UPPSALA
UNIVERSITET

Simulation

Laura Marie Feeney

Wireless embedded systems I

- operating system and hardware
  - interrupts, scheduling…
- wireless communication
  - radio propagation (relation between P_{TX} and P_{RX})
  - modulation/demodulation (constellations, BER, PER)
  - errors (error correcting codes, ACKs and retransmissions)
- MAC layer
  - coordinate transmissions to allocate resources, avoid collision
  - allow devices to operate at low duty-cycle (sleep as much as possible)

Wireless embedded systems 2

- routing
  - forwarding data to central controller / AP

- data collection
  - sensor data being collected over an area
  - application/data specific processing (e.g. averaging)

- application
  - devices cooperate to provide a specific sensing/control application
  - quantity (lifetime) and quality of data

Design and performance evaluation…

- Design, development and performance evaluation
  - individual components
  - entire systems
  - specific deployments

- Questions…

Design and performance evaluation…

- Design, development and performance evaluation
  - individual components
  - entire systems
  - specific deployments

- Questions…
  - what % of packets will be successfully received?
  - how big a network can I deploy (area, number of devices)?
  - what is the best modulation (or ACK timeout, or MAC layer backoff parameter) to minimize latency?
  - how many packets are lost to interference?
  - when I deploy my application, how long will the batteries last?

Methods

- analytic (mathematical) models
  - physics/first principles
  - empirical

- build something and see what happens?
  - severe practical limitations, but ground truth….

- **simulation**
  - monte carlo methods
  - **discrete event simulation** (Lab 3: cooja simulator)
  - practical advantages, but many pitfalls…

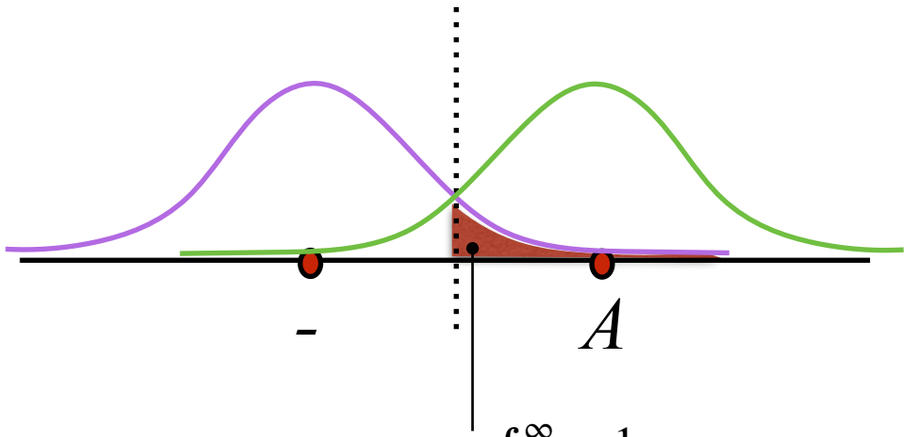
Analytic methods

- performance metric of interest can be represented by (a possibly very complex system of) mathematical equations that can be solved computationally

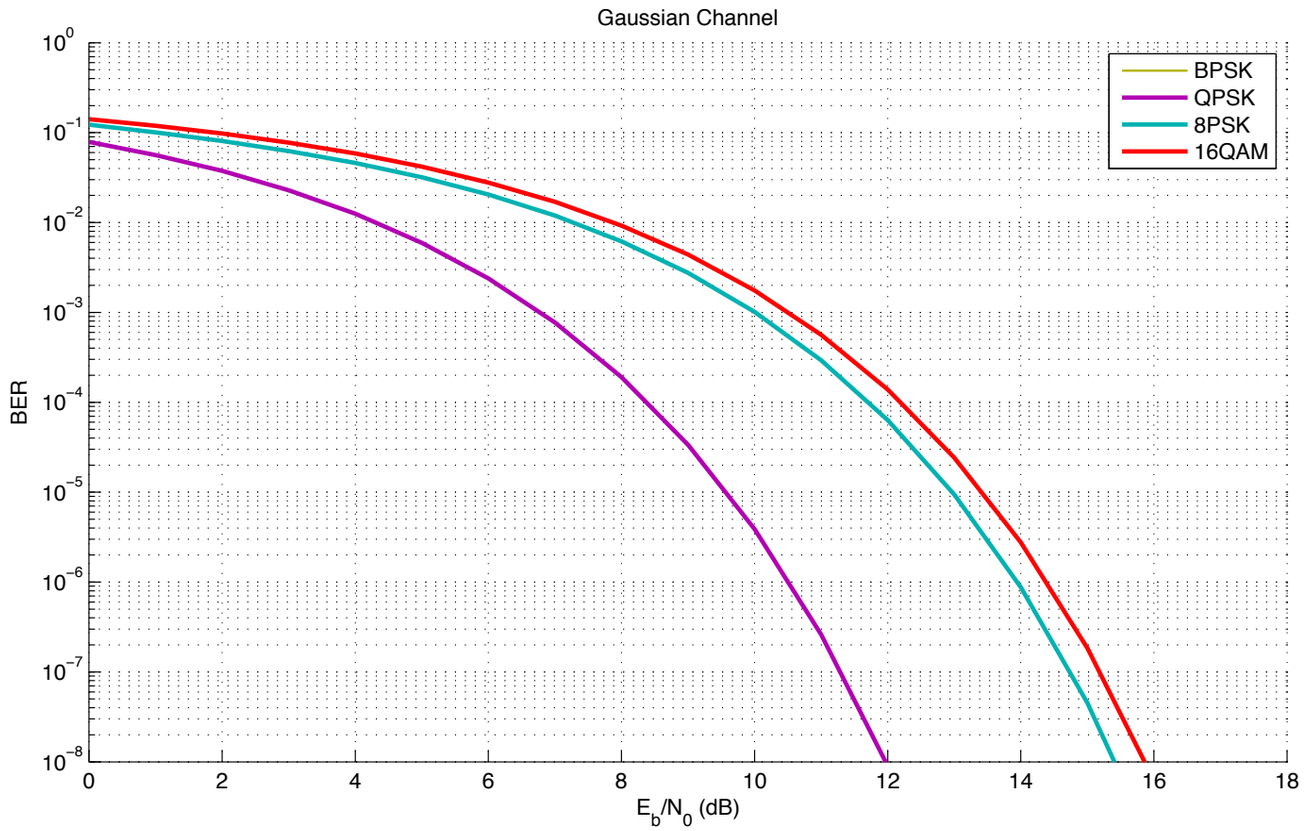
- example…

Analytic methods

- AWGN channel $\gamma_s = \frac{E_s}{N_0}$



$$P_b = P(x > 0) = \int_0^\infty \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x+A}{\sigma})^2}$$



