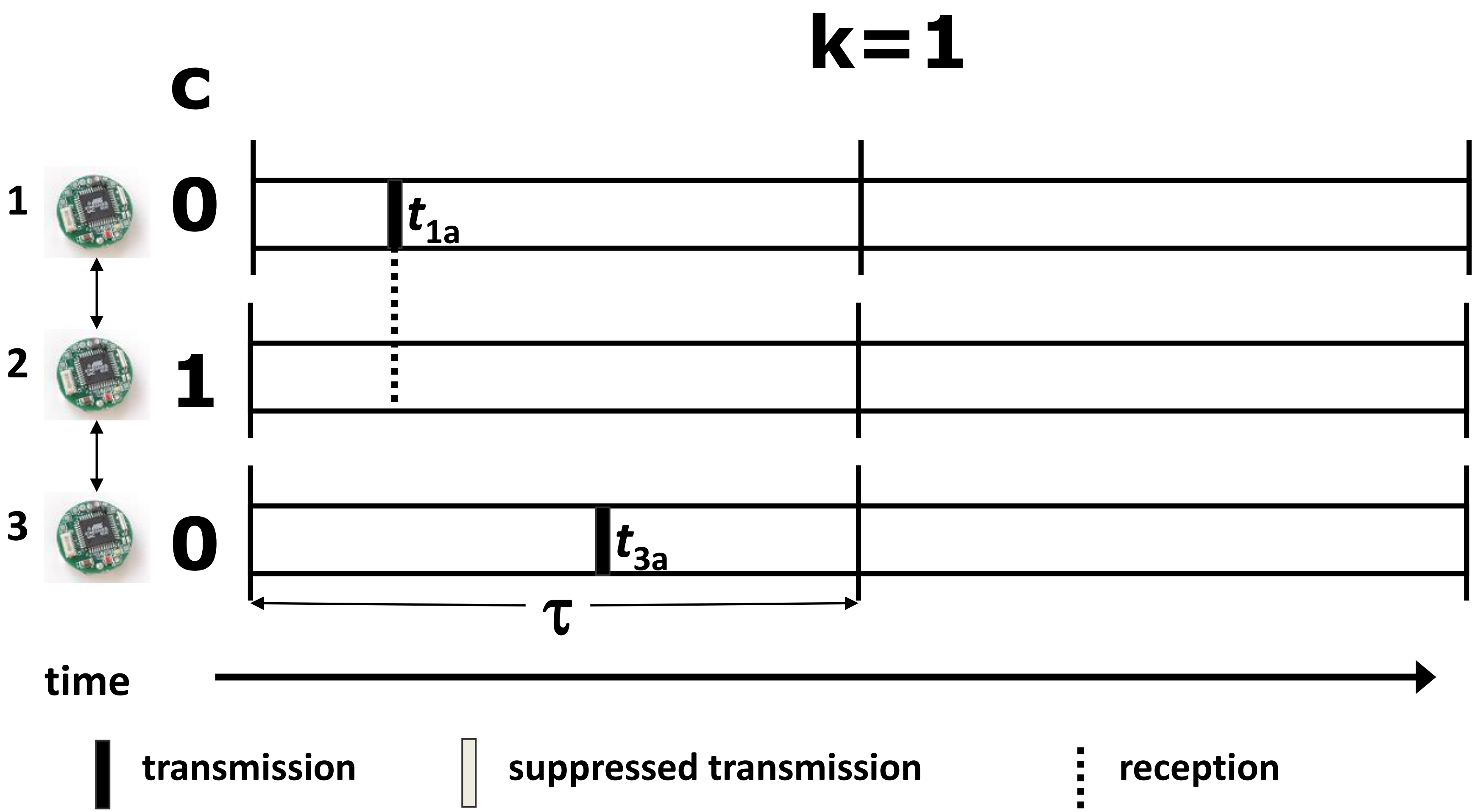
Example Trickle Execution



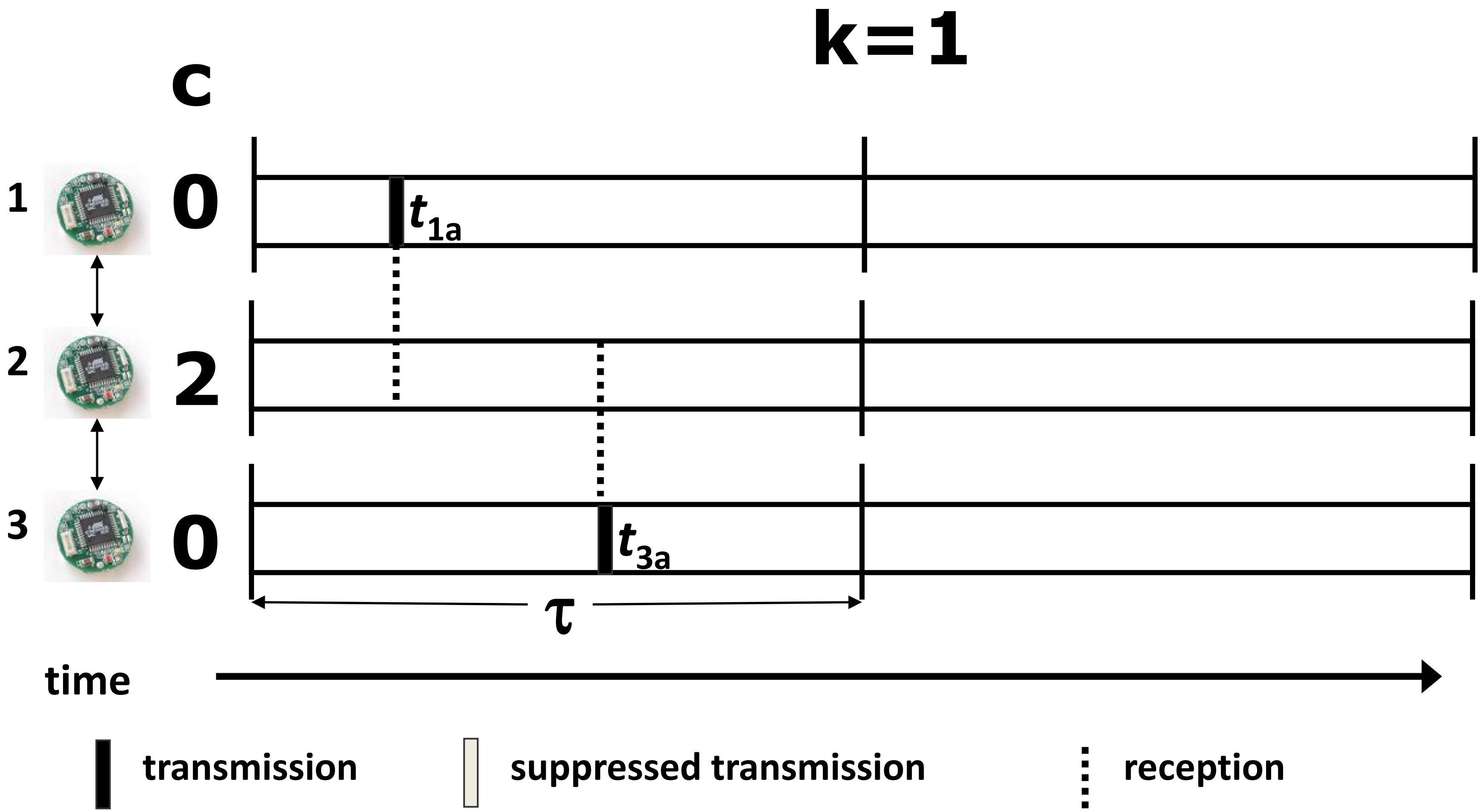
Example Trickle Execution



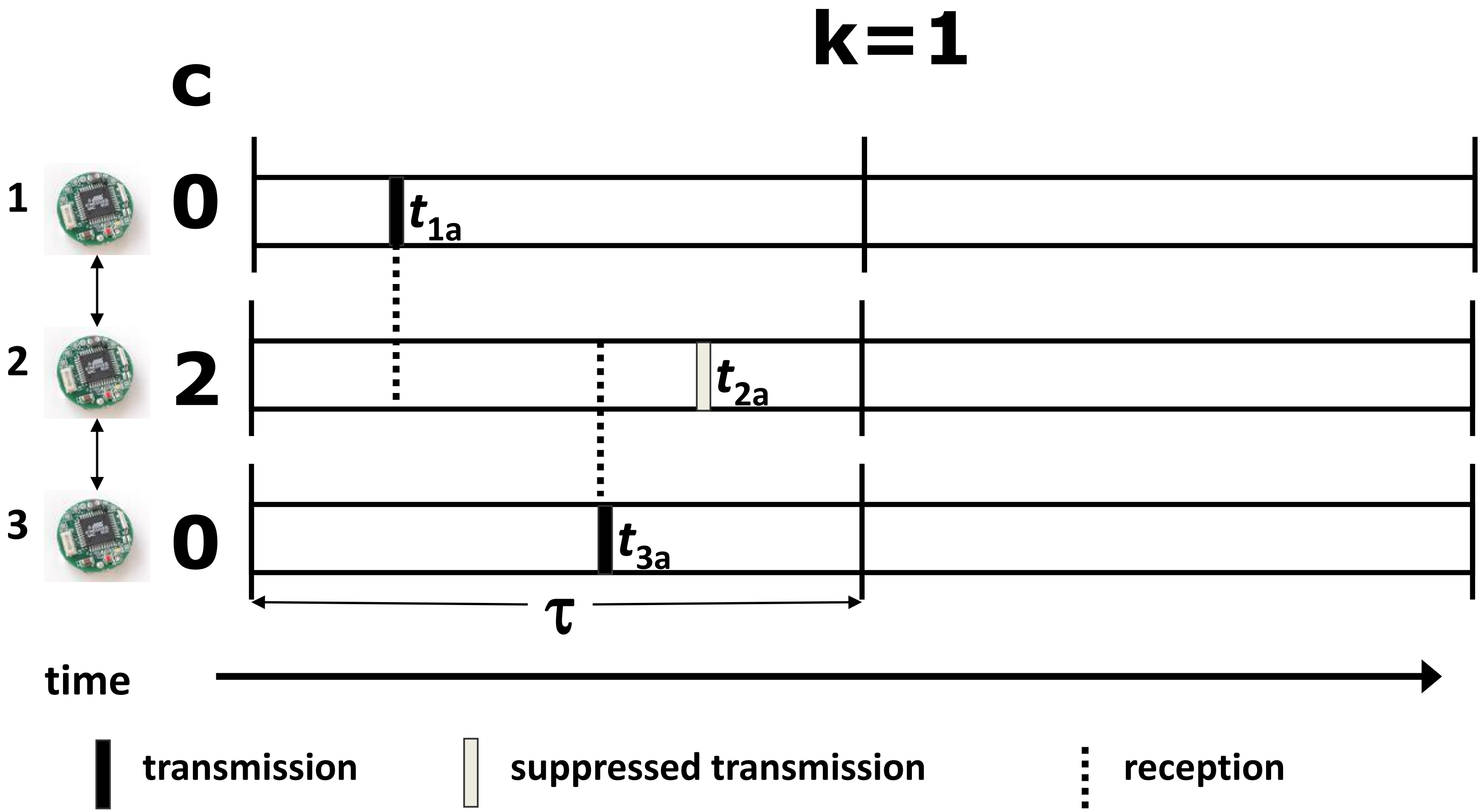
