MAURICIO DE OLIVEIRA BARROS

DIAGNÓSTICO EM NÍVEL DE SISTEMA PARA REDES DE SENSORES SEM FIO: UMA HEURÍSTICA

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.
Orientadora: Prof.ª Dra. Andréa Weber

CURITIBA

2015

MAURICIO DE OLIVEIRA BARROS

DIAGNÓSTICO EM NÍVEL DE SISTEMA PARA REDES DE SENSORES SEM FIO: UMA HEURÍSTICA

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.
Orientadora: Prof.ª Dra. Andréa Weber

CURITIBA

2015

Barros, Mauricio de Oliveira
Diagnóstico em nível de sistema para redes de sensores sem fio: uma heurística / Mauricio de Oliveira Barros. – Curitiba, 2015
83 f. : il.; tabs., grafs.

Dissertação (mestrado) – Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Informática.
Orientadora: Andrea Weber
Bibliografia: p.75-81

1. Redes de sensores sem fio. 2. Heuristica. Weber, Andrea. II. Título

CDD 004.68

Ministério da Educação
Universidade Federal do Paraná
Programa de Pós-Graduação em Informática

PARECER

Nós, abaixo assinados, membros da Banca Examinadora da defesa de Dissertação de Mestrado em Informática, do aluno Maurício de Oliveira Barros, avaliamos o trabalho intitulado, "Diagnóstico em Nível de Sistema para Redes de Sensores sem Fio: Uma Heurística", cuja defesa foi realizada no dia 27 de agosto de 2015, às 13:30 horas, no Departamento de Informática do Setor de Ciências Exatas da Universidade Federal do Paraná. Após a avaliação, decidimos pela:

(X) **aprovação** do candidato. ( ) **reprovação** do candidato.

Curitiba, 27 de agosto de 2015.

Profª. Dra. Andréa Weber
PPGInf - Orientadora

Prof. Dr. Linnyer Beatrys Ruiz
UEM – Membro Externo

Prof. Dr. André Luis Vignatti
PPGInf – Membro Interno

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PARANÁ
Setor de Ciências Exatas
Coordenação de Pós-Graduação em Informática

AGRADECIMENTOS

À Prof.ª Dr.ª Andréa Weber, minha orientadora, pela oportunidade de ingresso no programa de mestrado em informática da Universidade Federal no Paraná - UFPR, pela honra e confiança de trabalhar ao seu lado, pela paciência e disponibilidade de seu tempo em compartilhar seu conhecimento e pela impecável condução deste trabalho.

Ao colega Mestre Alexander Robert Kutzke, pela contribuição e paciência em compartilhar seus conhecimentos sobre o algoritmo ODTA desenvolvido em seu trabalho de dissertação, com os detalhes da heurística neste algoritmo até os experimentos executados e os resultados alcançados.

Aos meus pais, Maria Joanil e José Barros, por terem me educado para vida, ensinando o melhor caminho e por não terem deixado faltar nada em minha vida.

As minhas irmãs Mariana de Oliveira Barros e Fernanda de Oliveira Barros, que sempre me incentivavam e me confortavam com palavras fraternas para nunca desistir do meu objetivo.

À minha filha Amanda Caregnato de Oliveira Barros, que mesmo distante fisicamente é meu motivo para sempre continuar estudando.

Aos funcionários e professores do Programa de Pós-Graduação em Informática por todos ensinamentos passados.

Ao professor André Vignatti, que sempre esteve contribuindo com excelentes ideias na construção deste trabalho.

Aos professores Dr.º Elias Procópio Duarte Junior, Dr.º André Guedes, Dr.ª Michele Nogueira e Dr.º Aldri Luiz dos Santos pelos ensinamentos nesta longa jornada.

Ao amigo de mestrado Rodolfo Rodovalho, companheiro de disciplinas do curso, pela troca de experiências e por todos os momentos passados.

Ao amigo e companheiro Diego Henrique Pagani do Laboratório de Redes e Sistemas Distribuídos, que contribuiu com seus conhecimentos, experiência e sua paciência em ceder seu tempo no auxílio da execução das simulações no servidor do laboratório através das

máquinas virtuais.

Aos amigos Mestres Ivan Pires e Santiago Viertel no auxilio e nas dúvidas para execução dos experimentos deste trabalho, e também pelos momentos de descontração.

Ao casal Rui Barbosa e Noedi Lazarotto, que me acolheram em sua casa na cidade de Curitiba como se eu fosse um filho.

À Maraísa do Nascimento, que tive a oportunidade de conhecer na cidade de Curitiba - PR e descobrir uma pessoa fantástica e companheira de se conviver. Pela sua paciência e auxílio nas diversas leituras e contribuição deste trabalho. Pelos momentos de descontração e preocupação comigo.

A todos os colegas do mestrado e doutorado do DINF/UFPR, que tive a oportunidade de conhecer, conviver e trocar conhecimento e experiências.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

API Application Programming Interface

EETA Energy-Efficient Test Assignment

GLPK GNU Linear Programming Kit

GPS Global Positioning System

LEMON Library for Efficient Modeling and Optimization in Networks

MEMS Micro-Electro-Mechanical Systems

NAWMS Nonintrusive Autonomous Water Monitoring System

NPP Nuclear Power Plant

ODTA Optimal Design Test Assignment

PL Programação Linear

PLI Programação Linear Inteira

RSSF Rede de Sensores Sem Fio

SSCCR Set of Sensors Chosen by Centroid and Radius

TAWR Test Assignment Without Reciprocal Tests

WSN Wireless Sensor Network

RESUMO

Diagnóstico em nível de sistema é uma sub-área de tolerância a falhas. O objetivo de um algoritmo de diagnóstico em nível de sistema é reportar, para todas as unidades sem-falha de um sistema distribuído, o estado das demais unidades do sistema. A diagnosticabilidade do sistema depende de algumas propriedades topológicas do grafo de diagnóstico. Nesse contexto, um assinalamento de testes é um conjunto de testes mútuos entre as n unidades de um sistema. Um sistema com n unidades é dito t-diagnosticável se o número de unidades falhas não ultrapassar t e satisfizer as seguintes condições: (i) $n \geq 2t + 1$; e (ii) cada unidade for testada por, no mínimo, t outras unidades. Um sistema t-diagnosticável é definido como ótimo se $n = 2t + 1$. Considera-se o problema da definição de um assinalamento de testes para a identificação de nós com falha em uma Rede de Sensores Sem Fio - RSSF. Dado um conjunto de $2t + 1$ sensores, a abordagem *Optimal Design Testing Assignment* - ODTA [36] gera um assinalamento de testes ótimo do ponto de vista da diagnosticabilidade do sistema. Entretanto, o problema da escolha em termos da distância dos $2t + 1$ sensores que farão parte do assinalamento de testes tem características de um problema computacionalmente intratável. Devido à ausência de tal prova, apresenta-se o aprimoramento da heurística do ODTA para a escolha deste conjunto de sensores. Por meio da heurística Set of Sensors Chosen by Centroid and Radius - SSCCR apresentada, é possível selecionar em tempo polinomial tal conjunto não somente ótimo em termos de número de sensores, mas com uma considerável melhora dos resultados em termos de distância geográfica entre os sensores. Por fim, apresenta-se a comparação das duas heurísticas abordadas com a solução ótima obtida pela formulação do problema em programação linear inteira, na qual pode-se confirmar que a heurística SSCCR apresenta melhor desempenho em relação a heurística ODTA na escolha do conjunto de sensores com relação a distância entre eles e até mesmo, em algumas situações, pode proporcionar o alcance de valores ótimos e consequentemente obter a redução do consumo de energia na execução do diagnóstico de falhas.

ABSTRACT

System-level diagnosis is a subset of fault tolerance. The goal of a system-level diagnosis algorithm is to report the state of the units of a distributed system to all fault-free units of the system. The diagnosability of the system depends on some topological properties of the diagnostic graph. In this context, a test assignment is a set of mutual tests between n units of a system. A system with n units is called t-diagnosable if the number of faulty units does not exceed t and it satisfies the following conditions: (i) $n \geq 2t + 1$; and (ii) each unit is tested at least by t other units. A t-diagnosable system is said to be optimal if $n = 2t+1$. Consider the problem of defining a test assignment to identify faults in a wireless sensor network (WSN). Given a set of $2t + 1$ sensors, the approach *Optimal Design Testing Assignment* - ODTA [36] generates an optimal test assignment for the diagnosability of the system. However, the problem of the choice of $2t + 1$ sensors that will take part of the testing assignment has characteristics of a computationally intractable problem. Due to the absence of such proof, the improvement of ODTA heuristics is presented. According to the heuristic Set of Sensors Chosen by Centroid and Radius - SSCCR it is possible to select that set in polynomial time, optimal not only in terms of number of sensors, but with a considerable improvement of results in terms of geographical distance between the sensors. Finally, a comparison of the two heuristics with the optimal solution obtained by the problem formulated in integer linear programming is presented, which confirms that the heuristic SSCCR has better performance compared with ODTA heuristic, and in many cases achieves optimal values, and consequently achieve the reduction of energy consumption in the implementation of fault diagnosis.

CAPÍTULO 1

INTRODUÇÃO

Os avanços na área de tecnologia de informação, de redes de comunicação sem fio e nos projetos de sensores incentivaram e propagaram o uso das Redes de Sensores Sem Fio - RSSFs. Tais redes proporcionam interface entre o mundo físico e digital, e possibilitam o desenvolvimento de inúmeras aplicações em diferentes áreas, como no monitoramento para proteção de infraestruturas civis, monitoramento de habitat, agricultura de precisão, saúde, etc [34].

As RSSFs são caracterizadas como um tipo especial de redes *ad-hoc*, compostas por um conjunto finito de nós sensores, com limitações de *hardware*, interconectados por meio de componentes de rádio que proveem comunicação sem fio, com objetivo de monitorar, processar e analisar determinados tipos de eventos físicos em diferentes tipos de ambientes [1, 2, 40, 50]. As características do ambiente a ser monitorado têm um importante papel em determinar o tipo de implantação, topologia e o tamanho da rede. Por outro lado, as RSSFs têm diversas restrições com relação a recursos, tais como: energia limitada, dificuldade em recarga ou substituição de baterias, baixa banda, curto alcance de comunicação, e limitada capacidade de processamento e armazenamento de informações em cada nó [50].

Elas possuem vantagens com relação às redes sem fio *ad-hoc* convencionais, pois além da habilidade natural de cooperação entre os nós sensores, elas possuem características como rápida instalação, auto-organização e processamento inteligente. Contudo, essas características apresentam desafios para a implementação deste tipo de rede, como no projeto de *hardware*, desenvolvimento de aplicativos e protocolos de comunicação, de forma que se possa obter dados e serviços confiáveis, precisos, seguros e com eficiência energética suprindo as necessidades dos diversos tipos de aplicações [50].

Cada nó de uma RSSF é composto basicamente pela unidade de processamento,

memória, bateria e sensor. Um dispositivo de monitoramento pode possuir um ou diferentes tipos de sensores que são capazes de monitorar diferentes tipos de eventos físicos, tais como, temperatura, umidade, radiação no ambiente, pressão, velocidade, direção, movimento, luz, níveis de ruído, etc., permitindo uma enorme variedade de aplicações. Essas aplicações podem ser classificadas em cinco principais categorias: militar, ambiental, saúde, residencial e industrial [1, 2, 34].

No monitoramento de um determinado fenômeno físico por meio das RSSFs, sensores são capazes de emitir alarmes com informações relevantes sobre o evento monitorado. Essas informações são encaminhadas para uma unidade central com maior capacidade de processamento. A unidade central pode ser responsável por realizar a detecção de falsos positivos na rede, de maneira que problemas, falhas e prejuízos possam ser evitados. Neste trabalho, a abordagem escolhida para detectar a presença de falsos positivos em RSSFs é a de *diagnóstico em nível de sistema*, do inglês, *system-level diagnosis*. Diagnóstico em nível de sistema é uma área específica de tolerância a falhas, que visa reportar às unidades do sistema o estado, em termos da presença ou não de falhas, das demais unidades [5].

No modelo que inaugurou a área de *diagnóstico em nível de sistema*, conhecido por PMC, das iniciais dos nomes dos autores, Preparata, Metze e Chien [46], considera-se que um sistema consiste de n unidades independentes, não necessariamente idênticas, que não podem ser decompostas para efeitos de diagnóstico, capazes de realizar testes entre si.

Neste modelo, diz-se que um sistema com n unidades é t-diagnosticável se o número de unidades falhas não exceder a t e ele satisfizer as seguintes condições: $(c1)$ o número de unidades n do sistema deve ser maior ou igual a $2t + 1$, e $(c2)$ uma unidade deve ser testada por pelo menos t outras unidades. Um sistema que consiste de n unidades, é definido como *ótimo* do ponto de vista da diagnosticabilidade, se $n = 2t + 1$ [46].

Este trabalho considera uma RSSF composta por um conjunto de sensores, distribuídos de forma homogênea sobre um campo de sensoriamento, que tem como finalidade monitorar um fenômeno físico específico, e ser capaz de reportar mensagens de alarme para uma estação base. Com o objetivo de garantir que as mensagens de alarmes não sejam falsas, a estação base solicita um conjunto de testes entre os sensores presentes na região onde

o alarme foi gerado, e o resultado deste conjunto de testes, denominado de síndrome, é obtido e analisado por meio de um algoritmo de diagnóstico em nível de sistema, que identifica os sensores falhos. Espera-se que o assinalamento de testes do algoritmo tenha um número mínimo de sensores, e que a escolha dos sensores que participam do diagnóstico seja a melhor possível, de forma que o consumo de energia nos testes seja minimizado, permitindo assim, que a vida útil da rede possa ser prolongada.

