Institute for the Wireless Internet of Things at Northeastern University

EECE 5155 Wireless Sensor Networks (and The Internet of Things)

Prof. Francesco Restuccia
Email: f.restuccia@northeastern.edu
Office: 318 Dana

February 24, 2021

Contention-Free Mac Protocols: TRAMA

TRAMA

TRAMA: Energy Efficient Collision-Free MAC, V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, ``Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks,'' Proc. ACM SenSys 2003, LA, CA, Nov. 2003.

- Motivation:
  - Probability of collisions of both control and data packets in a contention-based scheme increases with traffic
  - This degrades channel utilization and reduces battery lifetime

- Idea:
  - Establish **transmission schedules** to avoid collisions at the receiver
  - Make schedules **dynamic, adaptive** to traffic patterns
  - Make nodes switch to **low-power mode** according to **dynamic** schedules, i.e., when there is no data packet intended for those nodes

TRAMA

- Time divided into period

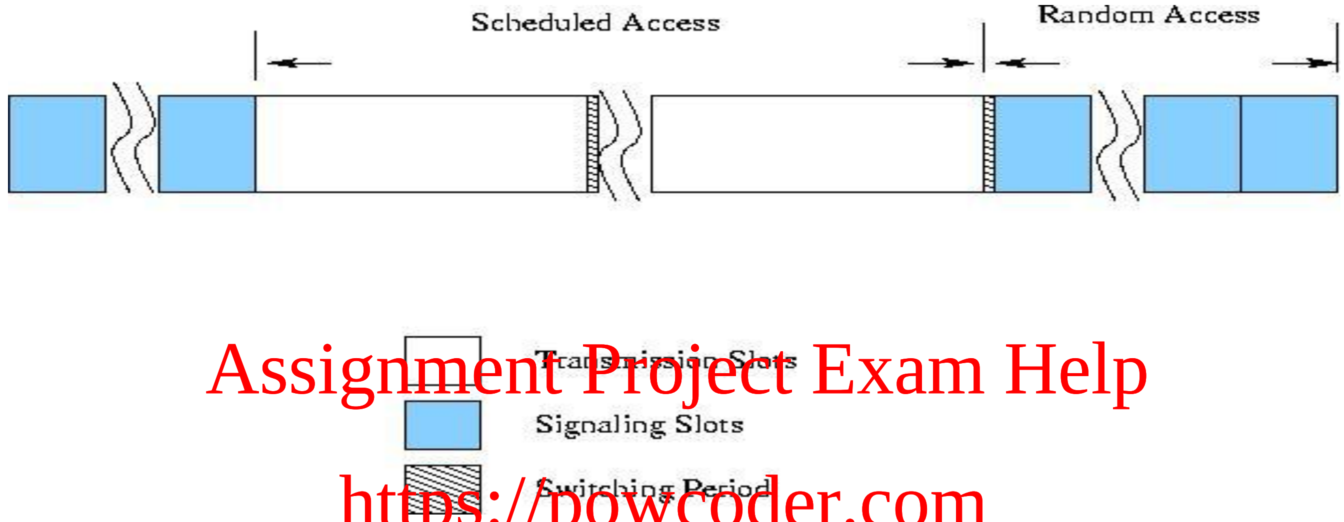
- Random Access Period
  - Used for **signaling**: updating two-hop neighbor information
  - Collisions are **possible**

- Scheduled Access Period
  - Used for contention free data exchange between nodes
  - Supports unicast, multicast and broadcast communication

TRAMA Components

- Neighbor Protocol (NP)
  - Gather 2-hop neighbors information
- Schedule Exchange Protocol (SEP)
  - Gather 1-hop traffic information for SCHEDULING
- Adaptive Election Algorithm (AEA)
  - Select transmitters, receivers for current time slot
  - Leave other nodes free to switch to low power mode using the NP and SEP results

TRAMA



- SIGNALING SLOTS
  - Used by NEIGHBOR PROTOCOL (NP) to propagate one-hop neighbor information among neighboring nodes during the random access period
  - In this way, a consistent two-hop topology information across all nodes is obtained

- TRANSMISSION SLOTS
  1. Used for collision-free data exchange
  2. Used for schedule propagation

Neighbor Protocol (NP)

- Gather two-hop neighborhood information by using **signaling packets** during the random access period
- If no updates, signaling packets are sent as "**keep-alive**" beacons
- A node times out if nothing is heard from its neighbor

Schedule Exchange Protocol (SEP)

- Each node computes a SCHEDULE INTERVAL (named SCHED) based on the rate at which packets are produced
- Quantity SCHED represents # of slots for which the node can announce the schedule to its neighbors according to its current state (queue)
- The node pre-computes # of slots in the interval

[t, t+SCHED]

for which it has the highest priority among its two-hop neighbors (contenders) → WINNING SLOTS

Schedule Exchange Protocol (SEP)

- The node announces the intended receivers for these slots

- The last winning slot is used for broadcasting the node's schedule for the next interval (example later)

If these winning slots cannot be filled by the node the remaining vacant slots can be released to other nodes

Schedule Exchange Protocol (SEP)

- EXAMPLE: Node **u** → SCHED is 100 slots
- During time slot 1000, u computes its winning slots between [1000,1100] - **HOW?**
- Assume: These slots are 1009, 1030, 1053, 1064, 1075, 1098
- **Node u** uses slot 1098 to announce its next schedule by looking ahead from [1098,1198]

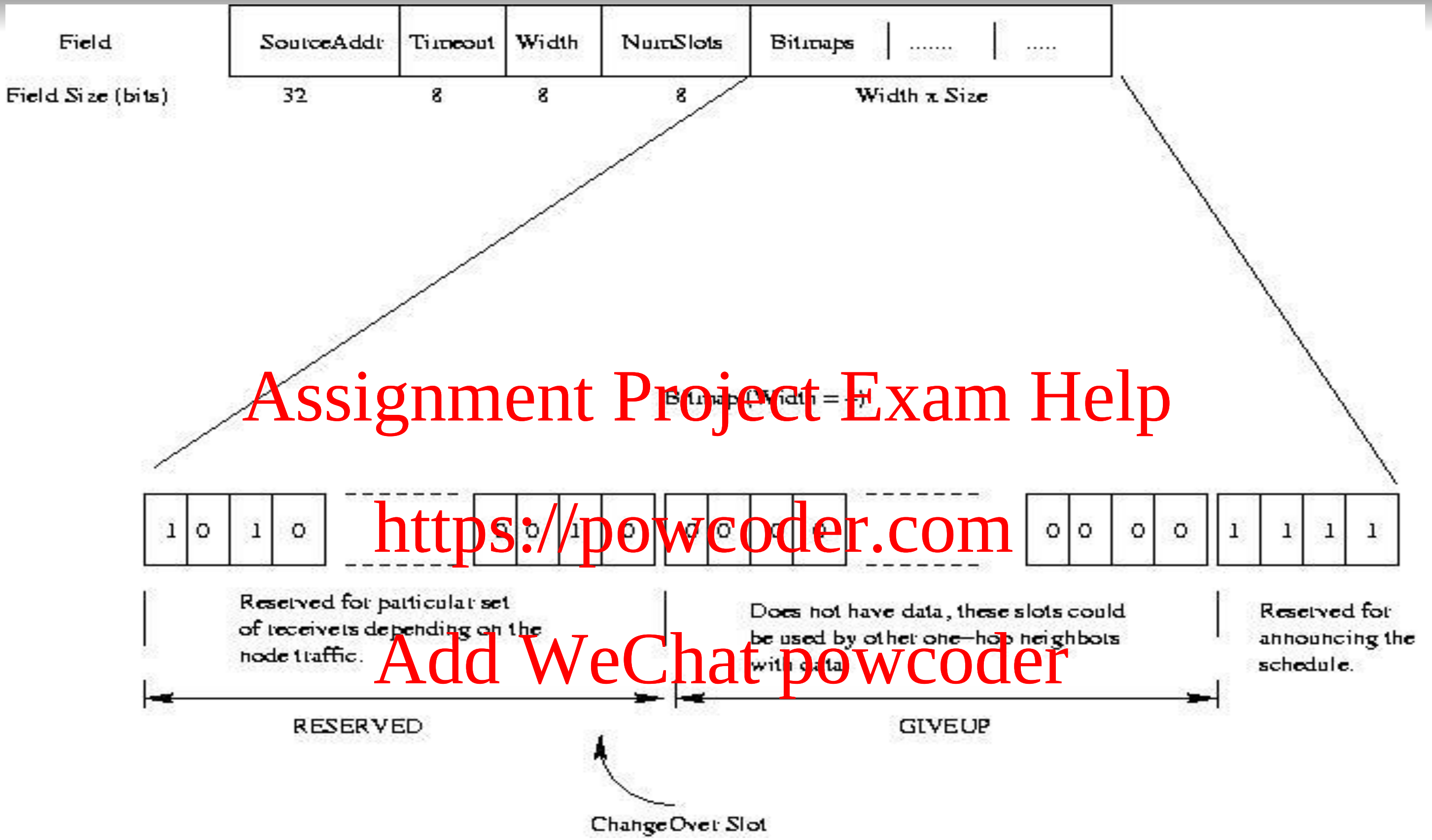
Schedule Exchange Protocol (SEP)

- Nodes announce their schedules via SCHEDULE PACKETS
- Use BITMAP: with the length equal to # of one-hop neighbors to indicate receivers
- Each bit corresponds to one particular receiver
- Example: One node with 4 neighbors 14,7,5 and 4
- BITMAP → size 4
- For broadcast: all bitmap bits are set to 1

Adaptive Election (AE)

- **Given:** Each node knows its two-hop neighborhood and their current schedules
- How to decide which slot (in scheduled access period) a node can use?
  - Use node identifier **x** and globally known hash function **h**
  - For time slot **t**, compute priority **p = *h* (x XOR t)**
  - Compute this priority for next **SCHED** time slots for node itself and all two-hop neighbors
  - Node uses time slots for which it has the highest priority
  - Gives up time slots for which it has no data to transmit

Schedule Packet Format



SourceAddr: Node announcing the schedule

Timeout: # of of slots for which the schedule is valid (starting from the current slot)

Width: Length of the neighbor bitmap (# of one-hop neighbors)

numSlots: total # of winning slots (# of bitmaps contained in the packet)

What are the main limitations of TRAMA?

Think-Pair-Share!

TRAMA Limitations

- Complex election algorithm and data structure
- Overhead due to explicit schedule propagation
- Higher queuing delay

- Energy savings in TRAMA depend on the workload situation
- Energy savings in S-MAC depend on duty cycle
- TRAMA has higher maximum throughput than contention-based S-MAC

- TRAMA disadvantage: substantial memory/CPU requirements for schedule computation

Institute for the Wireless Internet of Things at Northeastern University

EECE 5155
Wireless Sensor Networks (and The Internet of Things)

Prof. Francesco Restuccia
Email: f.restuccia@northeastern.edu
Office: 318 Dana

February 25, 2021

Hybrid Mac Protocols: Z-MAC

Z-MAC

Z(ebra)-MAC: A HYBRID MAC PROTOCOL

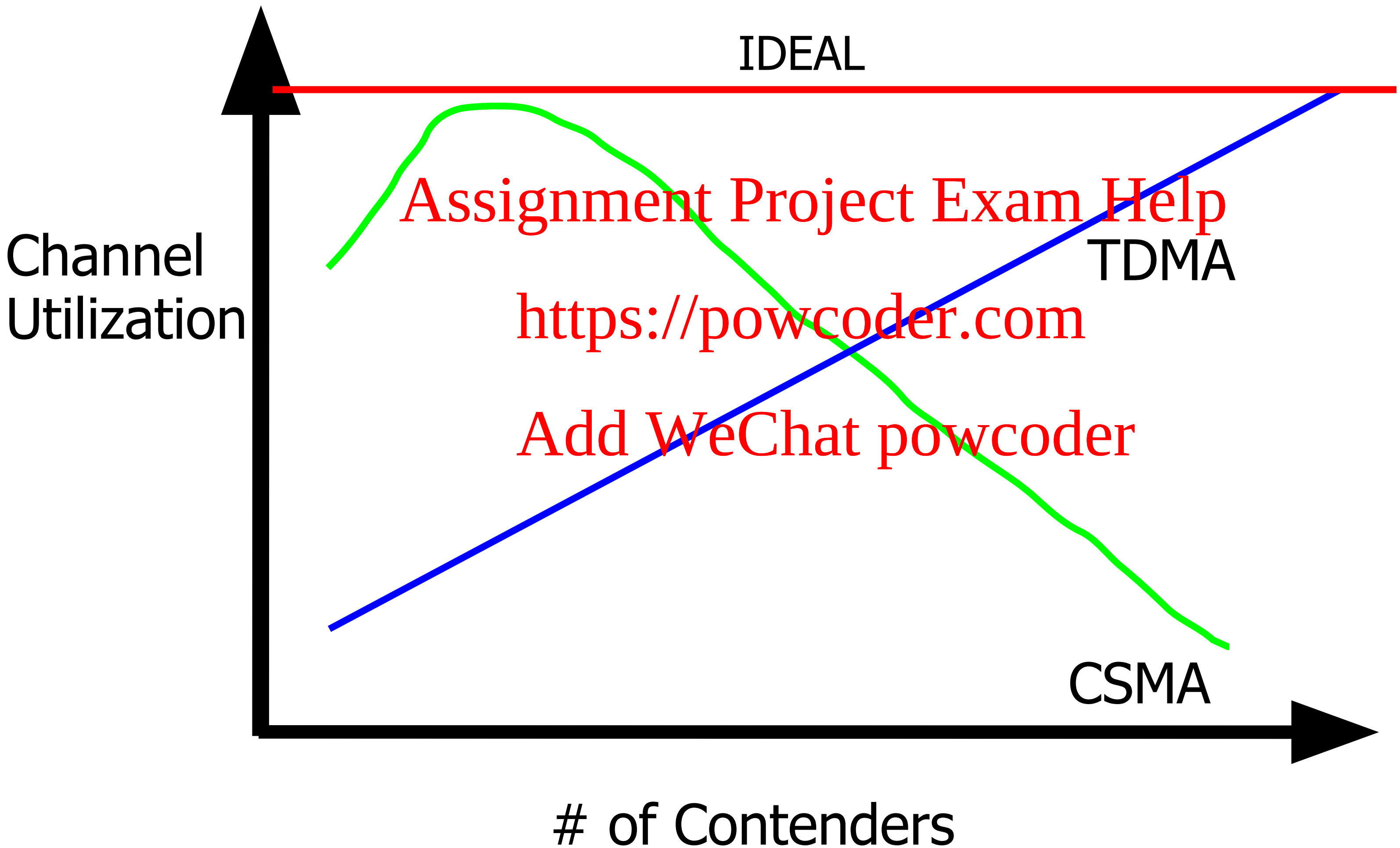
Rhee, A. Warrier,M. Aia, J. Min, ACM SenSys 2005, Nov 2005.