Example Trickle Execution



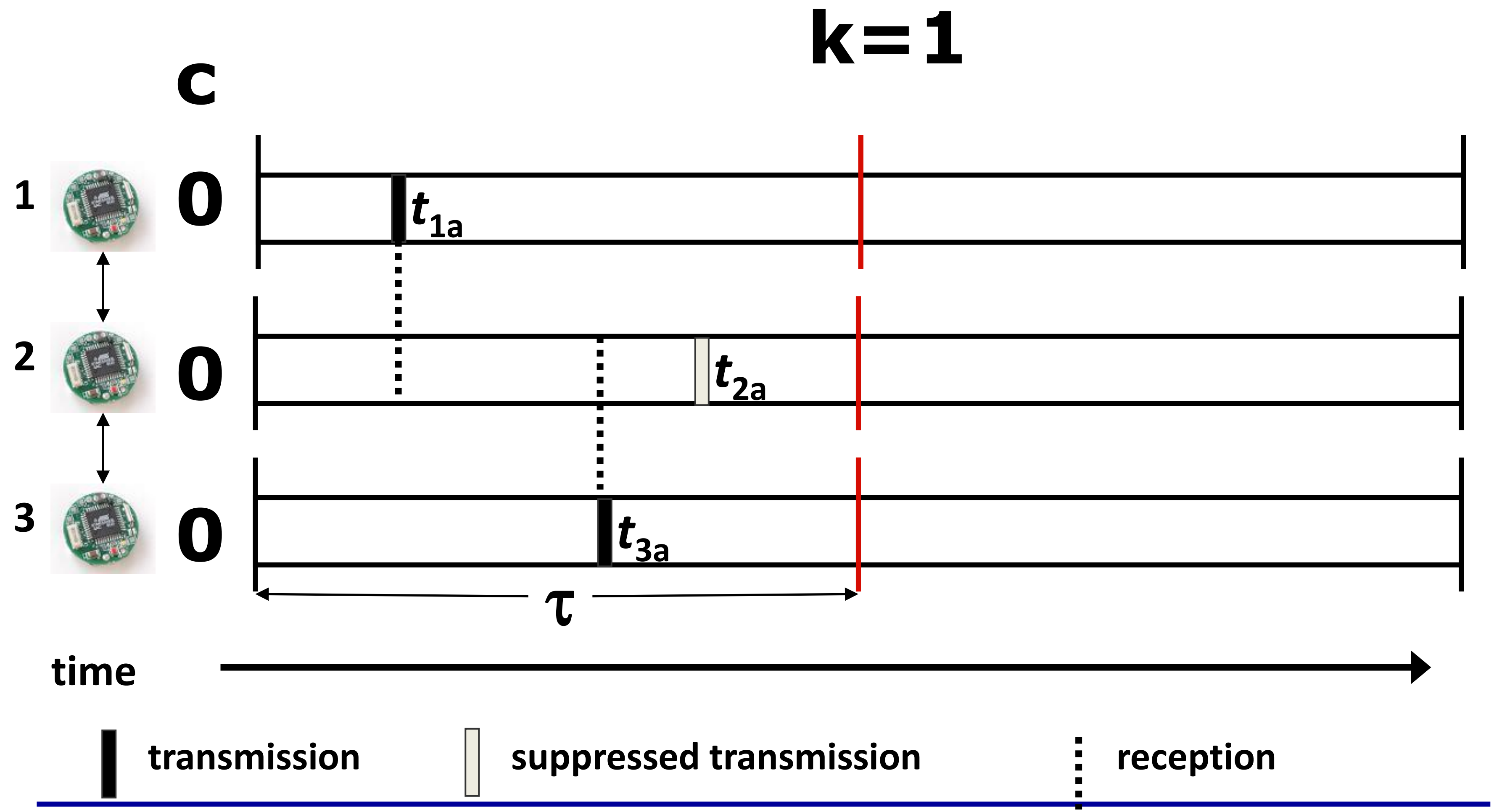
Example Trickle Execution



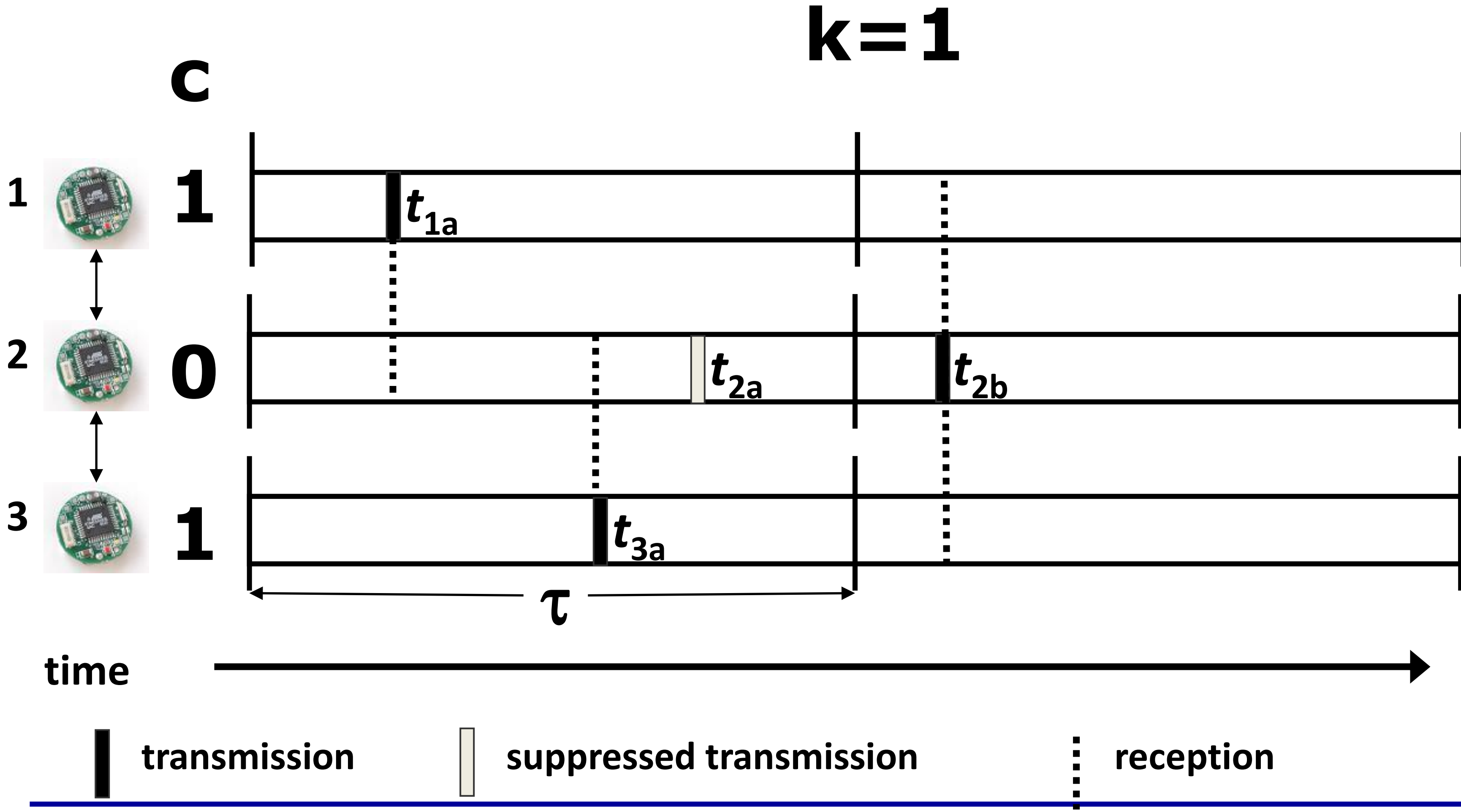
Example Trickle Execution



