Modeling and Analysis of WSNs

Joint work with :

F. Maraninchi, L. Samper, O. Bezet, W. Znaidi

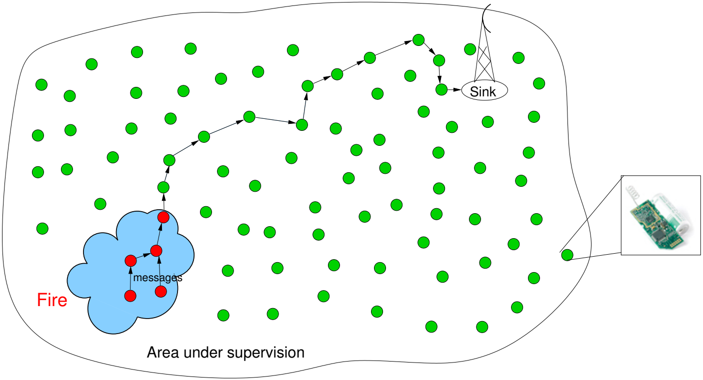
And many discussions within the ARESA project :
France-Télécom, Coronis Systems, LIG-Drakkar, CITI, TIMA

Overview

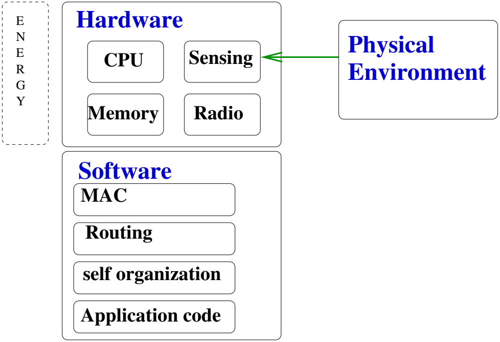
1. **Wireless Sensor Networks**
2. Design, programming and analysis techniques for WSNs
3. What has been done within VÉRIMAG ?
4. Where we could go in a near future ?

Wireles Sensor Network

A set of sensing devices (nodes), communicating through radio links, to collect/agreggate/transmit information about their external environment.



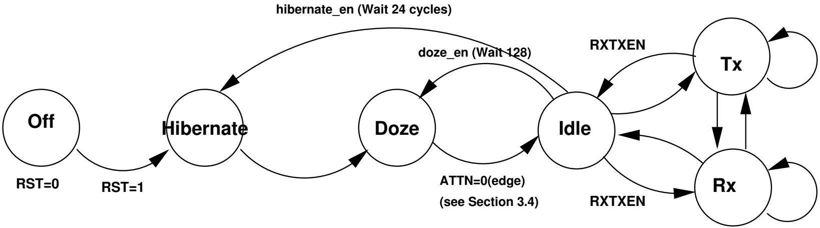
Node architecture



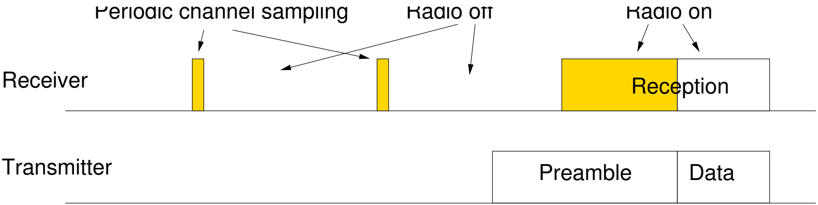
Typical hardware characteristics (e.g., Mica motes, Crossbow) :

- CPU : 8/16 bits, 8 MHz, $\sim$ 128 Kb pgm, $\sim$ 512 Kb data
- Radio : range $\sim$ 100 meters, 40.000 bits/sec.
- Energy : 2 AA batteries