Modulation	$P_s(\gamma_s)$	$P_b(\gamma_b)$
BPSK		$P_b = Q\left(\sqrt{2\gamma_b}\right)$
QPSK	$P_s \approx 2Q\left(\sqrt{\gamma_s}\right)$	$P_b \approx Q\left(\sqrt{2\gamma_b}\right)$
MPSK	$P_s \approx 2Q\left(\sqrt{2\gamma_s}\sin\left(\frac{\pi}{M}\right)\right)$	$P_b \approx \frac{2}{\log_2 M} Q\left(\sqrt{2\gamma_b \log_2 M}\sin\left(\frac{\pi}{M}\right)\right)$
M-QAM	$P_s \approx 4Q\left(\sqrt{\frac{3\overline{\gamma}_s}{M-1}}\right)$	$P_b \approx \frac{4}{\log_2 M} Q\left(\sqrt{\frac{3\overline{\gamma}_b \log_2 M}{M-1}}\right)$

Analytic methods

- advantages

- disadvantages

Simulation

- computational representation of a system for the purpose of studying its behavior
  - simplification

- only those aspects of the system that are relevant (to the question) need to be included
  - requires deep understanding and good judgement

- models need to have enough detail that conclusions are valid and useful
  - cannot be better than the underlying assumptions and models

- essential to understand limitations!

Why do simulation?

- Measuring real systems gives "ground truth"
  - even best simulation is just a representation and simplification

- Advantages for software development

- Advantages for performance evaluation

Wireless communication in embedded systems

- embedded devices support an application
  - sensing and/or control over a physical area, physical objects
  - user value: quantity (lifetime) and quality of data

- challenges
  - small devices: limited computing resources
  - wireless: limited communication resources
  - radio environment: fading, interference, shadowing
  - battery-powered: energy constrained
  - distributed application: complex interactions
  - deployed in some environment (factory, home, city, forest…)
  - sense and/or control of environment or physical objects

Why do simulation?

- Measuring real systems gives "ground truth"
  - even best simulation is just a representation and simplification

- Advantages for system development
  - much easier for debugging
  - controlled and deterministic (e.g. no interference)
  - stop system, visibility to all variables
  - hardware may not exist yet

- Advantages for performance evaluation
  - much faster and easier
  - access to all performance information
  - create many simulated deployments/environments
  - (automated) sweep over many configurations and parameters

Simulation

- monte carlo simulation
  - probabilistic, static (no time)

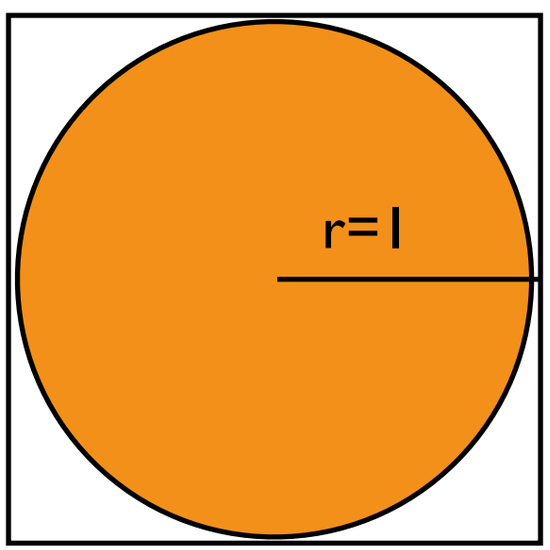
- trace-driven simulation

- **discrete event simulation**
  - event-driven, time
  - widely used technique for design and performance of wireless embedded networks (and **many** other things)
  - often includes analytic or probabilistic models as components
  - can be implemented at many different levels of resolution, depending on goal
  - requires deep understanding and good engineering judgement

Monte Carlo simulation

- model probabilistic phenomena
  - commonly used with markov chain models (MCMC)
  - complex mathematics

- can be used to evaluate non-probabilistic expressions using probabilistic methods
  - classic example: estimate pi

