Programação Genética Aplicada à Geração Automatizada de Aplicações para Redes de Sensores sem Fio

Renato Resende Ribeiro de Oliveira

Departamento de Ciência da Computação
Universidade Federal de Lavras

24 de janeiro de 2014

GRUBI

Sumário

GRUBI

Contextualização

Redes de Sensores sem Fio

- Uma rede composta por vários **nós sensores**.
- Redes afetadas por características de sistemas massivamente distribuídos.
- Ambiente físico e arquitetura de rede muito dinâmicos e diversificados.

Nós Sensores

- Microprocessador de baixo poder de processamento.
- Pouca memória.
- Rádio de comunicação sem fio de baixa potência.
- Baterias pequenas (fonte de energia limitada).
- Sensores de ambiente (temperatura, umidade, pressão, entre outros).

Motivação

Desenvolvimento de Aplicações para RSSF

- Linguagens de programação de baixo nível.
- Controle de características específicas de hardware.
- Especialização do programa para cada nó sensor baseado em características como posição geográfica e o papel do nó sensor na rede.
- Implantação física do programa é complicada.

Programação Automatizada + Implantação Automatizada
$\rightarrow$ Custos e esforço humano reduzidos

GRUBI

Objetivos

Objetivo Geral

Desenvolver um *framework* capaz de gerar, de forma automática, o código-fonte de aplicações para RSSF.

Estrutura do Framework Proposto

- *Middleware* de RSSF:
  - Provê uma linguagem de programação em *scripts* de alto nível.
  - Provê funcionalidades pré-programadas.
- Um módulo de Programação Genética (PG):
  - Evolui programas descritos na linguagem de *scripts* provida pelo *middleware*.
- Um módulo de simulação de RSSF:
  - Simula e calcula o *fitness* dos programas gerados pela PG.
  - Realizada uma simulação simplificada e rápida.

Referencial Teórico

Middlewares de RSSF

- Abstraem características específicas de hardware.
- Simplificam o processo de desenvolvimento provendo de linguagens de programação de alto nível.
- Automatizam o processo de implantação das aplicações.
- Fornecem funcionalidades específicas pré-programadas.

GRUBI

Referencial Teórico

Programação Genética

- Algoritmos evolutivos aplicados à geração de programas de computador.
- Baseada na teoria de seleção natural.
- Cria e evolui uma população de indivíduos (programas).
- Utiliza rotinas de seleção, recombinação, mutação e avaliação.

GRUBI

Referencial Teórico

Simuladores de RSSF

- Simular ambiente físico.
- Simular características de hardware e comunicação sem fio.
- Simular movimento e dinamicidade da rede.
- Processo de alto custo computacional.
- Exemplos:
  - NS-3
  - GNS-3
  - GRUBIX

GRUBi

Trabalhos Relacionados

Algoritmos evolutivos em geral:

- Otimização da densidade, conectividade e consumo de energia utilizando algoritmo genético. [Bhondekar et al. 2009]

Utilizando Programação Genética:

- Automatização da configuração de aplicações de rastreamento através de Programação Genética e rede reguladora de genes. [Markham and Trigoni 2011]
- Geração de algoritmos distribuídos utilizando Programação Genética. [Weise 2006]