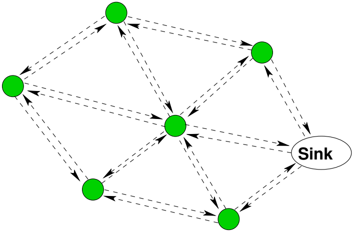
Typical radio device

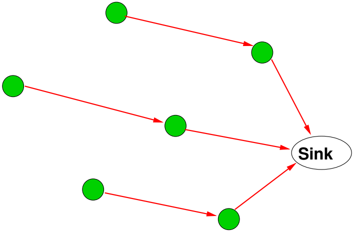


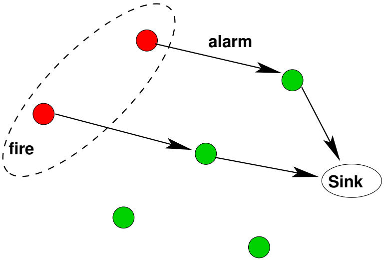
Typical MAC protocol



Typical routing protocol : directed diffusion







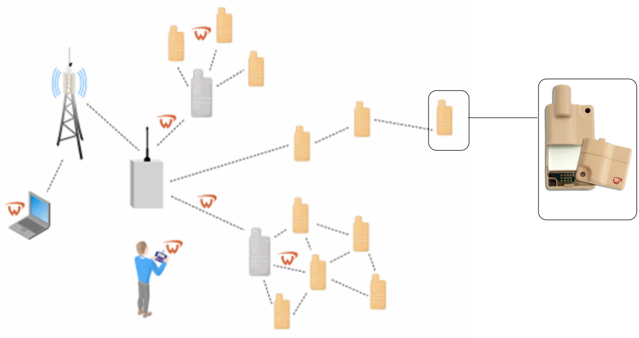
Typical (foreseen) WSN applications

- Environement/health monitoring
- Smart buildings (structure monitoring, home comfort, ...)
- Object tracking
- Traffic management
- Metering
- etc.

⇒ **some key parameters :**
node mobility, dynamic network (re)-configuration, energy supply, network size, security concerns, etc.

A real commercial application

Water counter metering, Coronis Systems



- periodic sampling (sensor at 32 hz, 1 measure/hour, 1 emission/day)
- expected lifetime : $\sim$ 10-15 years (using a 3.6 v Lithium battery)
- network : up to 2000 nodes, partly configured by hand
  (no collisions !)

Why is it still a challenging domain ?

WSN $=$ **embedded systems**

(tight constraints, difficult to update)

$+$ **wireless**

(multi-hop communications, timed behaviour)

$+$ **non-functional properties**

(energy, QoS, security, dots)

$\Rightarrow$ important need in **cross-layers development** and **tuning**

$\Rightarrow$ so far, **no "convincing" design and analysis techniques**

Some "related" topics :

- Ad hoc networks, vehicular networks (VaNet), ...
- smart cards, ...

Overview

1. Wireless Sensor Networks
2. **Design, programming and analysis techniques for WSNs**
3. What has been done within VÉRIMAG ?
4. Where we could go in a near future ?

Design and programming efforts

- **Hardware level :**
  Tight integration between low power CPU and radio transceivers
  (Berkeley's motes, Arch Rock, Coronis transceiver, WSN430 (Lyon CITI), ...)
- **Protocol level :**
  A huge number of MAC proposals, some "integrated" communication stacks
  (802.15.4, ZigBee, Wavenis, ...)
- **OS and application level :**
  A few dedicated languages, with programming and execution environments
  (nesC/TinyOS, Sun SPOT/Squawk VM, Contiki, etc.)

TinyOS (Berkeley)

An operating system and developemnt plateform for WSNs . . .

1. a **component library** (hardware interface + com. protocols)
2. a **programming language** (nesC)
3. a **code generator**
   - for specific hardware *motes*
   - for simulation tools

Generated code = software components + “built-in” scheduler

nesC (Berkeley)

nesC = C + component model + concurency model

Component interface :

- commands (methods accepted by the component)
- events (call backs sent by the component)

Concurency :

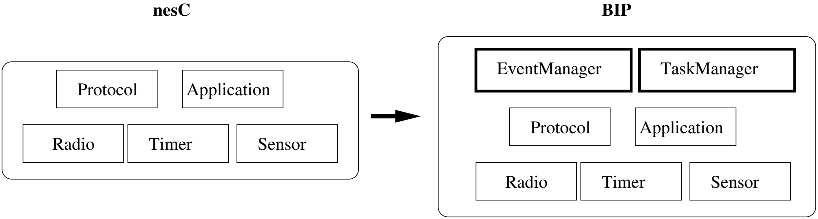
- "synchronous" tasks
  (executed when CPU is idle, can be preempted)
- events : hardware interuptions, split-phase operations
  (preempt the execution of running task or event).