Em [36], é apresentada uma abordagem denominada de *Optimal Design Test Assignment* (ODTA), responsável por obter um assinalamento ótimo em termos de números de sensores para realização dos testes. Em se tratando de comunicação sem fio, o consumo de energia em comunicação é proporcional à distância entre os nós. Com o objetivo de reduzir a energia consumida em comunicação entre os sensores que realizam os testes, uma heurística é utilizada no ODTA para escolha dos sensores que estejam próximos aos alarmes.

Neste trabalho, a heurística ODTA é aprimorada de forma a obter os sensores com a melhor posição geográfica em relação à distância entre eles. Como o impacto no consumo de energia despendida para a realização do diagnóstico é proporcional à distância entre os sensores, encontrar uma solução viável para esse problema é relevante. Encontrar um número mínimo de sensores, com a melhor posição geográfica para o assinalamento ótimo de testes, e que satisfaça as condições de diagnosticabilidade do sistema, tem características de ser um problema computacionalmente intratável [24].

Assim, devido a ausência de tal prova, é apresentada uma heurística, denominada Set of Sensors Chosen by Centroid and Radius - SSCCR para escolha deste conjunto de sensores com solução viável para o problema abordado neste trabalho. É realizada uma comparação do comportamento e dos resultados obtidos por meio das duas heurísticas, em relação ao ótimo obtido pela formulação do problema em programação linear inteira.

Este trabalho está organizado conforme descrito a seguir: o Capítulo 2 trata de Redes de Sensores Sem fio. No Capítulo 3 a área de diagnóstico em nível de sistema é abordada, e alguns dos trabalhos relacionados a essa pesquisa são apresentados. O Capítulo 4 apresenta a definição do problema e a descrição do algoritmo. No Capítulo 5 são apresentados

resultados experimentais. O Capítulo 6 conclui o trabalho.

CAPÍTULO 2

REDES DE SENSORES SEM FIO

As Redes de Sensores Sem Fio (RSSFs) emergiram há mais de uma década como uma das tecnologias mais promissoras para a área de monitoramento digital. Os avanços nos Sistemas Microeletromecânicos, tecnologia que compõe elementos mecânicos, sensores, comunicação sem fio, eletrônica digital em um pequeno chip, incentivaram o desenvolvimento de nós sensores inteligentes multifuncionais, com baixo custo de produção e consumo de energia eficiente [1, 2, 16, 50, 64]. No entanto, o projeto de redes de sensores sem fio apresenta enormes desafios, uma vez que a bagagem necessária de conhecimento abrange toda uma gama de tópicos na área de engenharia elétrica e ciência da computação [34].

Para Krishnamachari [34], as RSSFs representaram uma mudança fundamental de paradigma de comunicações pessoais inter-humanas tradicionais para as comunicações inter-dispositivos autônomos. Com as RSSFs são permitidas habilidades para observar e compreender em grande escala os fenômenos do mundo real, destinadas a fornecer informações sobre as características espaço-temporais (localização espacial de cada sensor de medição, tempo de medição) do mundo físico observado, e como resultado, têm também o potencial de gerar inovadores avanços tecnológicos.

Neste capítulo, abordaremos o conceito de RSSF, uma visão geral de suas características, limitações e aplicações em diversos ambientes.

2.1 Conceito de Redes de Sensores Sem Fio

RSSF é o conjunto de nós sensores, com restrições de *hardware*, interconectados por meio de seus componentes de comunicação, com a finalidade de monitorar (ou até mesmo reagir a) um determinado ambiente e processar esses dados com base no esforço de colaboração de uma considerável quantidade de nós. As RSSFs tem potencial em relação as rede de sensores tradicionais, os quais costumam ser implantados longe do fenômeno real,

de modo aleatório, ou com sua disposição topológica cuidadosamente planejada. Por outro lado, as RSSFs muitas vezes são densamente implantadas no interior de um fenômeno físico ou próximas a ele [1, 2, 47].

As RSSFs são um tipo especial de redes *ad-hoc*, uma subcategoria que também é projetada para um fim específico, onde não há uma infraestrutura de rede previamente instalada [7]. Redes *ad-hoc* são redes sem fio que permitem que os dispositivos possam criar e participar de redes em qualquer lugar, a qualquer hora, para praticamente qualquer aplicação, devido aos avanços de sofisticados protocolos, sistemas e experiências de implementação no mundo real [58].

Um nó sensor sem fio possui componentes de detecção, processamento, comunicação e capacidades de armazenamento [16]. Com isso, um nó sensor muitas vezes não é apenas responsável pela coleta de dados, mas também pela análise de rede, correlação e fusão de seus próprios dados de sensoriamento e dados de outros nós sensores. Embora um nó sensor sem fio seja equipado com dispositivos de detecção e de computação, transceptores de rádio e componentes de potência, os nós individuais em uma RSSF são recursos inerentemente limitados na velocidade de processamento, na capacidade de armazenamento, na largura de banda de comunicação e fonte de energia na forma de uma bateria limitada [40, 50]. A Figura 2.1 mostra os principais componentes básicos de uma RSSF.

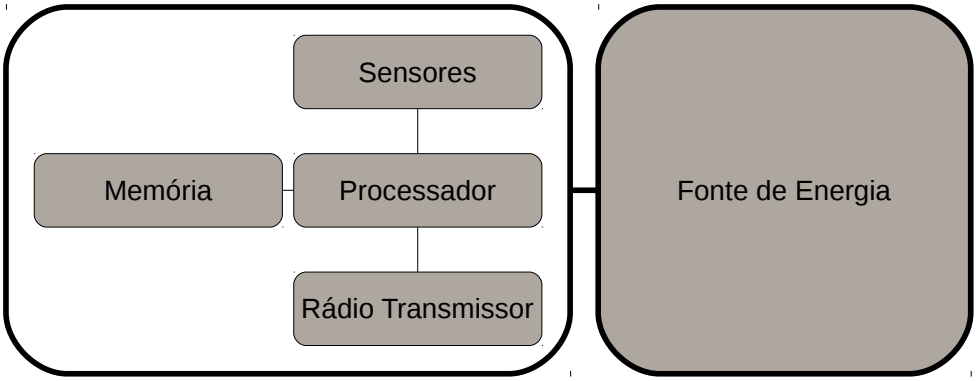


Figura 2.1: Esquema de um dispositivo de RSSF básico (adaptado de [34])

Os componentes básicos que constituem o hardware de um dispositivo de RSSF são: sensor, processador, memória, rádio transmissor e fonte de energia [34]. Com relação às características básicas:

O processador embutido no dispositivo possui restrições econômicas, no consumo de energia e no poder computacional, por exemplo, oito bits de 16 MHz. Devido a tais limitações no processamento, são utilizados sistemas operacionais específicos baseados em componentes, como por exemplo o TinyOS. A memória também possui limitações por considerações econômicas com relação a sua capacidade de armazenamento [34].

O rádio transmissor tem características de baixa taxa de transferência (10 - 100 kbps) e curto alcance (< 100m). É o componente que mais consome energia e que exige eficiência espectral, deve incorporar hibernação eficiente em termos de energia e modos de despertador. As características do componente sensor dependem da aplicação, e devido às limitações de largura de banda de comunicação e de potência é geralmente instalado um único tipo de sensor em cada dispositivo [34].

A fonte de energia na maioria dos dispositivos é concebida na forma de uma bateria finita (por exemplo, baterias AA Limh). É provável que este componente seja o gargalo mais crítico com relação aos recursos das aplicações em RSSFs [34]. Quando uma grande quantidade de sensores monitoram cooperativamente grandes ambientes físicos, eles formam uma RSSF. Nós sensores comunicam-se não só com os outros, mas também com uma estação base (conhecida como *sink*) usando seus rádios de comunicação sem fio, permitindo compartilhar seus dados de sensoriamento para processamento remoto, seja para visualização, análise ou armazenamento. A Figura 2.2 mostra três campos de sensores que monitoram três regiões geográficas diferentes e se conectam à Internet usando suas estações base.

As RSSFs são utilizadas para monitorar as condições físicas e ambientais, tais como temperatura, som, vibração, pressão, movimento ou poluentes. Transmitem cooperativamente os dados detectados por meio da rede para uma estação base (*sink)*, onde estes podem ser observados e analisados. Uma estação base atua como uma interface entre os usuários e a rede de sensores, ou seja, pode-se obter informações dos dispositivos de sensoriamento por meio de consultas aos resultados da estação base. Tipicamente uma rede de sensores sem fio possui centenas de milhares de nós sensores. Os nós sensores podem se comunicar entre si usando sinais de rádio [40].

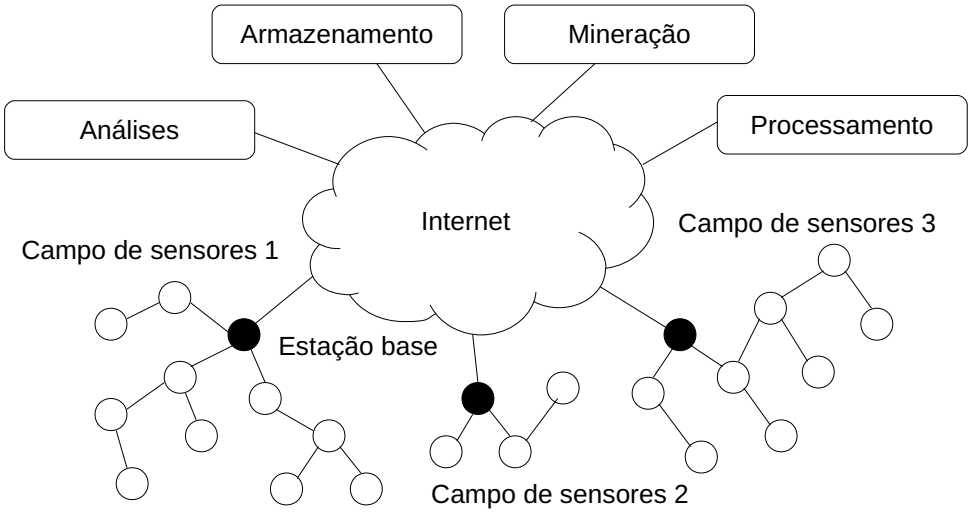


Figura 2.2: Modelo de Rede de Sensores Sem Fio - RSSF (adaptado de [16])

De acordo com Yick [64], as atuais RSSFs podem ser instaladas em terra, subsolo e debaixo d'água. Dependendo do ambiente, uma RSSF possui diferentes desafios e restrições. Existem cinco tipos de RSSFs: terrestre, subterrânea, subaquática, multimídia e móvel [64].

RSSFs terrestres consistem tipicamente de centenas ou milhares de nós sensores implantados em uma área de interesse, sejam em *ad-hoc* ou de forma pré-planejada. Em uma RSSF terrestre, a confiabilidade de comunicação é muito importante. Nós sensores devem ser capazes de transmitir efetivamente dados coletados para uma estação base.

RSSFs subterrâneas consistem de nós sensores que podem ser instalados no subsolo, em cavernas e minas para monitorar condições subterrâneas. As RSSFs subterrâneas são mais caras que as RSSFs terrestres, em termos de equipamentos, implantação e manutenção. O ambiente subterrâneo torna a comunicação wireless um desafio devido às perdas de sinal e alto nível de atenuação [64].

RSSFs subaquáticas são constituídas por nós sensores e veículos implantados debaixo d'água. Geralmente um número menor de sensores é implantado e seu custo de implantação também é mais elevado do que em RSSFs terrestres.

RSSFs multimídias permitem o monitoramento e rastreamento de eventos por meio de vídeos, áudio e imagens. São constituídas por nós sensores de baixo custo, equipados com

câmeras e microfones, e geralmente são implantados de forma pré-planejada no ambiente para garantir a cobertura desejada.

RSSFs móveis são constituídas por nós sensores que possuem a capacidade de se mover por conta própria e interagir com o ambiente físico. É possível a comunicação entre nós estáticos, mas a principal diferença é que nós móveis podem reposicionar-se e reorganizar-se na rede.

Nas RSSFs a maior preocupação é com a conservação de energia, sendo requerido o uso de protocolos de comunicação sofisticados e extremamente eficientes. Para cada tipo de aplicação de RSSF é exigida uma característica diferente entre os dispositivos de sensores e das redes. Em redes tradicionais com fio, não existe o problema com a conservação de energia, mas são consideradas algumas métricas de desempenho, tais como: tempo gasto pela rede para transportar um pacote do transmissor ao receptor, variação máxima do retardo do fluxo entre os pacotes, taxa de transmissão de pacotes que chegam e são entregues por unidade de tempo, taxas de perdas e taxas de erros [2].

2.2 Sensores e Sensoriamento

Segundo Dargie [16], sensoriamento é uma técnica utilizada para coletar informações do ambiente, seja um objeto físico ou um processo, incluindo a ocorrência de eventos, por exemplo a queda de temperatura ou pressão. Um dispositivo que executa a tarefa de monitorar é chamado de sensor. Um sensor é um dispositivo que traduz parâmetros ou eventos no mundo físico em sinais que podem ser medidos e analisados. Outro termo utilizado é transdutor, que tem como finalidade converter energia de uma forma para outra. Um sensor, portanto, é um tipo de transdutor que converte a energia do mundo físico em energia elétrica que pode ser enviada para um sistema de computação.

As capacidades dos nós sensores em uma RSSF podem variar amplamente, ou seja, nós sensores simples podem monitorar um único fenômeno físico, enquanto dispositivos mais complexos podem combinar diversas técnicas de sensoriamento (por exemplo, acústica, óptica, magnética). Os sensores também podem diferenciar-se em suas capacidades de comunicação, por exemplo, o uso de ultra-som, infravermelho, ou tecnologias de radio-

frequência com taxas de dados e latências variadas.