GRUBI

Arquitetura Geral

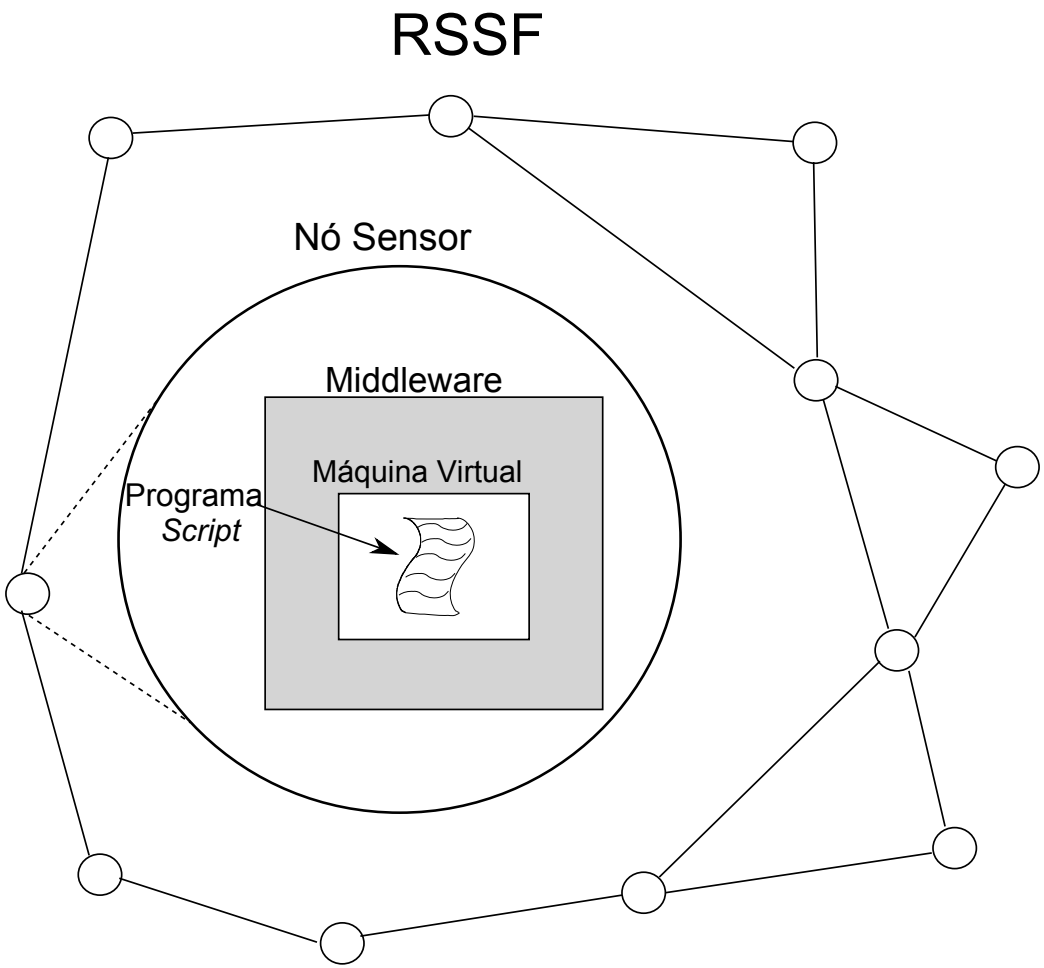


Figura : Ilustração esquemática de uma RSSF executando o sistema proposto.

GRUBi

System Overview

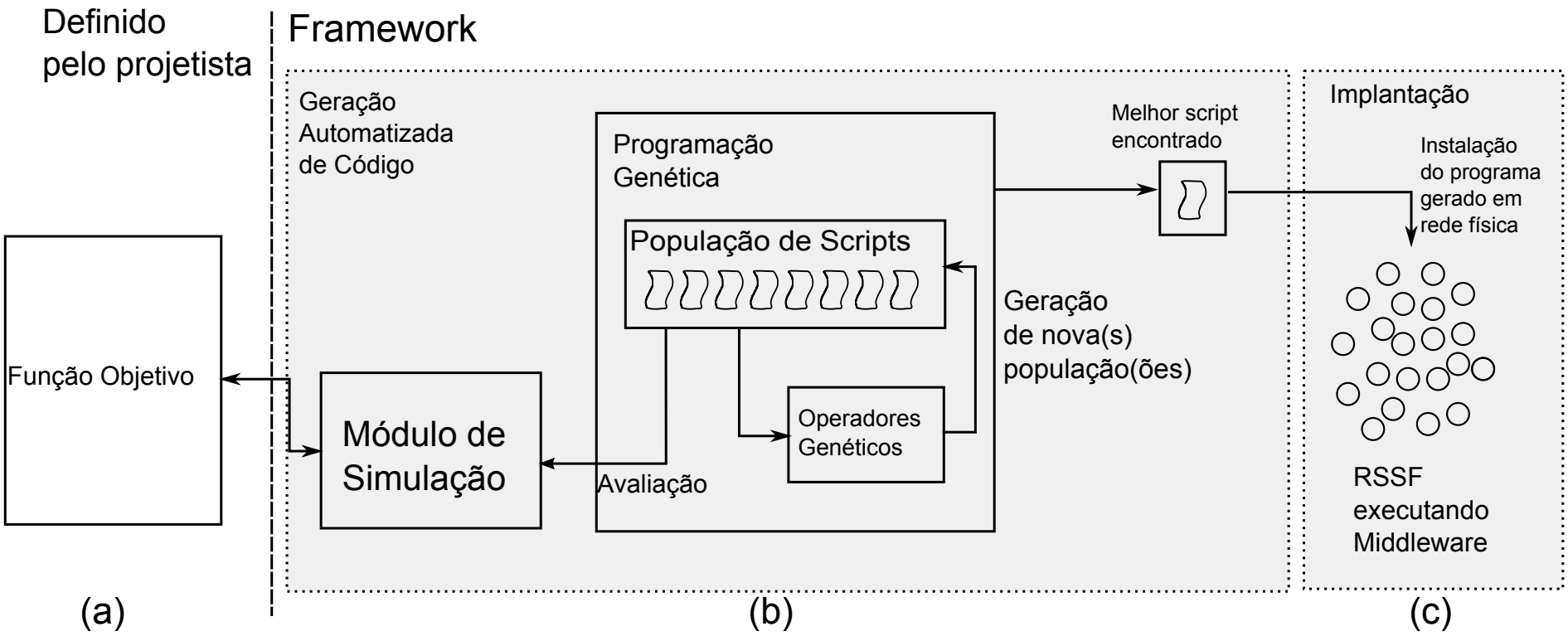


Figura : Visão geral da ferramenta proposta.