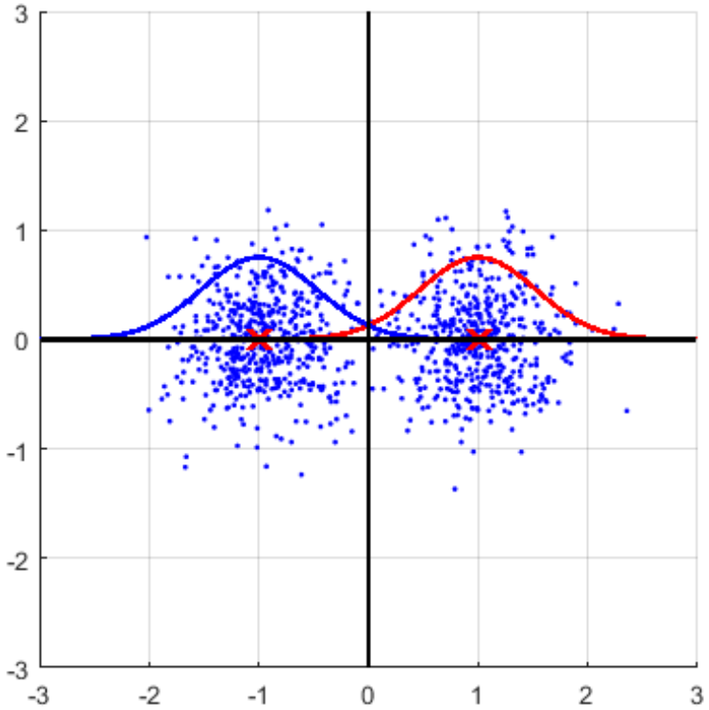


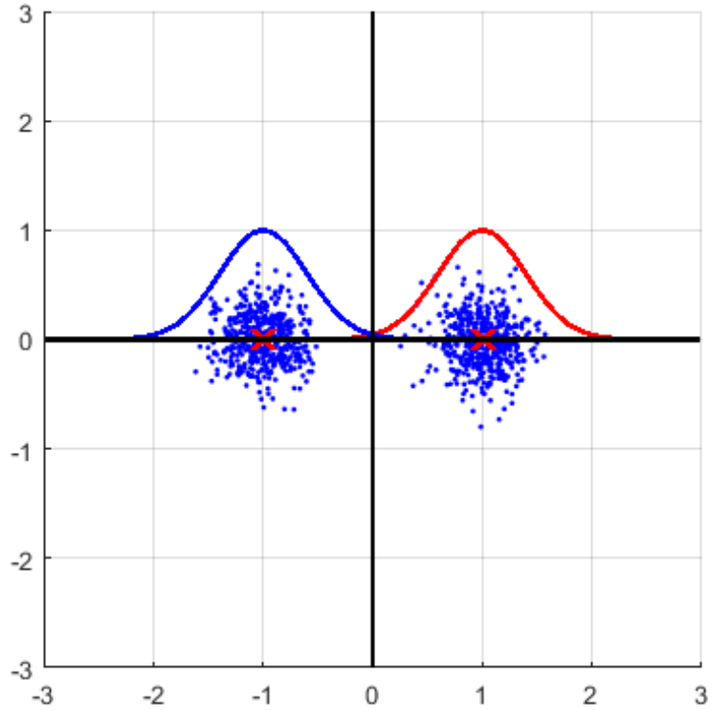
choose x,y values from uniform distribution

find proportion such that $x^2 + y^2 \leq 1$

Monte Carlo simulation

- AWGN channel — calculate BER





Discrete event simulation

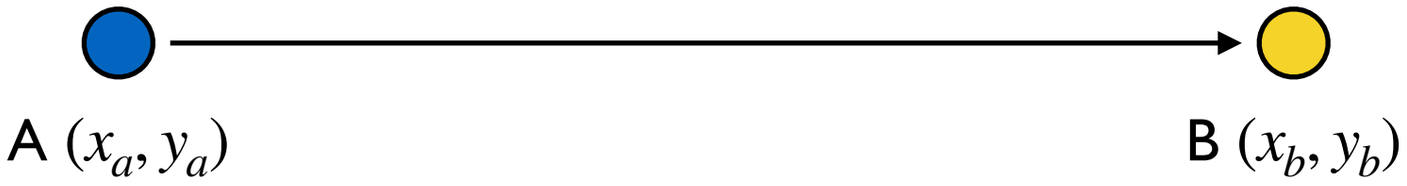
- operation of a system is a chronological sequence of events

- each event occurs at an instant in time and changes the state of the system

- generates an ordered queue of events

- take the first event and process it
  - creates one or more new events that are put in the queue