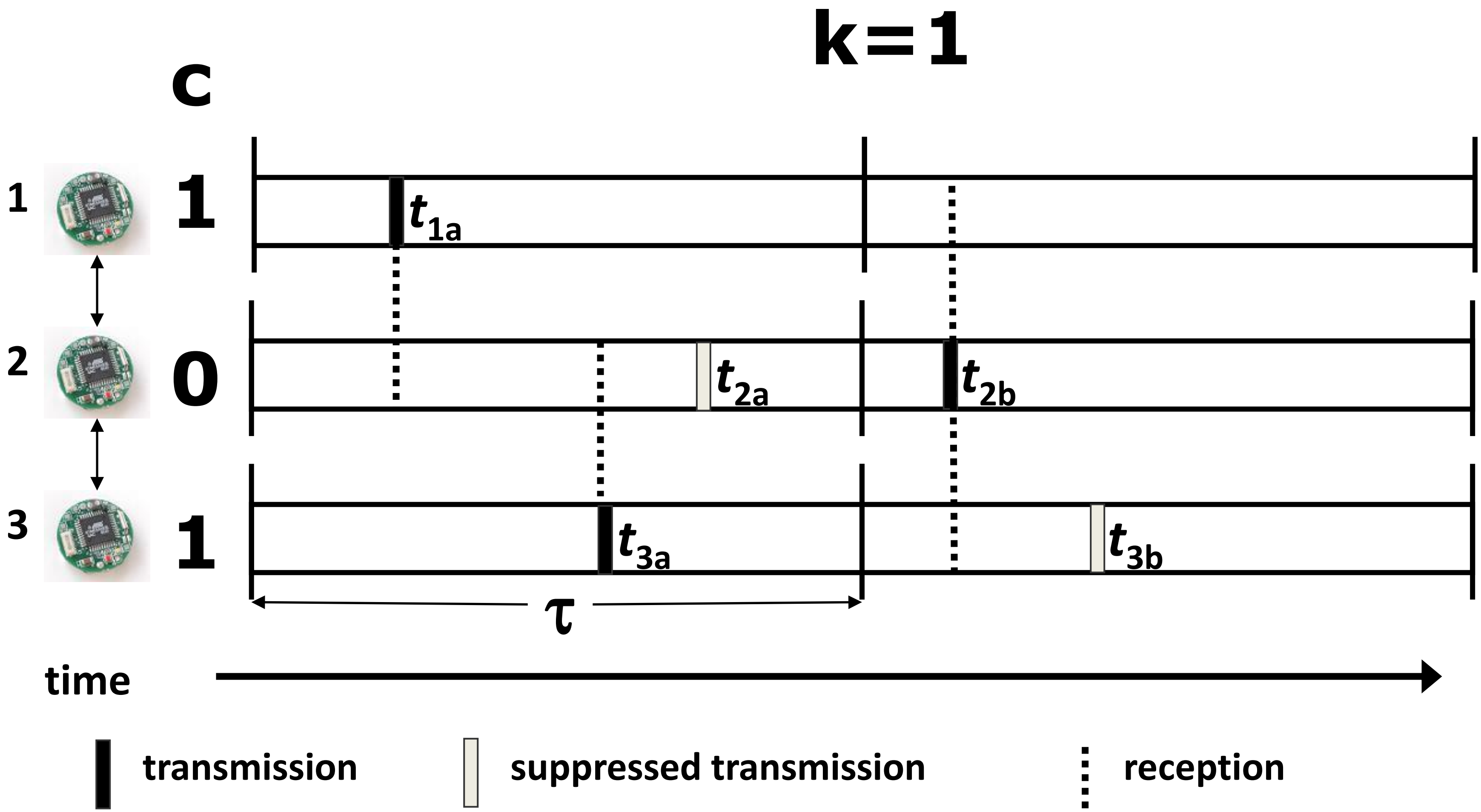
Example Trickle Execution



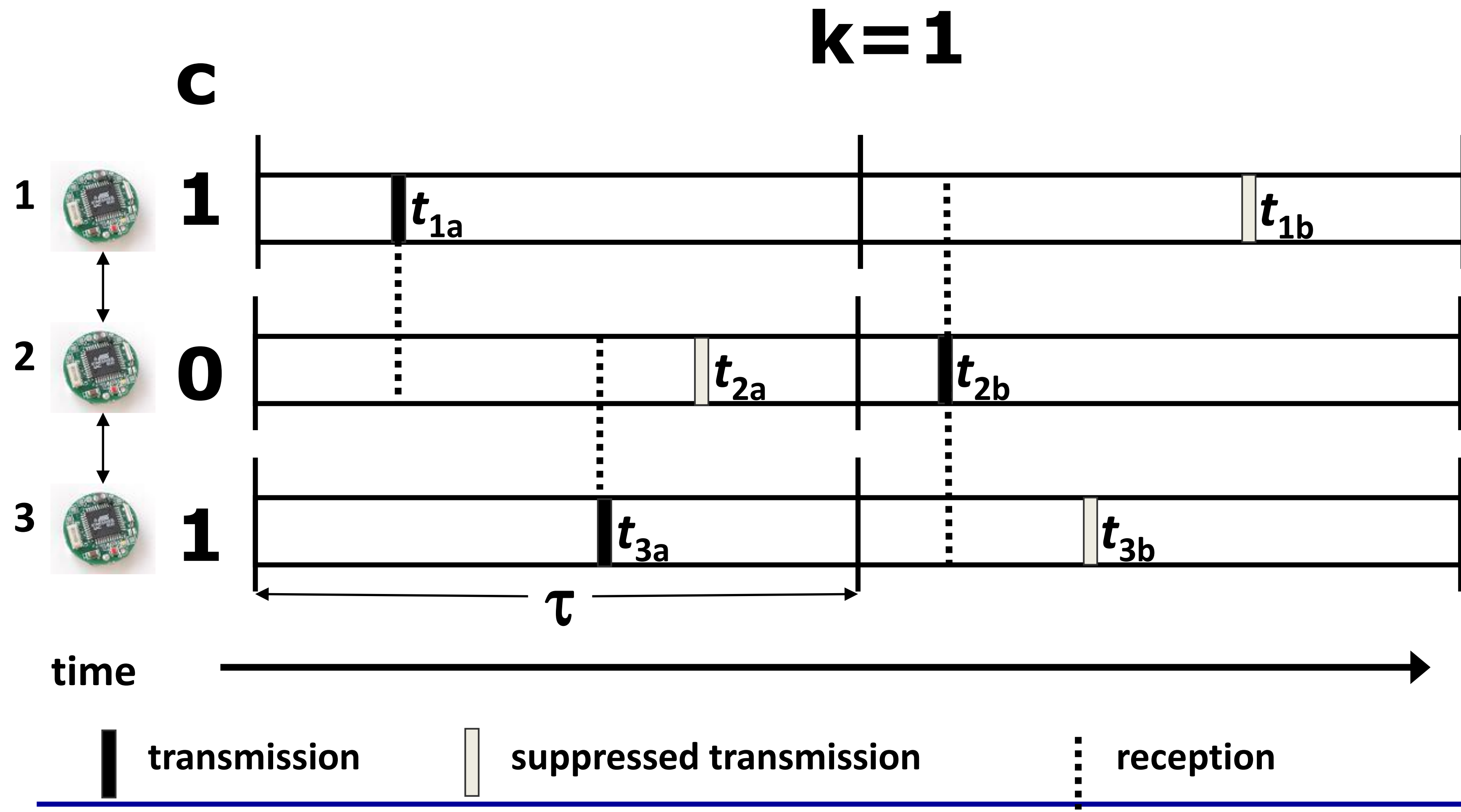
Example Trickle Execution



Example Trickle Execution



Example Trickle Execution