$\Rightarrow$ **low level programming model, possible race conditions, ...**

Modeling nesC/TinyOS with BIP

[joint work with Ananda, Joseph, Jacques (Pulou), Marc]

$\longrightarrow$ a translation of the nesC execution model into BIP



- partly automated (nesC behaviour to be translated by hand)
- shows that BIP is expressive enough
- compare to similar work performed with Ptolemy
- could be used for "intra-node" validation ?

Existing analysis techniques

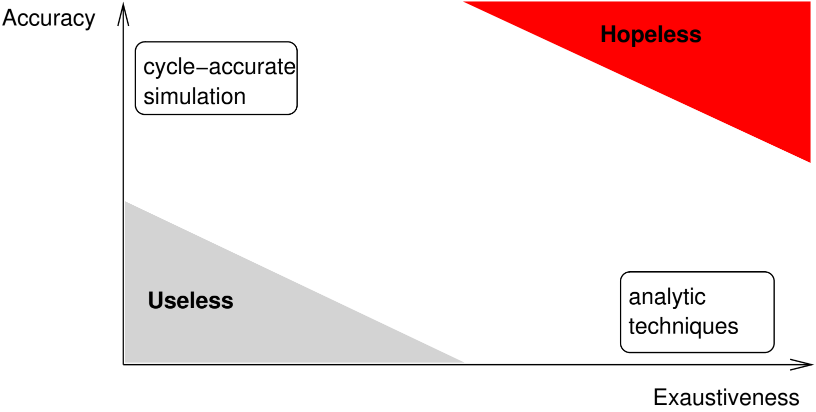
Simulation techniques

- operational model, can be executed
- can be made accurate, including "real" code (cycle-accuracy) : ("true" packet collisions, inst. energy consumed)
- but non exhaustive, and can be **simulator dependant**

Analytic techniques

- probabilistic model, descriptive
- requires some abstraction level, are they **sound** ? (prob. of collisions, energy = nb. of msgs sent)
- exhaustive analysis (worst case, mean case)

The current picture



Overview

1. Wireless Sensor Networks
2. Design, programming and analysis techniques for WSNs
3. **What has been done within VÉRIMAG ?**
4. Where we could go in a near future ?

Useless → Hopeless : how to fill the blank ?

1. Global and accurate **simulation model** for WSNs
   - to better understand the domain ...
   - to build an experimental simulation plateform
2. Experiments with existing **model-checking** tool
   - to know where are their limits
   - to propose the necessary extensions
3. Definition of a sound **abstraction relation**
   - taking into account the energy consumption
   - that can be applied "component-wised"

Glonemo : specification

A global, component-wise, and accurate **WSN model** :

- detailed node hardware description
- several software layers (com. protocols, application code)
- the physical environment (comm. canal, sensor inputs)

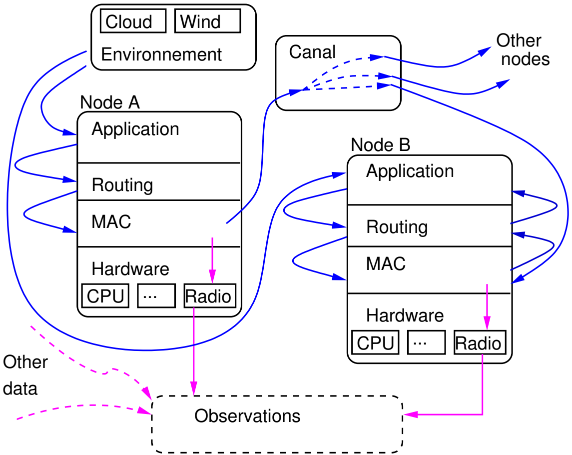
A **simulation engine** :

- able to collect various data during a run
  (e.g., energy, msg exchanged, ...)
- efficient enough ...

