Automatic Programming of Wireless Sensor Networks with Genetic Programming

Tales Heimfarth
Departamento de Ciência da Computação
Universidade Federal de Lavras

Renato R. R. de Oliveira, Ariel F. F. Marques
Raphael W. de Bettio, Jesimar S. Arantes

Sumário

The Node

- Low processing power CPU, small amount of memory
- Low power communication radio
- Constrained Energy
- Different sensors (temperature, GPS, humidity, movement, etc).

The Network

- High cost, low speed communication links
- Data-centric networking

Challenges

- Constrained nodes
- Low level programming languages
- Complex network topologies
- Massive distributed system

Approaches

- Higher level abstractions (operating system, middleware)
- Simulators

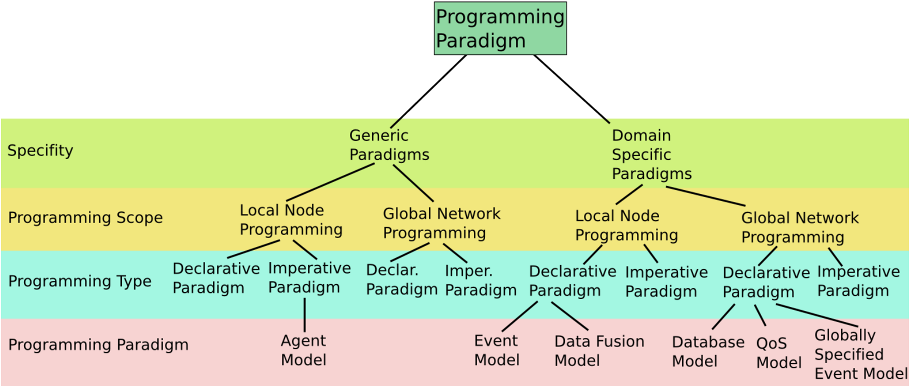

Figura: *Middleware* classification

Middleware

- Programming of the WSN as a aggregate, not as a set of individual nodes (middleware)
- Options:
  - Declarative Programming (Database model)
  - Imperative Programming (Agent insertion)

...nevertheless either application specific or requiring programming a massive distributed system

Related Work

Genetic Programming

The genetic programming is a collection of evolutionary computer techniques which aims to solve problems automatically [Poli et al. 2008]

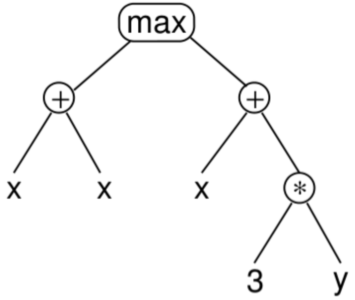

Figura: GP syntax tree representing $max(x + x, x + 3 * y)$. Source: [Koza 1992]

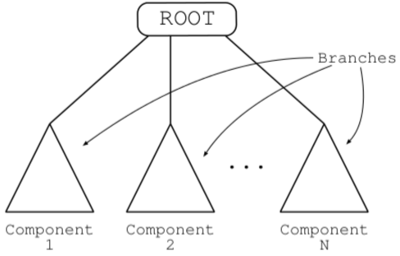

Figura: Multi-component program representation. Source: [Koza 1992]

Evolutionary approaches in general:

- WSN topology is optimized using GA to determine the radio power [Ferentinos and Tsiligiridis 2007].
- Construction of clusters with diverse parameters guided by GA [Turgut et al. 2002]
- The network density, its connectivity and energy consumption are optimized using a GA [Bhondekar et al. 2009]

Using GP:

- Automatizes the configuration of WSN for a tracking application. Based on gene network regulators [Markham and Trigoni 2011].
- Generates distributed algorithms with GP. Problems: MCD, node election [Weise 2006].

Objectives

To introduce a framework capable to generate automatically the source code of WSNs applications. Necessary developments:

- a *script* language for application description.
- a *middleware* with a virtual machine
- a simulation environment for the fitness function evaluation
- set of tools for evolving applications by means of genetic progamming

Metodology
Overview

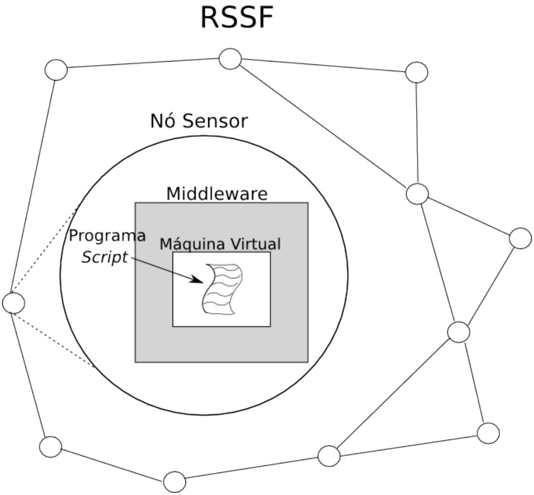

Figura: Proposed *Middleware*.