Ideal case

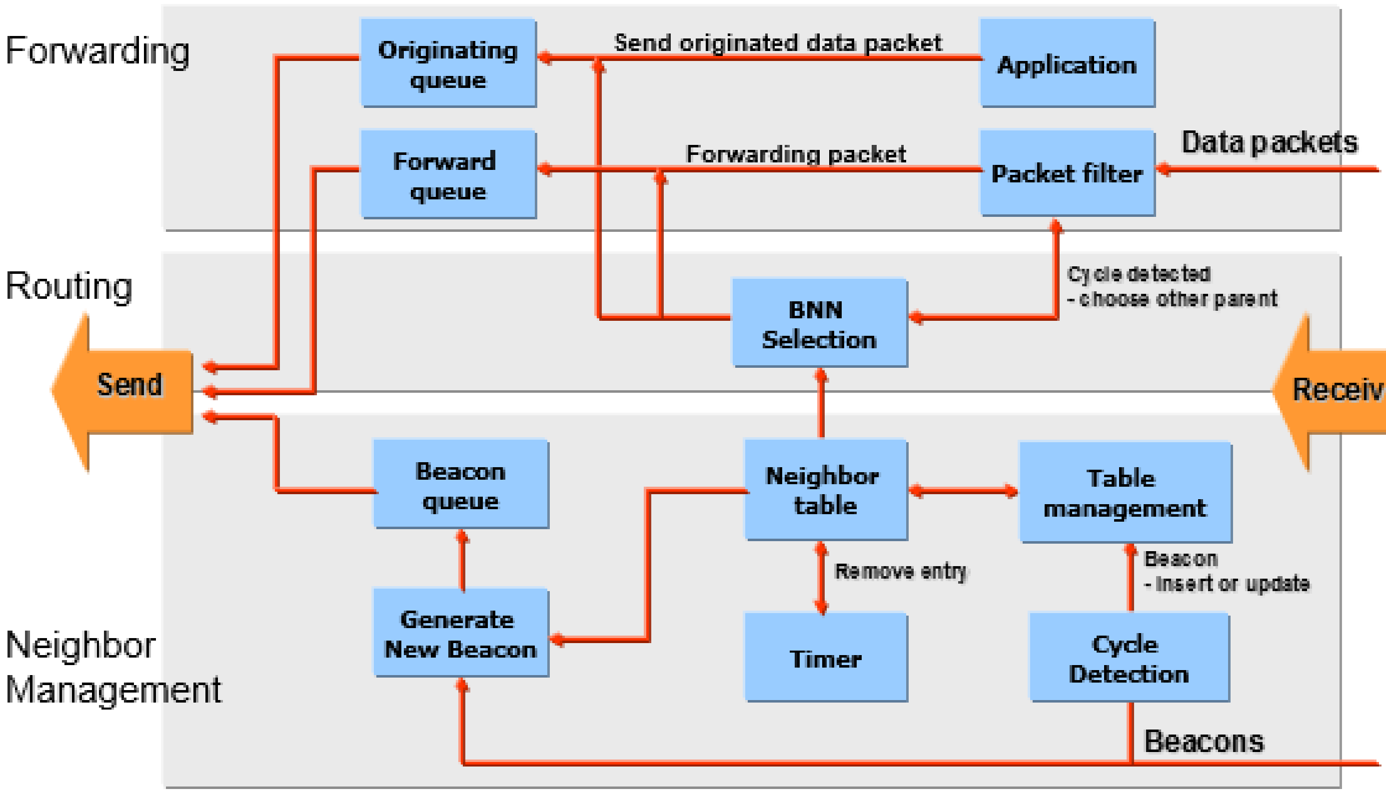
- ***k*** transmissions per interval
- First ***k*** nodes to transmit suppress all others
- Independent of density