⇒ a **"benchmark application"** :

monitoring and tracking of a *pollution cloud*

Glonemo : model architecture



Implementation

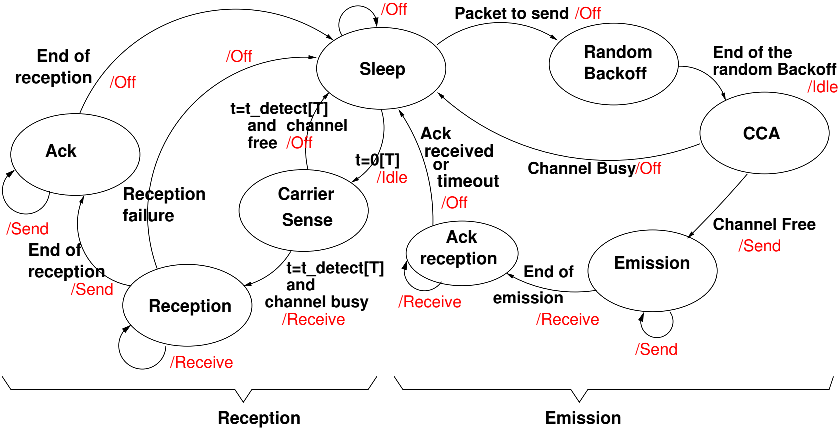
Written in **Reactive ML** (L. Mandel, LRI)

- based upon Caml (expressive, high-level, data structure)
- parallelism is a top-level primitive
  (1 process per component, 8 components per node)
- synchronous reactive model (discret global time)
  $\Rightarrow$ fixed step simulation

Energy model

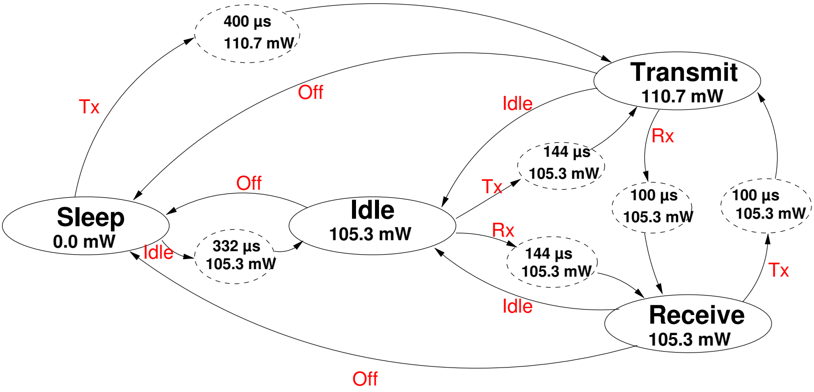
- synchronous observers associated to each hardware element
  (1 state = 1 instantaneous consumption value)
- mimics the functional behaviour
- a global observer integrates the energy consumed / node

MAC protocol

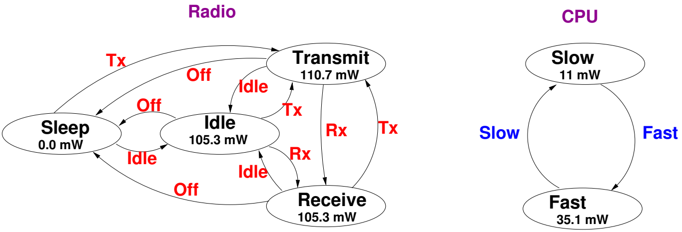


CCA : Clear Channel Assessment

Radio energy consumption (Motorola MC13192)



Global energy consumption (example)