MAC	Channel Utilization	
	Low Contention	High Contention
CSMA	High	Low
TDMA	Low	High

- Combines the strengths of both CSMA and TDMA at the same time offsetting their weaknesses
- High channel efficiency and fair

Effective Throughput

CSMA vs. TDMA



Z-MAC

- Uses the TDMA schedule as a 'hint' to schedule transmissions
- The owner of a time-slot has always priority over the non-owners while accessing the medium
- Unlike TDMA, non-owners can 'steal' the time-slot when the owners do not have data to send

Z-MAC

- This enables Z-MAC to switch between CSMA and TDMA depending on the level of contention

- Hence, under low contention
  - Z-MAC acts like CSMA
  - High channel utilization and low latency
- Under high contention,
  - Z-MAC acts like TDMA
  - **High channel utilization, fairness and low contention overhead**

Schedule TDMA-like with DRAND

- Z-MAC requires a conflict-free transmission schedule or a TDMA schedule
- Uses DRAND, a distributed TDMA scheduling scheme
- DRAND is distributed, and is a distributed implementation of RAND, a famous centralized channel scheduling scheme
- Let G = (V, E) be an input graph, where V is the set of nodes and E the set of edges.
- An edge e = (u, v) exists if and only if u and v are within interference range
- Given G, DRAND calculates a TDMA schedule in time **linear** to the maximum node degree in G

Rhee, I., Warrier, A., Min, J. and Xu, L., 2009. **DRAND: Distributed randomized TDMA scheduling for wireless ad hoc networks**, *IEEE Transactions on Mobile Computing*, *8*(10), pp.1384-1396, 2009.

Transmission Control

- Slot Ownership
  - If current timeslot is the node's assigned time-slot, then it is the Owner, and all other neighboring nodes are Non-Owners
- If Low Contention Level (LCL) is detected:
  - Nodes compete in all slots, albeit with different priorities
- Before transmitting:
  - If I am the Owner:
    take backoff = Random(T_o)
  - Else if I am the Non-Owner:
    take backoff = T_o + Random(T_{no})
- After backoff, sense channel, if busy repeat above, else send

Transmission Control

- Switches between CSMA and TDMA automatically depending on contention level

- Performance depends on specific values of T_o and T_{no}

- Usually, $T_o = 8$ and $T_{no} = 32$ are used

Explicit Contention Notification (ECN)

- With ECN, a node informs all nodes within two-hop neighborhood not to send during its time-slot

- When a node receives ECN message, it sets its High Contention Level (HCL) flag

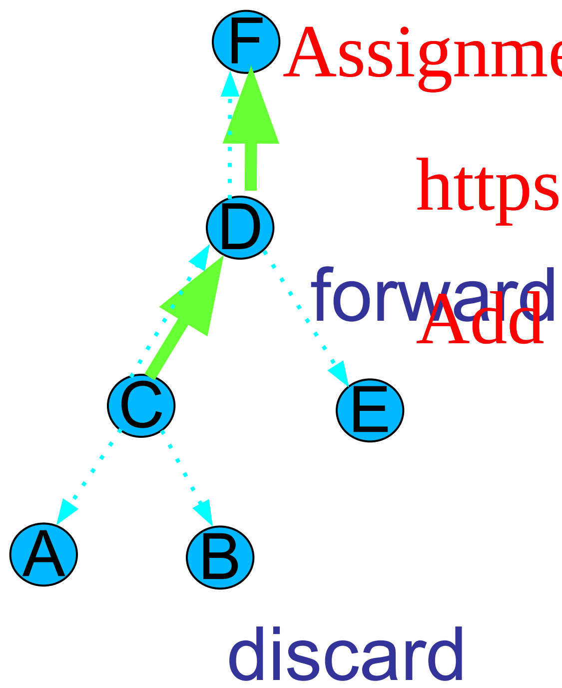
- **BTW, HOW DO WE DETECT HIGH CONGESTION?**

- High contention is detected by lost ACKs or repeated backoffs

- On receiving one-hop ECN from a node i, forward two-hop ECN if it is on the routing path from node i

Explicit Contention Notification - Example

Thick Line – Routing Path
Dotted Line – ECN Messages



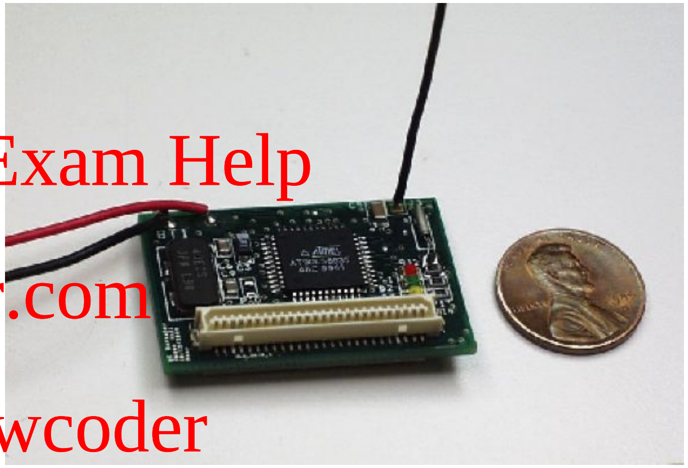
- C experiences high contention
- C broadcasts one-hop ECN message to A, B, D
- A, B not on routing path (C->D->F), so discard ECN
- D is on routing path, so it forwards ECN as two-hop ECN message to E, F
- D and F will **not** compete during C's slot as Non-Owners
- A, B and E are eligible to compete during C's slot, albeit with lesser priority as **Non-Owners**

Performance Evaluation

DRAND and ZMAC have been implemented on both NS2 and on Mica2 motes

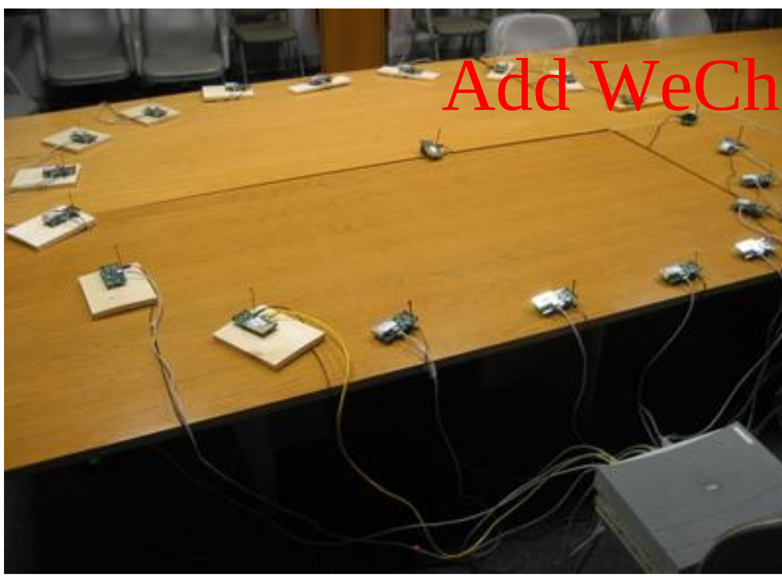
Performance Results

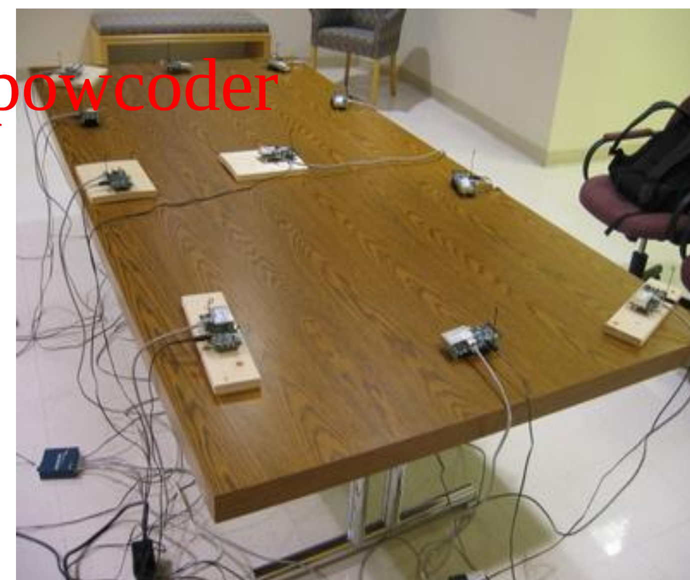
- Platform:
  - Mica2
  - 8-bit CPU at 4MHz
  - 8KB flash, 256KB RAM
  - 916 MHz radio (ISM)
  - TinyOS event-driven



Experimental Setup – Single Hop

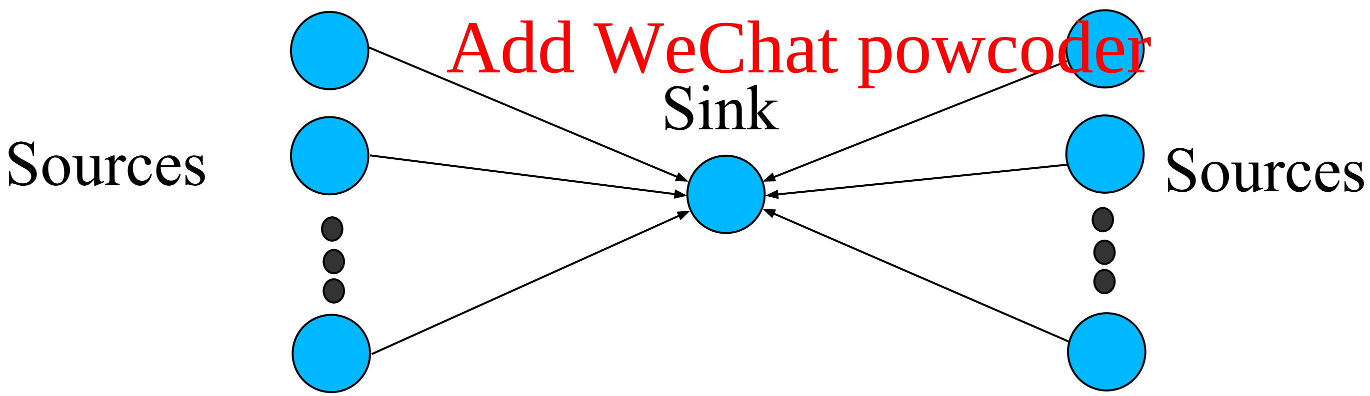
- **Single-Hop Experiments:**
  - Star network configuration
  - Tests repeated 10 times and average/standard deviation errors reported (confidence intervals)