Link Characteristics

- RSSI/LQI given by hardware
- How can we consider a link good or bad?
  - Based on RSSI/LQI?
  - Based on PRR?
- Neighbor Management
  - Policy: Add/Remove
  - Information Exchange: Link Estimation Exchange Protocol (LEEP): measure the link quality based on number of receiving:
    - Data Packet
    - Beacon

Routing Design

- ❑ Including 3 sub-layers: Neighbor Management, Routing and Forwarding



Simple Address-Free Flooding Protocol

- Root broadcasts a "new" message to local neighborhood
- Each node performs a simple rule

```
if ("new" incoming message) then {
      take local action
      retransmit modified message
}
```

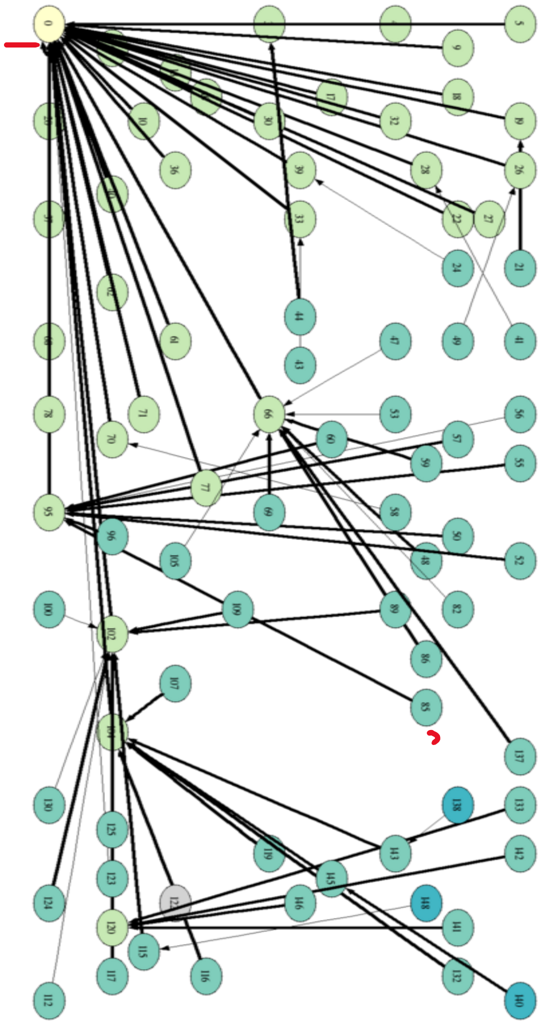
- No underlying routing structure required
  - The connectivity over physical space determines it.

CTP: Collection Tree Protocol (TEP123)

- The Collection Tree Protocol (CTP) is a tree-based protocol with some tree root nodes
- CTP is address free
- Proactive Routing & Distance Vector
- Nodes generate routes to root using rooting gradient
- CTP assumes that the data link layer provides:
  - efficient local broadcast address
  - synchronous ACKs for unicast packets
  - protocol dispatch field (support higher-level protocols)
  - single-hop source and destination fields

CTP: Collection Tree Protocol

- CTP assumes that it has link quality estimates of some number of nearby neighbors
- CTP has several mechanisms in order to improve delivery reliability (not promise 100%)
- CTP designed for relatively low traffic



CTP: Routing metric

- ETX (*Expected number of transmission*): measure each link's delivery probability with broadcast probes (& measure reverse)

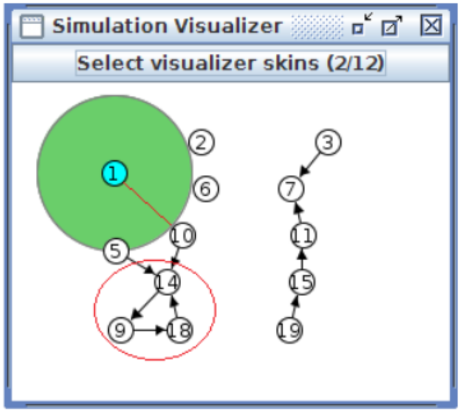
$$P_{delivery} = P_{data} * P_{ACK}$$

$$ETX_{link} = 1 / P_{delivery}$$

$$ETX_{route} = \Sigma (ETX_{link})$$

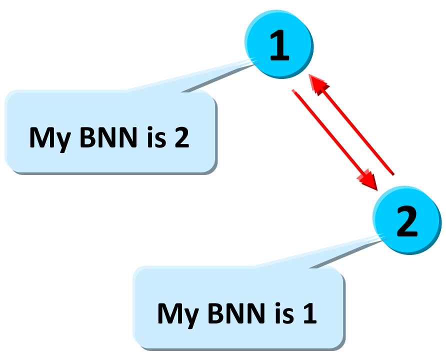
- CTP uses Expected Transmissions (ETX) as a routing metric
  - $ETX_{root} = 0$ and $ETX_{node} = ETX_{parent} + ETX_{link\text{-}to\text{-}parent}$
  - CTP should choose the route with the lowest Route-ETX (routing metric)
  - CTP represents ETX as 16-bit fix-point real number with precision of hundredths

- Main problems
  - Routing loops
  - Packet duplication
  - Network partitioning

