Politecnico di Milano

Advanced Network Technologies Laboratory

TinyOS SIMulator

Simulate a Wireless Sensor Network with TOSSIM

Motivations

- WSN require large scale deployment
- Located in inaccessible places
- Apps are deployed only once during network lifetime
- Little room to re-deploy on errors

System evaluation

- Check correctness of application behavior

- Sensors are hard to debug!
  - "… prepare to a painful experience" [Tinyos authors' own words]

Simulation Pros and Cons

- Advantages
  - Study system in controlled environment
  - Observe interactions difficult to capture live
  - Helps in design improvement
  - Cost effective alternative
- Disadvantages
  - May not represent accurate real-world results
  - Depends on modeling assumptions

General concepts

- TOSSIM is a discrete event simulator
- It uses the same code that you use to program the sensors
- There are two programming interfaces supported: Python and C++

Key requirements

- Scalability
  - Large deployments (10^3 motes)
- Completeness
  - Cover as many interactions as possible
  - Simulate complete applications
- Fidelity
  - Capture real-world interactions
  - Reveal unexpected behavior
- Bridging
  - Between algorithm and implementation

RadioTossim

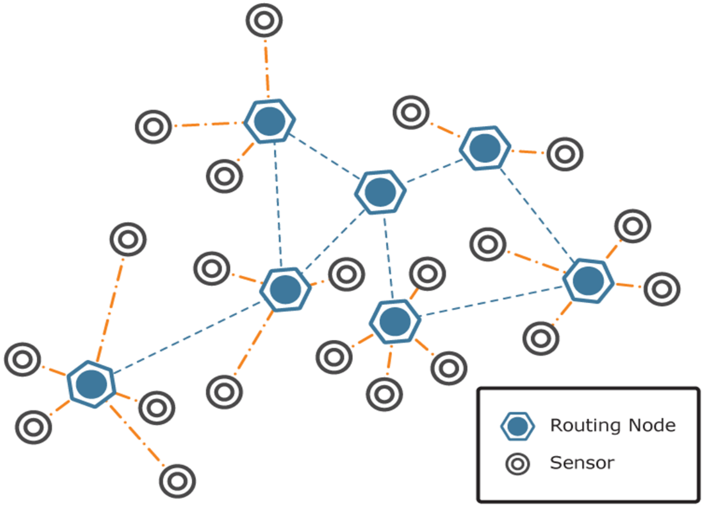
- Let’s simulate the RadioCountToLeds example with Tossim
- The updated code is in the folder RadioTOSSIM
- Behaviour:
  - Send a BROADCAST message with a counter
  - Turn on/off the LEDs according to the counter

TOSSIM Files

- Let’s use RadioTOSSIM example
  - TinyOS files:
    - `RadioTossC.nc`
    - `RadioTossAppC.nc`
    - `RadioToss.h`
  - Topology file: `topology.txt`
  - Noise file: `meyer-heavy.txt`
  - Simulation script: `RunSimulationScript.py`