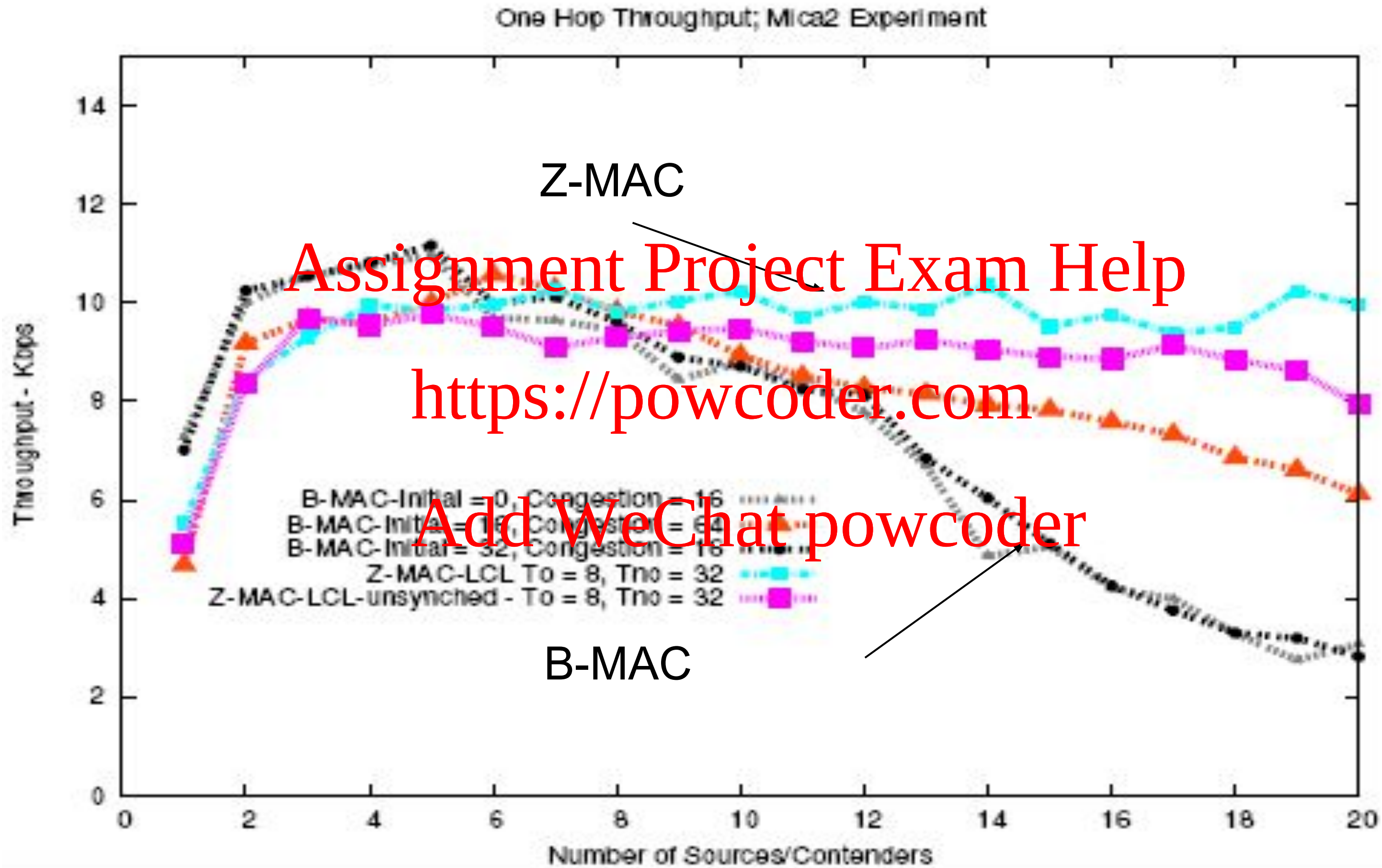


Z-MAC – Two Hop Experiments

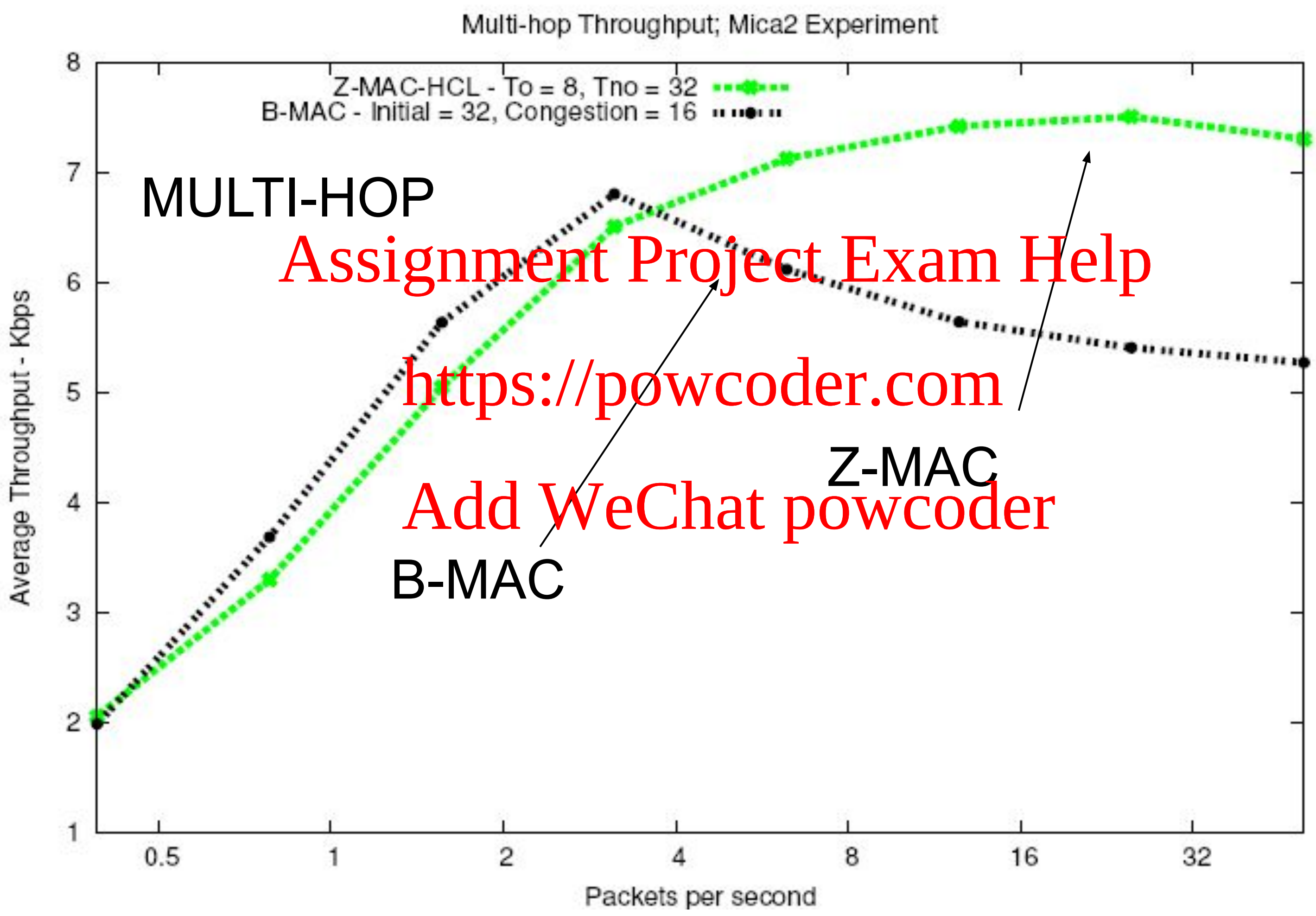
- Setup – Two-Hop
  - Dumbbell shaped topology
  - Transmission power varied between low (50) and high (150) to get two-hop situations
  - Aim – See how Z-MAC works when Hidden Terminal Problem manifests itself



Single-Hop Throughput

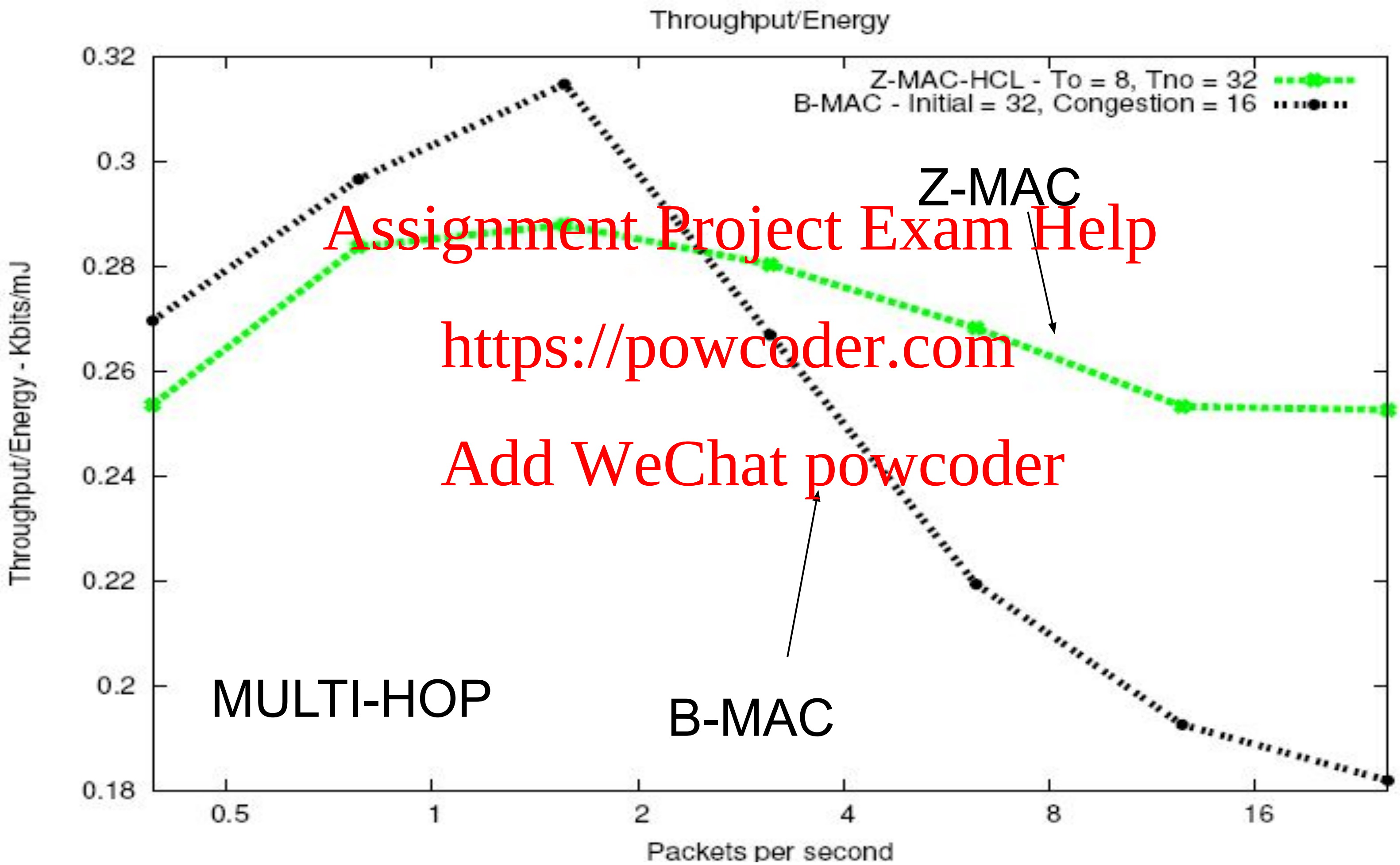


Multi Hop Results – Throughput



Multi Hop Results – Energy Efficiency

(KBits/Joule)



What are the pros and cons of ZMAC?

Think-Pair-Share!

Overhead (Hidden Costs)

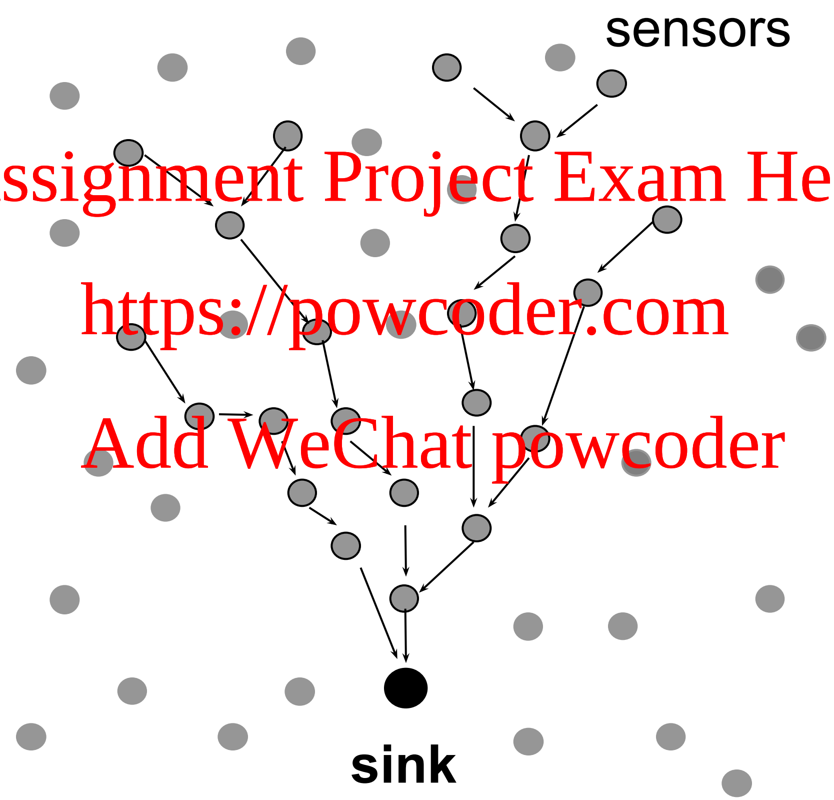
Operation	Average (J)	StdDev
Neighbor Discovery	0.73	0.0018
DRAND	4.88	3.105
Local Frame Exchange	1.33	1.39
Time Synchronization	0.28	0.036

Total energy: 7.22 J – 0.03% of typical battery (2500mAh, 3V)

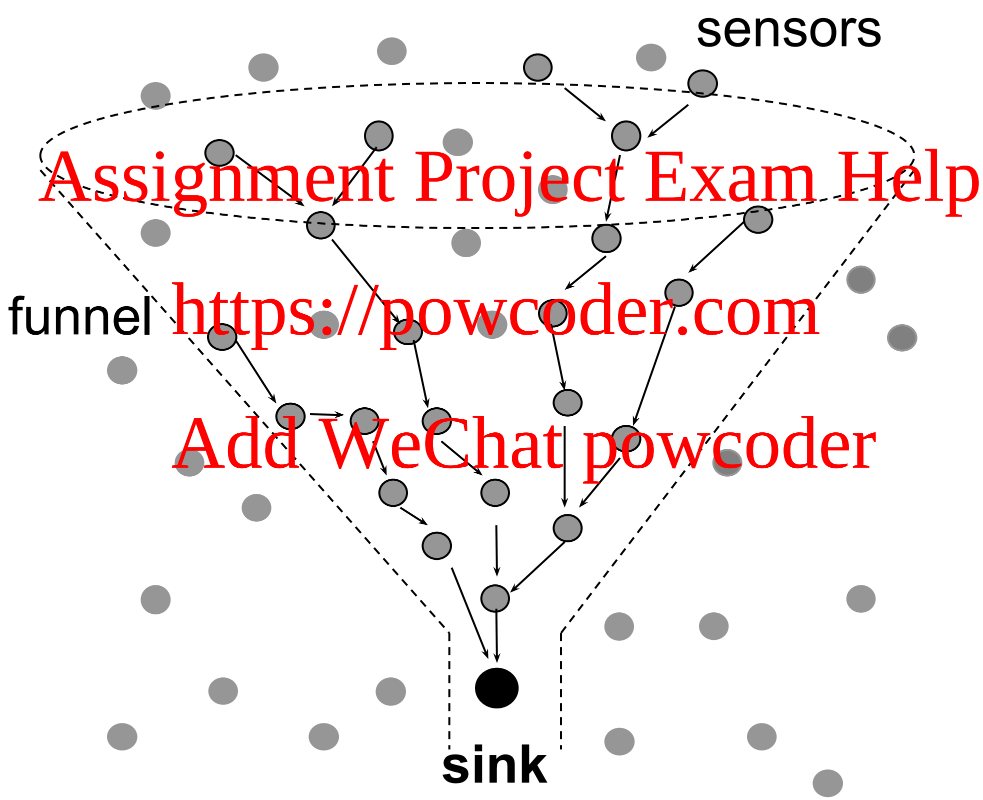
Hybrid Mac Protocols: Funneling-MAC

Gahng-Seop Ahn, Emiliano Miluzzo, Andrew T. Campbell, Se Gi Hong, and Francesca Cuomo, "Funneling-MAC: A Localized, Sink-Oriented MAC For Boosting Fidelity in Sensor Networks", In *Proc. of Fourth ACM Conference on Embedded Networked Sensor Systems (SenSys 2006)*, Boulder, Colorado, USA, Nov 1-3, 2006