Discrete event simulation

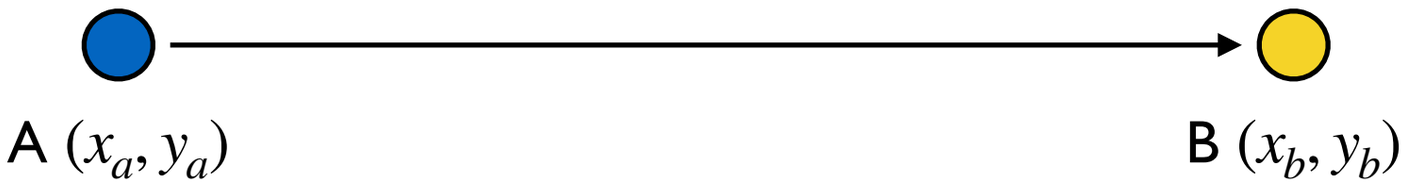


$t = 0$: A sends a packet to B

$t = 0$: A sends a packet to B

$t = 4$: A times out waiting for ACK

Discrete event simulation



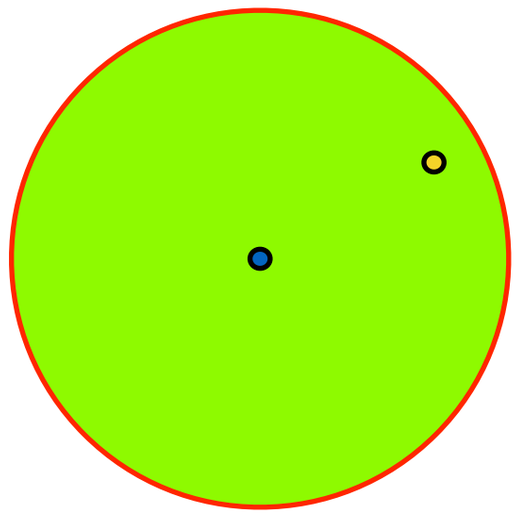
$t = 0$: A sends a packet to B

$t = 1$: B receives packet

$t = 0$: A sends a packet to B

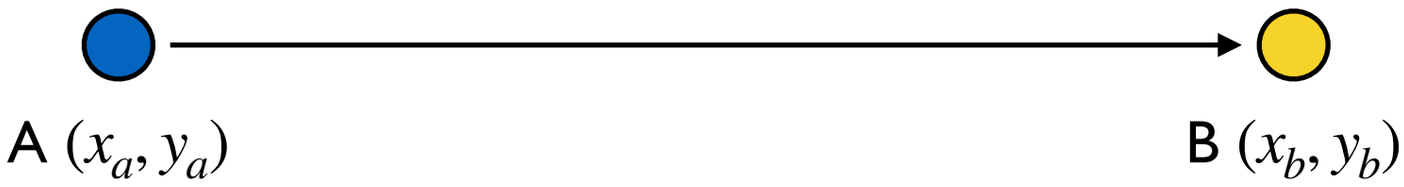
$t = 4$: A times out waiting for ACK

unit disk graph model



$d(A, B) \leq 1$

Discrete event simulation



$t = 0$: A sends a packet to B

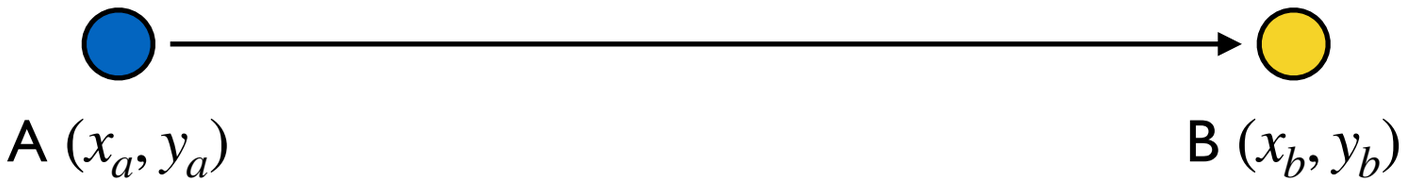
$t = 1$: B receives packet

$t = 2$: B sends ACK

$t = 1$: B receives packet

$t = 4$: A times out waiting for ACK

Discrete event simulation



$t = 0$: A sends a packet to B

$t = 1$: B receives packet

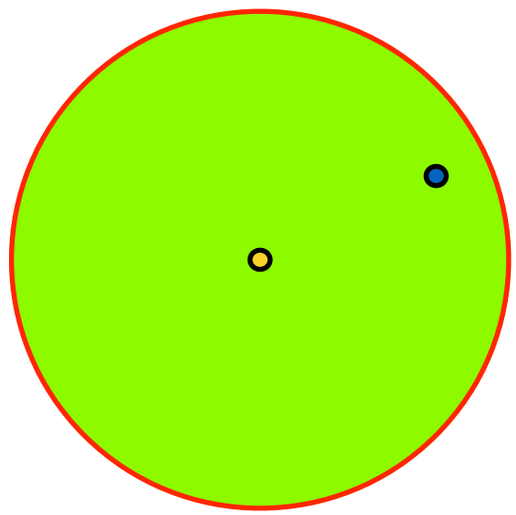
$t = 2$: B sends ACK

$t = 3$: A receives ACK

$t = 4$: A times out waiting for ACK

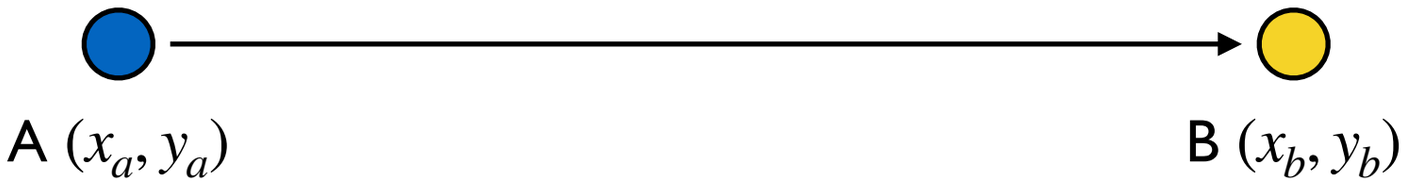
$t = 1$: B sends ACK

unit disk graph model



$d(B, A) \leq 1$

Discrete event simulation



$t = 0$: A sends a packet to B

$t = 1$: B receives packet

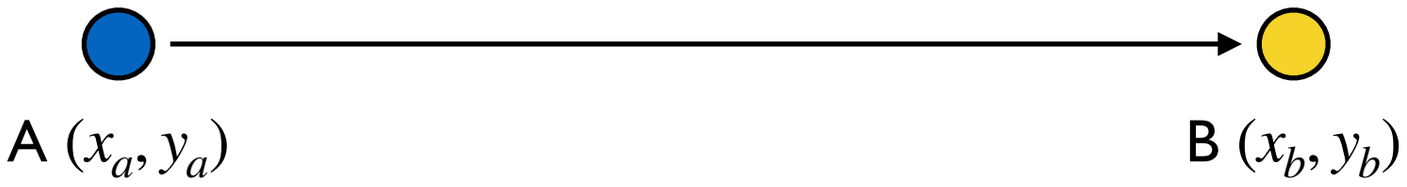
$t = 2$: B sends ACK

$t = 3$: A receives ACK

~~$t = 4$: A times out waiting for ACK~~

Note that there is one event queue for the whole simulation.

Discrete event simulation



$t = 0$: A sends a packet to B

$t = 1$: B receives packet

$t = 2$: B sends ACK

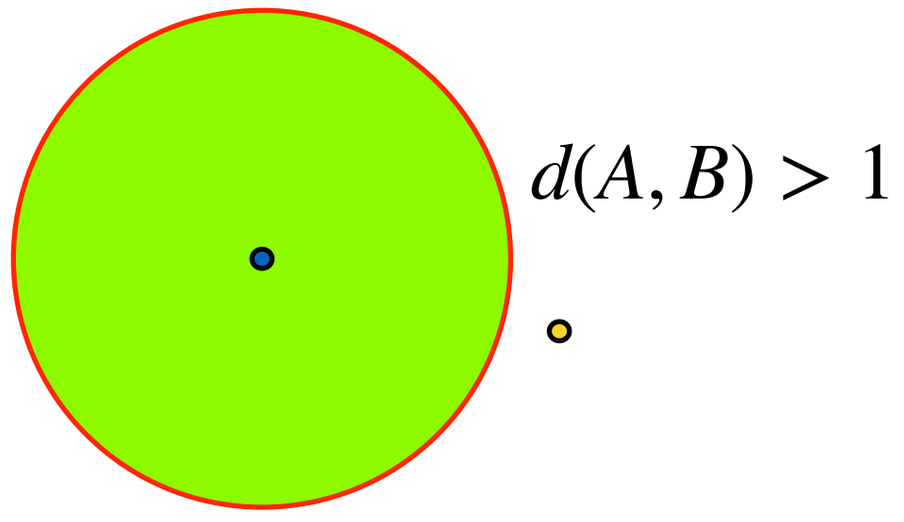
$t = 3$: A receives ACK

$t = 4$: A times out waiting for ACK

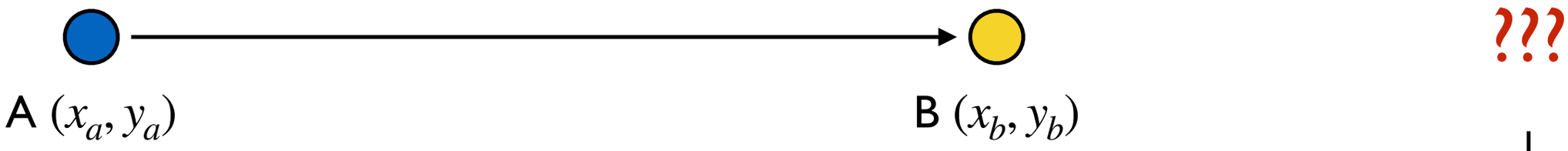
$t = 5$: A retransmits packet to B

$t = 4$: A times out waiting for ACK

unit disk graph model



Discrete event simulation



t = 0: A transmits with P_{tx}, N bits, and bit-rate R

t = t + d(A,B)/c: packet begins at B —> P_{rx}

t = t + N/R: packet ends at B —> B received?

- start-end events are more typical
- lots and lots of events

What could possibly go wrong :-)

- model not accurate enough for purpose
- poor choice of simulation parameters
- models aren't verified (right model, bug in implementation)
- simulation doesn't run long enough
  - e.g. first few packets are fine, but later there is a backlog
- not enough different runs
  - if there is a probability element, like BER, need different sequences of random numbers)…

Challgenges

- Challenges
  - what parts are hardest to simulate?