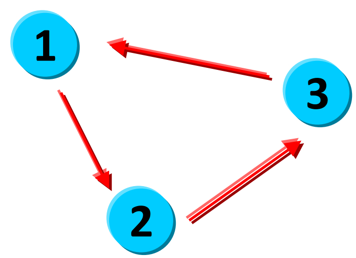


Loop – Duplication Problems

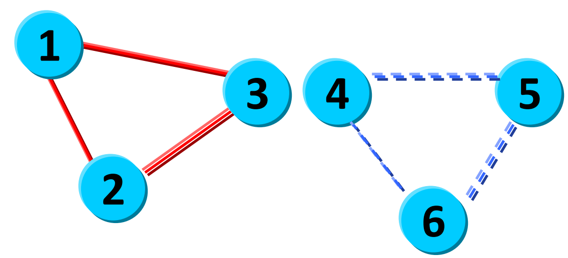
- **Beacon frame**
  - Loop: using **Sender** field to avoid. If a node receives a Beacon with **Sender** field equals to its ID: a loop occurs.
  - Duplicate: using **Sequence** field to suppress. If a node receives a Beacon with lower **Sequence** value -> not rebroadcast this beacon
- **Data frame**
  - Duplicate: using **Sequence** field in header of Data packet
  - Loop: two-hop, multi-hop and network partitioning. So, each node will have 2 BNNs if possible



If **Sender** field in packet and node's BNN are equal, try to use another BNN



If a mote receives its own packets, tries another BNN

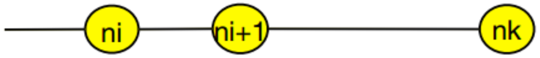


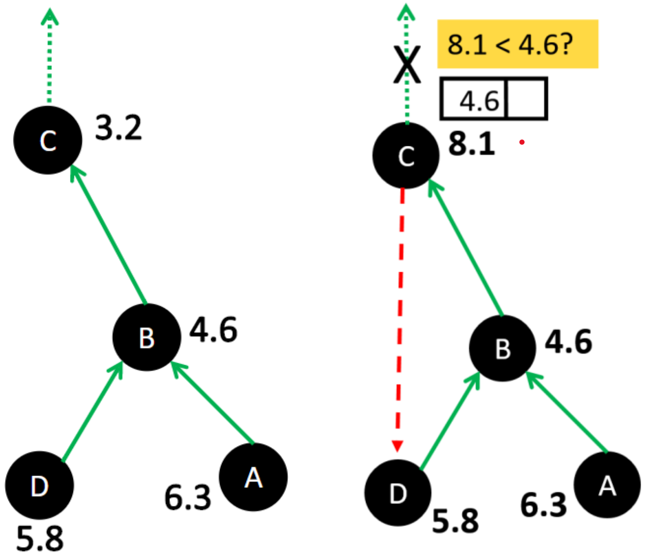
If TTL equals to a max value, discards packets because of network partitioning

CTP: Routing loop

- Occur when a node choose a new route with higher ETX than its old one
- Two mechanisms to address this problem:
  - CTP packet contains a node's current cost
    - the data frame with lower cost indicates inconsistency
    - Do not drop packets
    - try to solve inconsistency by broadcasting a beacon frame
  - Not consider routes with an ETX higher than a reasonable constant
- Data path validation
  - Cost in the packet
  - Receiver checks

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





CTP: Packet Duplication

- Occurs when a node receives a packet successfully, but the ACK is not received by the sender
- The sender retransmits the packet and the receiver receives it a second time
- The duplication is exponential
- CTP data frames have Time Has Lived (THL) field which was incremented by routing layer

CTP Data Frame

- **P** (Routing Pull)
  - allows nodes to request routing information
  - if **P** is set the node should transmit a routing frame
- **C** (Congestion notification)
  - if a node drops a CTP data frame it must set the **C** bit field on the next data frame
- **THL** (Time Has Lived)
  - if a node generates a CTP data frame, it must set THL to 0
  - if a node receives a CTP data frame must increment the THL

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
P	C	reserved						THL							
ETX															
origin															
seqno								collect_id							
data...															

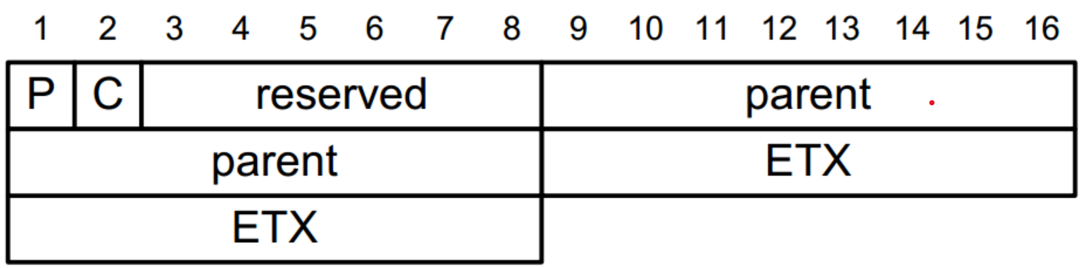
CTP Data Frame

- **ETX** (Expected Transmissions)
  - the ETX is the routing metric of the *single-hop* sender
  - node send a CTX data frame must put the ETX of its routes
  - node receives a packet with lower gradient must schedule a routing frame
- **Seqno**
  - origin sequence number
- **Collect_id**
  - Higher-level protocol identifier
- **Data**
  - the data payload, of zero or more bytes

<table>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td></tr>
<tr><td>P</td><td>C</td><td colspan="6">reserved</td><td colspan="8">THL</td></tr>
<tr><td colspan="16">ETX</td></tr>
<tr><td colspan="16">origin</td></tr>
<tr><td colspan="8">seqno</td><td colspan="8">collect_id</td></tr>
<tr><td colspan="16">data...</td></tr>
</table>

CTP Routing frame (beacon)

- **P** (Routing Pull): same as data frame
- **C** (Congestion Notification): same as data frame
- **Parent**: the node's current parent
- **Metric** (ETX): the node’s current routing metric value