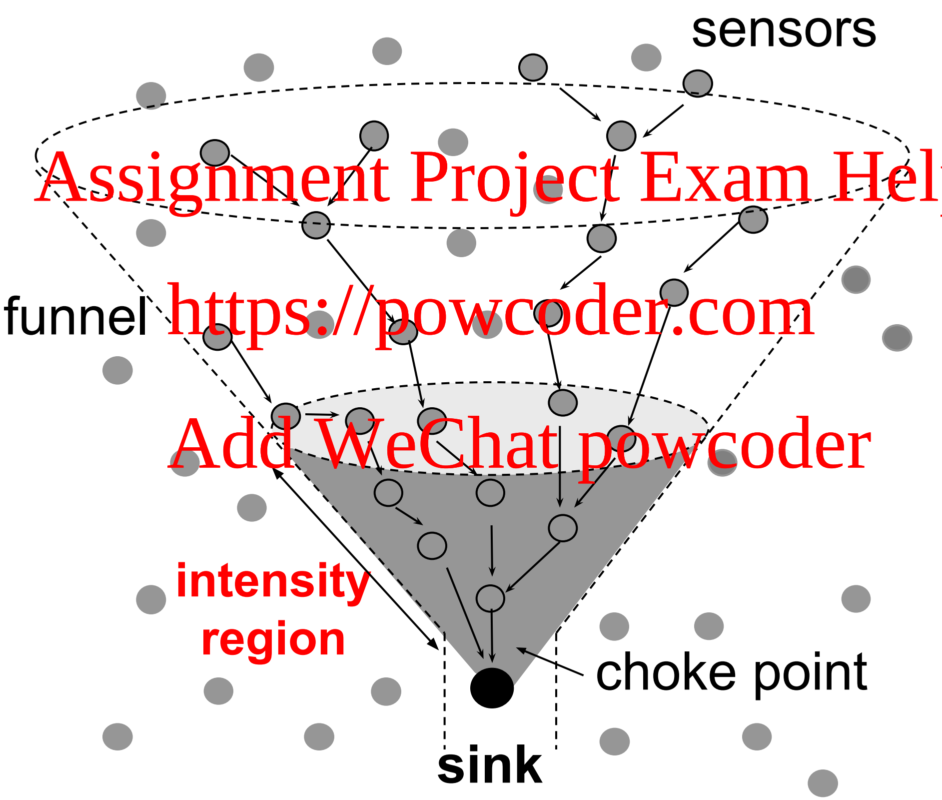
The Funneling Problem



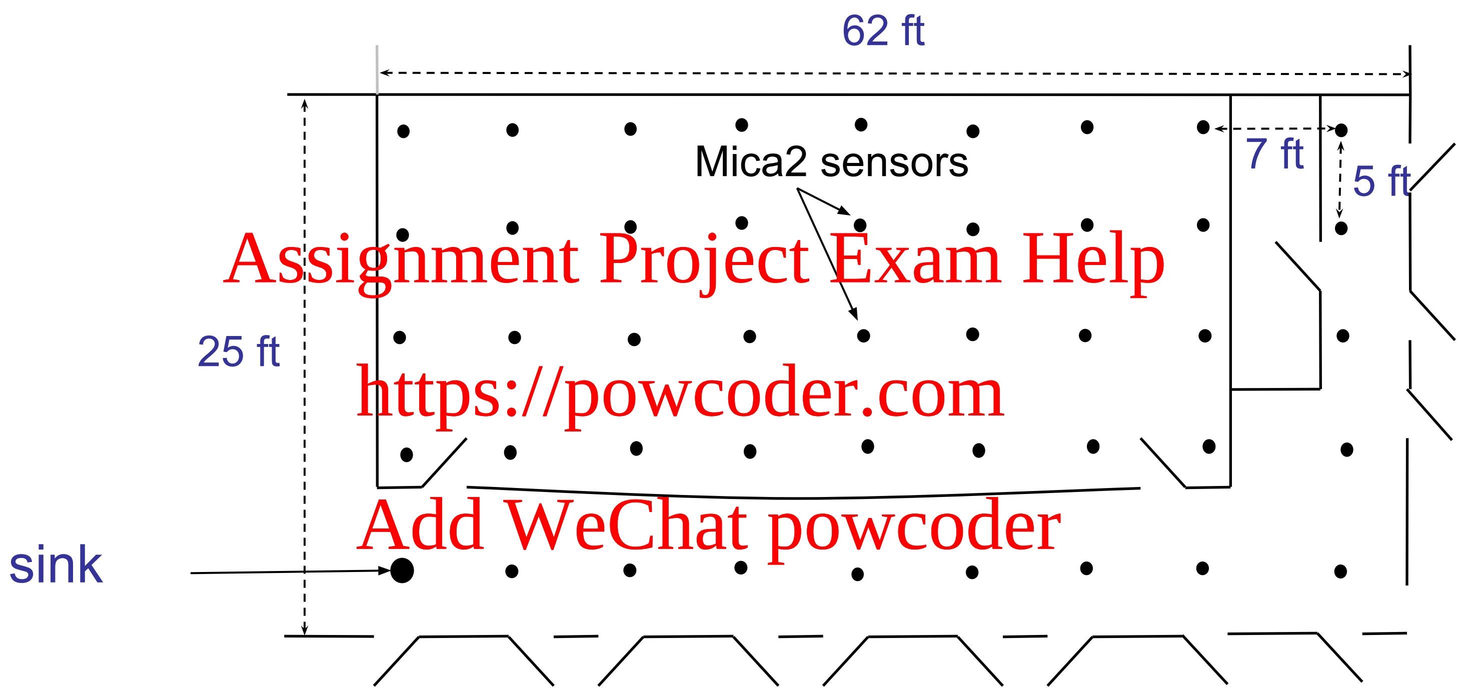
The Funneling Problem



The Funneling Problem

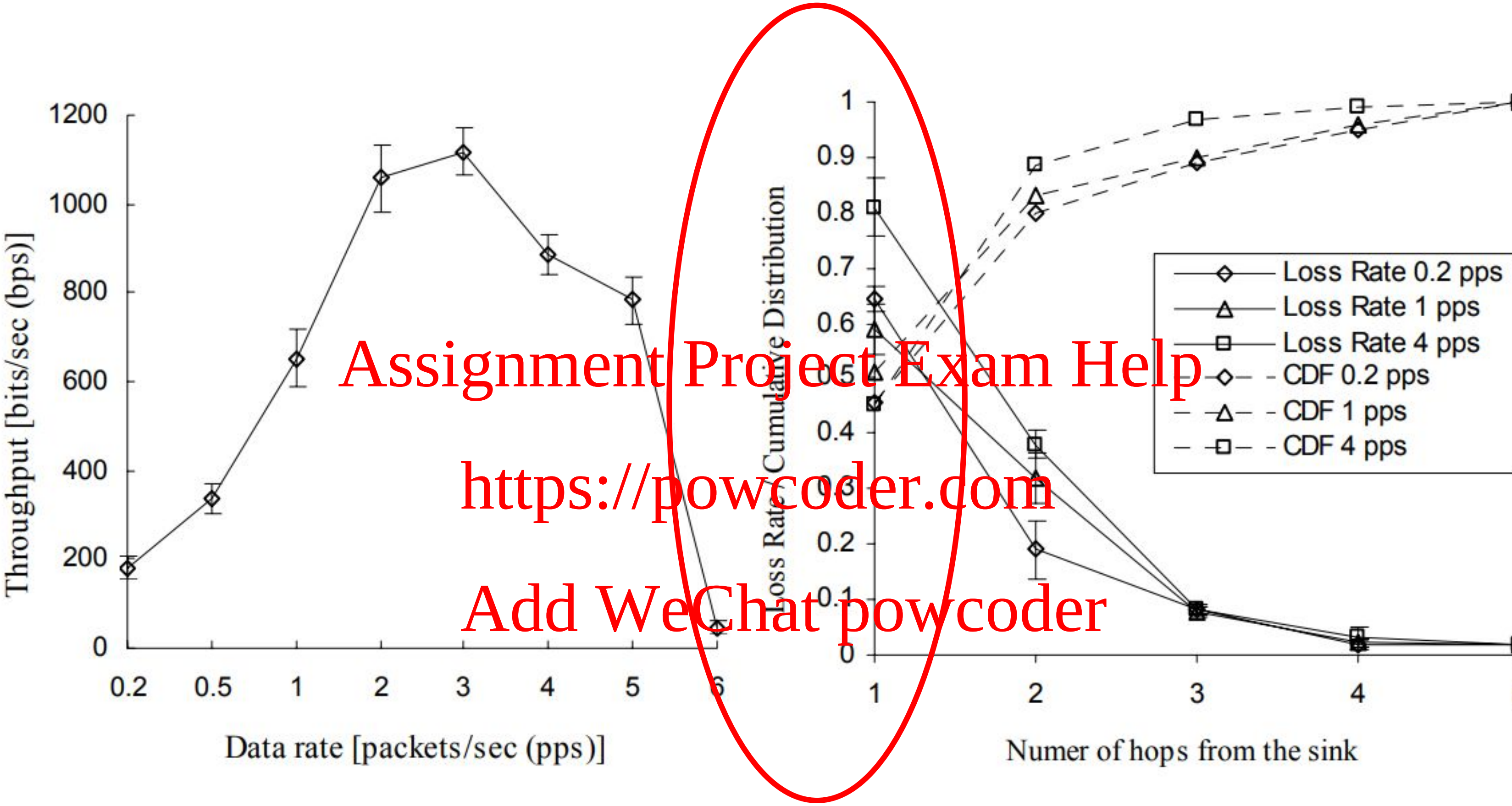


Quantifying the Funneling Effect



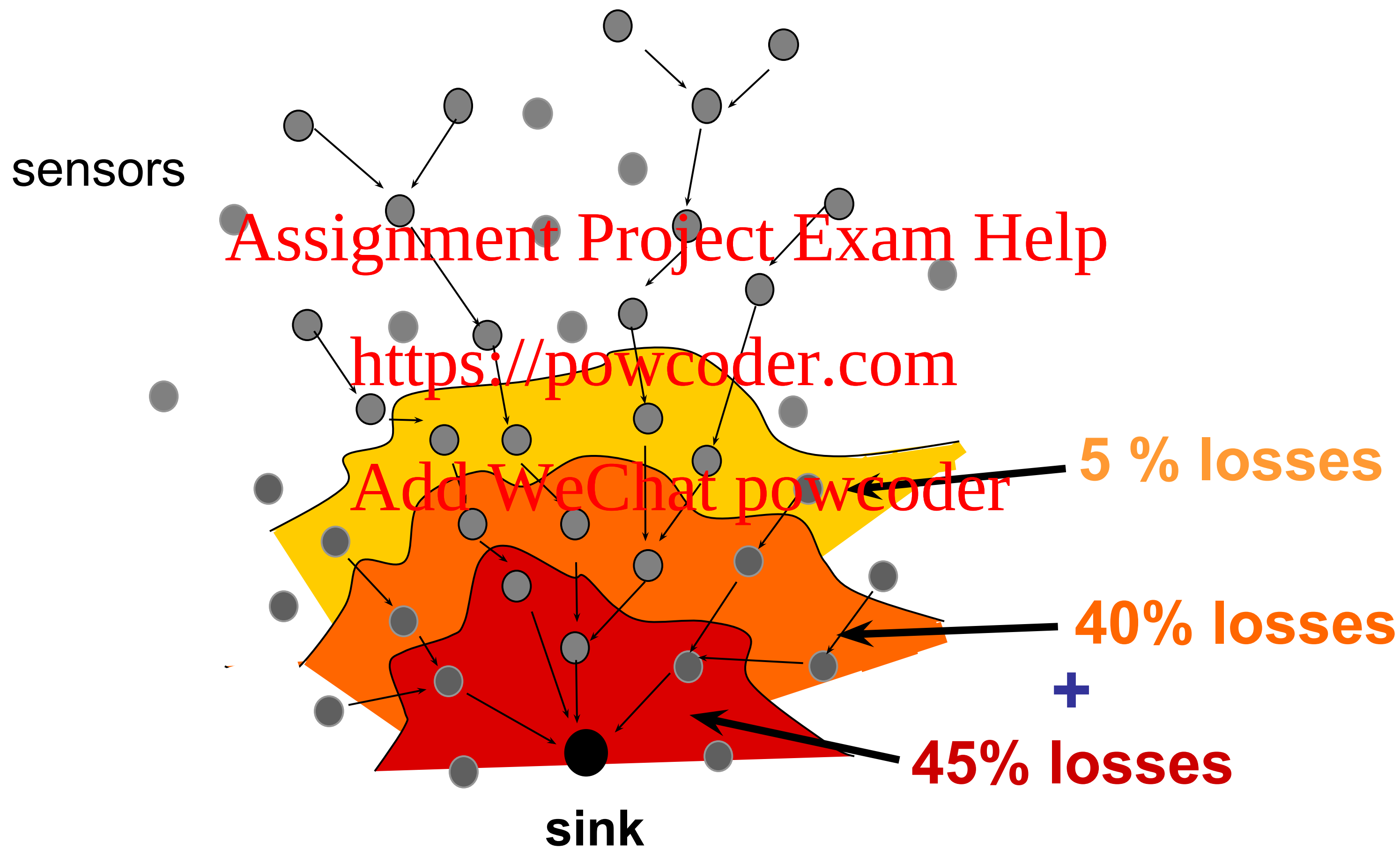
- 45 Mica2 in a 9x5 grid topology
- Grid calibration: 1 hop → > 80% of total nodes, 2-hop → < 20%
- TinyOS 1.1.15