Problema de Detecção de Eventos (PDE)

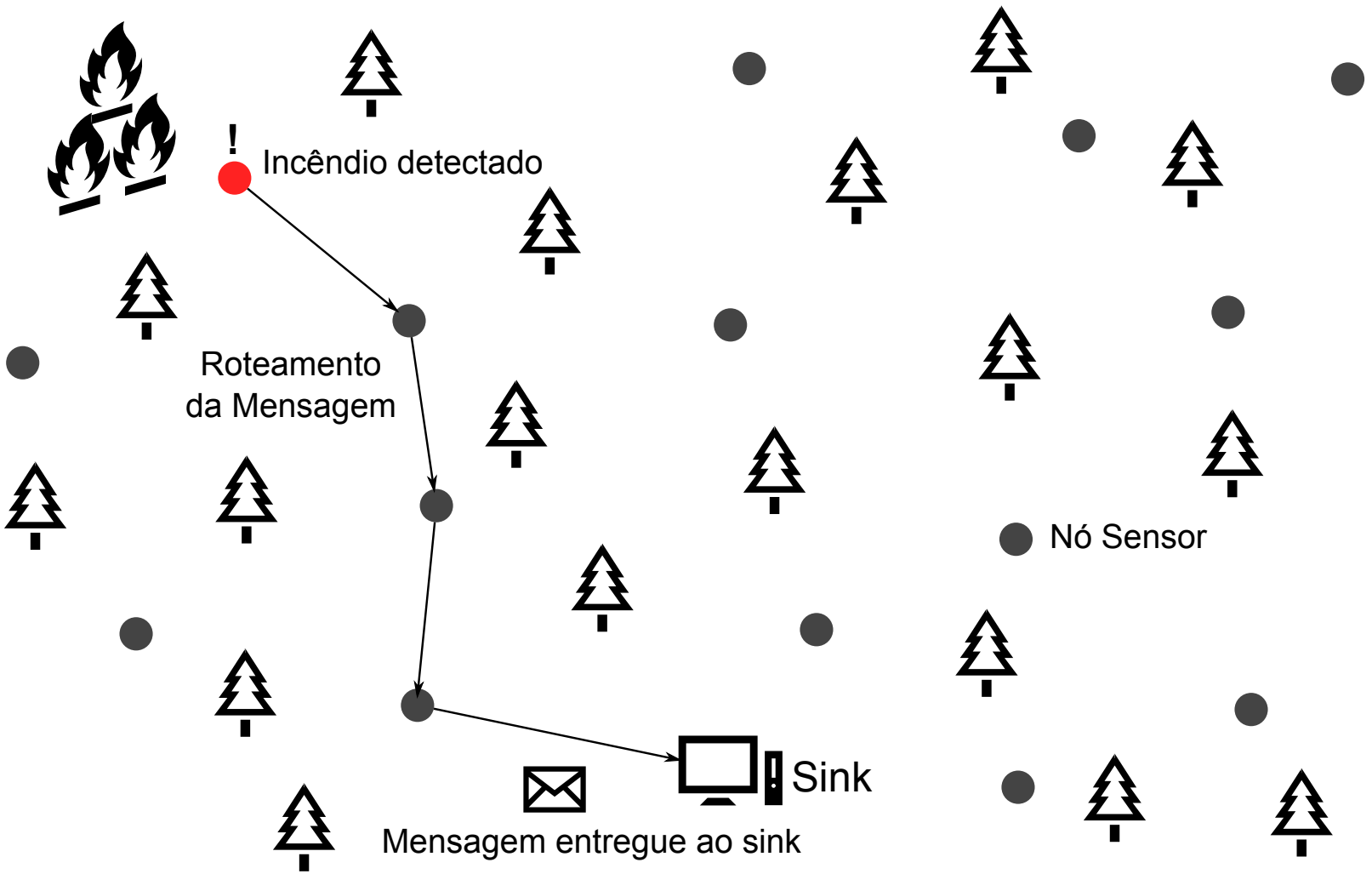
- Problema clássico.
- Avaliar a ferramenta proposta.
- Exemplo: Monitoramento de florestas para detecção de incêndios.

Características

- A rede fica "parada" (sem enviar informações entre os nós) realizando medições periódicas no ambiente.
- Quando um determinado comportamento do ambiente é detectado, um evento ocorre.
- A detecção do evento gera uma mensagem que deve ser enviada até o nó *sink* comunicando a ocorrência do evento.

GRUBI

Problema de Detecção de Eventos



GRUBi

Função Objetivo

Parâmetros da função objetivo.

C_{el}	Custo de eventos que não forem entregues ao *sink*.
C_{ms}	Custo do envio de mensagens.
C_{pm}	Custo do envio prematuro de mensagens.
C_{pa}	Custo de ações prematuras.
C_{mwd}	Custo de mensagens enviadas para direção incorreta.
C_{ed}	Custo da distância mínima entre a mensagem e o *sink*.