A Tabela 2.1 apresenta um exemplo de configuração de *hardware* de um sensor Mica2 Motes, com objetivo de ilustrar algumas das características de um nó sensor.

Tabela 2.1: Configuração de *hardware* de um nó sensor do tipo Mica2 Motes. Adaptado de [54])

Componentes	**Características**	
Rádio CC1000	Modulação: FSK, 300 a 1000 MHz, Taxa de transferência: 76,8 kbps	Tx: 16,5 mA (868 MHz), Rx 9,6 mA
Processador, Memória	ATMEGA128L, 8bits, 128 kbytes Flash,4 Kbytes RAM, ADC: 10 bits, 2.7 a 3,6V	active: 8mA, idle: 1,6 mA, sleep: 8 μA
Sensores	Térmico, áudio, acelerômetro, magnético	active: 5mA, sleep: 5
Fonte de energia	Não recarregáveis; insubstituíveis	Baterias recarregáveis; Substituíveis

Após o lançamento aleatório de nós sensores sobre o ambiente a ser monitorado, eles são responsáveis por uma auto-organização de infraestrutura de rede, na maioria das vezes por meio de uma comunicação multi-*hop* entre eles. Assim, as RSSFs são auto-configuradas, ou seja, detectam os nós sensores que estão dentro do alcance do sinal de rádio, realizam trocas de informações importantes para estabelecerem comunicação e formar uma topologia dinâmica da rede. Em seguida, os sensores iniciam a coleta de informações de interesse. Dispositivos de sensores sem fio também respondem a perguntas enviadas a partir da estação base para executar instruções específicas ou fornecer amostras de detecção.

Os nós sensores podem trabalhar de duas formas: contínua (monitoramento constante do fenômeno em que o objetivo é ter o máximo de precisão dos dados coletados) ou evento conduzido (o monitoramento do fenômeno pode ser realizado em momentos específicos, quando o sensor pode ser desligado após a detecção do evento, e religado no intervalo de tempo programado para uma nova detecção). Dispositivos de sensores sem fio podem ser equipados com atuadores para agir sob certas condições [40].

2.3 Características e Limitações das RSSFs

A habilidade natural em cooperar com outros nós sensores em uma RSSF traz muitas vantagens com relação às redes sem fio *ad-hoc* convencionais, incluindo a instalação rápida,

auto-organização e processamento inteligente. No entanto, as características de RSSF apresentam desafios no projeto de hardware, aplicativos e protocolos de comunicação, de forma que possa suprir as necessidades e desafios nos diversos tipos de aplicações imaginadas [50].

As características de RSSFs e suas diferenças com relação a outras redes sem fio, tais como *ad-hoc*, têm sido discutidas por vários pesquisadores [1, 2, 16, 33, 34, 35, 45, 50, 64]. A Tabela 2.2 apresenta de forma resumida as principais diferenças entre os dois tipos de rede sem fio, as quais trazem muitos desafios técnicos para concepção e implementação de tais sistemas.

Tabela 2.2: Diferenças entre RSSFs e Redes Ad-hoc Sem Fio Convencionais (Adaptado de [33])

Características	**RSSF**	**ad-hoc sem fio convencional**
Número de nós	centenas de milhares	moderada
Densidade dos nós	Alta	Relativamente baixa
Redundância de dados	Alta	Baixa
Fonte de energia	Não recarregáveis; insubstituíveis	Baterias recarregáveis; Substituíveis
Taxa de dados	Baixo; 1-100 kb/s	Alta
Mobilidade dos nós	Baixo	Pode ter alta mobilidade
Direção dos fluxos	Unilateral; sensores $\Rightarrow$ *sink*	Bidirecional; fluxos de ponta-a-ponta
Encaminhamento de pacote	Muitos para um; centrada em dados	Centrada em endereço de ponta-a-ponta
Natureza de consulta	Baseado no atributo	Baseado no nó
Disseminação de consulta	Transmissão	Salto por salto ou transmissão
Endereçamento	ID único não global	ID único global
Ciclo de trabalho ativo	Pode ficar abaixo de 1%	Alto

A arquitetura de rede *ad-hoc* é muito interessante para as RSSFs, por diversos motivos [33], tais como:

- a arquitetura *ad-hoc* supera as dificuldades de configuração de infraestrutura pré-determinada dos outros tipos de redes sem fio. Nas RSSFs a funcionalidade da rede não é afetada quando novos nós são inseridos ou removidos da rede;
- as redes *ad-hoc* podem atender aplicações específicas por ter uma característica de fácil adaptação;
- para alcançar a eficiência energética, ao invés de utilizar a comunicação de roteamento dos nós sensores com a estação base por meio de único salto (*single hop*), ou seja, de longo alcance, pode ser utilizada a comunicação de múltiplos saltos (*multi*

hop), ou seja, de curto alcance, com o intuito de diminuir a intensidade de propagação do sinal de rádio entre os sensores numa comunicação a curta distância, consequentemente reduzindo o consumo de energia da rede;

- esta arquitetura é altamente robusta para falhas de nós simples devido à redundância de nós e sua natureza distribuída.

A Tabela 2.3 apresenta importantes desafios e mecanismos necessários para o funcionamento esperado de uma RSSF. Nós sensores têm limitações de recursos, incluindo energia, memória e as capacidades computacionais. A fonte de energia limitada dos nós sensores na rede impõem restrições à vida útil da RSSF. O problema da escassez de recurso energético em cada nó sensor (e/ou da RSSF) pode ser contornado com o uso eficiente dos dispositivos de comunicação, processamento e memória, a fim de maximizar a vida útil da rede [50].

Tabela 2.3: Desafios em Relação aos Mecanismos Necessários em RSSFs (Adaptado de [50])

Desafios	**Mecanismos necessários**
Limitações de recursos	Uso eficiente dos recursos
Condições ambientais dinâmicas e extremas	Operação de rede adaptativa
Redundância de dados	Fusão de dados e processamento localizado
Comunicações sem fio não confiáveis	Confiança
Sem identificação global (ID) para nós sensores	Paradigma de comunicação centrada em dados
Nós são propensos a falhas	Tolerante a falhas
A implantação em larga escala	Baixo custo dos sensores de pequeno porte com a auto-configuração e auto-organização

Para Krishnamachari [34], uma característica importante da RSSF é a possibilidade de processamento inteligente na rede. Nós intermediários ao longo do caminho não agem apenas como encaminhadores de pacotes, mas também podem examinar e processar o conteúdo dos pacotes que passam por eles. Isso geralmente é feito com o propósito de compressão de dados ou para processamento de sinal para melhorar a qualidade das informações coletadas.

2.3.1 Requisitos para Redes de Sensores Sem Fio

Em [35, 45], discutem-se alguns requisitos característicos de um sistema composto por nós sensores sem fio. O sistema deve ser:

- **Tolerante a falhas:** o sistema deve ser robusto contra falha dos nós sensores (problemas com o *hardware* ou *software* do nó sensor, falta de energia, etc.).

- **Escalável:** o sistema deverá suportar um grande número de nós sensores para atender a diferentes aplicações.

- **Durável:** o tempo de vida de cada nó sensor define a sobrevivência da RSSF, a qual espera-se que seja longa o suficiente. Substituir os recursos de energia ou recarregar os milhares de nós na rede não é viável, portanto as funções executadas por cada nó sensor, como operações de monitoramento, processamento, comunicação devem ser realizadas de forma eficiente, a fim de economizar o máximo de energia.

- **Programável:** a reprogramação de nós sensores no campo de sensoriamento pode ser necessária para melhorar a flexibilidade da rede, mas é quase impossível.

- **Acessível:** utilizar dispositivos de baixo custo em função de que a rede é composta por milhares de nós sensores. Instalação e manutenção dos elementos do sistema também devem ser de baixo custo.

- **Tempo-real:** rigorosas restrições de tempo para sensoriamento, processamento e comunicação devem ser suportadas pela RSSF por estar fortemente relacionada com os fenômenos do mundo real. Por exemplo, em uma detecção em tempo real de um determinado evento, os estímulos são capturados, processados, identificados e os resultados encaminhados imediatamente por meio da rede em um intervalo mínimo de tempo.

- **Seguro:** tão importante quanto as outras características citadas, espera-se que os nós sensores sejam seguros com relação ao controle de acesso (evitar tentativas não autorizadas de acesso aos nós), integridade da mensagem (detectar e prevenir alterações não autorizadas na mensagem), confidencialidade (criptografar mensagens para que apenas os nós que possuem a chave secreta possam ouvir a mensagem) e proteção de *replay* na entrega de mensagens (proteger contra a interceptação e reutilização de mensagens autênticas, evitando a negação de serviço (DoS) e o ataque à

rede). Este recurso de proteção permite que os dispositivos possam verificar se todos os pacotes enviados foram recebidos anteriormente em sequência, caso contrário o dispositivo rejeita o pacote.

Os requisitos exigidos pela aplicação é que definem os objetivos para implantação de uma RSSF, que deve atender à necessidade do ambiente a ser monitorado, às características e às limitações dos recursos entre os nós sensores [35, 45].

2.4 Aplicações de Redes de Sensores Sem Fio

As redes de sensores podem consistir de diferentes tipos de sensores tais como sísmico, térmico, visual, infravermelho, acústico, de radar, dentre outros, que são capazes de monitorar uma grande variedade de condições ambientais que incluem: temperatura, umidade, pressão, velocidade, direção, movimento, luz, composição do solo, níveis de ruído, presença/ausência de certos tipos de objetos, e níveis de desgaste mecânico de peças acopladas no interior de uma máquina. Como consequência, uma enorme variedade de aplicações é possível [1, 47]. Muitas das aplicações existentes para RSSF ainda estão em pesquisa e desenvolvimento, tanto por meio das universidades como pela indústria [34]. Esta gama de aplicações pode ser classificada em cinco principais categorias: militar, ambiental, saúde, residencial e industrial [1, 2, 47].

Já para Yick [64], as RSSFs podem ser classificadas em duas categorias: monitoramento e rastreamento, conforme a Figura 2.3. Aplicações de monitoramento incluem monitoramento ambiental interno ou externo, saúde e bem estar, energia, localização de estoques, processos de automação em fábricas, sísmico e estrutural. Aplicações de rastreamento incluem rastreamento de objetos, animais, humanos, e veículos.

Explica Kuorilehto [35] que, independente do domínio da aplicação, as principais funcionalidades de uma RSSF são coletar e processar dados. O tipo de coleta de dados e o tratamento dessas informações dependem da aplicação. Apesar da variedade de aplicações e domínios, são identificadas quatro principais funções independentes do domínio de aplicação: monitoramento (determinar o valor de um parâmetro em um de-

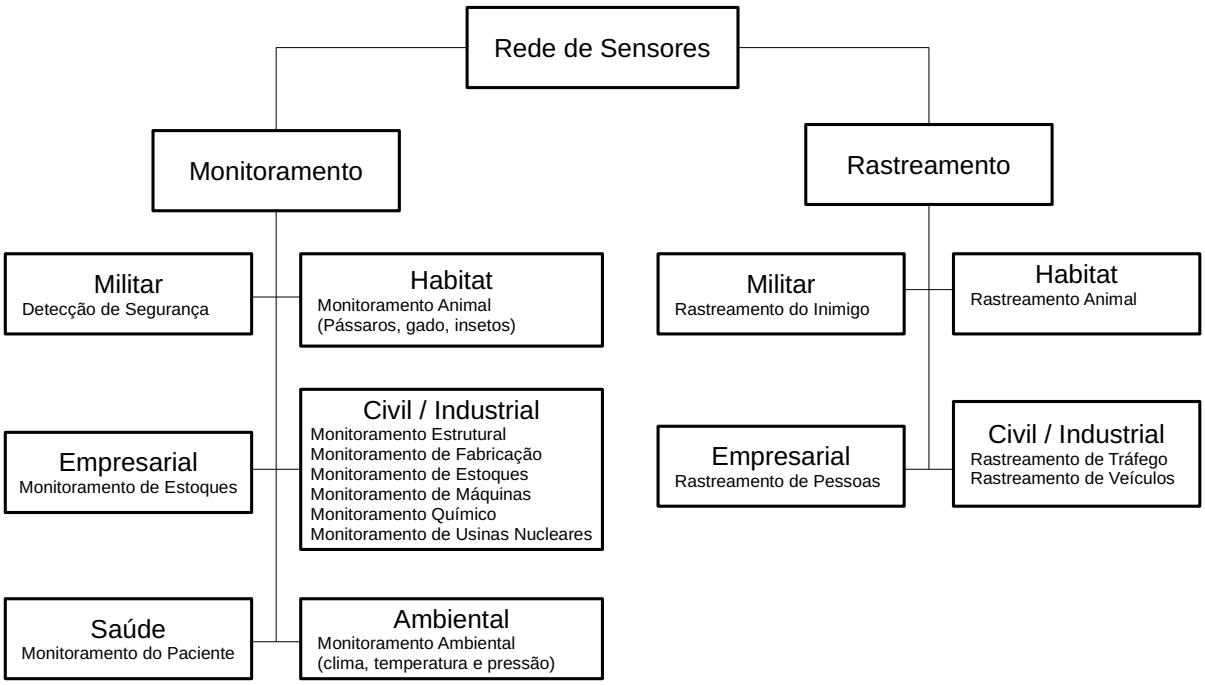


Figura 2.3: Visão geral das aplicações em sensores (adaptado de [64])

terminado local ou área de cobertura; normalmente a tarefa é executada por meio de medições periódicas); detecção por evento (detectar a ocorrência de eventos de interesse); classificação de objeto (identificar um objeto ou evento); e rastreamento de objetos (obter os traços de movimento e posição de um objeto móvel dentro da área de cobertura da rede).