Metodology

Overview

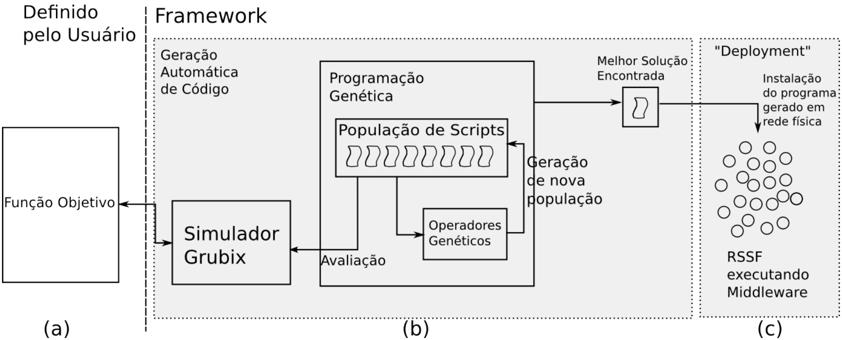

Figura: Proposed *framework*.

Metodology

Middleware architecture

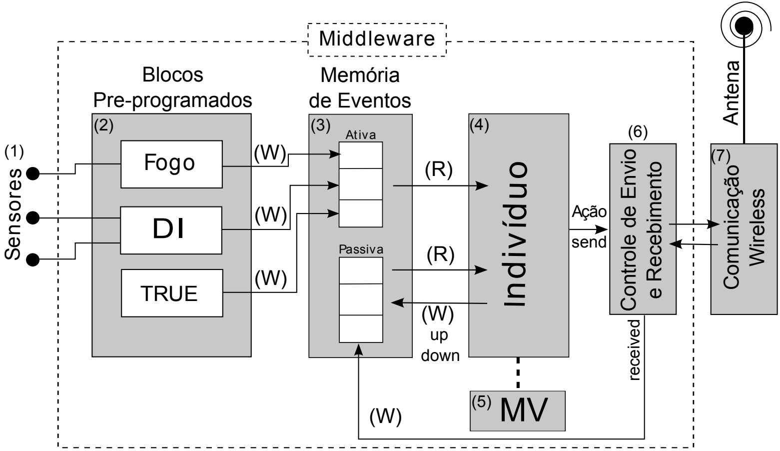

Figura: *Middleware* Architecture.

A1 **and** A2	P1 **or** A2	P3 **and** P2
up(P2)	down(P1)	up(P2)
down(P3)	send(P2,→)	down(P1)
send(P3,↑)		send(P1,↓)
down(P1)		

Figura: Example of *script*.

Metodology

Genetic Algorithm

```
begin
    initialize(population);
    simulationAndEvaluation(population);
    actualGeneration ← 1;
    while actualGeneration < totalGenerationsNumber do
        mattingPool ← ∅;
        insert(bestIndividual(population), mattingPool);
        for i ← 2 to numberOfIndividuals do
            (parent1, parent2) ← tournament(population);
            if probability(crossoverRate) then
                newIndividual ← crossover(parent1, parent2);
            else
                newIndividual ← parent1;
            if probability(mutationRate) then
                newIndividual ← mutation(newIndividual);
            insert(newIndividual, mattingPool);
        simulationAndEvaluation(mattingPool);
        population ← mattingPool;
        actualGeneration ← actualGeneration +1;
```