Funneling Effect Impact

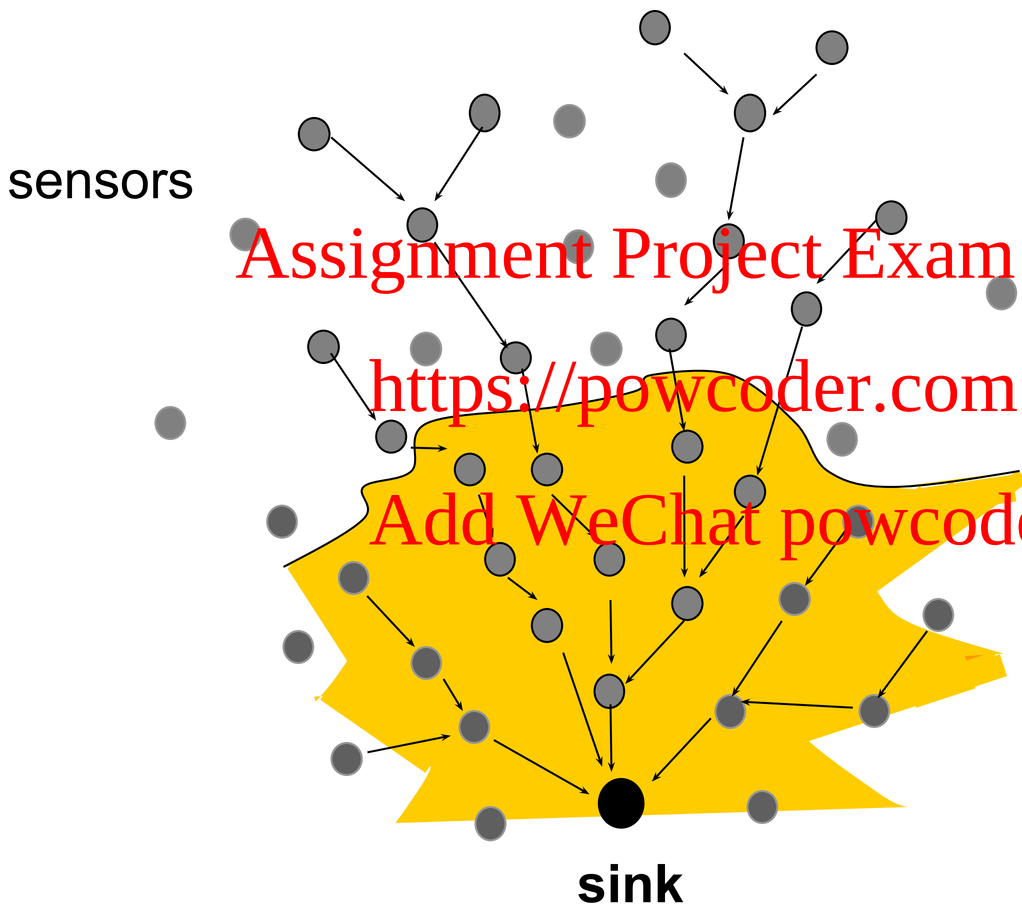


- At the sink overall loss rate: between 67% and 95%

Is there a simple solution to this problem?



Is there a simple solution to this problem?



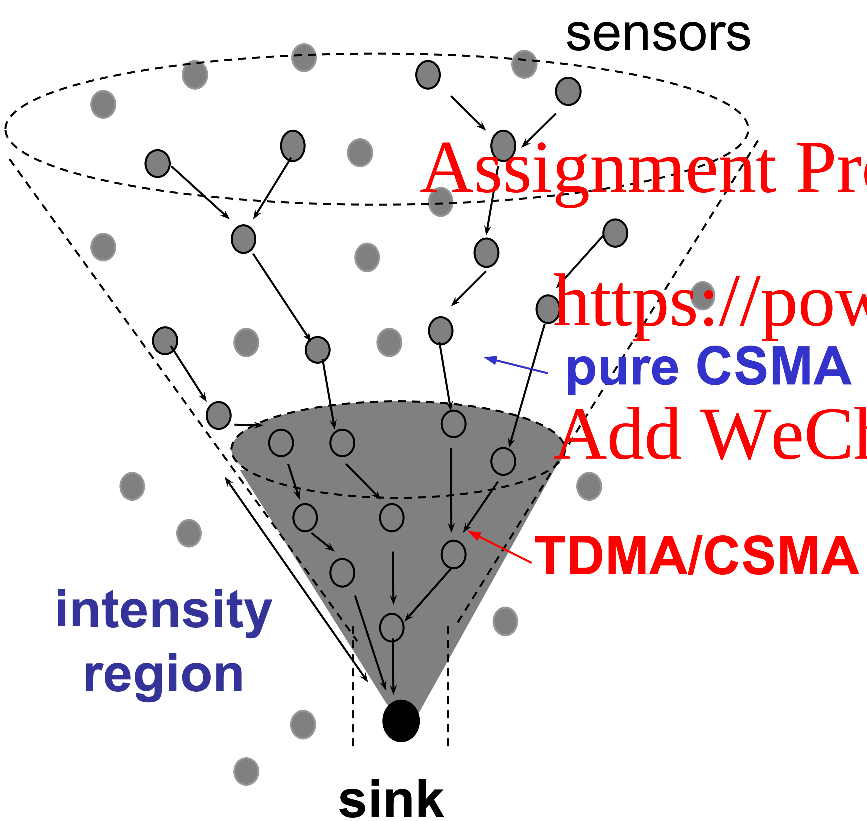
low losses within the first 2 hops

Answer

- Yes, it is possible and the Funneling-MAC is built to
  - Exploit localized control over the intensity region
  - Reacting dynamically to network conditions

- Such that it addresses scalability while proposing an efficient scheduling protocol

Funneling-MAC Design Considerations



- Hybrid **TDMA/CSMA** scheme **inside** the intensity region
- Pure **CSMA** scheme **outside** the intensity region
- Uses **Sink-oriented** TDMA scheduling
- Maintenance of the intensity region **dynamically** operated by the sink

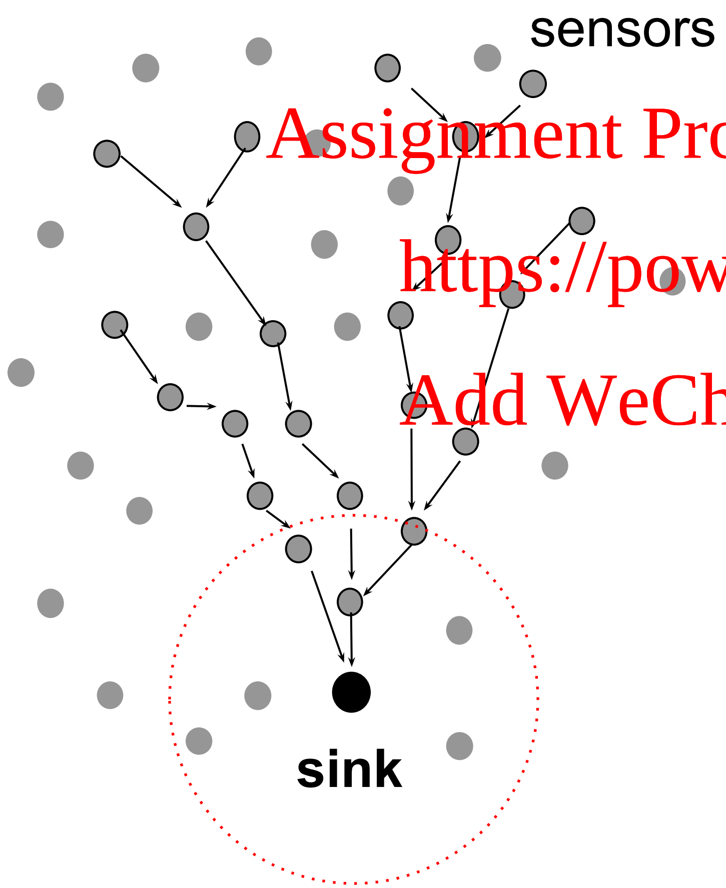
Funneling-MAC algorithm

- On-demand beaconing
- Dynamic-depth tuning
- Sink-oriented scheduling

On-demand Beaconing

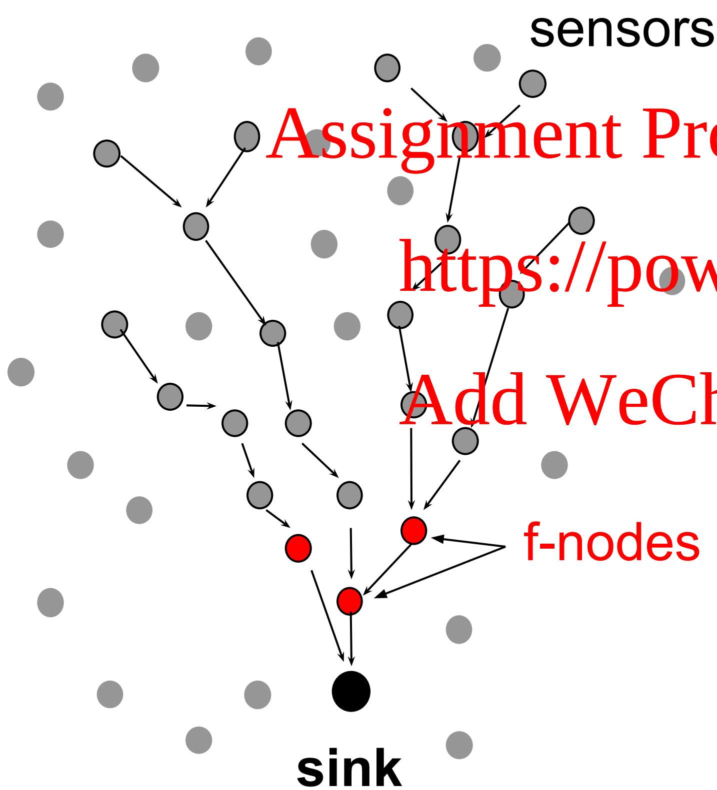
- To dynamically drive the depth of the intensity region
- To synchronize the nodes inside the intensity region

On-demand Beaconing



- The sink periodically broadcasts a *Beacon*
- At the bootstrap of the network or when starting with low traffic the Beacon transmission power is the same as the sensors

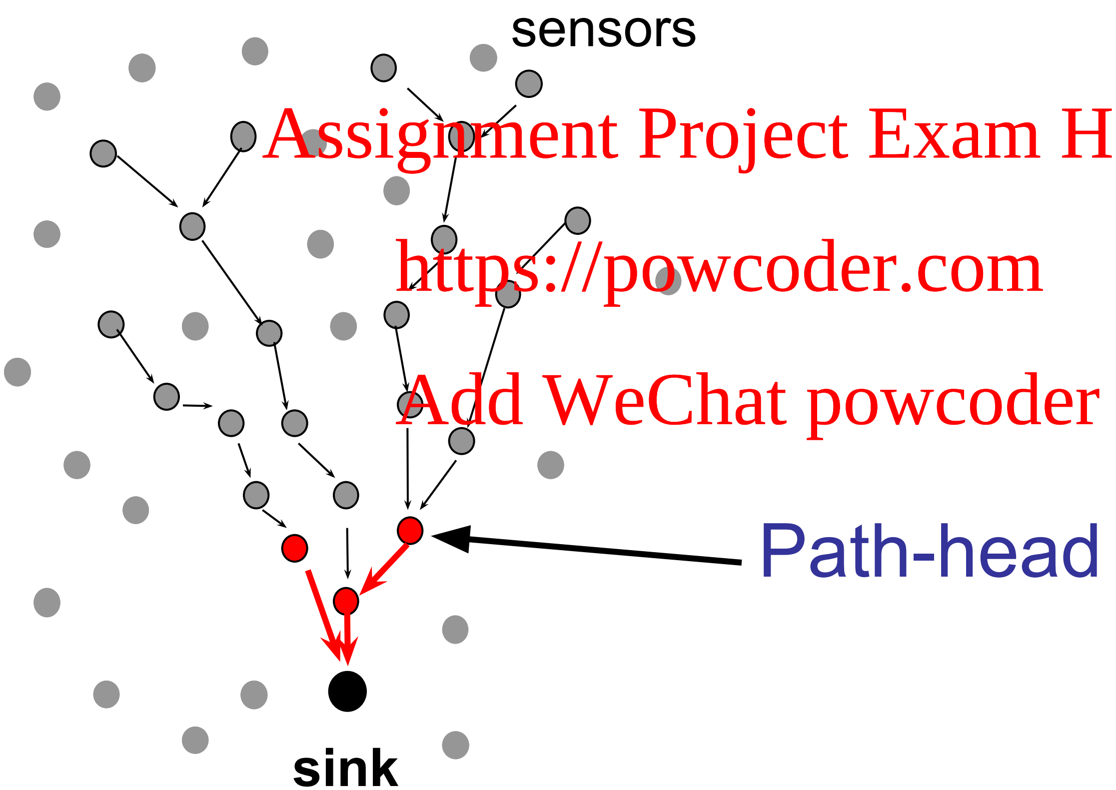
On-demand Beaconing



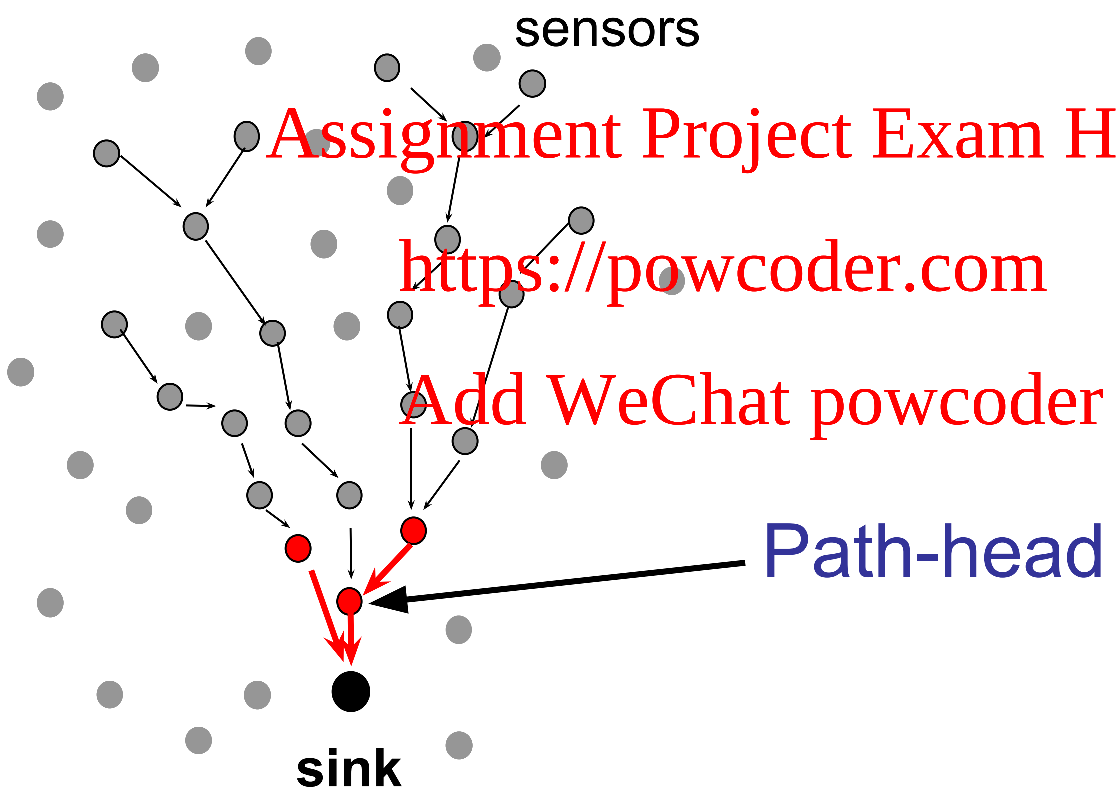
- Sensors receiving a *Beacon* become f-nodes and consider themselves inside the intensity region