Afirma Kuorilehto [35] que, além dessas quatro funcionalidades, RSSFs podem implementar o controle ou atuação sobre o ambiente monitorado. O controle pode ser integrado na rede ou controlado por uma estação base e/ou entidade central. O controle integrado toma as decisões sobre a atuação com base apenas nas informações locais disponíveis, enquanto a estação base e/ou entidade central pode explorar e analisar dados de toda a rede para tomar as decições e reagir com os atuadores. Porém, o controle integrado é efetivamente mais rápido, pois depende apenas dos dados locais e não de dados que são encaminhados por toda a rede.

Dentre os vários autores pesquisados [1, 2, 12, 22, 23, 26, 27, 30, 32, 35, 42, 50, 64], observamos as seguintes categorias de aplicações em RSSFs que são utilizadas em diferentes áreas como: aplicação militar, saúde, monitoramento residencial, segurança,

controle de tráfego, monitoramento ambiental, monitoramento comercial e industrial.

Nosso trabalho pode contemplar diferentes categorias de aplicações em RSSFs, mas citamos como exemplo específico o monitoramento de radiação em usinas nucleares. As RSSFs podem oferecer baixo custo na área de monitoramento dos níveis de radiação no ambiente interno e externo das instalações de usinas nucleares, sem necessitar de intervenção humana [27]. Também podem oferecer um sistema de alerta para detecção precoce de liberação de radiação [26]. Além disso, o grande volume de equipamentos que requerem monitoramento e a limitação de espaço suficiente nos conduítes em que os cabeamentos são instalados, criam oportunidades excepcionais para a utilização de RSSFs [30]. No entanto, preocupações com segurança, confiabilidade e interferência eletromagnética e de radiofrequência são as principais razões para a resistência do uso de RSSF em usinas nucleares [30]. Mais especificamente, o impacto da interferência eletromagnética e de radiofrequência nas instalações das usinas sobre o desempenho dos sistemas sem fio, e vice-versa, é uma preocupação que vem impedindo a adoção das RSSFs em plantas nucleares [30]. Contudo, indústrias nucleares começaram a fazer o uso de comunicação sem fio em plantas nucleares [38]

Usinas nucleares costumam ser próximas à costa e sensores próximos ao mar ficam expostos a maresia e propensos a erros. Falsos alarmes podem ser induzidos por meio do uso inadequado do sistema de detecção falhas e diagnóstico (FDD), conduzindo carga de trabalho de manutenção desnecessária ao gerar alarme e realização do diagnóstico para identificar falsos positivos [38].

Além disso, alertas antecipados de falhas incipientes permitem que medidas de correção sejam tomadas antes de situações críticas ocorrerem [38]. Uma rede neural pode também ser formada de tal maneira que ela obtém as medições instantâneas de um maior número de sensores como entradas, e diagnostica a causa do problema transiente. Este esquema é mais complexo, mas pode diagnosticar o problema em seu estágio inicial.

CAPÍTULO 3

DIAGNÓSTICO EM NÍVEL DE SISTEMA NO CONTEXTO DE REDES DE SENSORES SEM FIO

Neste capítulo, fazemos uma revisão das pesquisas desenvolvidas na área de diagnóstico em nível de sistema (do inglês, *system-level diagnosis*) no contexto de Redes de Sensores Sem Fio. Apresentamos o Modelo PMC que estreou a área de diagnóstico em nível de sistema, bem como alguns algoritmos propostos posteriormente, para redes representáveis por grafos completos e para redes de topologia arbitrária. São apresentadas algumas abordagens para diagnóstico em RSSFs, tais como: WSNDiag, TAWR, EETA e ODTA que buscam a tolerância a falhas bem como a eficiência energética.

3.1 Diagnóstico em Nível de Sistema

Considere-se um sistema composto de unidades autônomas e que não podem ser decompostas, não necessariamente iguais, que podem assumir dois estados de funcionamento: falho e sem-falha. O diagnóstico em nível de sistema tem objetivo de identificar de maneira precisa os estados de funcionamento dessas unidades.

O termo *dependability*, traduzido como "confiança no funcionamento", é a capacidade de um sistema oferecer um determinado serviço de forma confiável e segura, propriedade que permite aos usuários de um sistema computacional terem uma "confiança justificada no serviço fornecido pelo mesmo" [5].

Conforme Avizienis et al. [5], *dependability* é um conceito que vem sendo desenvolvido nas últimas três décadas e que engloba cinco atributos para um sistema oferecer serviços de modo confiável e seguro. São eles: *availability* (disponibilidade), *reliability* (confiabilidade), *safety* (segurança), *integrity* (integridade) e *maintainability* (capacidade de manutenção). A disponibilidade é a entrega do serviço correto de forma contínua. A confiabilidade é a capacidade do sistema fornecer corretamente o serviço. O atributo de

segurança é responsável por evitar consequências trágicas para o ambiente e/ou usuário. A integridade é o atributo que não permite alterações indevidas nos dados ou no sistema. Por último, a capacidade de manutenção é o atributo que permite que o sistema possa sofrer modificações e reparos.

Além dos atributos definidos, a caracterização das ameaças e os meios que garantem a *confiabilidade* são apresentados na Figura 3.1, com a taxonomia para a *confiabilidade* e a *segurança* em um sistema [5].

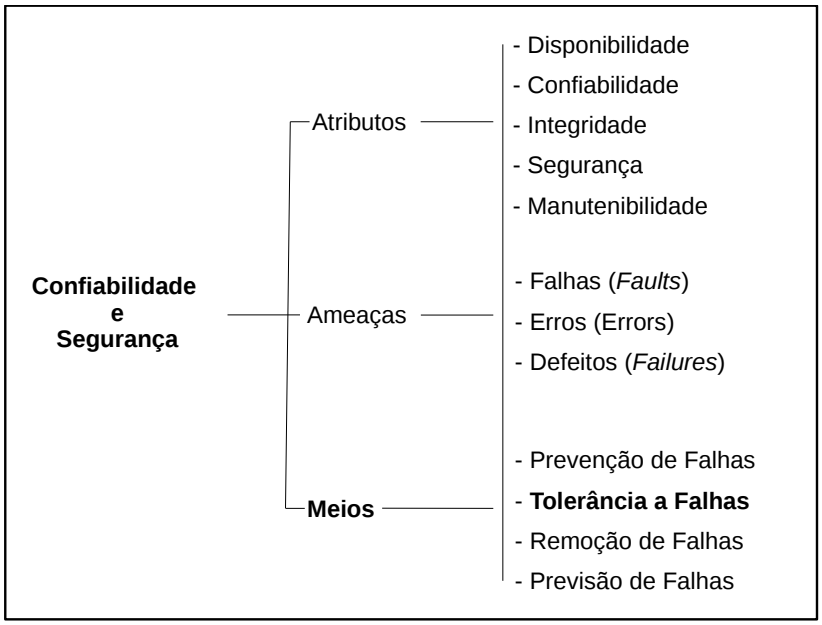


Figura 3.1: Árvore de confiabilidade e segurança [5]

De acordo com Avizienis [5], as ameaças são definidas em três níveis pelos termos em inglês *fault*, *error* e *failure*, traduzidas respectivamente como falha, erro e defeito. *Falhas* podem ocorrer tanto no *hardware* como no *software*, introduzidas de forma ingênua ou maliciosa, que podem ou não se manifestar, ou seja, podem estar *ativas* ou *dormentes*. O *erro* ocorre como a consequência da manifestação da falha, que pode ficar apenas interno ao sistema. O defeito ocorre devido à manifestação da falha e do erro na interface do sistema, apresentando resultados fora da especificação.

Em [3, 55] são classificadas as falhas em RSSFs em quatro tipos: nível do nó sensor (hardware e software), nível de rede, nível do sink e nível de aplicação. Este trabalho tem foco nas falhas em nível do nó sensor, em que as mesmas podem ocorrer a nível de

hardware, ou seja, nos componentes do nó sensor: bateria, memória, processador, rádio de comunicação ou sensores, e a nível de software, ou seja, roteamento, MAC e aquisição de dados.

A Figura 3.2 exemplifica a diferença entre falha, erro e defeito. Um serviço de sensoriamento em execução sobre o nó A aguarda periodicamente para enviar as informações de medição dos seus sensores para o nó B, que executa o serviço de agregação. No entanto, o nó A sofre um impacto acarretando um mau contato em um dos seus sensores. Uma vez que a implementação do código do serviço do nó A não foi concebido para detectar e superar tais situações, um estado não esperado ocorre quando o serviço do sensor tenta adquirir dados a partir do sensor. Devido a este estado, o serviço não envia dados do sensor para o serviço de agregação dentro do intervalo de tempo especificado. Essa situação resulta em um defeito de omissão ou *crash* do nó A observado pelo nó B [55].

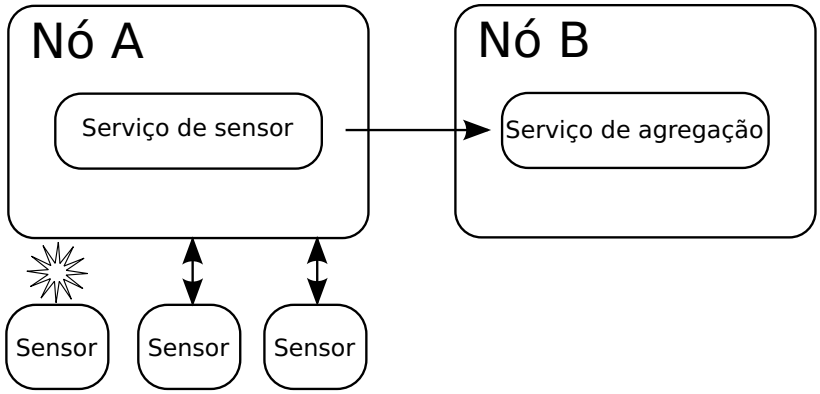


Figura 3.2: Defeito causado por um sensor com mau contato [traduzido de [55]]

No cenário exemplificado, a falha é o mau contato do sensor. O erro é o estado do serviço após tentar ler os dados do sensor e o defeito ocorre quando o aplicativo não envia os dados do sensor dentro do intervalo de tempo especificado [55].

Conforme Luciana Sá et al. [55], a confiabilidade de RSSFs pode ser afetada por falhas que podem ocorrer devido a várias razões, tais como, mal funcionamento do hardware, falhas de software, deslocamento indevido dos dispositivos no ambiente monitorado, ou riscos ambientais, como inundação ou fogo. Uma RSSF que não está preparada para lidar com tais situações de riscos pode consequentemente sofrer uma redução em seu tempo

de vida ou levar a consequências perigosas em contextos de aplicações críticas. Uma das principais técnicas de recuperação de falhas é a exploração de redundância, que muitas vezes é uma condição padrão em RSSF. Outra abordagem importante é a utilização do *sink* ou nós projetados especificamente para manter as operações após a ocorrência de falhas.

3.2 Tolerância a Falhas

Os meios para alcançar a confiabilidade e a segurança foram agrupadas em quatro técnicas: prevenção de falhas (*fault prevention*), tolerância a falhas (*fault tolerance*), remoção de falhas (*fault removal*) e previsão de falhas (*fault forecasting*). A técnica explorada por esse trabalho é a tolerância a falhas, que destina-se a evitar o fracasso, por meio da detecção de erros (*error detection*) e recuperação do sistema (*system recovery*). As técnicas de tolerância a falhas são apresentadas na Figura 3.3. A recuperação do sistema subdivide-se em duas técnicas: tratamento de erros (*error handling*) e tratamento de falhas (*fault handling*) [5].

Um nó sensor pode apresentar diferentes tipos de falhas, a nível de hardware ou software. Para manter o sistema tolerante a falhas é necessário detectar falsos positivos em um conjunto de sensores que geraram alarmes ou sob demanda para avaliar a acuidade dos sensores.

O tratamento de falhas, por sua vez, se inicia por meio do diagnóstico, cujo objetivo é "determinar as causas das falhas, em termos de localização e natureza". Em seguida são aplicadas as técnicas para recuperação do sistema: isolamento de falhas, reconfiguração e reinicialização do sistema [5].

O *diagnóstico em nível de sistema*, uma subárea de tolerância a falhas, é uma teoria que surgiu em 1967, no contexto de circuitos integrados. Nos últimos trinta anos passou a ser aplicada em sistemas distribuídos. O objetivo principal do diagnóstico é identificar quais unidades de um sistema estão falhas. Uma característica comum a todas as soluções de diagnóstico é que são baseadas em testes que são realizados por meio das unidades que compõem o próprio sistema [19].