Radio	in	Idle	-	Tx	-	-	Off
	state	Sleep	Idle	Idle	Tx	Tx	Tx
	conso (mW)	0.0	105.3	105.3	110.7	110.7	110.7
CPU	in	Fast	-	-	-	-	Slow
	state	Slow	Fast	Fast	Fast	Fast	Fast
	conso (mW)	11	35.1	35.1	35.1	35.1	35.1
Node	(mW)	11	140.4	140.4	145.8	145.8	145.8
	mJ **(1 step** $= 10^{-2}$**s)**	0.11	1.514	2.918	4.376	5.834	**7.292**

Environment modelling

Realistic sensor inputs are spatially and temporarally correlated ($\neq$ e.g., Poisson law)

$\Rightarrow$ need to be explicitely modeled ...

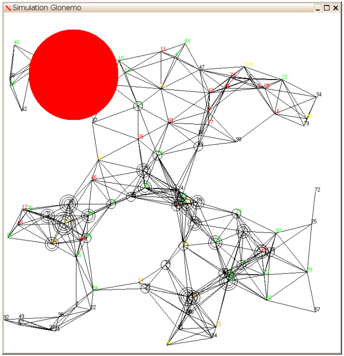
Connection between Reactive ML and Lucky

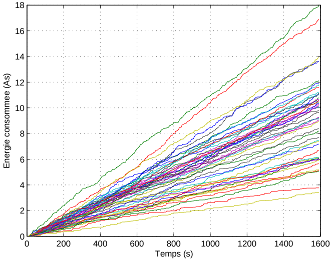
- a constraint-based language
- description and simulation of stochastic reactive systems

Application : processes `wind` and `cloud`

$\Rightarrow$ experimental results showed more pessimistic network lifetimes than with classical distribution laws

In practice . . .





efficient : up to 100.000 nodes, faster than real-time below 1000 nodes

modular : set of replacable "components"

Some experiments with Glonemo

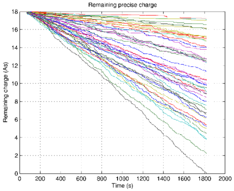
“accurate” vs. “abstract” modelling

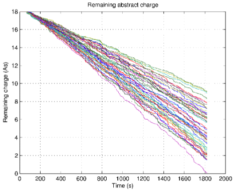
→ compare the lifetime of each node when energy E :

1. is computed accurately (w.r.t. hardware current state)
2. is abstracted :

$$E = \text{number of emissions} \times \lambda \times \text{fixed emission cost}$$

$$\lambda = \text{average number of collisions (estimated by simulations)}$$





⇒ mean values not always representative . . .

Comparison with a real network

→ "calibrate" the simulator by comparison with an existing WSN

Coronis :

- network deployed since 2004
  (water counter metering, Les Sables d'Olonnes)
- monitoring + precise energy consumption estimation

Work in-progress :

1. simulate this application with Glonemo (2000 nodes)
2. compare the energy consumptions
3. evaluate a possible auto-configuration protocol

A new auto-organisation protocol

→ evaluate a new protocol (T. Watteyne, FT/CITI)

- "geographic" routing from **virtual** node coordinates
- node coordinates randomly initialized, updated during msg emissions
- analytic an simulation techniques are encouraging (good performance, robust, energy efficient)

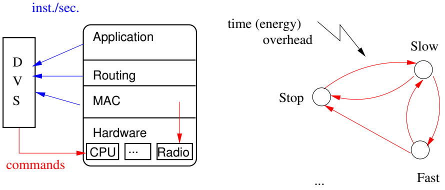
work in-progress :

1. implement this protocol within the "cloud" benchmark
2. compare with other classical MAC/routing protocol
3. better tune the organisation phase

Using Dynamic Voltage Scaling (DVS) ?

How to reduce the CPU consumption inside a node ?