- Upon receiving a beacon f-nodes synchronize with each other by initializing their clock

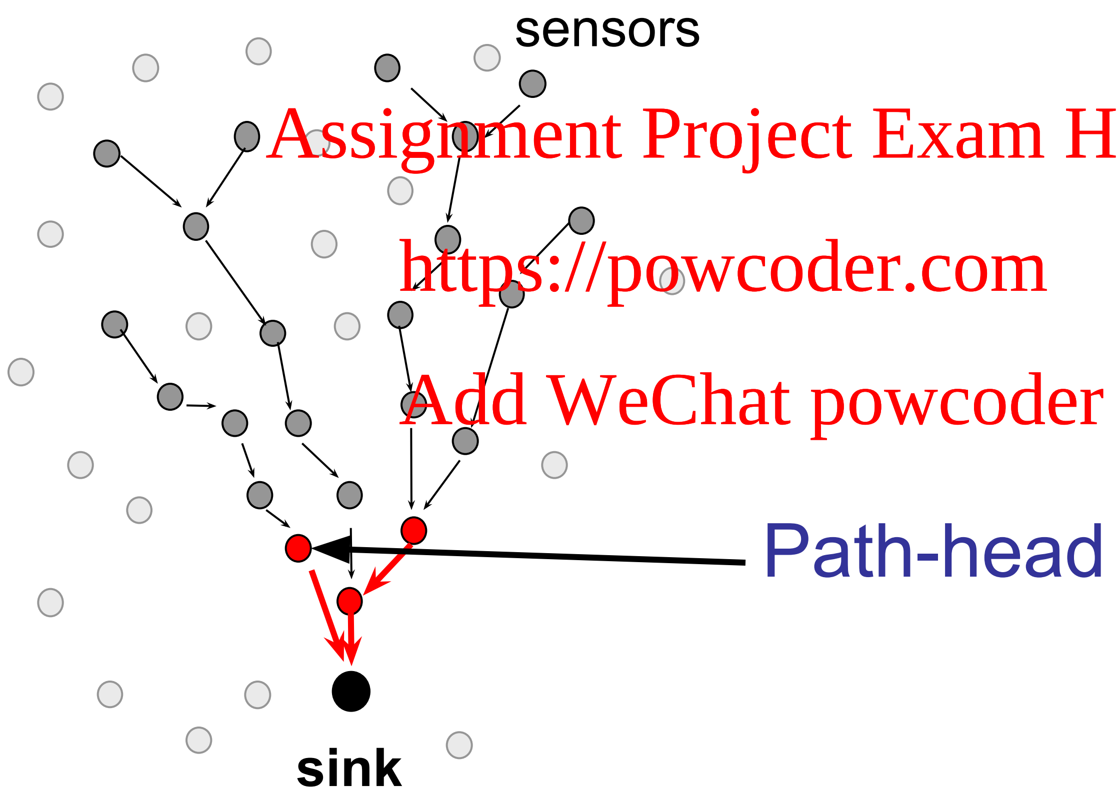
On-demand Beaconing



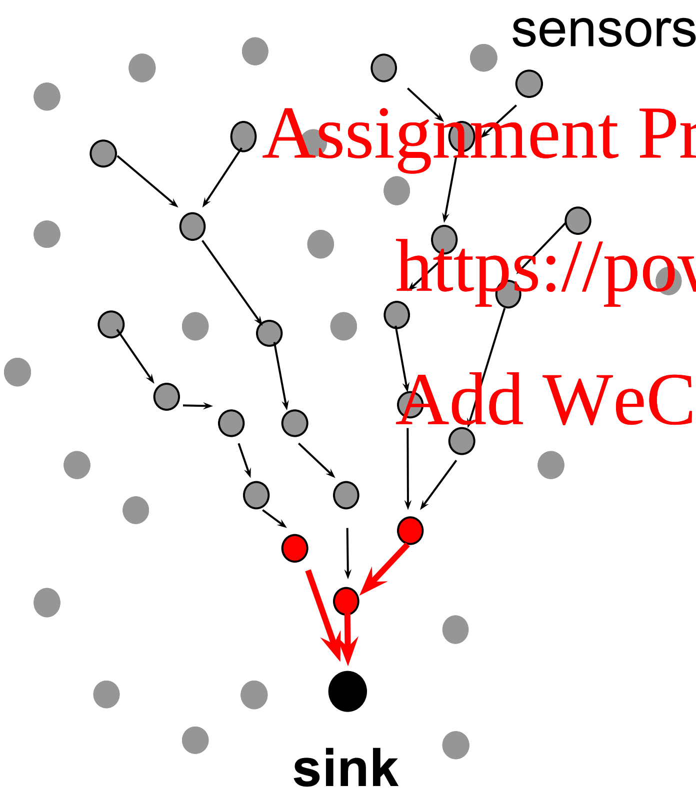
On-demand Beaconing



On-demand Beaconing

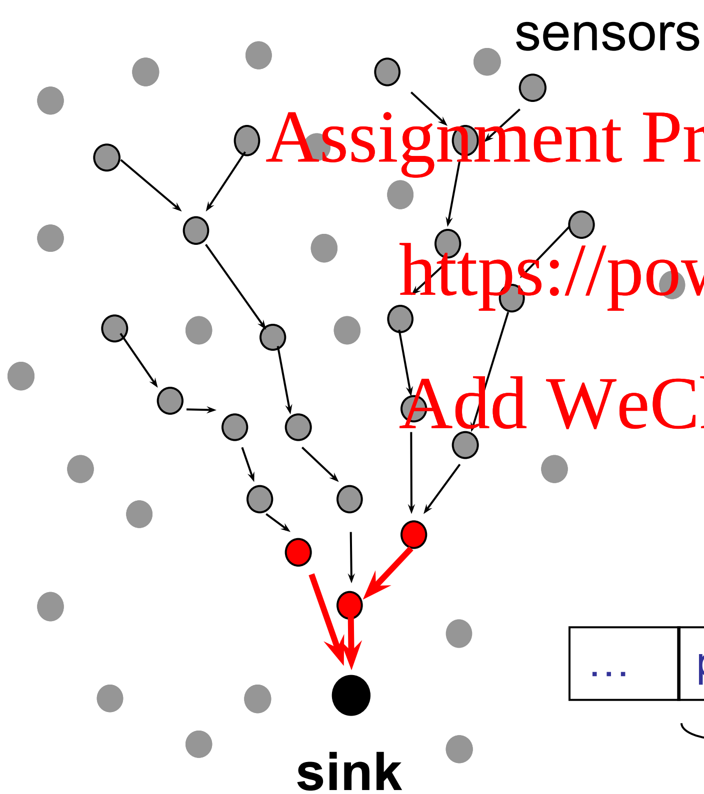


On-demand Beaconing



Path-heads operate a *passive registration* by which the sink knows the number of path-heads and how many hops they are far away from the sink for scheduling purposes

On-demand Beaconing



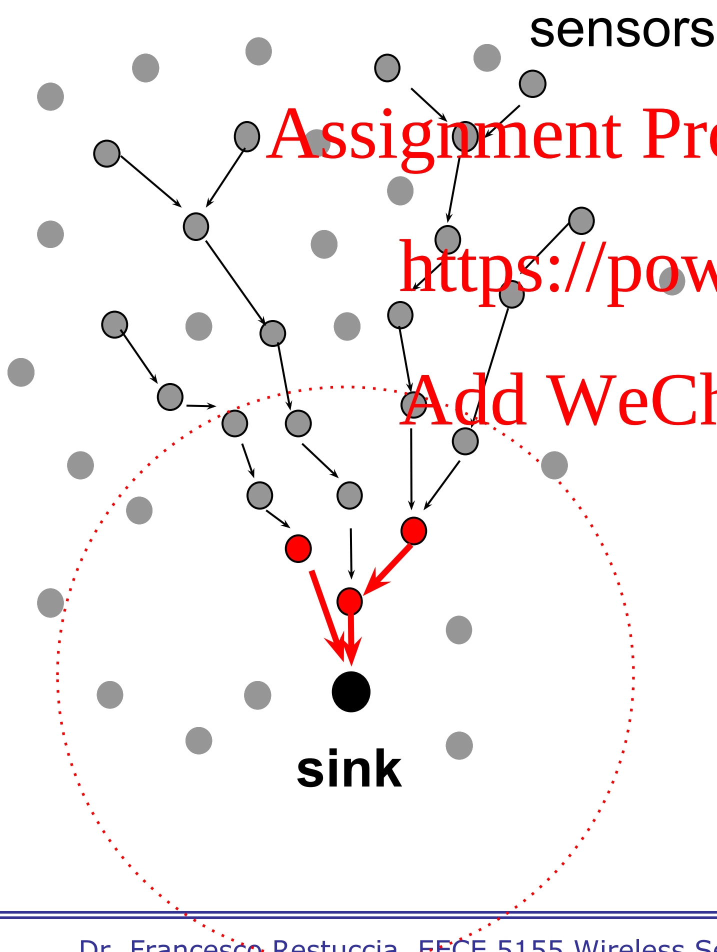
Path-heads operate a *passive registration* by which the sink knows the number of path-heads and how many hops they are far away from the sink for scheduling purposes

…	path-head	No of hops	…

3 bytes

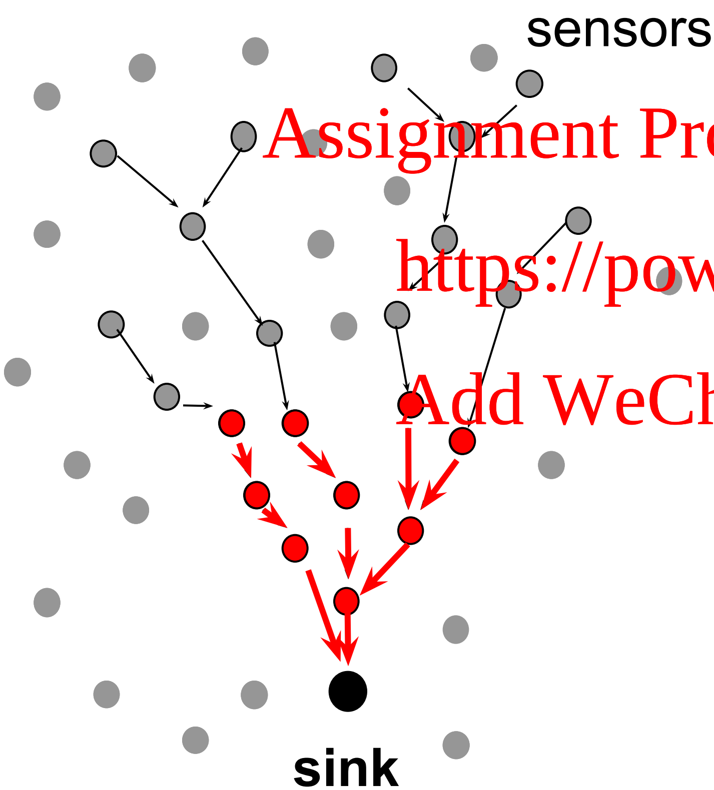
Data packet

On-demand Beaconing



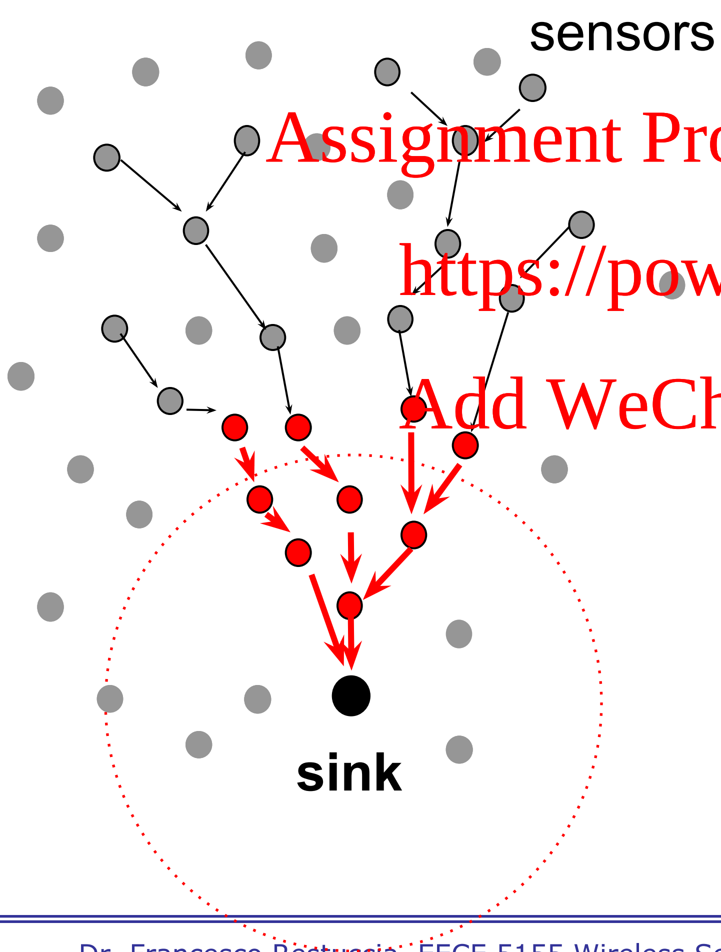
If the sink realizes that it can schedule more nodes, it increases the transmission power of the Beacon to expand the intensity region