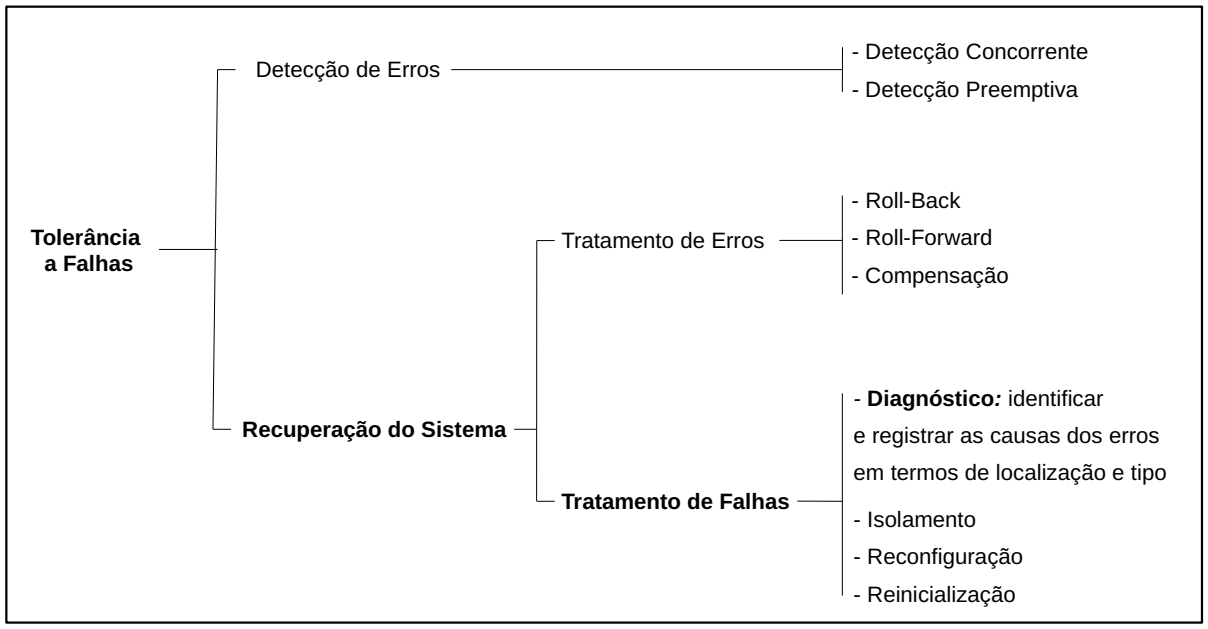


Figura 3.3: Técnicas de Tolerância a Falhas [5]

Para Duarte [19], existem limites óbvios ao diagnóstico que podem torná-lo impossível de ser realizado: se todas as unidades do sistema estiverem falhas ou se as unidades mudam rapidamente de estado (sistemas dinâmicos).

3.3 O Modelo PMC

No modelo que inaugurou a área de *diagnóstico em nível de sistema*, conhecido por PMC, o que corresponde às iniciais dos nomes dos autores, Preparata, Metze e Chien [46], considera-se um sistema S que consiste de n unidades independentes, não necessariamente idênticas, que não podem ser decompostas para fins de diagnóstico, capazes de realizar testes entre si.

De acordo com Preparata et al. [46], o diagnóstico correto pode ser alcançado por meio da detecção de qualquer padrão de falha diagnosticável, obtido por um determinado arranjo de enlaces de testes conhecido como assinalamento de testes (*connection assignment*), em que cada unidade do sistema S é responsável por testar um subconjunto de outras unidades. Assume-se que quando uma unidade testadora está sem-falha ela é capaz de determinar e reportar os resultados com precisão. Por outro lado, quando uma

unidade testadora está falha ela reportará resultados arbitrários.

Os testes realizados entre as unidades do sistema são a base do diagnóstico. O conjunto dos resultados de todos os testes é conhecido por *síndrome*. Os testes são realizados de modo distribuído, mas o diagnóstico é centralizado. O observador central tem a tarefa de coletar os resultados dos testes e efetuar o diagnóstico [46].

O sistema é representado por um grafo direcionado onde os vértices representam as unidades do sistema, e as arestas representam os enlaces de comunicação. Um assinalamento de testes existe entre as unidades u_i e u_j, se entre elas existir um enlace de comunicação, e u_i for designada para testar u_j [15]. A combinação dos resultados dos testes pode ser exibida como um grafo direcionado com pesos binários $a_{ij} = \{0, 1\}$. a_{ij} é igual a *zero* se, sob hipótese de que u_i não ser falho u_j também não é falho; a_{ij} é igual a *um* se, sob a mesma hipótese, u_j é falho. No caso em que u_i é falho, o resultado do teste é arbitrário e a_{ij} pode assumir *zero* ou *um*, independente da situação de u_j [46].

No modelo PMC, um sistema é diagnosticável desde que o número de unidades defeituosas não exceda um certo limite. Além disso, a chamada *diagnosticabilidade* do sistema está sujeita a algumas propriedades topológicas do grafo de diagnóstico. Mais especificamente, diz-se que um sistema de n unidades é t-diagnosticável se o número de unidades falhas não exceder a t. Além disso, ele deve satisfazer as seguintes condições: (i) : o número de unidades n do sistema deve ser maior ou igual a $2t + 1$, e (ii) : uma unidade deve ser testada por pelo menos t outras unidades. Um sistema S, que consiste de n unidades, é definido como *ótimo* do ponto de vista da diagnosticabilidade, se $n = 2t + 1$ [46].

Hakimi e Amin [28] provaram que, no modelo PMC, se duas unidades não se testam mutuamente, as condições (i) e (ii) são suficientes para a diagnosticabilidade do sistema. Por outro lado, em casos em que testes mútuos ocorrem, uma terceira condição deve ser satisfeita, para a qual um corolário é obtido: $(c3)$*:* seja G um grafo direcionado do sistema S com n-unidades, e $\kappa(G)$ a conectividade do grafo G (o número mínimo de vértices cuja remoção desconecta G); se $\kappa(G) \geq t$ então o sistema é t-diagnosticável [28, 29].

Muitas outras abordagens e algoritmos para diagnóstico em nível de sistema foram propostas ao longo dos anos [8, 9, 18, 48, 57], mas o fato do modelo PMC depender da

presença de um supervisor central para o processamento das informações de diagnóstico faz com que encontre aplicabilidade em RSSF devido à presença, naquelas redes, da estação base ou *sink*.

3.4 Algoritmos de Diagnóstico em Nível de Sistema

Na literatura encontram-se vários algoritmos para diagnóstico em redes representáveis por grafos completos e em redes de topologia arbitrária. O Modelo PMC [46] foi o primeiro algoritmo concebido na área de diagnóstico. Posteriormente, novos algoritmos foram propostos para essa finalidade, porém os mesmos devido a determinadas características tornam-se inviáveis para aplicação em RSSF.

3.4.1 Algoritmos de Diagnóstico para Redes Representáveis por Grafos Completos

Algoritmos de diagnóstico para redes representáveis por grafos completos pressupõe uma rede completamente conectada, ou seja, que cada sensor tenha acesso a todos os demais sensores da rede ponto-a-ponto, o que é impraticável em uma RSSF.

Diversos algoritmos com essas características foram concebidos: algoritmo Hakimi e Nakajima [29], New-SELF proposto por Hosseini et al. [31], Adaptive-DSD proposto por Bianchini e Buskens [10], Hi-ADSD proposto por Duarte Jr. e Nanya [18] e HeartbeatComplete apresentado por Subbiah e Blough [57]. Tais abordagens citadas são inexequíveis para a realidade de RSSFs, pois podem gerar sobrecargas de testes sobre uma única ou mais unidades do sistema, gerar alto número de troca de mensagens periodicamente por *broadcast* e alto consumo de energia.

3.4.2 Algoritmos para Diagnóstico em Redes de Topologia Arbitrária

Na literatura também podem ser encontrados algoritmos de diagnóstico para redes de topologia arbitrária. Porém, tais algoritmos tornam-se impraticáveis em RSSFs, pois os

mesmos executam testes periódicos na rede, e apresentam alto consumo de energia para realizar o diagnóstico.

O algoritmo de Bagchi e Hakimi [6], algoritmo Adpt, apresentado por Stahl et al. [56], Algoritmo RDZ, proposto por Rangarajan, Dahbura e Ziegler [48], Algoritmo NBND, concebido por Duarte Jr. [20], Algoritmo ForwardHeartbeat, elaborado por Subbiah e Blough [57], Algoritmo DNR, proposto por Duarte Jr., Weber e Fonseca [21], são algoritmos de diagnóstico concebidos para redes de topologia arbitrária, que devido as suas características na abordagem do diagnóstico tornam-se inviáveis para o problema abordado no contexto de RSSF apresentado neste trabalho.

3.5 Diagnóstico em RSSF

Em seguida são apresentadas algumas abordagens para diagnóstico em RSSF. Tais estratégias visam, de forma eficiente, gerar assinalamentos de testes com diagnosticabilidade suficiente para validar um conjunto de alarmes. As estratégias apresentadas garantem que, para um dado alarme gerado por t sensores, um grafo de testes no mínimo t-diagnosticável seja gerado, baseado em assinalamentos sem testes recíprocos e que buscam a economia de energia durante o processo de diagnóstico.

3.5.1 WSNDiag

Em [52], Chessa e Santi consideram o problema de identificar falhas do tipo *crash* em nós de uma RSSF. Este problema é de fundamental importância em cenários de RSSF onde a bateria pode ser substituída. As informações de diagnóstico obtidas por meio dos sensores podem ser utilizadas por um operador externo para promover o reparo e/ou reconfiguração da rede, estendendo sua vida útil. Os autores desenvolveram e analisaram um protocolo de diagnóstico de falhas especificamente para RSSF chamado de WSNDiag, provaram que o mesmo pode ser ótimo e eficiente em energia em algumas situações. No modelo adotado, as falhas do tipo *crash* são permanentes e sensores que apresentam esse tipo de falha não podem realizar qualquer tipo de comunicação.

O processo de diagnóstico é iniciado por meio de um sensor sem-falha, ou por meio de uma unidade externa, que solicita o diagnóstico. Assim, o diagnóstico é executado sob demanda, resultando em uma economia de energia para todo o sistema. O protocolo é capaz de diagnosticar corretamente um sistema com até t unidades falhas, onde $t < \kappa(G)$ e $\kappa(G)$ representa a conectividade do sistema. Dois tipos de mensagens são trocadas durante a sua execução, *I'm alive* - IMA e mensagens de diagnóstico. Uma árvore abrangendo todos os nós sensores sem-falha é construída durante a propagação de mensagens IMA. Após o tempo de espera, os nós sensores que não responderam com a sua mensagem IMA são diagnosticados como falhos [52].

Uma vez que um nó sensor tem visão local do diagnóstico, ou seja, do estado de seus vizinhos distantes de um salto, ele espera pelas informações locais dos seus filhos na árvore de busca. Uma vez que essas informações locais tenham sido recebidas, o nó sensor atualiza as informações e então seletivamente as envia para seu pai na árvore de busca. Quando a unidade externa da rede, ou o nó sensor que iniciou o diagnóstico receber as informações de diagnóstico de todos os seus filhos na árvore de busca, é então gerado o diagnóstico global da rede, por meio da combinação das informações de todos os seus filhos na árvore. O diagnóstico global é então disseminado para toda a árvore por meio de um protocolo de transmissão, a fim de garantir que cada nó sensor tenha as informações de todos os sensores que estão sem-falha na rede [52].

Conforme Chessa e Santi [52], esse problema de identificar as falhas do tipo *crash* em nós de um sistema distribuído tem sido exaustivamente estudado na literatura [6, 11, 31], mas no âmbito de redes de computadores com fio baseadas no paradigma de comunicação um-para-um, onde o consumo de energia não é considerado um problema. Por outro lado, em RSSF, onde o paradigma de comunicação é de um-para-muitos, em que um nó transmite, e todos os sensores dentro de seu alcance podem receber a mensagem, exige-se um protocolo de comunicação que seja eficiente no consumo de energia, independente da possibilidade de troca das baterias, pois sua substituição ainda é uma operação difícil e cara, sendo assim, o consumo de energia ainda é considerado um problema. Consequentemente, os protocolos apresentados em [6, 11, 31] tornam-se inviáveis ou ineficientes, em

virtude do alto consumo de energia quando aplicados em RSSFs.

No protocolo WSNDiag, nós sem-falhas executam a agregação das informações de diagnóstico com objetivo de reduzir a redundância de dados e evitar que as mensagens de diagnóstico sejam perdidas no caminho até o nó que iniciou o diagnóstico. O protocolo possui característica de obter o diagnóstico global do sistema por meio da troca de mensagens que são realizadas por uma árvore de busca. Tal característica gera um desafio no objetivo em tentar equilibrar a sobrecarga de trabalho entre os nós da rede, a fim de evitar que os nós sobrecarregados esgotem rapidamente a energia das baterias. O nó responsável por iniciar o diagnóstico, assim como os nós com mais filhos na árvore de busca, consomem mais energia em comparação ao restante dos nós da rede. É utilizada uma estratégia para reduzir esse problema de sobrecarga, de forma que na construção da árvore de busca, cada nó seja limitado ao grau cinco ou mesmo podendo ser personalizado. No entanto, tal estratégia implica no aumento de latência do diagnóstico.

Embora o WSNDiag seja eficiente em termos de consumo de energia, ele faz o diagnóstico de toda a RSSF, o que pode não ser viável em algumas situações.

3.5.2 TAWR - Test Assignment Without Reciprocal Tests

Em [60], é apresentada uma estratégia para o assinalamento de testes entre os sensores de uma RSSF, a fim de detectar alarmes falsos (falsos positivos). Dada uma determinada região de uma RSSF em que t alarmes tenham sido disparados, foi desenvolvida uma estratégia de testes entre os sensores desta área, de tal modo que o grafo do sistema que representa a região de alarmes garanta os níveis desejáveis de diagnosticabilidade. Foi adotado um modelo de diagnóstico em nível de sistema, em que todos os sensores na região onde os alarmes foram disparados cooperam para executar testes mútuos. Os resultados dos testes são passados como entrada para o algoritmo de diagnóstico que identifica os sensores falhos.

A estratégia de testes considera um modelo em que os sensores são implantados em uma área de monitoramento com distribuição uniforme, e que a topologia da rede é conhecida pelo *sink*. Assume-se que cada sensor conhece suas coordenadas geográficas.

A RSSF é modelada como um sistema representado por um grafo $G = (V, E)$ onde cada vértice v em V representa um sensor e uma aresta (v_i, v_j) existe se e somente se v_i e v_j estão dentro do alcance de transmissão entre si. Alarmes são disparados na ocorrência de qualquer evento anormal durante o sensoriamento, e tais informações de alarmes são encaminhadas para o *sink*. Assume-se que um conjunto T (de cardinalidade t) de sensores dispararam alarmes, e então, o algoritmo desenha um retângulo mínimo R que contém os sensores de T. O conjunto de sensores em R é definido por V_R e por n a cardinalidade de V_R.