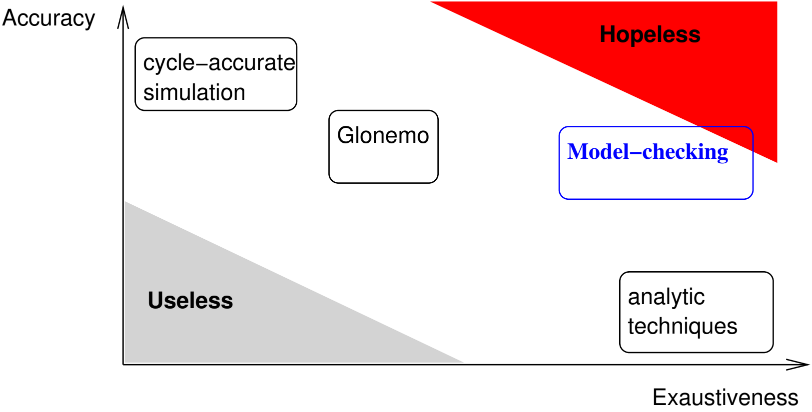
- V divided by $n \Rightarrow$ speed divided by n, E divided by n^2
- WSN nodes are essentially idle or weakly busy (weak duty cycle)
- synchronous CPU :
  - discrete working states
  - time (and energy) latency when state changes



Node activity prediction ? Strong deadlines ? Need for an OS ?

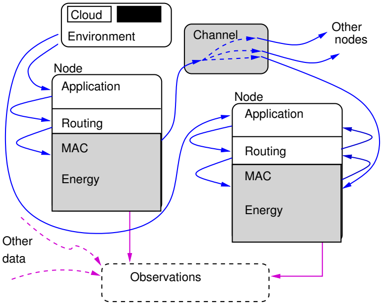
$\Rightarrow$ **preliminary answers through simulation . . .**

The current picture



WSN model-checking with IF : a case study

- Detection of a pollution cloud
- Comparison of the network lifetime w.r.t two routing algorithms : directed diffusion and flooding.
- Other elements (MAC, hardware, channel, ...) more abstract



IF model (details)

- Network lifetime : a global clock, never reset
- MAC layer and channel :
  - $\rightarrow$ real network :
    - Unicast : acks $\Rightarrow$ msgs re-emitted when lost
    - Broadcast : no acks, no re-emissions
  - $\rightarrow$ IF model :
    - Unicast : arbitrary (finite) emission delays
    - Broadcast : non-deterministic failures, fixed emission delays
- Energy : 1 emission = 3 units ; 1 reception = 2 units
- Environnement :
  - périodic stimulation of the current node
  - move to a neighbour node

Computation of worst-case network lifetime

A node is *dead* (fail-silent behaviour) when out of energy

Several criteria do define network lifetime :

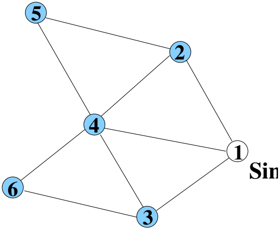
- elapsed time before the death of X % of the nodes
- elapsed time before the network is partitioned

$\Rightarrow$ notion of "dead" states

worst-case scenario =
shortest path (w.r.t Lifetime clock) from initial to a "dead" state

Implementation :
IF exploration engine + A* shortest path search algo (`min-cost` tool)

Experimental results



Worst case lifetime	1st node dead	partitioned network
Flooding	14	21
Directed diffusion	41	52

1st node dead	worst-case	random simulation				
Flooding	14	40	32	18	21	19
Directed diffusion	41	80	78	123	108	101

Model-checking with IF : conclusion

In favor :

- WSNs routing algorithms could be modeled
- Possible to define and compute lifetime properties
- performs like similar tools (Uppaal, RT-Maude)

Against :

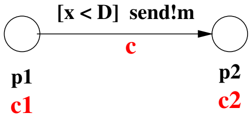
- strong and uncontroled abstractions
  (energy consumed $\sim$ number of msg transmitted)
- network configuration too small
  (12 nodes $\Rightarrow 3.10^6$ states $\Rightarrow$ 1 day of CPU)

Model-checking with IF : what else could be done ?