On-demand Beaconing



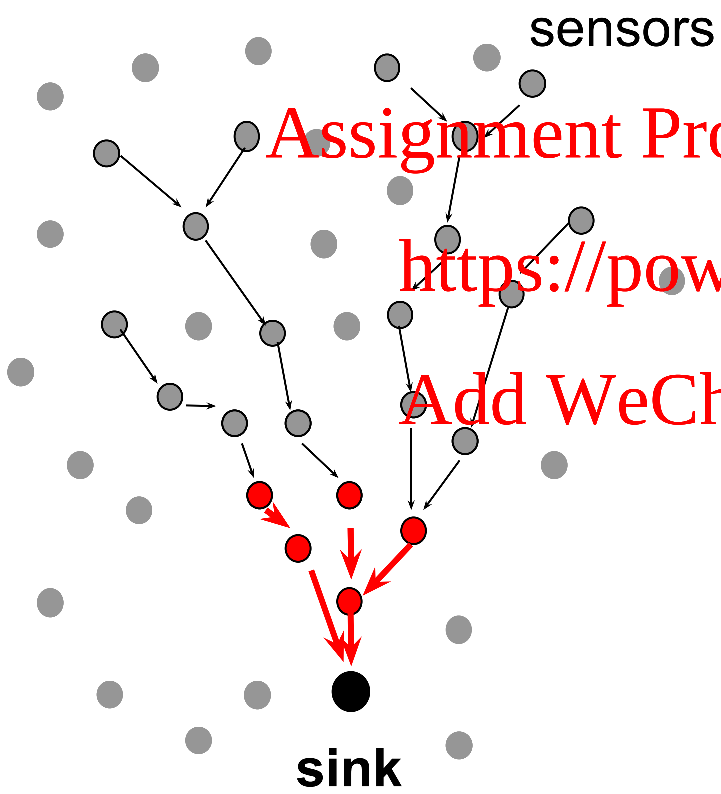
If the sink realizes that it can schedule more nodes, it increases the transmission power of the Beacon to expand the intensity region

On-demand Beaconing



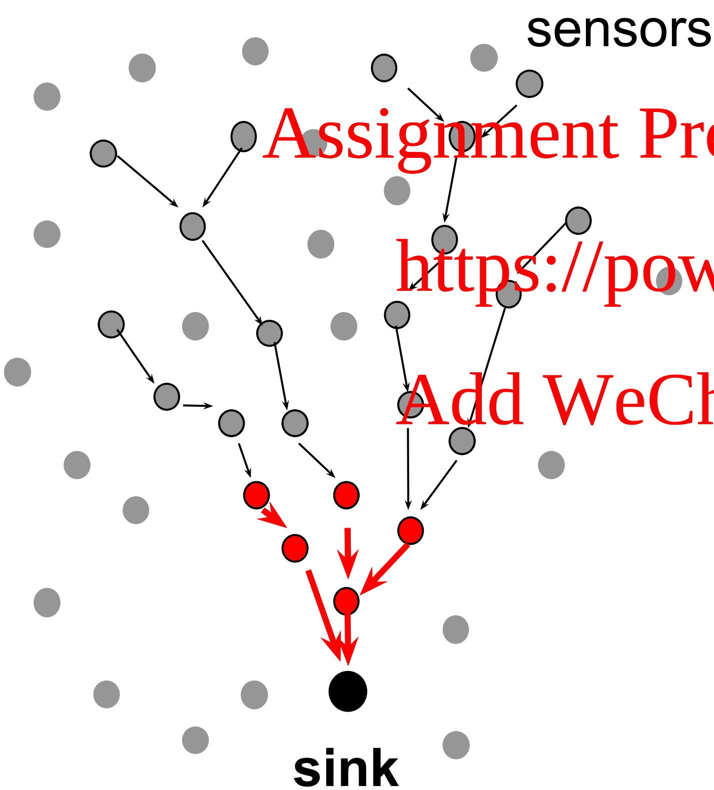
If the sink realizes that the number of f-nodes attempting the registration exceeds the maximum number of nodes that can be scheduled, then the sink reduces the beacon transmission power

On-demand Beaconing



If the sink realizes that the number of f-nodes attempting the registration exceeds the maximum number of nodes that can be scheduled, then the sink reduces the beacon transmission power

On-demand Beaconing



If the sink realizes that the number of f-nodes attempting the registration exceeds the maximum number of nodes that can be scheduled, then the sink reduces the beacon transmission power

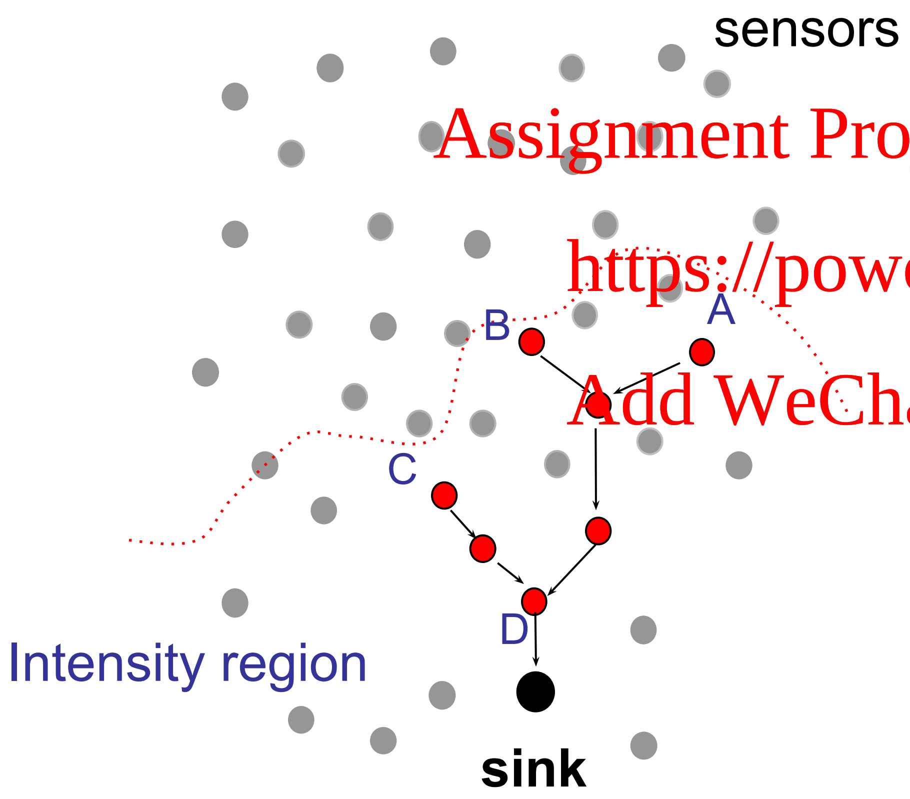
The beacon transmission power is determined by the ***Dynamic-depth tuning*** algorithm

Dynamic-depth Tuning - More formally

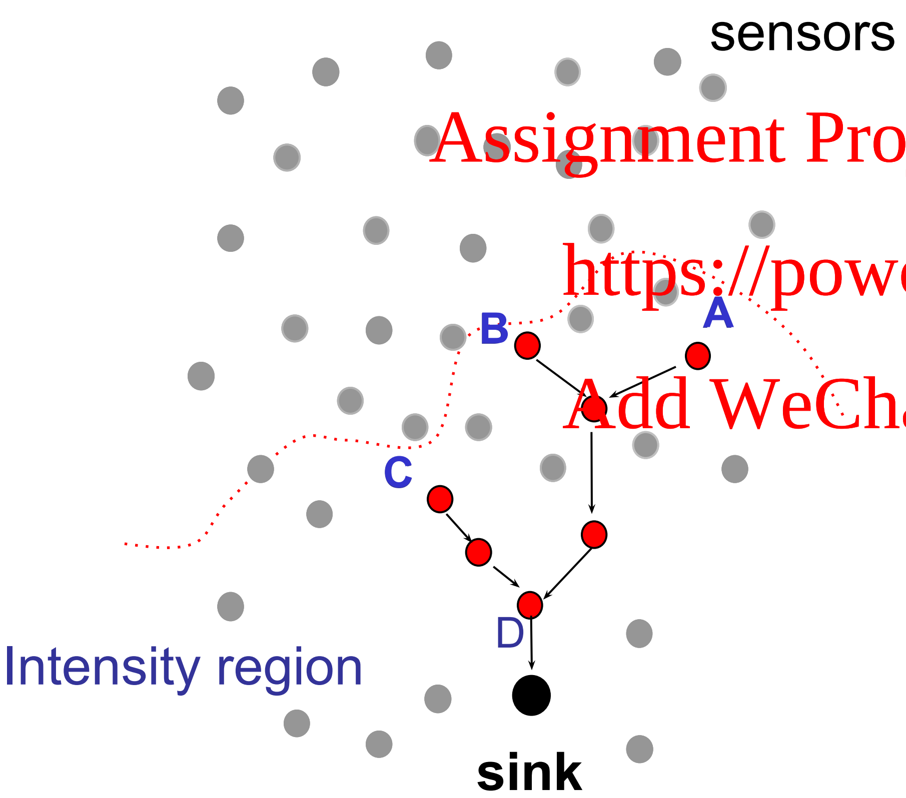
- $\mathbf{A}_{\mathbf{max}}$ → max number of slots that can be assigned given the TDMA capacity
- **A** → number of slots required to schedule path-heads' traffic

- if $\mathbf{A} \leq \mathbf{A}_{\mathbf{max}}$ → sink increases beacon transmission power
- if $\mathbf{A} > \mathbf{A}_{\mathbf{max}}$ → sink decreases beacon transmission power