GRUBI

Função Objetivo

Dados de simulação utilizados pela função objetivo.

PS	Tamanho do *script* gerado.
EL	Número de eventos que não foram comunicados ao *sink*.
PM	Número de mensagens enviadas antes que algum evento ocorra.
PA	Número de ações executadas antes que um evento seja detectado.
MWD	Número de mensagens enviadas para a direção contrária do *sink*.
ED	Distância mínima entre a mensagem comunicando o evento do nó *sink*.
MS	Número total de mensagens enviadas durante a simulação.

GRUBI

Função Objetivo

Considerando estes parâmetros e dados de simulação, a função objetivo é definida a seguir na Equação 1.

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

GRUBI

Módulo de Simulação

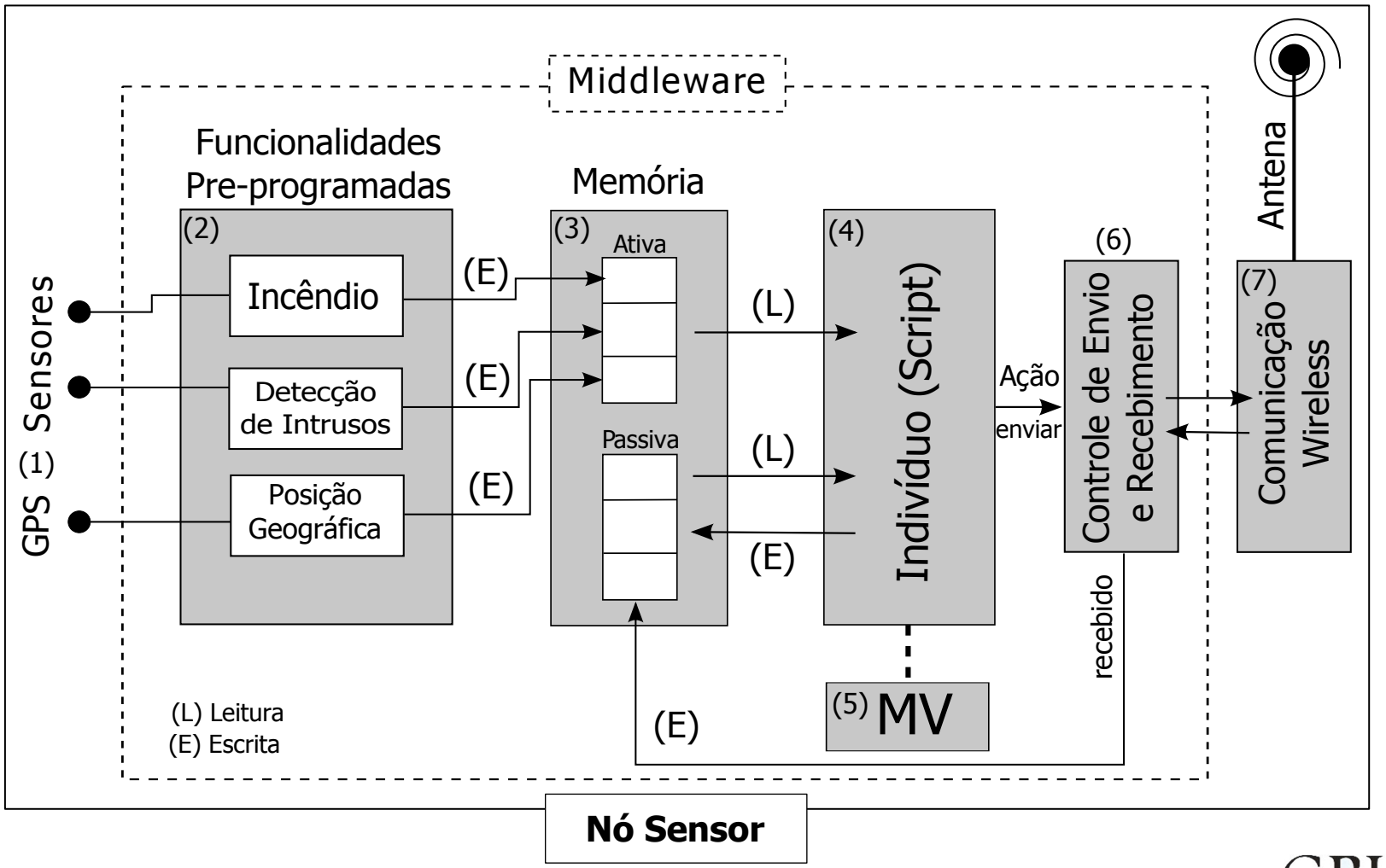
Pseudocódigo da execução do simulador.

```
Data: script
begin
    middleware ← Implementação simulada.;
    for node ∈ sensorNodes do
        node.instalaMiddleware(middleware);
        node.middleware.implantaScript(script);
        node.middleware.inicializa();
    for currentTime ← 1 até simulationMaxTime do
        for event ∈ eventTriggers do
            if event.ocorreu(currentTime) then
                sensorNodes.disparaEvento(event);
                eventTriggers.removeEvento(event);
        for node ∈ sensorNodes do
            node.middleware.executaScript();
```

GRUBI

Middleware Proposto

Arquitetura geral do middleware.