- **Modeling langage**
  - communication primitives
    (Fifo queues vs radio broadcasts + RDV)
  - network topology
  - $\Rightarrow$ a WSN BIP profile ?
- **Exploration engine**
  - dedicated data structures for energy consumption
    state = (control, discrete data, clocks, energy)
    e.g., symbolic "priced zones" (Linear Priced Timed Automata)
  - dedicated exploration algo ? Bounded model-checking ?
- **Abstraction techniques**
  - combine various abstraction degrees :
    a detailled node + a few less detailed ones + rest of the net

Linearly Priced Timed Automata (LPTA)

Timed automaton extend with **prices** (on states and transitions)



$$(p1, \eta, p) \xrightarrow{\delta} (p1, \eta + \delta, p + c1.\delta)$$
$$(p1, \eta, p) \xrightarrow{\text{send !m}} (p2, \eta, p + c)$$

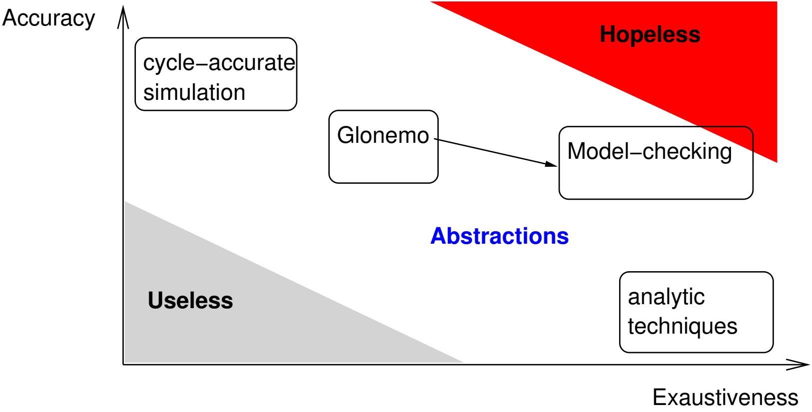
$\Rightarrow$ **costs** are associated to each run of the automaton

Symbolic state space representations : **priced zones**

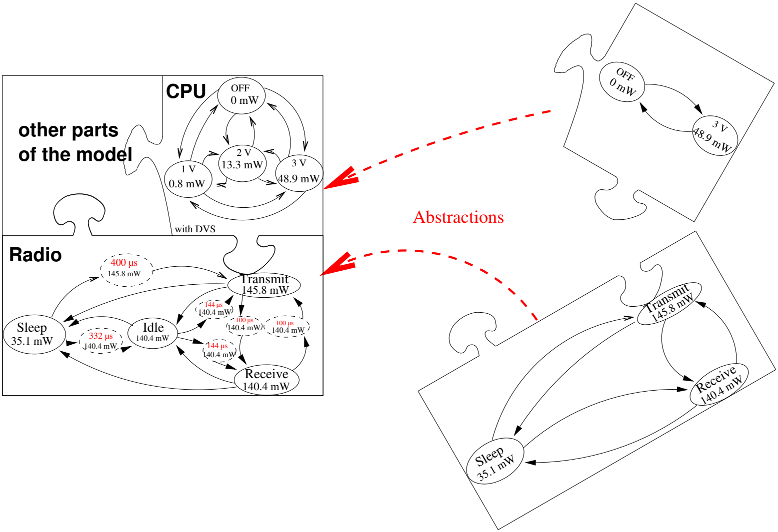
- capture the **minimal** cost to reach a location
- minimum cost reachability pb is **decidable**

Efficiency in practice ? Application to maximal cost ?

The current picture



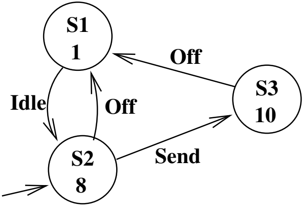
Energy preserving abstractions



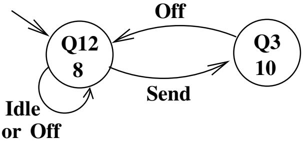
Abstraction definition (1/3)

M_1 **less abstract than** M_2 iff

1. same inputs $\rightarrow$ same outputs (functional equivalence)
2. M_2 consumes as much as M_1 (worst-case lifetime)