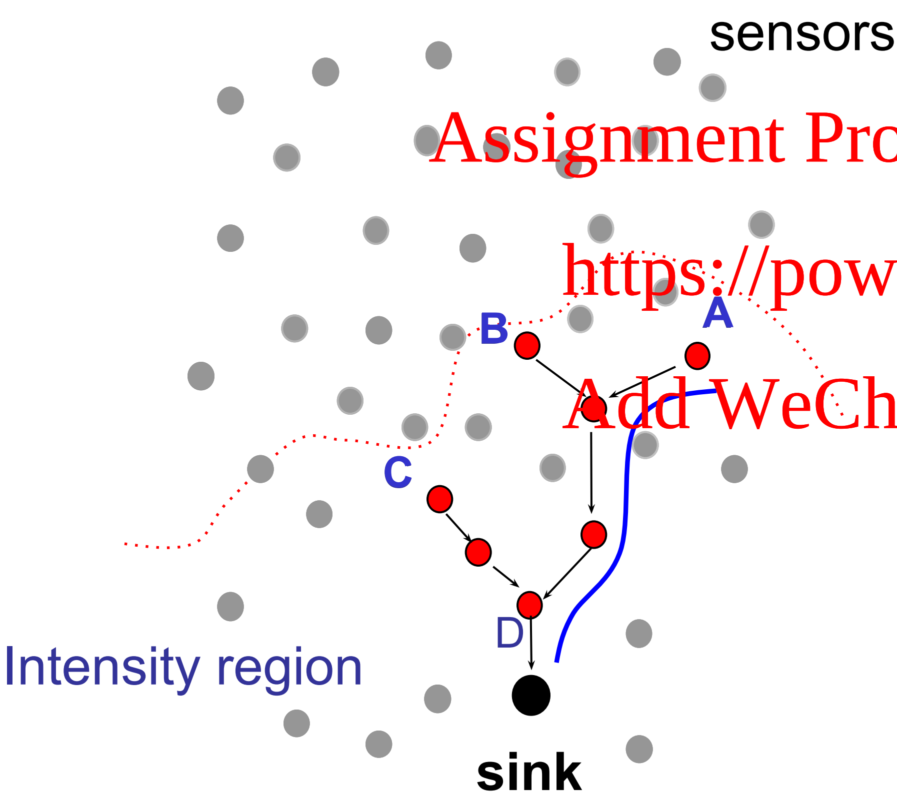
Sink-oriented Scheduling



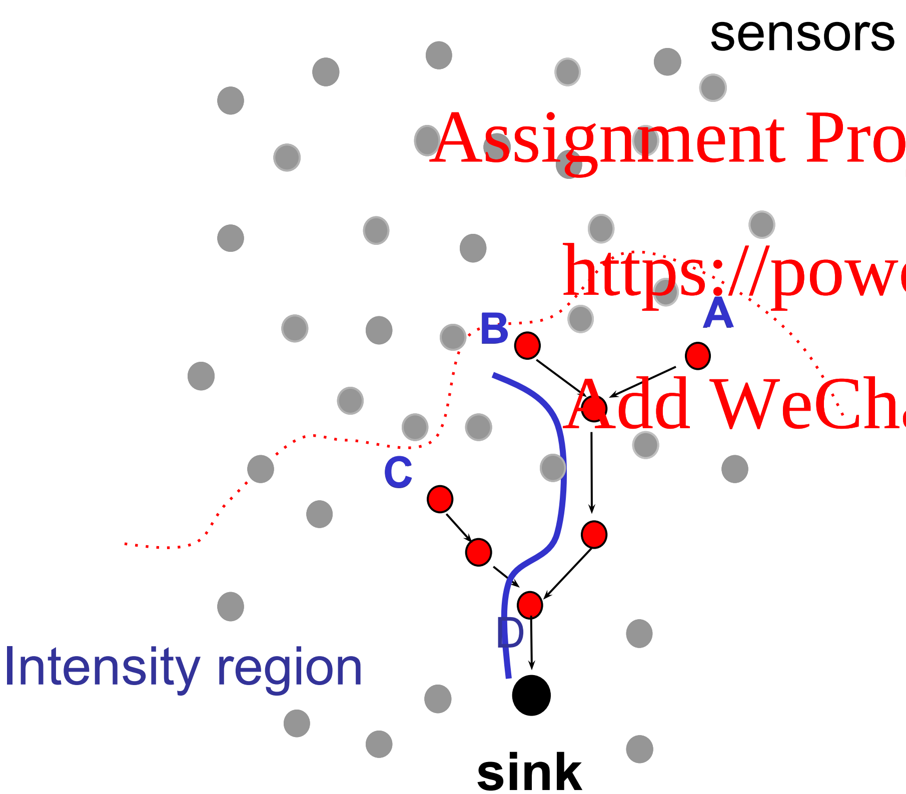
Sink-oriented Scheduling



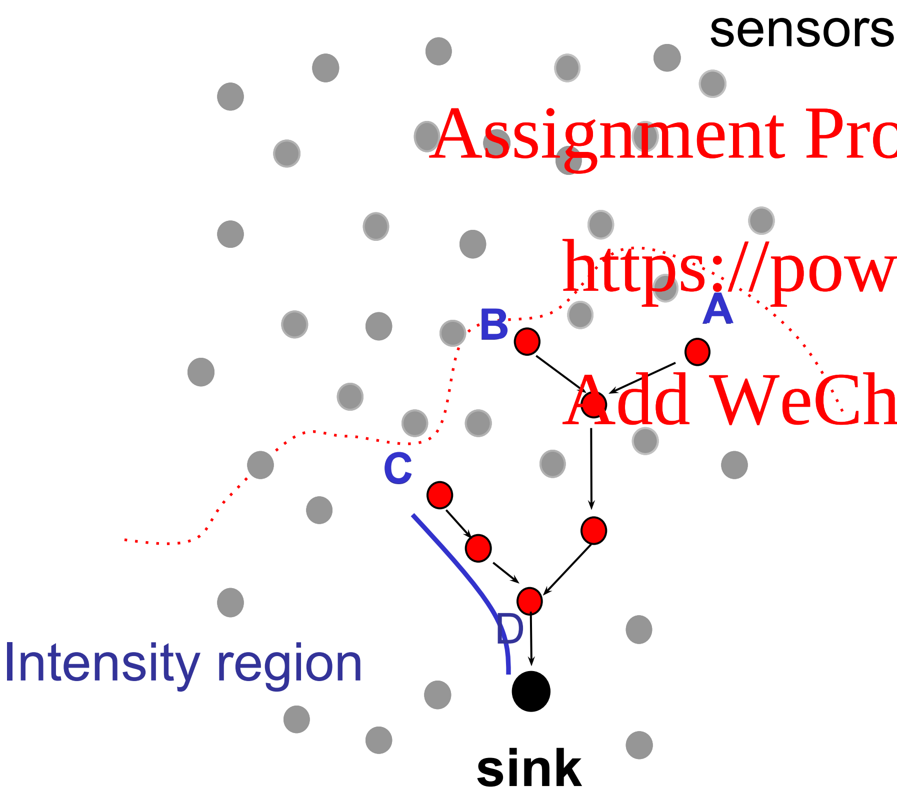
Sink-oriented Scheduling



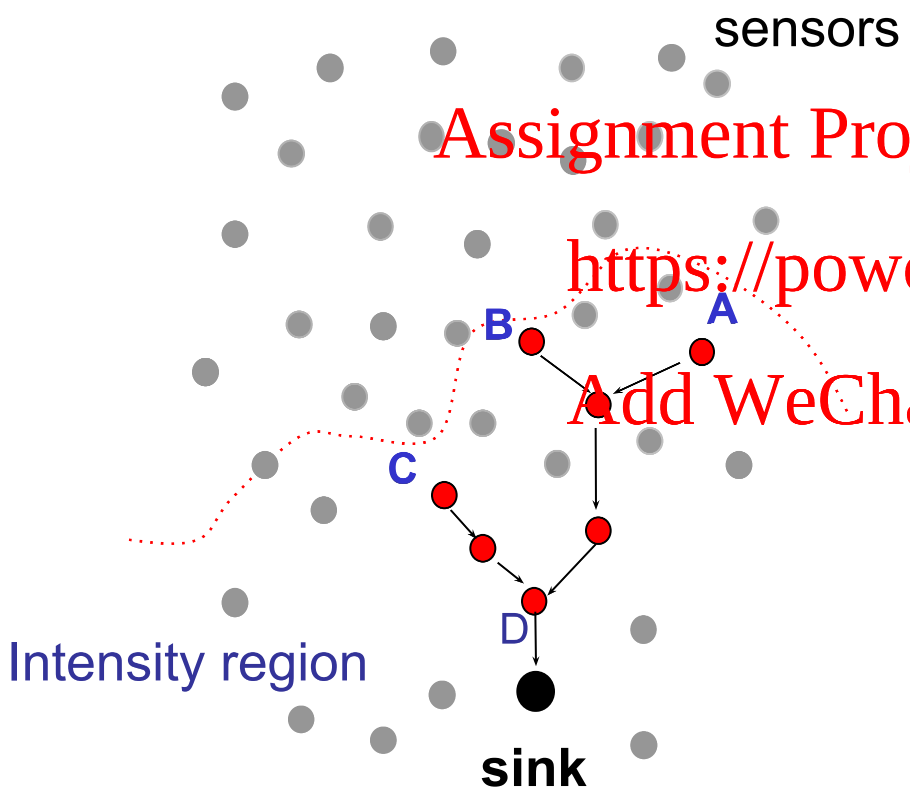
Sink-oriented Scheduling



Sink-oriented Scheduling



Sink-oriented Scheduling

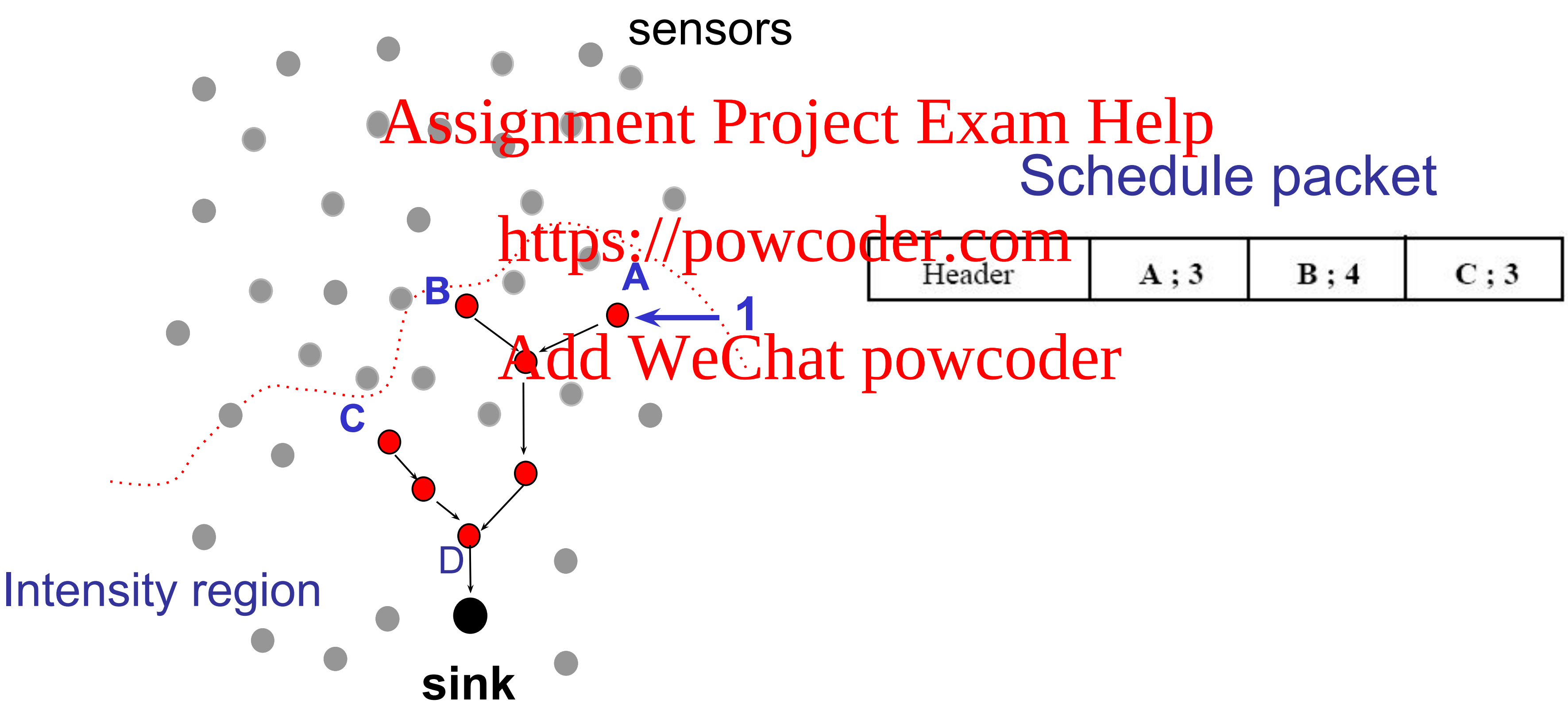


Schedule packet

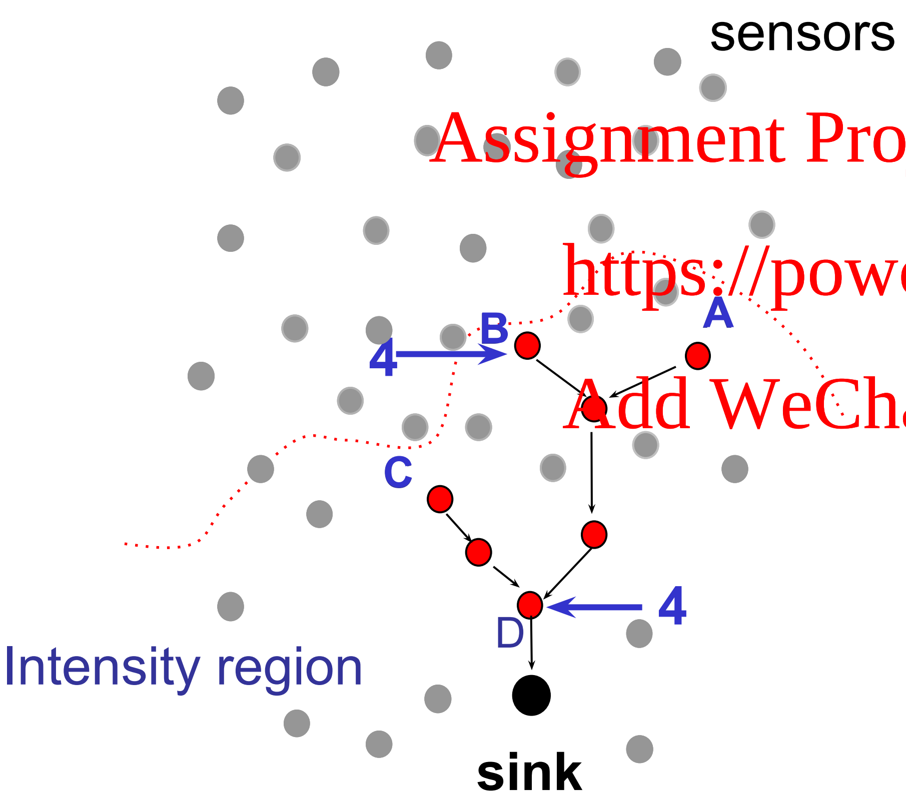
Header	A ; 3	B ; 4	C ; 3

- B starts 3 slots after A
- C starts 7 slots after A

Sink-oriented Scheduling



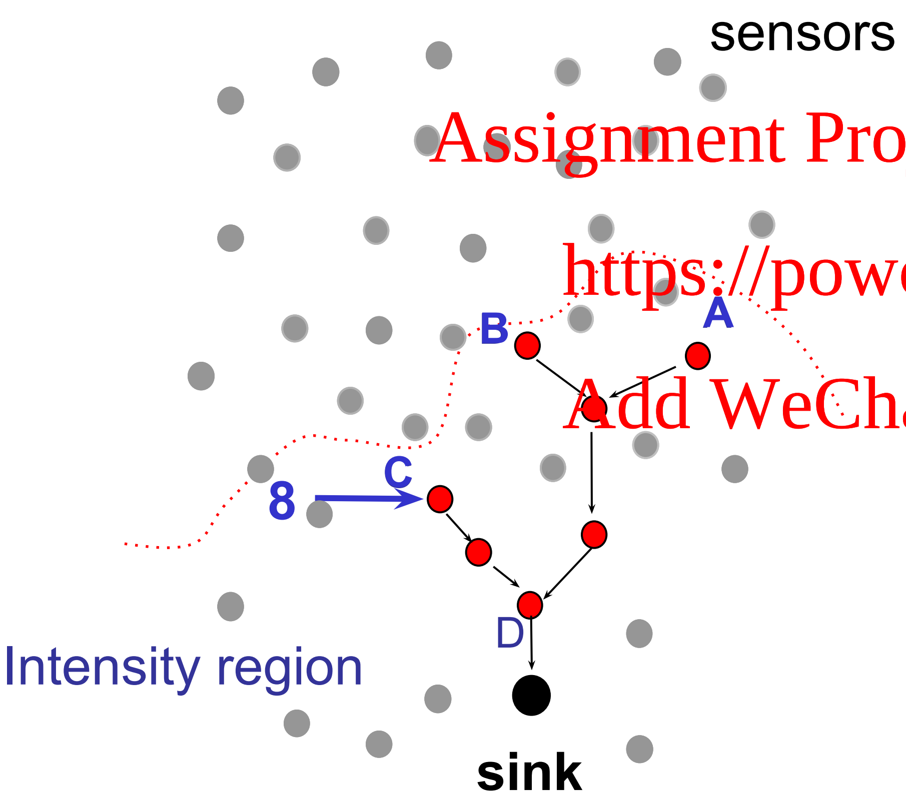
Sink-oriented Scheduling



Schedule packet

Header	A ; 3	B ; 4	C ; 3

Sink-oriented Scheduling



Schedule packet

Header	A ; 3	B ; 4	C ; 3

Conclusion

- Contribution
  - Boosts reliability by mitigating the funneling effect in choke points
  - Provides a lightweight, robust, and efficient hybrid TDMA/CSMA scheme
  - Shows that multiple medium access schemes can seamlessly coexist

- Funneling-MAC could more generally operate on multiple sinks/hierarchical sensor networks

Any other approaches to tackle the Funneling Effect?

Think-Pair-Share!