GRUBI

Middleware Proposto

Estruturas existentes na linguagem proposta.

Estrutura	Exemplo	Descrição
trigger	if(*ev*1 *op* *ev*2)	Estrutura condicional composta de dois operandos e um operador.
send command	send(*ev*,*dest*)	Envia um dado para um nó vizinho.
up event command	up(*ev*)	Guarda o valor *true* em um evento.
down event command	down(*ev*)	Guarda o valor *false* em um evento.

GRUBI

Middleware Proposto

Exemplo de script na linguagem proposta.

if *A1 and A2* **then**
- up(P2);
- down(P3);
- send(P3, *up*);
- down(P1);

if *P1 or A2* **then**
- down(P1);
- send(P2, *right*);

if *P3 and P2* **then**
- up(P2);
- down(P1);
- send(P1, *down*);

GRUBI

Programação Genética

Representação do indivíduo.

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)		

GRUBI

Programação Genética

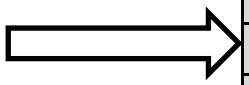
Operador de recombinação de um ponto em triggers.

Pai 1

A1 and A2	P1 or A2	P3 and P2
up(P2)	down(P1)	up(P2)
down(P3)	send(P2, right)	down(P1)
send(P3, up)	up(P3)	send(P1, down)
down(P1)		

Pai 2

P2 or A1	P3 and A1	P1 and A2
down(P2)	down(P1)	up(P1)
up(P3)	send(P3, left)	down(P3)
send(P1, down)	up(P2)	send(P4, up)
	down(P3)	



Filho

A1 and A2	P3 and A1	P1 and A2
up(P2)	down(P1)	up(P1)
down(P3)	send(P3, left)	down(P3)
send(P3, up)	up(P2)	send(P4, up)
down(P1)	down(P3)	

Programação Genética

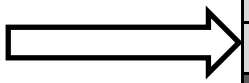
Operador de recombinação de um ponto em comandos.

Pai 1

A1 and A2	P1 or A2	P3 and P2
up(P2)	down(P1)	up(P2)
down(P3)	send(P2, right)	down(P1)
send(P3, up)	up(P3)	send(P1, down)
down(P1)		

Pai 2

P2 or A1	P3 and A1	P1 and A2
down(P2)	down(P1)	up(P1)
up(P3)	send(P3, left)	down(P3)
send(P1, down)	up(P2)	send(P4, up)
	down(P3)	



Filho

A1 and A2	P1 or A2	P1 and A2
up(P2)	down(P1)	up(P2)
down(P3)	send(P3, left)	down(P3)
send(P1, down)	up(P3)	send(P4, up)

Programação Genética

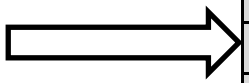
Operador de recombinação uniforme em trigger.

Pai 1

A1 and A2	P1 or A2	P3 and P2
up(P2)	down(P1)	up(P2)
down(P3)	send(P2, right)	down(P1)
send(P3, up)	up(P3)	send(P1, down)
down(P1)		

Pai 2

P2 or A1	P3 and A1	P1 and A2
down(P2)	down(P1)	up(P1)
up(P3)	send(P3, left)	down(P3)
send(P1, down)	up(P2)	send(P4, up)
	down(P3)	



Filho

A1 and A2	P3 and A1	P3 and P2
up(P2)	down(P1)	up(P2)
down(P3)	send(P3, left)	down(P1)
send(P3, up)	up(P2)	send(P1, down)
down(P1)	down(P3)	

Programação Genética

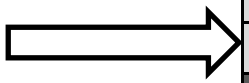
Operador de recombinação uniforme em comandos.

Pai 1

A1 and A2	P1 or A2	P3 and P2
up(P2)	down(P1)	up(P2)
down(P3)	send(P2, right)	down(P1)
send(P3, up)	up(P3)	send(P1, down)
down(P1)		

Pai 2

P2 or A1	P3 and A1	P1 and A2
down(P2)	down(P1)	up(P1)
up(P3)	send(P3, left)	down(P3)
send(P1, down)	up(P2)	send(P4, up)
	down(P3)	



Filho

A1 and A2	P1 or A2	P1 and A2
up(P2)	down(P1)	up(P2)
down(P3)	send(P3, left)	down(P3)
send(P1, down)	up(P3)	send(P1, down)
	down(P3)	