Assume-se que a região R é a menor área retangular que contém todos os sensores que enviaram alarmes para o *sink*. Assume-se que o alcance do raio de transmissão dos sensores é de no máximo o tamanho da região. A fim de definir o conjunto de testes, o algoritmo proposto divide a região R em quatro quadrantes de mesmo tamanho. Sensores presentes em um mesmo quadrante não executam testes recíprocos. Cada quadrante possui dois quadrantes vizinhos, que, em sentido anti-horário, são chamados de quadrantes predecessor e sucessor. Os sensores em um determinado quadrante solicitam por testes em seu quadrante sucessor, e eles são solicitados a testar sensores em seu quadrante predecessor.

A fim de que o grafo de diagnóstico D seja t-diagnosticável, as condições $(i) : n \geq 2t+1$ e (ii) : o grau de cada vértice em D deve ser no mínimo t, ou, em outras palavras, cada sensor é testado por no mínimo t outros sensores, devem ser satisfeitas.

Em resposta a um alarme disparado pelos sensores em T, o *sink* primeiro verifica a veracidade dos alarmes enviando um conjunto de testes mútuos entres os sensores na região R. Um teste consiste de um conjunto de estímulos que são produzidos por um sensor testador v_i e que então é enviado para um sensor a ser testado v_j. Por sua vez v_j produz um resultado do teste que é devolvido para v_i. Finalmente v_i compara os resultados produzidos por v_j com o resultado esperado e então é gerado um resultado binário do teste: é 0, se o sensor testado está sem-falha e 1 caso contrário. Baseado no modelo PMC, assume-se que os resultados de testes realizados por um sensor sem-falha são sempre confiáveis, e completamente incertos se um nó testador é falho. Todos os

resultados dos testes são finalmente coletados pelo *sink* e decodificados por meio de um algoritmo de diagnóstico.

Os autores realizam experimentos de simulação e mostram que a estratégia satisfaz à condição de assegurar que cada nó na rede seja testado por no mínimo t sensores, embora com alta cardinalidade de nós ou com longo alcance de transmissão.

3.5.3 EETA - Energy-Efficient Test Assignment

Em [61] é apresentada uma evolução da abordagem anterior, a qual preocupa-se basicamente com a redução do número de testes requeridos para o processo de diagnóstico, e consequentemente com a redução do custo energético utilizado em tal processo. Para isso, foram consideradas duas questões: a escolha do conjunto de sensores para participar do processo de diagnóstico e a distância geográfica entre os sensores. A energia gasta em comunicação entre os sensores em um teste é diretamente proporcional à distância geográfica entre os sensores, assim quanto mais próximos os sensores estiverem entre si, menor o custo energético [44, 49].

Por sua vez, essa abordagem, denominada *EETA*, é baseada na escolha de um conjunto de sensores para produzir um assinalamento de testes, definido pelo *sink*, de menor custo energético que na abordagem TAWR. O *sink* após receber as informações de alarmes gerados pelos sensores, executa o processo de formação do grafo de testes. É identificada uma área geográfica retangular, denotada como R, que contém todos os sensores que geraram alarmes. Os sensores que participarão do diagnóstico são escolhidos como os mais próximos do centro da região R, de forma que os sensores selecionados para os testes fiquem próximos aos alarmes gerados.

Na abordagem EETA também é considerada uma RSSF composta por sensores implantados em uma área de monitoramento com uma distribuição uniforme e assume-se que o observador central conhece a topologia da rede. Cada sensor conhece suas coordenadas geográficas dentro da área de monitoramento, e a mesma também é conhecida pelo observador central. A rede é modelada como um grafo $G = (V, E)$, onde cada vértice $v \in V$ representa um sensor e uma aresta $(v_i, v_j) \in E$, se e somente se, v_i e $v_j \in V$ e os

mesmos estão dentro do alcance de transmissão.

O observador central é responsável por definir um assinalamento de testes para ser usado pelos sensores para realizar os testes. O assinalamento é um grafo de teste $D = (V_D, E_D)$, onde $V_D \subset V$ e $E_D \subset E$, e uma aresta $(v_i, v_j) \in E_D$, se e somente se, v_i testa v_j. Define-se n como a cardinalidade de V_D.

Foi implementado com base na abordagem TAWR [60], o método de dividir em quadrantes a região retangular R que envolve os sensores que geraram alarmes. O centro da região R, denotado por R_C, é utilizado como referência para a escolha dos sensores mais próximos, não pertencentes ao conjunto que gerou alarmes e que vão participar do diagnóstico. O procedimento garante que os sensores de um determinado quadrante testem sensores de outro quadrante de forma cíclica, de maneira que testes recíprocos sejam evitados. Uma heurística é utilizada para trocar sensores com custos mais elevados de testes por outros sensores com custos de testes reduzidos. O número de sensores utilizados no processo de diagnóstico é equivalente a $4t$, pois t sensores são selecionados em cada quadrante, assegurando assim as condições de diagnosticabilidade do sistema.

Assim como na abordagem TAWR [60], testes são executados em uma ordem predefinida entre os quadrantes, que garantem que testes recíprocos não ocorram. Cada quadrante tem dois vizinhos que, em sentido anti-horário, são chamados de quadrante predecessor e quadrante sucessor.

O custo de energia necessário para execução dos testes aumenta polinomialmente com a distância geográfica entre os sensores. Se os sensores estão geograficamente próximos, estes são selecionados para V_D, apresentando alta probabilidade de gerar um grafo de testes D com um valor do custo do diagnóstico reduzido.

Finalmente, o observador central é responsável por encaminhar a esse conjunto de sensores selecionados, quais testes cada sensor deve realizar. O conjunto dos sensores escolhidos para participar do diagnóstico, denotado como V_D, possui cardinalidade igual a quatro vezes o número de alarmes gerados, bastante menor se comparada à abordagem anterior.

3.5.4 ODTA - Optimal Design Test Assignment

Em [36], é apresentada uma terceira abordagem para o problema do assinalamento de testes para a identificação de nós falhos em RSSFs, denominada de ODTA. O observador central presente na rede é responsável por localizar $t + 1$ sensores próximos da região em que foram gerados alarmes, possibilitando assim, a formação de um assinalamento de $2t + 1$ sensores, considerado ótimo em termos de números de sensores.

No contexto de diagnóstico em nível de sistema, um sistema S, consistindo de n unidades é definido como ótimo se $(i) : n = 2t + 1$, onde t é o número de unidades falhas em S e (ii) : uma unidade deve ser testada por no mínimo t outras unidades [46].

Na abordagem ODTA, assume-se que os enlaces são bidirecionais, mas considera-se que os testes são realizados por enlaces unidirecionais, isto é, v_i testa v_j ou v_j testa v_i. Assim, a diagnosticabilidade pode ser derivada pelas duas condições (i) e (ii) [36].

Como nas abordagens anteriores, TAWR e EETA, assume-se, no modelo de sistema do ODTA, que os sensores são implantados em uma área de monitoramento com distribuição uniforme. Além disso, cada sensor conhece suas coordenadas e o observador central conhece a topologia da RSSF. Como ocorre no EETA, o observador central é responsável por gerar o assinalamento de testes e informar quais são os sensores responsáveis pelos testes [36].

A RSSF é definida como um grafo $G = (V, E)$, onde cada vértice $v \in V$ representa um sensor e uma aresta $(v_i, v_j) \in E$, se e somente se, v_i está dentro do alcance de transmissão de v_j ou v_j de v_i. Apesar do modelo PMC assumir enlaces bidirecionais e um grafo completamente conectado, esta asserção pode ser enfraquecida, desde que existam todos os enlaces necessários para os testes ocorrerem. Na asserção definida para o ODTA, em que testes são realizados por meio de enlaces unidirecionais, é considerado que cada sensor pode ajustar corretamente as faixas de alcance de transmissão, de forma a possibilitar que os testes existam [36].

Assume-se que um conjunto T, com cardinalidade t, de sensores em uma RSSF gera alarmes e os mesmos são recebidos pelo observador central. Por sua vez, o observador central define um assinalamento de testes a fim de diagnosticar a região da rede onde os

alarmes foram gerados. Então, o observador central informa cada sensor selecionado para realizar os testes [36].

Dado que a RSSF é definida como um grafo G, o assinalamento de testes é um grafo de diagnóstico $D = (V_D, E_D)$, onde $V_D \subset V, E_D \subset E$ e uma aresta $(v_i, v_j) \in E_D$, se e somente se, v_i testa v_j. É definido que n é a cardinalidade de V_D e $T \subset V_D$.

Para que o grafo de diagnóstico D seja t-diagnosticável, as condições (i) e (ii) devem ser cumpridas [46]. Para garantir o grafo de testes no ODTA, admite-se que sensores podem aumentar o alcance de transmissão para executar os testes [36].

Um sistema S pertence a um *design*, ou grafo $D_{\delta t}$ quando um teste (u_i, u_j) existe em S, se e somente se, $(j - i) mod\ n = (\delta m) mod\ n$ com m assumindo valores $1, 2, ..., t$ [46]. É provado em Preparata et. al., que, uma vez que um sistema S constrói um grafo D_{1t}, então S é t-diagnosticável. A característica cíclica do grafo do tipo D_{1t} produz grafos de testes t-diagnosticáveis sem que existam testes recíprocos. É provado também que um grafo $D_{\delta t}$ gera assinalamentos t-diagnosticáveis, se δ e t são relativamente primos [46].

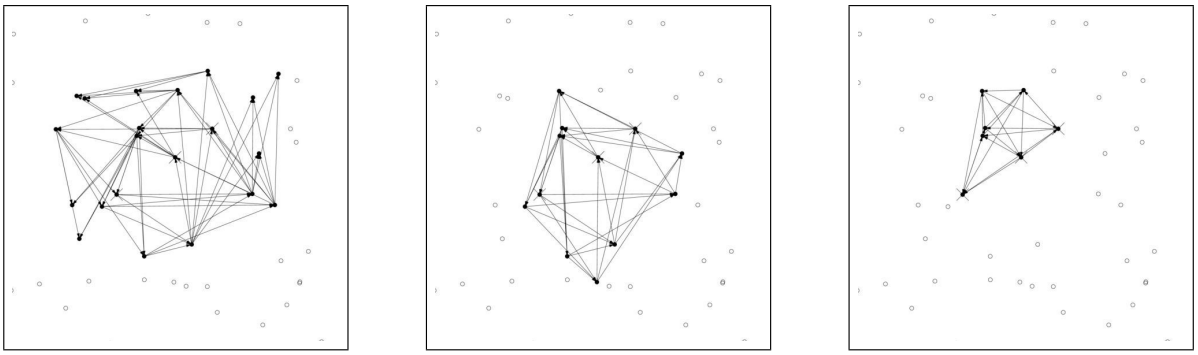
O algoritmo ODTA tem como objetivo definir um conjunto de arestas que criam um grafo D_{1t} para um sistema composto de V_D unidades. Para isso, assume-se que o conjunto V_D tenha sido definido previamente e que $n = 2t + 1$. Para definir um grafo de teste D pertencente a família de grafos D_{1t}, cada sensor presente em V_D recebe um identificador único i, onde $i = 0, ..., 2t$. Estes identificadores são escolhidos aleatoriamente pelo observador central, que cria uma rede lógica sobre os sensores de V_D e as arestas que devem existir de acordo com o grafo de teste na definição do *design* D_{1t}. O alcance de transmissão dos sensores que pertencem a V_D pode ser incrementado, com o propósito de assegurar que todas as arestas necessárias em E_D existam [36].

No ODTA, o grafo de diagnóstico D é então construído da seguinte maneira: cada sensor irá testar os t próximos sensores seguindo a ordem crescente dos seus identificadores. Então, um sensor v_i, onde i é seu identificador na rede lógica, testa os sensores $v_{(i+1)\ mod\ n}, ..., v_{(i+t)\ mod\ n}$ sobre a rede lógica. Portanto, cada sensor irá testar t sensores com identificadores maiores que seu, e será testado por t outros sensores com identificadores menores que o seu [36].

A ausência de testes recíprocos é proporcionada pelo fato de que um sensor v_i deve testar o próximo sensor, e este deverá ser testado por seus sensores anteriores na rede lógica. Isso ocorre com todos os sensores, os quais, de uma maneira cíclica, evitam testes recíprocos [36]. O custo total dos testes é igual, mesmo com a atribuição aleatória de identificadores, porque cada sensor se relaciona com todos os demais do assinalamento de testes ou como testado, ou como testador.

Para essa heurística de seleção de sensores, aplicada no ODTA, o melhor caso ocorre quando sensores em T estão próximos uns dos outros, ou seja, a heurística irá apresentar um conjunto V_D de sensores que estão próximos de R_C e os mais próximos dos sensores do conjunto T, minimizando assim o custo total do diagnóstico com relação a distância entre os sensores [36].

Em [36], foi realizada uma comparação das três abordagens, conforme a Figura 3.4. A heurística TAWR (a) apresentou consumo energético total superior às demais estratégias em todos os experimentos, com relação à energia consumida para a comunicação entre os nós sensores na realização dos testes, visto que uma grande quantidade de sensores são escolhidos para a realização dos testes.



(a) TAWR - Custo = x (b) EETA - Custo = 4t (c) ODTA - Custo = 2t+1

Figura 3.4: Comparação do grafo de testes entre as estratégias TAWR - EETA - ODTA, com $t = 3$, apresentado em [36].