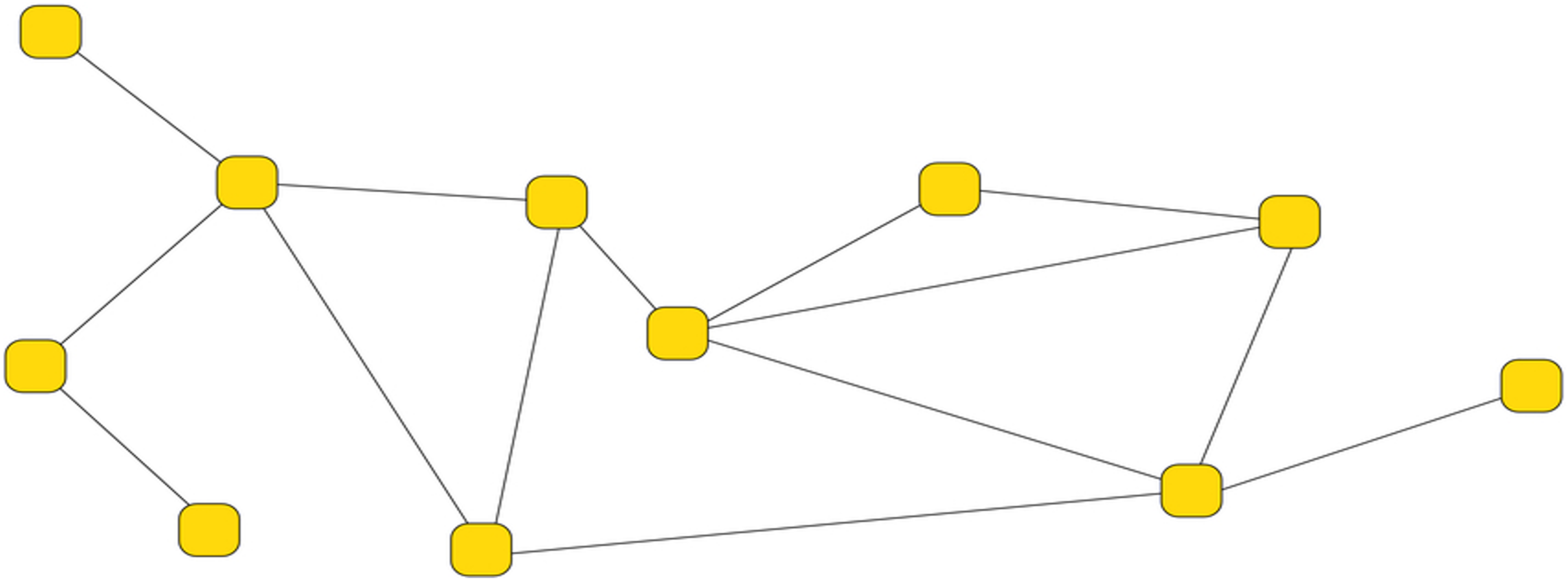
Configuring a Network

- It’s easy to simulate large networks
- You must specify a *network topology*
- The default TOSSIM radio model is signal-strength based



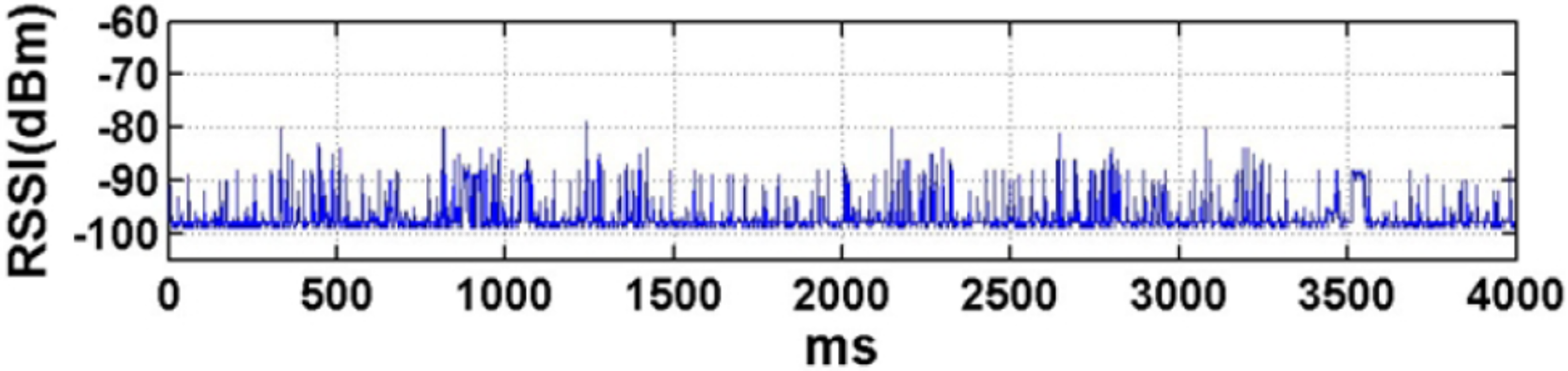
Network Topology

- You can create network topologies in terms of channel gain
  - source, the destination and the gain (for example "1 2 -54.0")



Radio Channel

- You need to feed a noise trace
- The directory tos/lib/tossim/noise contains some sample models



- On reception, SNR is evaluated for each bit of the message

How to Run TOSSIM

- To compile TOSSIM you pass the sim option to make
  - `make micaz sim`

- To run Tossim use the RunSimulationScript. It must be in the same folder of the TinyOS files
- `python RunSimulationScript.py`

Debugging Statements

- The output is configurable by channels
  - <ch> identifies the output channel
  - <text> text debugging and application variables

- dbg(<ch>,<text>) → DEBUG(ID):<text>
- dbg_clear → <text>
- dbgerror → ERROR(ID):<text>

- A channel can have multiple outputs