Programação Genética

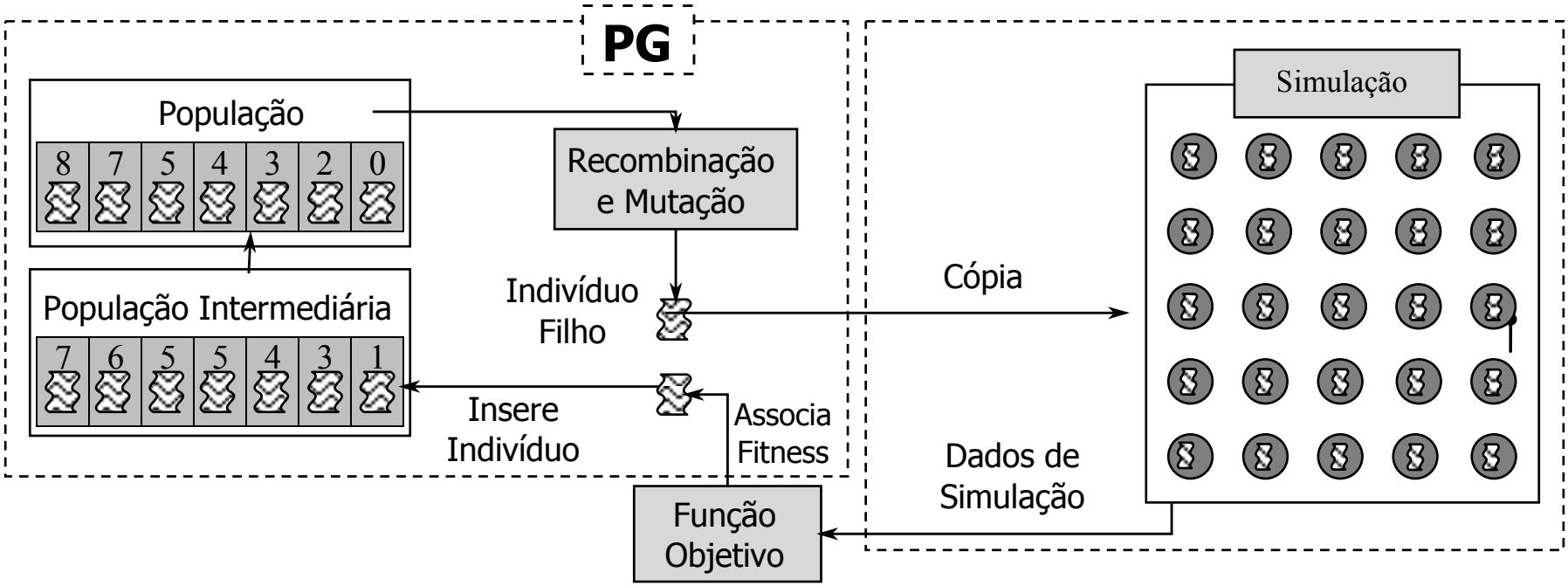
Operadores de mutação.

Substituir comando:	Sorteia um comando de forma aleatória e troca este por outro comando gerado também aleatoriamente.
Reiniciar comandos:	Faz o mesmo que "Substituir comando", porém ele substitui todos os comandos de um *trigger* escolhido aleatoriamente.
Remover e inserir:	Sorteia um comando em um *trigger* aleatório, remove o comando deste *trigger* e o insere em uma posição aleatória em outro *trigger*.
Trocar comandos:	Sorteia dois comandos dentro de um *trigger* aleatório e troca os dois de posição na lista de comandos deste *trigger*.
Alterar comando:	Sorteia um comando e modifica os parâmetros do mesmo, como eventos e destinos.
Trocar *triggers*:	Troca a ordem de dois *triggers* sorteados aleatoriamente na lista de *triggers* de um indivíduo.
Alterar cabeçalho:	Modifica os parâmetros do cabeçalho de um *trigger* sorteado aleatoriamente, alterando os operadores e operandos.

GRUBI

Programação Genética

Algoritmo evolutivo.