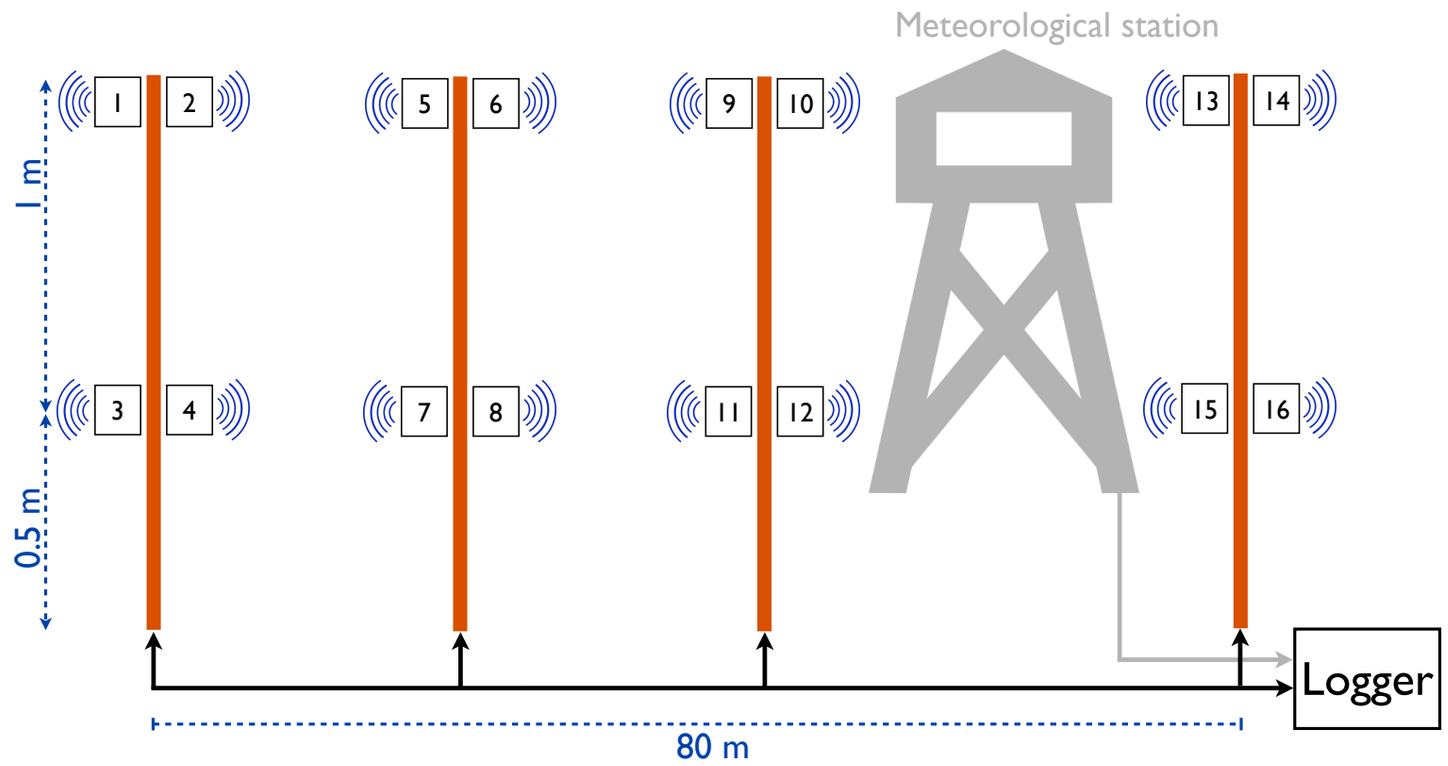
- Advantages

- Disadvantes

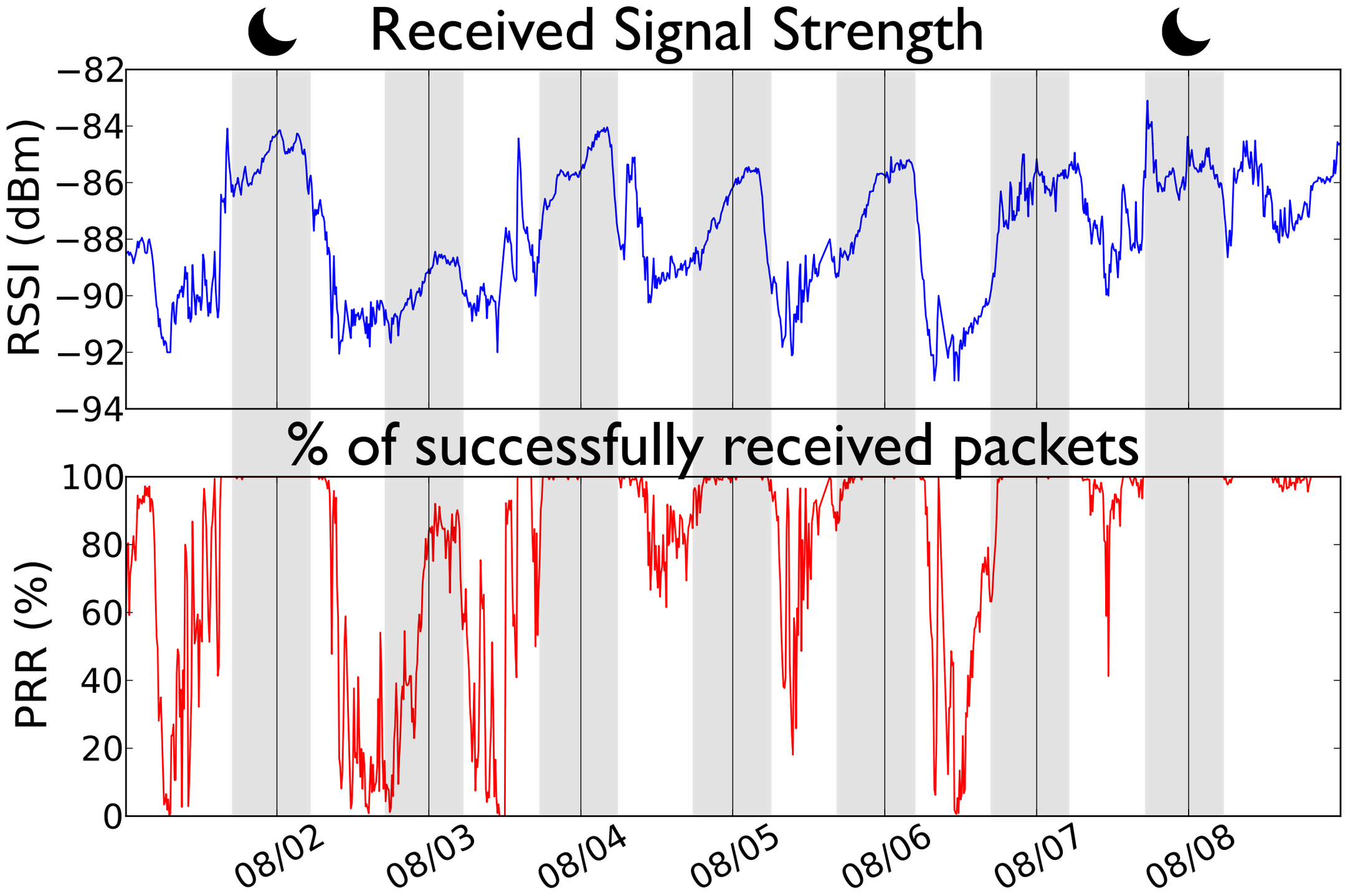
Challenges

- Useful simulation is very very hard
  - huge amounts of engineering judgement — what part of reality needs to be included and how is it represented/modeled
  - protocols and applications are pretty easy to model (run the same algorithm (or even the same code) that is in your real system
  - radio model…. very very hard
  - sensed phenomena, traffic loads — also hard