Detailed radio model, M_1

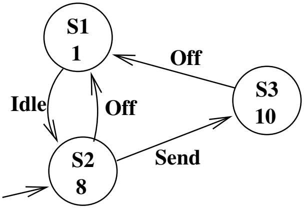


Abstract radio model, M_2

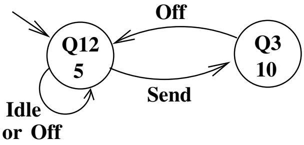
$$M_1 \preceq M_2 =_{def}$$
$$\forall \text{ input } e.\ Out(M_1, e) = Out(M_2, e) \wedge E(M_1, e) \leq E(M_2, e))$$

But, very **strong** abstraction ...

Abstraction definition (2/3)



Detailed radio model, M_1



Abstract radio model, M_2

Finer abstraction, but true only under certain conditions :

$$\exists e \text{ such that } E(M_1, e) \not\leq E(M_2, e)$$

$\Rightarrow$ Need to restricts the set of inputs using a **context** K

Abstraction definition (3/3)

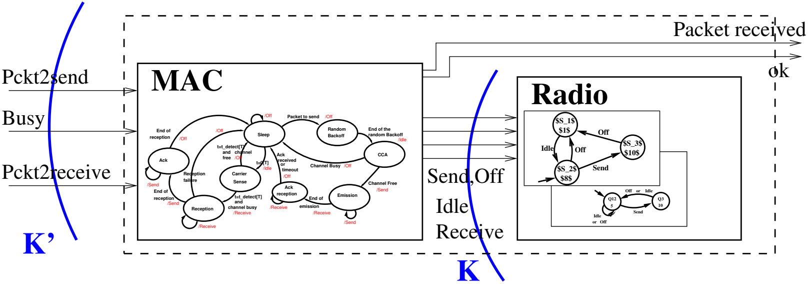
Abstraction might be true only for some "relevant" inputs (not for all combinations, only after a certain time, etc.)

Context K = set of input traces (e.g., $K \in 2^{\{\texttt{Send},\texttt{Idle},\texttt{Off}\}^*}$)

$$M_1 \preceq_K M_2 \iff \forall e \in K, \begin{cases} \mathrm{Out}(M_1, e) = \mathrm{Out}(M_2, e) \\ \wedge \\ E(M_1, e) \leq E(M_2, e) \end{cases}$$

$\rightarrow$ But radio abstraction still have to be preserved by composition with the MAC model ...

Abstraction : model composition



If, $\forall$ input $e \in K'$, $Out(Mac, e) \in K$ **then**

$$Radio_1 \preceq_K Radio_2 \Rightarrow Mac \parallel Radio_1 \preceq_{K'} Mac \parallel Radio_2$$

More generaly :

$$Out(M, K') \subseteq K \Rightarrow (M_1 \preceq_K M_2 \Rightarrow M \parallel M_1 \preceq_{K'} M \parallel M_2)$$

Abstraction : what to do next ?

- **decision procedure** for relation $\preceq_K$
  - "weighted" trace inclusion
  - enumerative vs symbolic techniques ?
- transpose this work in the **"asynchronous framework"**
  - general com. primitives, dense time, non-determinism
  - weighted simulation relation (e.g. *amortized simulation*)
  - simulation relations between LPTA
  - decision procedure
- **methodology**
  - how to guess correct contexts and abstract models ?
  - assume-guarantee paradigm for energy related properties ?

Overview

1. Wireless Sensor Networks
2. Design, programming and analysis techniques for WSNs
3. What has been done within VÉRIMAG ?
4. **Where we could go in a near future ?**

Some more open perspectives

WSN = relevant/challenging application domain for design and analysis techniques

- **Programing**
  - dedicated language (Gals systems, more abstract than nesC ?)
  - code generation for specific platforms
    (asics, lightweight run-time environment)
- **Analysis**
  - behavioural vs analytic techniques
    (probabilities, is worst case really an issue)
  - security properties
- **Next ?**
  - experiment plateform ?