Por outro lado, as abordagens EETA (b) e ODTA (c) apresentaram custos energéticos totais, em termos de comunicação entre os nós, menores que TAWR, com relação ao assinalamento de testes. A estratégia EETA apresenta um assinalamento de testes com $4t$ sensores, enquanto a ODTA apresenta com $2t + 1$. Em particular, a estratégia ODTA

foi a que apresentou o menor consumo de energia para a maioria das simulações realizadas, pois o menor número de sensores utilizados nos testes, bem como a escolha dos sensores que estejam mais próximos entre si, contribuem para diminuir o custo energético.

O alto número de combinações possíveis para a escolha dos $t + 1$ sensores não pertencentes a T, a fim de garantir que o custo total do diagnóstico seja mínimo, sugere um problema NP-Difícil [24, 36].

3.5.5 Outras abordagens

Em Taghikhaki [65] é apresentada uma abordagem distribuída para melhora da vida útil de uma RSSF em termos de energia. Falhas são detectadas localmente por meio de informações contidas em nós cabeça dos *clusters*. O tipo das informações de falhas é identificado por meio de conceitos de confiança e reputação. Os sensores que pertencem ao mesmo *cluster* compartilham e comparam suas informações. A partir dessas comparações, cada sensor gera um conjunto de possíveis sensores vizinhos com defeito. O nó cabeça do *cluster* verifica quais os sensores apresentam mais sinais de alarme para determinar o conjunto de possíveis sensores defeituosos no *cluster*.

Em Venkataraman [59] é apresentada outra abordagem também baseada em *cluster* visando a economia de energia e a degradação do desempenho com o objetivo de detectar com antecedência as falhas que podem causar perda de conectividade.

Várias são as soluções encontradas para realizar o diagnóstico dos elementos de um sistema. Algoritmos citados anteriormente neste trabalho, para detecção de falhas em sistemas distribuídos com topologia cabeada, onde energia não é um ponto determinante, quando utilizados em RSSF não são eficientes devido a características das mesmas. Esses algoritmos realizam execução periódica de testes e transmissão de mensagens redundantes, trocam enormes quantidades de mensagens de diagnóstico na rede.

Alguns utilizam a técnica de *cluster-head* [18, 59] para agrupar as informações de elementos vizinhos e não se preocupam com o sobrecarregamento destes elementos no sistema. Essa abordagem não é recomendada para ser aplicada em RSSF, pois a ideia de centralizar as informações de diagnóstico em alguns sensores *cluster-head*, além de sobre-

carregá-los, são passíveis de falhas o que pode comprometer o desempenho do diagnóstico.

Em Ruiz [51] é apresentada uma arquitetura de gerenciamento para aplicação em RSSFs, que leva em consideração as características específicas desse tipo de rede. A arquitetura MANNA atua em três áreas de gerenciamento: área funcional, níveis de gerenciamento e funcionalidade da RSSF. Esta arquitetura foi proposta para ser simples e coesa à rede [51].

As atividades ou funções que são executadas com maior frequência são localizadas pelo gerenciamento de serviços. Esses são executados por um conjunto de funções, definidas por meio de políticas específicas, que, por exemplo, podem utilizar um serviço de manutenção de uma área de cobertura que obtém a energia e as condições do alcance do monitoramento dos nós na rede. Para todos os serviços, a prioridade é minimizar o consumo de energia, tais como: mapa de energia e mapa da topologia [51].

Sabe-se que em uma RSSF as condições da rede pode variar drasticamente em alguns momentos, neste caso, a utilização de modelos estabelecidos pela MANNA é de fundamental importância para gerenciamento destas redes, embora o seu ciclo de atualização possa ser extremamente complexo e dinâmico. Com base nas informações obtidas com esses modelos, os serviços e funções são executadas de acordo com as políticas de gerenciamento da arquitetura [51].

Para o gerenciamento são consideradas informações do tipo estática e dinâmica. As informações estáticas descrevem a configuração do serviço, tanto como rede, como elemento de rede. As informações de gerenciamento dinâmico são descritas pelos modelos de RSSF e precisam ser obtidas frequentemente. Para adquirir essas informações é gerado um custo energético, e esse é um aspecto importante para determinar um momento adequado, uma frequência e fidelidade para atualizar as informações. Como exemplo de modelos dinâmicos encontram-se: mapas da área de cobertura de monitoramento e de comunicação, modelos de comportamento da RSSF e topologia da rede e energia residual. A colaboração entre os sensores é realizada de ponta-a-ponta, e neste caso, dois nós sensores cooperam um com o outro, em um determinado momento [51].

Para a arquitetura MANNA são identificadas como dependentes da área funcional

de configuração: as falhas, a segurança, o desempenho e área funcional de contabilidade. São decorrentes da área de configuração da RSSF todas as características administrativas, de manutenção operacional, execução adequada das atividades de configuração, monitoramento, processamento e comunicação. O serviço oferecido pode ser afetado se ocorrer qualquer situação diferente da identificada na fase de configuração, já que o objetivo de um nó sensor é de monitorar e controlar o ambiente. Assim, o gerenciamento de configuração é a área funcional de grande importância no gerenciamento de RSSF [51].

Como gerenciamento de configuração foram definidos o auto teste de nó e estado operacional do nó. Já para o gerenciamento de desempenho os principais itens considerados são a qualidade de aquisição da informação e a distribuição de serviços. No gerenciamento de desempenho deve existir um equilíbrio entre o alto número de parâmetros gerenciados, o alto consumo de energia e o baixo tempo de vida da rede. Uma das grandes preocupações do gerenciamento é a enorme quantidade de tráfego produzida com relação à solicitação de operações e transmissão de notificações [51].

É originado um problema de concentração de tráfego com a utilização de um gerenciador central, o que desfavorece a aplicação, para RSSF assim como ad-hoc tradicionais, do gerenciamento centralizado. É possível um alto volume de respostas recebidas por operações e eventos provenientes do gerenciamento. Como solução é possível selecionar um conjunto de agentes para retornar as respostas. Outra solução é adotar um abordagem com resposta programada [51].

Para melhor precisão das informações de gerenciamento podem ser necessárias modificações do estado da rede. Dentro do monitoramento devem ser realizadas medidas probabilísticas com maior granularidade, consequentemente evitando o efeito de monitoramento (*probe effect*), que levaria um quantidade finita de energia do sistema e assim, que pode acarretar a mudança do estado da rede [51].

Na arquitetura MANNA é utilizada uma limitação do escopo para diminuir o consumo de energia para que, no mesmo momento, a base de informações de gerenciamento possa ser atualizada. Assim é definido um espaço físico, para a limitação temporal, onde os dados serão considerados para fins de gerenciamento, bem como para limitação funcional,

onde são selecionados os dados corretos da rede [51].

São implementados agentes inteligentes ou agentes móveis na arquitetura física, para delegar as funcionalidades de gerenciamento do sistema. De forma local, as funções de gerenciamento são executadas, e são transmitidas apenas o código do problema ao invés das informações processadas. Consequentemente é possível verificar benefícios importantes, como a melhora na estabilidade [51].

CAPÍTULO 4

UMA NOVA HEURÍSTICA

Neste capítulo, abordamos as considerações preliminares para a implementação da heurística Set of Sensors Chosen by Centroid and Radius - SSCCR apresentada neste trabalho, a definição do problema e a formulação do problema em programação linear inteira. São expostas as definições do modelo de sistema e asserções, bem como o modelo de energia adotado. Descrevemos e especificamos o algorítimo da heurística SSCCR, responsável pela escolha de um conjunto de sensores que farão parte do assinalamento de testes do algoritmo ODTA.

4.1 Considerações Preliminares

Para garantir o funcionamento correto de uma RSSF em um ambiente crítico, é de fundamental importância que exista um sistema de diagnóstico para monitorar o desempenho dos nós sensores na rede e garantir um sistema tolerante a falhas. O diagnóstico em nível de sistema é utilizado para detectar e identificar as unidades falhas do sistema, para que medidas possam ser tomadas antes que as ameaças prejudiquem o bom funcionamento e a confiança dos serviços executados. Alguns algoritmos implementados para RSSFs [36, 52, 59, 60, 61, 65] abordam formas de lidar com falhas, assim, garantem a entrega confiável do serviço e a eficiência energética.

O algoritmo proposto neste trabalho também pode ser aplicado em sistemas de gerenciamento de RSSF, de forma a complementar suas as funcionalidades, como, por exemplo, a arquitetura MANNA [51].

Desta forma, com objetivo de garantir um sistema de tolerância a falhas em RSSF aplicada em missão crítica, este trabalho busca por meio da heurística SSCCR a eficiência energética na escolha do assinalamento de testes.

4.2 Definição do Problema

Em uma RSSF as informações de monitoramento são coletadas por sensores e transmitidas para o observador central da rede. Na ocorrência de comportamentos anômalos, um ou mais alarmes são gerados para reportar tais eventos, por meio de sensores localizados na região onde houve a detecção do fenômeno monitorado.

O diagnóstico pode ser iniciado quando o observador central recebe as informações de alarmes ou sob demanda, a fim de verificar o comportamento de sensores suspeitos, que venham a apresentar qualquer tipo de anormalidade na obtenção de dados de sensoriamento. Periodicamente ou por demanda nós sensores podem coletar amostras das leituras de sensores vizinhos, e a partir dessas informações, a detecção de anormalidades pode ser realizada por meio da comparação de amostragem temporal ou espacial da leitura de sensores vizinhos.

O observador central é responsável por verificar esse conjunto de nós sensores que reportaram alarmes e descartar a presença de falsos positivos. Essa verificação é realizada por meio do observador central que executa um algoritmo para obter um assinalamento de testes sobre um conjunto de sensores selecionados. Após execução dos testes pelos sensores, os mesmos encaminham os resultados para o observador central, que executa um algoritmo de diagnóstico em nível de sistema para decodificar a síndrome, e então finalmente identificar se existem falhas dentro do conjunto de sensores que geraram os alarmes.

No algoritmo ODTA é gerado um assinalamento de testes, ou seja, um grafo D direcionado de testes, que é ótimo em termos de número de sensores escolhidos para o diagnóstico.

Para definição do grafo de testes do diagnóstico, foram utilizados os conceitos de sistemas ótimos e *designs* ótimos do modelo PMC [46], em que um sistema S, composto por um conjunto de n unidades, é definido como ótimo se $n = 2t + 1$, onde t é o número de unidades falhas presentes no sistema, e cada unidade do conjunto S é testada por exatamente t outras unidades.

A formação do grafo de diagnóstico do ODTA possui uma característica cíclica com

ausência de testes recíprocos. Sobre uma rede lógica, um sensor v_i deve testar t sensores com identificadores subsequentes e deve ser testado por t sensores com identificadores precedentes, de maneira que todos os sensores contidos no conjunto S evitem testes recíprocos.

Uma heurística é utilizada para a escolha do conjunto de sensores, que tomarão parte no assinalamento de testes. O elevado número de possíveis combinações para a escolha de $t+1$ sensores que participarão do diagnóstico tem características de ser um problema computacionalmente intratável [24].

A heurística utilizada pelo algoritmo ODTA, é responsável por escolher um conjunto de $t+1$ sensores em sua maioria próximos geograficamente entre si, com base na região central, denotada como R_C, da menor área retangular, denotada como R, que envolvem todos os sensores que pertencem ao conjunto T de alarmes, com cardinalidade t. A R_C, obtida como centro de R, é a referência para escolha dos sensores, de maneira que os mesmos estejam o mais próximos entre si, bem como próximos dos sensores que geraram alarmes. Ao final do processo de escolha, um conjunto ótimo em termos de números de sensores, composto por $2t+1$, sensores que incluem o conjunto T que obrigatoriamente devem participar do diagnóstico, juntamente com $t+1$ sensores escolhidos pela heurística.

Porém, a heurística utilizada pelo algoritmo ODTA para a escolha dos $t+1$ sensores não garante necessariamente arestas com custo mínimo com relação à distância entre os sensores. Assim, a heurística SSCCR for ODTA, para a escolha de tais sensores é apresentada com o intuito de selecionar um conjunto de sensores que estejam o mais próximos geograficamente entre si, e entre os sensores que geraram alarmes (desconsiderando-se as possíveis obstruções), e posteriormente ser aplicado o assinalamento de testes definido pela heurística ODTA, que tem como característica ser ótimo em termos de número de sensores, tendo sido especificados os sensores que farão parte do mesmo.

Para realizar a comparação do custo total do somatório das distâncias entre os sensores selecionados por meio da heurística ODTA e da heurística SSCCR, foi formulado o problema em programação linear inteira para se obter o conjunto de sensores com o resultado ótimo em termos de distância entre os mesmos.

4.2.1 Programação Linear

Em programação linear o objetivo é encontrar um vetor não negativo x com valores em $\mathbb{Q}$ que minimiza a função objetivo linear em função dos elementos do vetor x do dado problema, sujeita às restrições que são desigualdades lineares em função de x. Formalmente, dado um vetor c de tamanho n com valores em $\mathbb{Q}$, um vetor b de tamanho m com valores em $\mathbb{Q}$ e uma matriz $A = (a_{ij})$ de dimensões $m \times n$ pertencente a $\mathbb{Q}^{m \times n}$, a solução ótima do programa linear é um vetor x de tamanho n que minimiza a função objetivo $\sum_{j=1}^{n} c_j x_j$ sujeito ao conjunto de restrições: $\sum_{j=1}^{n} a_{ij} x_j >= b_i$, para $i = 1, \ldots, m$ e $x_j >= 0$, para $j = 1, \ldots, n$ [63].