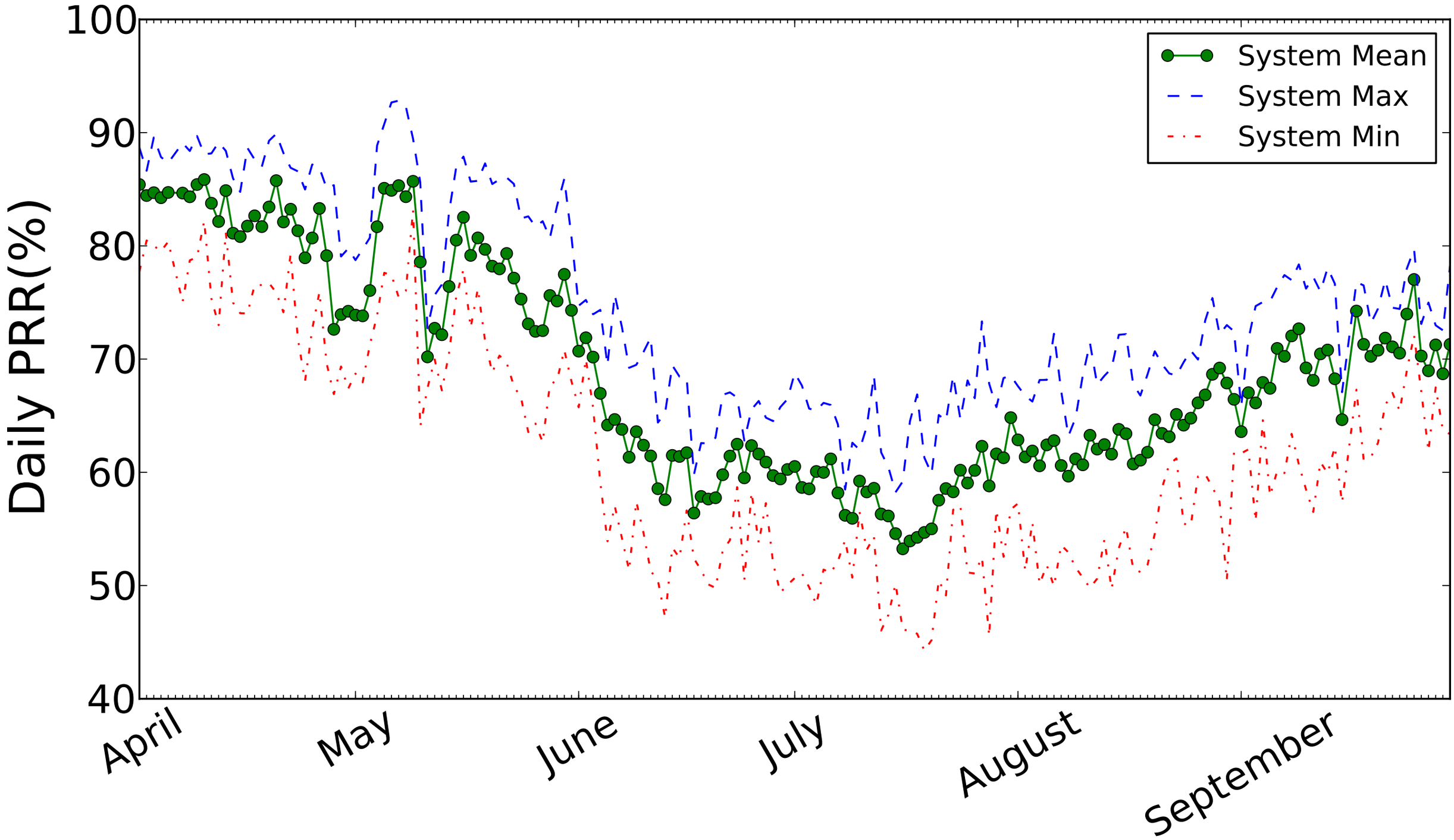
WSN Deployment



Measuring Link Quality



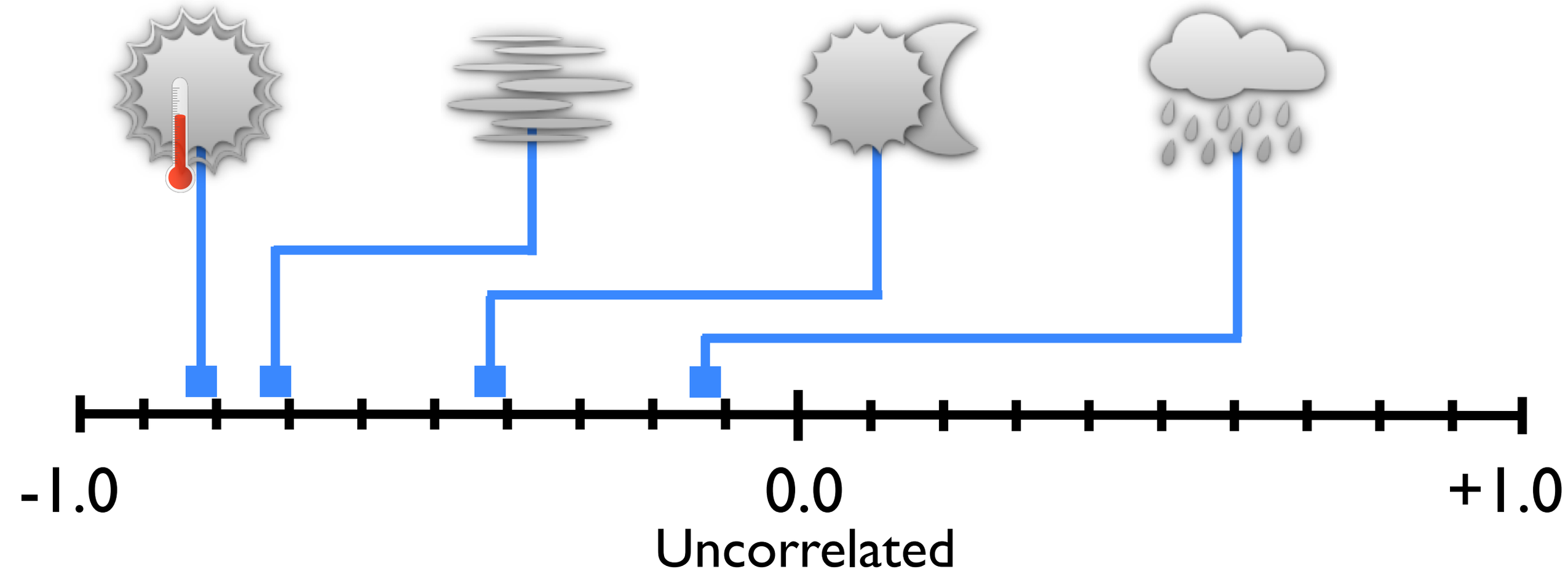
Seasonal Variation