Metodology

Crossover

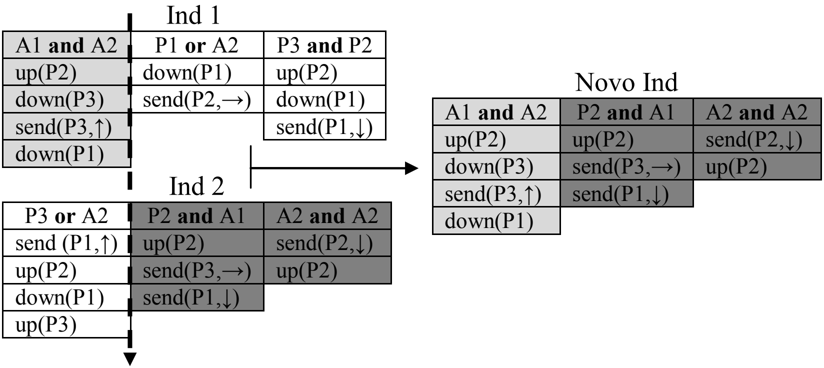

Figura: One point crossover in *triggers*

Metodology

Crossover

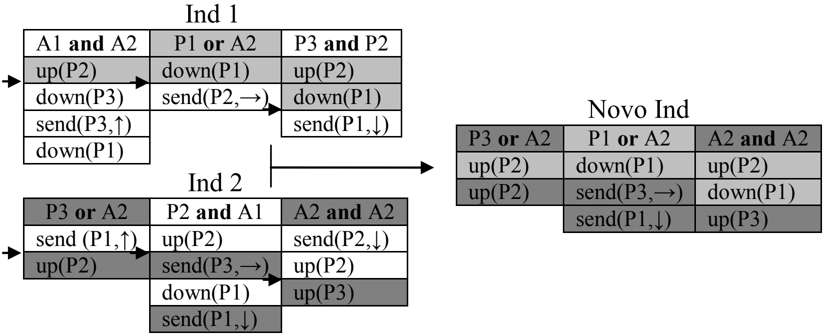

Figura: One point crossover in command

Metodology

Mutations

- Replace command: randomly select a command and replace it by another command randomly chosen.
- Replace list of commands: randomly select a trigger and replace all commands in its list.
- Insert: randomly select a command and insert it in another trigger randomly chosen.
- Swap commands: select two commands in the same trigger and swap them.
- Modify: randomly select a command and modify its parameters like events or destinations.
- Swap triggers: select randomly two triggers and swap them.
- Header: randomly select a trigger and modify its header parameters like logical operators or terms.

Metodology
Objetive Function

Parameters:

- C_{el} Cost of the events that didn't reached the sink node.
- C_{ms} Cost of messages sent.
- C_{pm} Cost of premature messages sent.
- C_{pa} Cost of premature actions.
- C_{mwd} Cost of messages sent to the wrong direction.
- C_{ed} Cost of the distance of the event from the sink node.

Simulation Data:

- EL Events lost. The number of occurred events that didn't reached the sink node.
- PM Premature messages. The number of sent messages before an event occur.
- MWD Messages sent to the wrong direction. The number of messages sent to the opposite direction of the sink node.
- ED Event distance from sink. The minimum distance from the sink node reached by the event.
- MS Messages sent. The total number of sent messages during the entire simulation.

$$
\begin{aligned}
Min\ F(\cdots) = \quad & C_{ms} \cdot MS + C_{el} \cdot EL + C_{pa} \cdot PA \\
+ \quad & PS + C_{pm} \cdot PM + C_{ed} \cdot (ED)^2 \\
+ \quad & C_{mwd} \cdot MWD \cdot EL
\end{aligned}
\tag{1}
$$

GRUBI

Event Detection Problem

Grid Topology

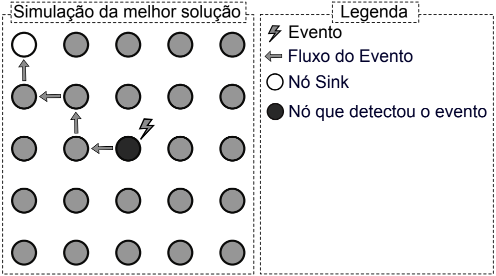

Figura: Example of the problem

Configurations

Tabela: Instances tested

Instance	Grid Size	Field Dimention	Range of Nodes	Area
25 nodes	5 x 5	40m x 40m	12m	1.600m^2
49 nodes	7 x 7	60m x 60m	12m	3.600m^2
225 nodes	15 x 15	140m x 140m	12m	19.600m^2
625 nodes	25 x 25	240m x 240m	12m	57.600m^2

Results

Tabela: Obtained Results

Number of nodes	Optimal fitness	Generations until Optimal	Time to Optimal	Total Time(s)
25	25	122	5,5	65,3
49	41	314,5	25,6	124,1
225	105	596,7	244,7	667
625	185	592,9	811	2.196,6

Conclusions

Conclusions

- A GP method for automatic programming of WSNs proposed
- For the problem presented, the method was capable of generating programs for networks with more than 500 nodes.
- The user just have to define the objective function

Further developments

- Different scenarios with random topology
- On the fly adjustments for autonomic computations (distributed GP)
- New problems, enhance the expression power of the script language

GRUBi

Referências

Bhondekar, A. P., Vig, R., Singla, M. L., Ghanshyam, C., and Kapur, P. (2009).
Genetic algorithm based node placement methodology for wireless sensor networks.
In *Proceedings of the International MulticConference of Engineers and Computer Scientists*.

Ferentinos, K. P. and Tsiligiridis, T. A. (2007).
Adaptive design optimization of wireless sensor networks using genetic algorithms.
Comput. Networks, 51:1031–1051.

Koza, J. R. (1992).
Genetic programming: on the programming of computers by means of natural selection.
MIT Press, Cambridge, MA, USA.

Markham, A. and Trigoni, N. (2011).