Experimentos Computacionais

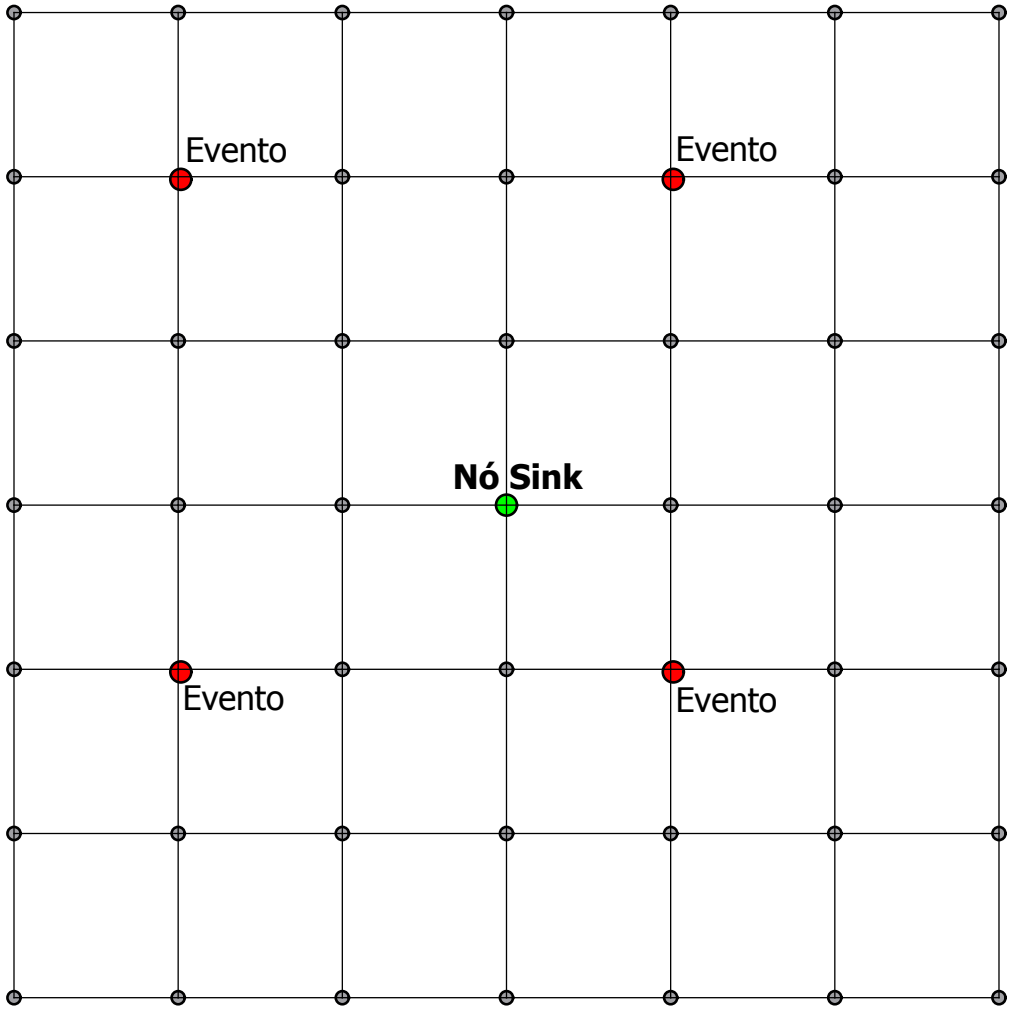
Instâncias de teste criadas.

Instância	Num. de Nós	Dimensões da Área	Topologia
G25	25	40m x 40m	Em Grade
G49	49	60m x 60m	Em Grade
G225	225	140m x 140m	Em Grade
G625	625	240m x 240m	Em Grade
R25	25	40m x 40m	Randômica
R49	49	60m x 60m	Randômica
R225	225	140m x 140m	Randômica
R625	625	240m x 240m	Randômica

GRUBI

Experimentos Computacionais

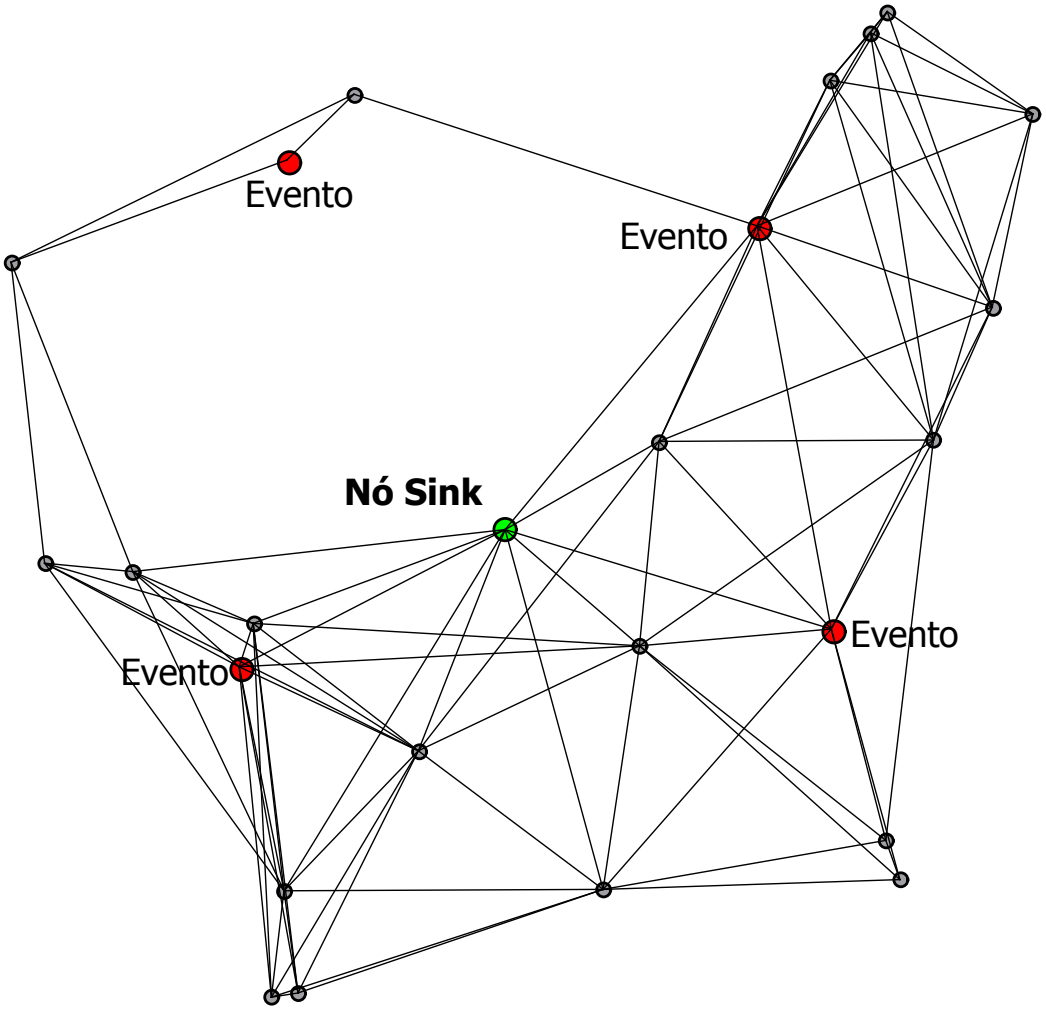
Ilustração da RSSF representada pela instância G49.