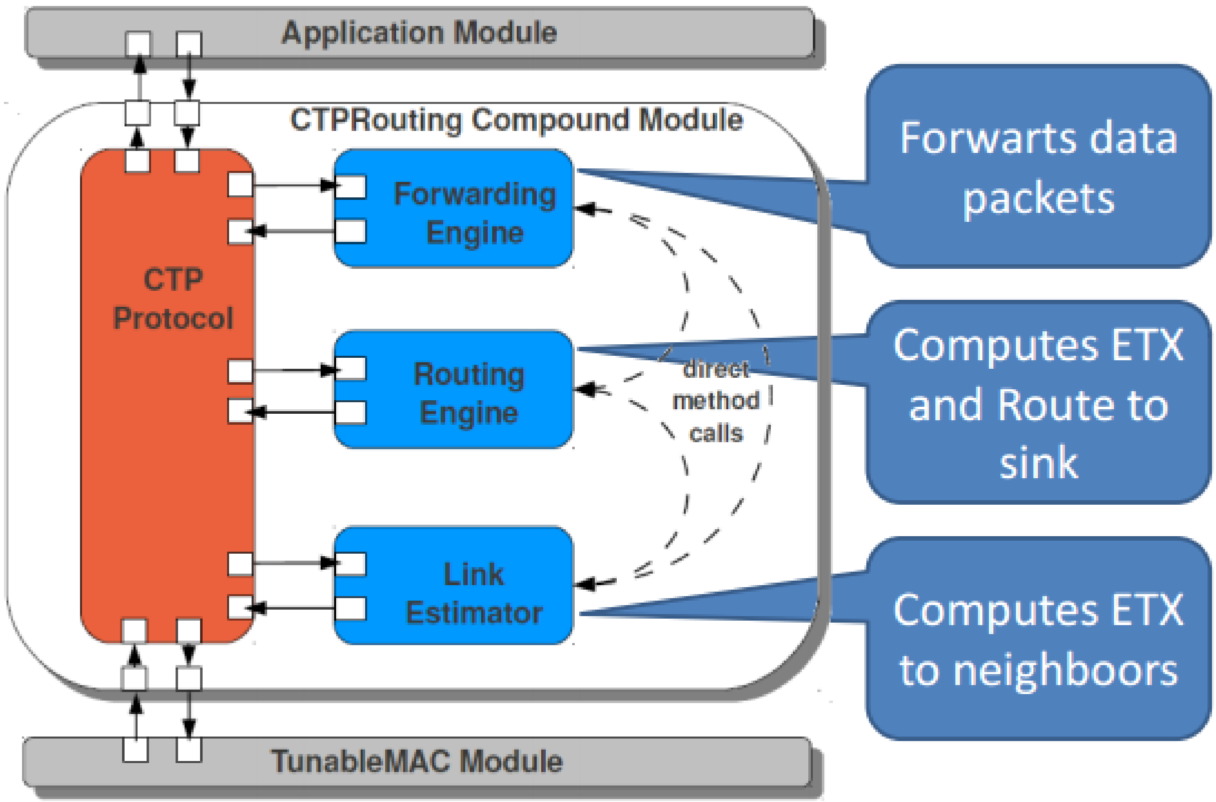


CTP: Operation

- After a node hears a routing frame (beacon), it must update its routing table
- If a node ETX value changes significantly, should transmit broadcast/beacon frame to notify other nodes
- The parent can detect when a child ETX is significantly below its own
  - parent must schedule a routing frame

CTP: Architecture

- Enable control and data plane interaction
- Two mechanisms for efficient and agile topology maintenance
- Data path validation



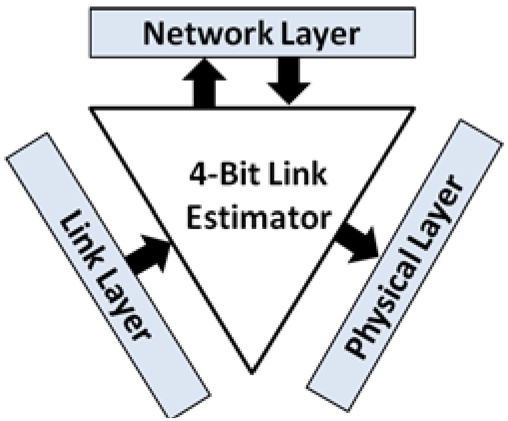
CTP: Implementation

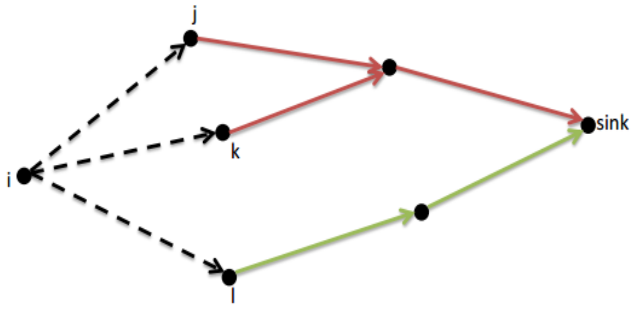
- Three major subcomponents
  - ***link estimator***
    - responsible for estimating the single-hop ETX of communication with single hop neighbors
  - ***routing engine***
    - uses link estimates to decide which neighbor is the next hop routing hop
  - ***forwarding engine***
    - maintains queue of packets to send
    - decides when and if to send them

CTP: Link Estimator

Two mechanism to estimate the link quality

- ❑ Periodic Link Estimation Extension Protocol (LEEP) packets.
  - ❖ sends routing beacons as LEEP seeds the neighbor table
  - ❖ similar to Trickle based dissemination
- ❑ Data packets
  - ❖ direct measure of ETX
  - ❖ estimator produces ETX estimate after 5 successfully acknowledged packet transmission





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

CTP: 4B Link Estimator

The ETX values are separately calculated for the sent unicast packets and received beacons

- The unicast ETX value is updated every k_{uw} **unicast packets**. k_{uw} is called the unicast update window.
  If a out of k_{uw} packets are acknowledged by the receiver, the unicast ETX estimate is:

$$\boldsymbol{ETX_u = \frac{k_{uw}}{a}}$$

- The beacon ETX value is updated every k_{bw} **beacons** (of which some might be missed). k_{bw} is called the beacon update window. The calculation is similar but involves an extra step. First the packet reception ratio (PRR) is calculated based on the number of received beacons R_b and failed beacons F_b:

$$\boldsymbol{PRR_{last} = \frac{R_b}{R_b + F_b}}$$

" Four Bit Wireless Link Estimation " by Rodrigo Fonseca, Omprakash Gnawali, Kyle Jamieson, and Philip Levis. In *Proceedings of the Sixth Workshop on Hot Topics in Networks (HotNets VI)* , 2007.

CTP: 4B Link Estimator

This instantaneous PRR value is dampened using an exponentially weighted moving average (EWMA) function:

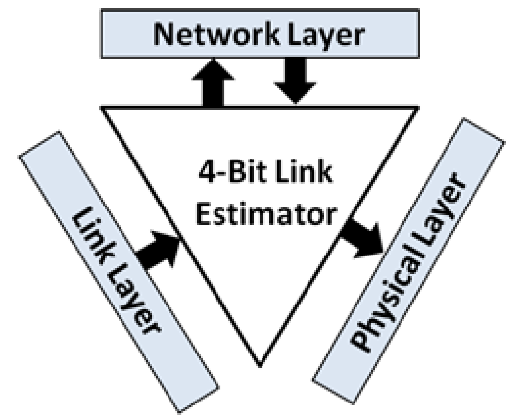
$$PRR_{new} = \alpha\, PRR_{old} + (1-\alpha)\, PRR_{last}$$