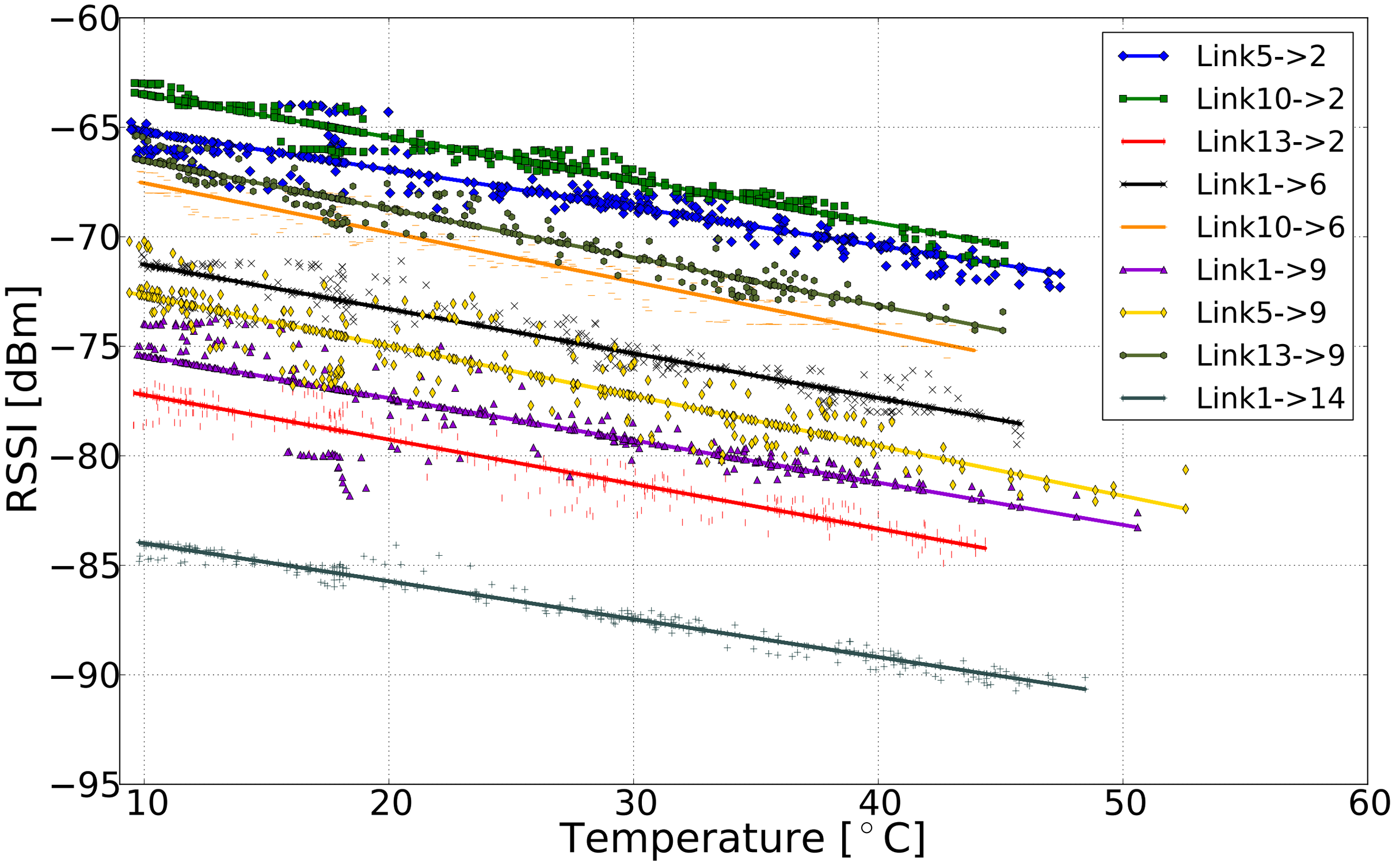
Meteorological Impact

Temperature correlates the most with the quality of the link!