GRUBI

Experimentos Computacionais

Ilustração da RSSF representada pela instância R25.



GRUBI

Experimentos Computacionais

Resultados obtidos.

	Fitness da Melhor Solução			Coeficiente	Tempo
Instância	Média	Mínimo	Máximo	de Varia.	Total(s)
G25	22,6	20	27	13,55%	55,1
G49	29	29	29	0,00%	116,6
G225	61	61	61	0,00%	654,7
G625	101	101	101	0,00%	2.135
R25	32,9	30	53	22,22%	88,7
R49	84,8	43	189	70,75%	208,4
R225	1.157	376	2.722	77,74%	1.927,3
R625	2.756.2	461	4.384	47,27%	4.447,4

GRUBi

Experimentos Computacionais

Melhor script encontrado para a instância G49.

if *P0 and A3* **then**
  send(P0, down);
if *P0 and A1* **then**
  send(P0, up);
if *A2 and P0* **then**
  send(P0, right);
if *A4 and P0* **then**
  send(P0, left);
if *A0 or P0* **then**
  up(P0);

GRUBi

Conclusões e Trabalhos Futuros

Conclusões

- O *framework* mostrou-se promissor.
- O sistema é capaz de solucionar o PDE para redes com topologia em grade e topologia randômica.
- A ferramenta consegue soluções boas mesmo em redes grandes (mais de 600 nós sensores).
- Para topologias randômicas, o método é um pouco instável quando executado por apenas 2.000 gerações.

GRUBI

Conclusões e Trabalhos Futuros

Publicações

- 2013 IEEE Congress on Evolutionary Computation [de Oliveira et al. 2013a].
- Workshop on Software Technologies for Future Embedded and Ubiquitous Systems [Heimfarth et al. 2013].
- III Workshop de Sistemas Distribuídos Autonômicos [de Oliveira et al. 2013b].

Trabalhos Futuros

- Tratar ocorrência de mais de um tipo de evento simultaneamente.
- Realizar testes com nós sensores reais.
- Estender abordagem para tratar outros tipos de problemas de RSSF, como rastreamento de alvos.

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*.

de Oliveira, R. R. R., Heimfarth, T., de Bettio, R. W., Arantes, M. S., and Toledo, C. F. M. (2013a).
A genetic programming based approach to automatically generate wireless sensor networks applications.
In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 1771–1778, Cancún, México. IEEE Publishing.

GRUBi

Referências

de Oliveira, R. R. R., Heimfarth, T., Marques, A. F. F., de Bettio, R. W., and Arantes, J. S. (2013b).
Programação automática de redes de sensores sem fio utilizando programação genética.
In *III Workshop de Sistemas Distribuídos Autonômicos (WoSiDA)*, pages 45–48, Brasília, Brasil.

Heimfarth, T., de Oliveira, R. R. R., de Bettio, R. W., Marques, A. F. F., and Toledo, C. F. M. (2013).
Automatic generation and configuration of wireless sensor networks applications with genetic programming.
In *Software Technologies for Future Embedded and Ubiquitous Systems (SEUS), Workshop on*, Paddeborn, Germany. IEEE Publishing.

GRUBI

Referências

Markham, A. and Trigoni, N. (2011).
The automatic evolution of distributed controllers to configure sensor networks operation.
The Computer Journal, 54(3).

Weise, T. (2006).
Genetic programming for sensor networks.
Technical report, University of Kassel, University of Kassel.

GRUBI