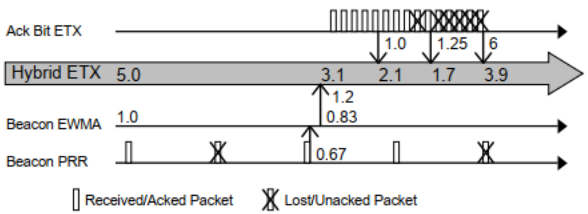
with α being a weighting factor between 0 and 1.
The resulting PRR value is then inversed to turn it into an ETX value:

$$ETX_b = \frac{1}{PRR}$$

These two streams of ETX values are combined in a second EWMA:

$$ETX_{hybrid} = \alpha\, ETX_u + (1-\alpha)\, ETX_b$$





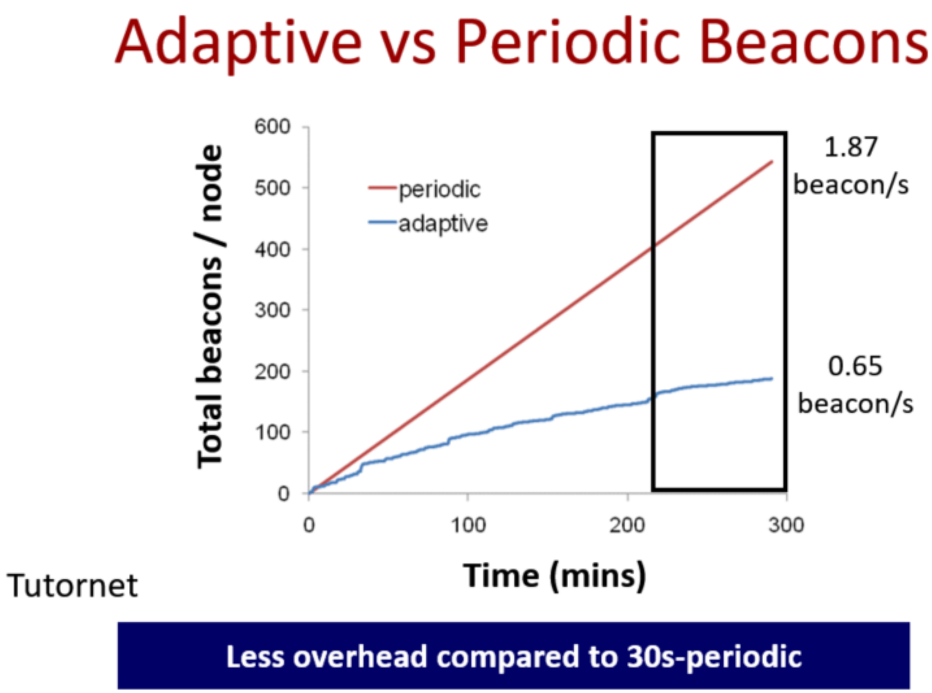
When there is heavy data traffic, unicast estimates dominate.
When the network is quiet, broadcast estimates dominate

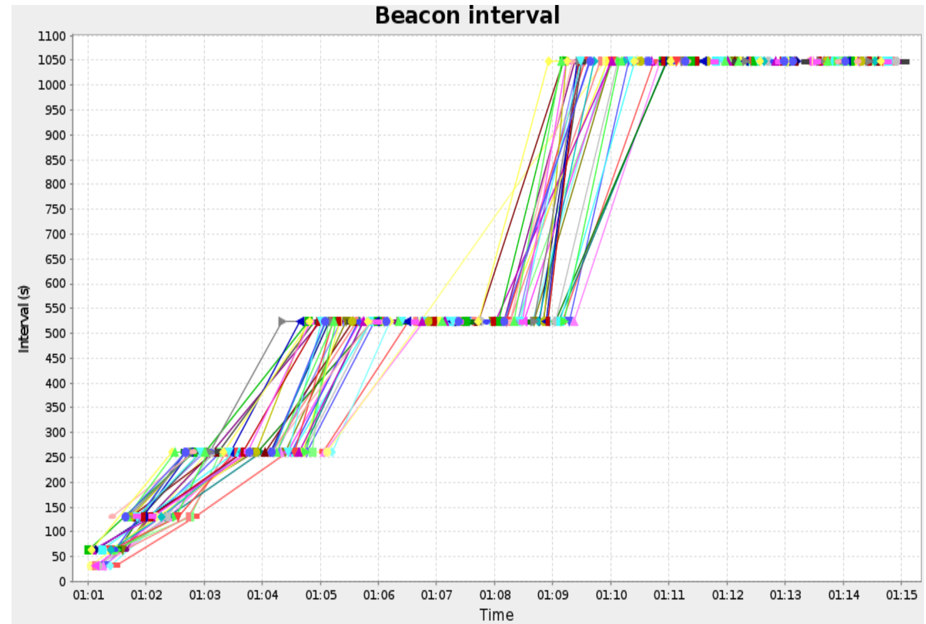
" Four Bit Wireless Link Estimation " by Rodrigo Fonseca, Omprakash Gnawali, Kyle Jamieson, and Philip Levis. In *Proceedings of the Sixth Workshop on Hot Topics in Networks (HotNets VI)* , 2007.

CTP: Adaptive Beaconing

- **Adaptive Beaconing:** Routing protocols typically broadcast control packets at a fixed interval (e.g., every 30 seconds). This interval poses a basic tradeoff.
- A small interval, i.e., frequent beacons, makes the protocol more responsive to the changes in the network, but uses more bandwidth and energy. A large interval uses less bandwidth and energy but can let topological problems persist for a long time.
- CTP uses adaptive beaconing to break this tradeoff. When the topology is inconsistent and has problems, it sends beacons *faster*. Otherwise, it *decreases* the beaconing rate exponentially. Thus, CTP can quickly respond to adverse wireless dynamics while incurring low control overhead in the long term

CTP: Adaptive Beaconing



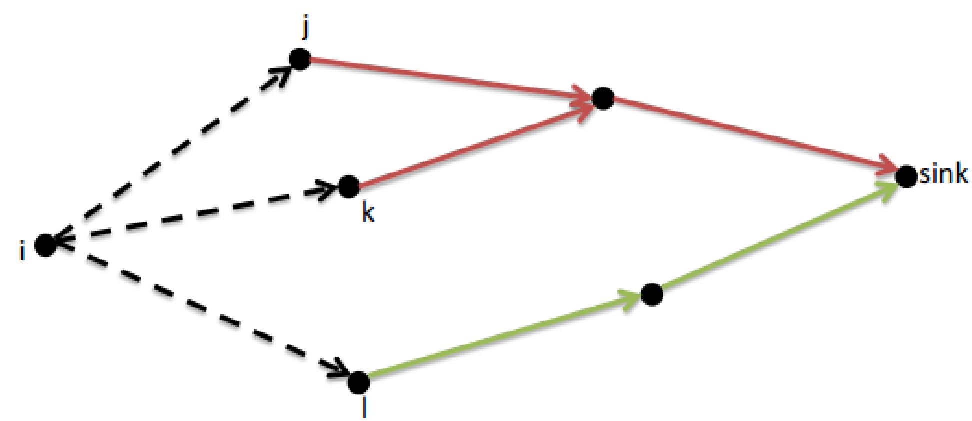


Routing Engine

- Picking the next hop for data transmission
- Keeps track of the path ETX of the subset of the nodes
- The minimum cost route has the *smallest sum*
    - the path ETX from that node
    - the link ETX of that node

Link estimator:

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



Routing Engine:

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

➡ **Choose the route having min $ETX_{multihop}$**

Forwarding Engine

- Transmitting, retransmitting packets to the next hop and passing ACK based information to the link estimator
- Deciding when to transmit packets to the next hop
- Detecting routing inconsistencies and informing the routing engine
- Maintaining a queue of packets to transmit (local and forwarded)
- Detection singe-hop transmission duplicates

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

CTP: Summary

- Advantages
  - Consistent routing
  - Suitable for many-to-one application
  - High PRR
  - Evaluation:
    - CTP delivers >90% of packets (usually 99.9%)
    - CTP sends 73% fewer beacons than others
    - CTP reduces topology repair latency by 99.8%
- Disadvantages
  - Do not support any-to-any routing (e.g. IP application)