Temperature Abs. Humidity Sunlight Precipitation

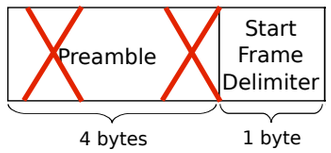


Signal Strength vs. Temperature



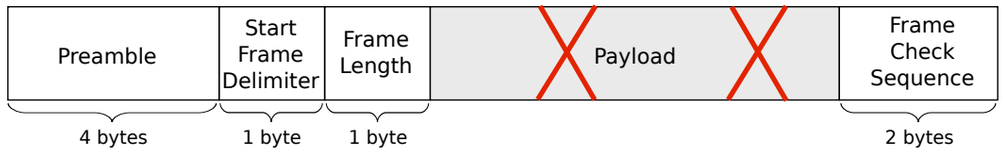
Weakening Links

just for illustration



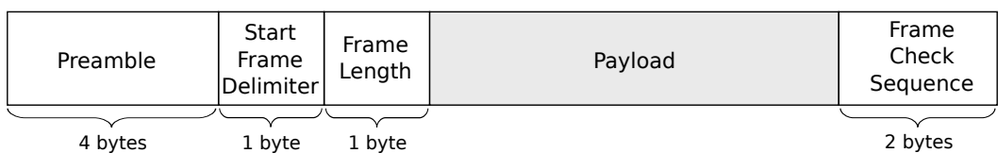
Missed

35.9%



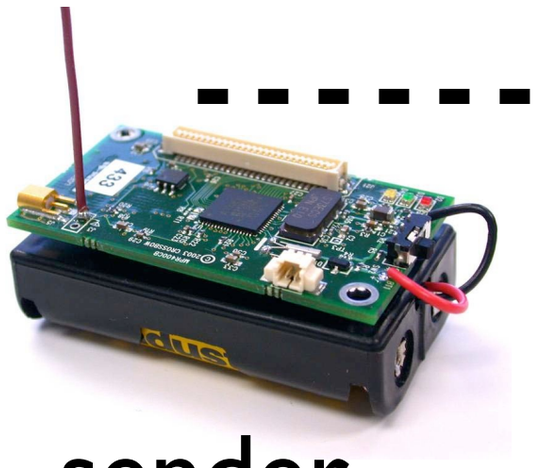
Broken

14.2%



Successful

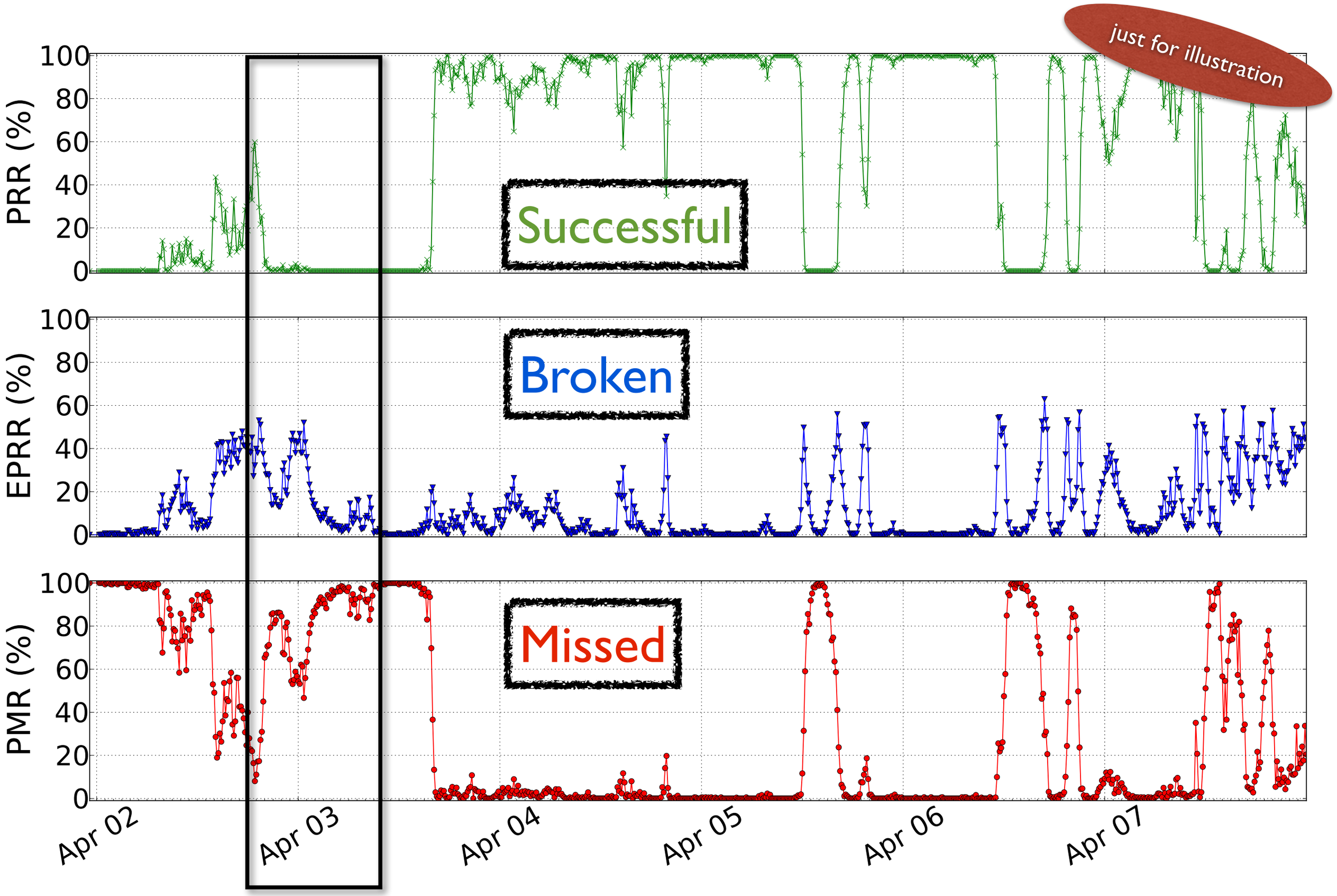
49.9%



sender

transmission with little or no interference

receiver


just for illustration
Successful
Broken
Missed
PRR (%)
EPRR (%)
PMR (%)
100
80
60
40
20
0
Apr 02
Apr 03
Apr 04
Apr 05
Apr 06
Apr 07

Discrete event simulators

- COOJA
  - specialized simulator
  - part of Contiki OS
  - open source (Contiki and Cooja @ SICS and UU)
- OMNET++
  - optimized general-purpose discrete event engine (many application areas)
  - many wireless simulation tool kits
  - open source for academic

Cooja simulator

- developed at SICS / Uppsala
- it is part of Contiki and for simulating networks of contiki nodes

- simulates at 3 different levels:
  - application level (Java)
  - OS level (contiki)
  - hardware level (MSPsim emulator)
- “plug-ins”
- radio models
  - “unit disk graph” — ray-tracing