As linhas de restrições do programa linear definem, na verdade, um conjunto de restrições. Cada restrição desse conjunto é um somatório de elementos que é normalmente composto de n elementos, ou seja, a quantidade de células existentes no vetor x. O vetor x é chamado de variável. Qualquer x que satisfaça as restrições pode ser dito como solução viável, e se um tal x existe, então o programa linear é viável. Resolver um problema em programação linear é obter uma solução ótima. Se não existir qualquer x viável para a solução, então a programação linear é inviável [63].

$$minimizar \sum_{j=1}^{n} c_j x_j$$

$$sujeito\ a \sum_{j=1}^{n} a_{ij} x_j \geq b_i, \qquad i = 1, \ldots, m, \tag{4.1}$$

$$x_j \geq 0, \qquad j = 1, \ldots, n. \tag{4.2}$$

O termo *Linear Programming* é frequentemente abreviado como LP (em português, Programação Linear - PL). A PL é uma das técnicas mais utilizadas para formular problemas de otimização, que tem como característica a distribuição eficiente dos limitados recursos de maneira que se maximize lucros ou minimize custos. O cálculo dos custos ou lucros são definidos por uma função objetivo. Para definir quais atividades consomem recursos, e em que proporções os mesmos são consumidos, são utilizadas equações line-

ares. Para cada recurso são utilizadas desigualdades lineares denominadas restrições do modelo [63].

Diversas são as formas de distribuir os recursos escassos de um dado problema, de modo que as distribuições tenham coerência com as restrições do modelo. Porém, em um problema formulado em PL, o que se procura é a solução ótima de uma função objetivo, ou seja, maximizar lucros ou minimizar custos [63].

A programação linear pode ter outras variações, mas, para resolver o problema abordado neste trabalho, foi utilizada a programação linear inteira. Essa variação restringe os valores dos elementos do vetor x apenas a inteiros. Assim, há possibilidade de declarar x_j como pertencente ao conjunto dos números naturais ou limitar a um subconjunto de números inteiros como, por exemplo, $x_j \in \{0, 1\}$. Entretanto, ao contrário da programação linear que admite valores em $\mathbb{Q}$, ainda não são conhecidos algoritmos polinomiais que solucionam programação linear inteira. O problema em solucionar programação linear inteira é conhecido com um problema de otimização NP-Difícil [24]. Contudo, a programação linear é uma técnica útil para modelar de maneira compacta problemas de otimização combinatória [63].

Um problema NP-Difícil pode também ser inicialmente formulado por meio de um programa linear relaxado. Tal programa permite que as variáveis assumam valores fracionários. O valor ótimo do programa relaxado pode ser melhor ou igual ao valor ótimo do problema em programação inteiro [24].

O uso do relaxamento da solução no problema abordado neste trabalho não é viável, pois trata-se de um problema de otimização combinatória cujas variáveis são discretas. Em tal problema, não é permitida a decomposição de sensores. Para que fosse possível o uso da solução de um programa linear relaxado, seria necessário que tal solução fosse arredondada. Tal tratamento não foi realizado para limitar o escopo do trabalho.

4.2.2 Formulação do Problema em Programação Linear Inteira

É dado um grafo $G = (V, E)$, onde V é um conjunto não vazio de sensores denominados vértices e E é um conjunto de todos os subconjuntos de V de tamanho 2, chamados arestas.

Além disso, é dada a função $d : E \rightarrow \mathbb{R}^+$ que devolve valores contidos no conjunto dos números reais positivos que representam o peso d, ou distância entre pares de sensores. O conjunto de sensores que geraram alarmes é denotado por T e está contido no conjunto V. O tamanho do conjunto T é equivalente ao número de t alarmes gerados na rede.

O objetivo é selecionar um conjunto S com tamanho igual a $2t + 1$, de maneira que o conjunto T esteja contido em S e o somatório das distâncias de todos os pares de sensores contidos nesse conjunto seja mínimo.

As variáveis de decisão do problema são dadas pelo vetor x onde cada célula x_i pode assumir 1 ou 0 para representar, respectivamente, se o sensor i está ou não contido no conjunto S. Outra variável é o vetor y, cujas células y_{ij} assumem 1 ou 0 e representam, respectivamente, se os sensores x_i e x_j estão ou não contidos no conjunto S. Com o intuito de obter o conjunto S de sensores com a menor distância entre si, a função objetivo é minimizar o somatório das distâncias entre os sensores i e j, tal que $\{i, j\}$ são todos os subconjuntos de V de tamanho igual a 2 formados pelos identificadores de todos os sensores na rede.

Definição:

$$G = (V, E) \qquad d : E \rightarrow \mathbb{R}^+ \qquad T \subseteq V \qquad |T| = t$$

Objetivo:

$$\text{Selecionar } S \supseteq T \text{ tal que } \quad |S| = 2t + 1 \quad \text{ e } \quad c(S) = \sum_{i,j \in S} d_{ij} \quad \text{seja mínimo;}$$

Formulação em Programação Linear:

$$\begin{cases} y_{ij} = 1, & \text{se ambos } i \text{ e } j \in S \\ y_{ij} = 0, & \text{caso contrário} \end{cases}$$

$$\begin{cases} x_i = 1, & \text{se } i \in S \\ x_i = 0, & \text{caso contrário} \end{cases}$$

Função Objetivo:

$$minimizar \sum_{i \text{ e } j \in V} d_{ij}.y_{ij}$$

$$sujeito\ a \sum_{i \in V \setminus T} x_i = t + 1, \tag{4.3}$$

$$x_i = 1, \qquad \forall_i \in T, \tag{4.4}$$

$$x_i + x_j \leq 1 + y_{ij}, \qquad \forall_{ij} \in V, \tag{4.5}$$

$$x_i \geq y_{ij}, \qquad \forall_{ij} \in V. \tag{4.6}$$

A função objetivo está sujeita ao seguinte conjunto de restrições: (4.3) o somatório de sensores x_i em V, que não pertencem a T, mas devem fazer parte da solução é igual a $t + 1$, ou seja, essa restrição permite selecionar $t + 1$ sensores dentre os que não geraram alarme; (4.4) se o sensor pertence ao conjunto T, obrigatoriamente x_i deve fazer parte da solução, ou seja, essa restrição visa garantir a presença dos nós que geraram alarmes; (4.5) e (4.6) garantem que, se um enlace está presente, os nós adjacentes fazem parte da solução.

4.3 Definições do Modelo de Sistema e Asserções

O sistema é composto por n dispositivos sensores, denominados de nós sensores, que monitoram uma determinada região com o objetivo de coletar, processar e disseminar informações de um evento específico. Os nós sensores possuem uma característica básica, composto por unidade de processamento, memória, rádio comunicador e bateria limitada. Os sensores podem assumir o estado falho ou sem-falha. Devido a diversos problemas, como acidente ou descarregamento da bateria, nós podem estar no estado falho e ficar isolado sem comunicação com o restante do sistema. Nós permanecem falhos até serem reparados ou substituídos.

Assumimos em nossa abordagem que o observador central pode possuir mobilidade para coletar informação de diagnóstico com menor custo para os nós sensores e conhece a topologia da rede. Consideramos que os nós sensores dispostos na rede conhecem suas coordenadas na área de cobertura monitorada e cada um possui uma identificação única, a qual permite identificar-se e comunicar-se com seus nós vizinhos. Assumimos que nós são implantados de forma homogênea no campo de monitoramento e não possuem mobilidade.

Todas as transmissões a partir de qualquer nó sensor são omni-direcionais. Assim, qualquer mensagem enviada por um sensor pode ser recebida por qualquer nó de sua vizinhança, ou seja, dentro do seu alcance de transmissão. O custo associado com a transmissão de uma mensagem é superior aquele associada à sua recepção. Assumimos que a energia dissipada para os nós ouvirem o canal é desprezível. O custo associado com a troca de mensagens é proporcional ao seu tamanho $O(log\ n)$. Admite-se que a síndrome é composta pelo identificador do nó testado, mais um bit referente ao resultado de seu teste, ou seja, 0, caso sensor testado esteja no estado sem-falha ou 1, se falho.

A estação base é capaz de transmitir mensagens para qualquer nó da rede em um único salto (*one-hop broadcast*), pois a mesma é dotada de um hardware mais robusto com relação a processamento, armazenamento, comunicação e energia. Por outro lado, devido às características de uma unidade de nó sensor, assumimos que nós distantes do observador central só conseguem transmitir mensagens para nós que estão dentro da sua área de comunicação. Desta forma, nós distantes dependem de nós vizinhos para enviar suas informações para o observador central (*multi-hop*).

Para detectar as ameaças em uma RSSF um algoritmo de diagnóstico em nível de sistema é executado por meio do observador central. O mesmo é encarregado de distribuir os testes entre os nós sensores da rede periodicamente ou sob demanda, ou seja, quando um ou mais alarmes ocorrerem. Após a realização dos testes, os sensores enviam seus resultados para o observador central, de forma que, por meio de um algoritmo, ele possa decodificar a síndrome dos testes realizados e identificar os sensores falhos.

O modelo PMC possui aplicabilidade em RSSF devido à presença de um *sink*, que faz o papel do observador central na realização do diagnóstico. Neste modelo, a execução de

um teste requer um enlace bidirecional entre v_i e v_j. Nas RSSFs, os testes podem ser executados na presença de um enlace unidirecional. Um sensor pode iniciar um auto teste sobre um conjunto predefinido de estímulos e enviar os resultados para o seu testador. Assim, é necessário que apenas um nó testado seja capaz de enviar os resultados de seu teste para o seu testador. Para isso consideramos que os sensores podem aumentar sua faixa de transmissão, ou seja, calibrar a intensidade do sinal de rádio, de maneira que se possa permitir que o enlaces de testes existam. Nós assumimos que os enlaces podem ser bidirecionais, mas consideramos que os testes possuem enlaces unidirecionais, ou seja, não recíprocos. Apesar do modelo PMC assumir enlaces bidirecionais e um grafo completamente conectado, esta suposição pode ser enfraquecida, desde que sejam fornecidos todos os enlaces unidirecionais necessários para que os testes existam.

4.3.1 Modelo de Energia Adotado

O principal objetivo dos modelos de energia é verificar o comportamento de consumo de cada nó sensor de uma RSSF. O modelo de energia é composto pela bateria, com uma capacidade finita de energia, e as unidades consumidoras de energia, tais como: o rádio comunicador, processador e dispositivos de sensoriamento. Cada unidade consumidora é responsável por informar seu consumo de energia à unidade de gerenciamento e provedora de energia [43].

Para a realização do diagnóstico em nível de sistema é utilizada a troca de mensagens entre sensores, por meio de seus seus componentes de rádio transmissor. Dentre os componentes de um sensor sem fio, a unidade que mais consome energia é a do rádio comunicador [4]. Por esse motivo, o cálculo estimado para o consumo de energia na comunicação entre sensores é baseado no modelo empírico de atenuação de sinal, o modelo *one slope*, utilizado na área de redes sem fio [44, 53]. Este modelo assume uma dependência linear entre a atenuação de sinal e o logaritmo da distância d entre o transmissor e o receptor:

$$L(d)|_{dB} = l_0 + 10\alpha log_{10}(d) \tag{4.7}$$

onde:

- $L(d)|_{dB}$ Diferença em decibéis da potência do sinal no transmissor e sua potência ao alcance do receptor;
- l_0 Atenuação de sinal (distância de referência equivalente a 1 metro);
- 10α fator de atenuação que caracteriza o ambiente;
- $log_{10}(d)$ Referência média de atenuação de sinal com relação a distância d.

Esse modelo é de fácil aplicação, pois os parâmetros de entrada para a fórmula que calcula a estimativa do consumo de energia são: a distância entre o transmissor e o receptor e o fator de atenuação do ambiente.

Para assegurar a comunicação entre o sensor transmissor t e um sensor receptor r, colocado a uma distância d entre os dois, é necessário que o pacote enviado por t alcance r com um nível de potência superior à sensibilidade do receptor. De outro modo, seja E_t o poder de transmissão, E_r a potência do sinal do receptor (onde E_r depende de E_t e da distância d) e E_m é a sensibilidade do receptor, $E_r > E_m$ [44].

Considerando que $L(d)|_{dB}$ é equivalente a perda por propagação em função da distância, ou seja, igual a diferença em decibéis da potência do sinal no transmissor e sua potência ao alcance do receptor, dada a Equação 4.7, pode ser obtido:

$$10\ log_{10}\left(\frac{E_t}{E_r}\right) = l_0 + 10\ \alpha\ log_{10}\ d \tag{4.8}$$

Logo:

$$E_r = \frac{E_t}{10^{(\frac{l_0}{10}\ +\ \alpha\ log_{10}\ d)}} \tag{4.9}$$

Para ser obtida a potência mínima de transmissão E_t no transmissor, de maneira que seja assegurado o alcance do pacote pelo receptor com energia suficiente, foi adicionado na equação acima $E_r > E_m$:

$$E_t = E_m 10^{(\frac{l_0}{10}\ +\ \alpha\ log_{10}\ d)} \tag{4.10}$$

Consequentemente, a Equação 4.10 está sujeita à distância d entre o transmissor e receptor, à sensibilidade do receptor E_m e aos parâmetros de atenuação de sinal l_0 e o fator de atenuação que caracteriza o ambiente α [53]. Dessa forma, após o desenvolvimento da equação foi obtido:

$$E_t = E_m 10^{\frac{l_0}{10}}\ d^{\alpha} \tag{4.11}$$

A equação acima, mostra que o consumo de energia cresce polinomialmente conforme a distância d, com expoente α. Portanto, nesse trabalho o custo de energia é baseado na distância geográfica entre os sensores.

4.4 Especificação do Algoritmo

É apresentado nessa sessão o pseudocódigo referente ao Algoritmo 1 que descreve os passos da heurística SSCCR implementada nesse trabalho.

4.4.1 Descrição do Algoritmo

Seja V o conjunto que contém os valores inteiros formados pelos identificadores de todos os sensores da RSSF. Seja T o conjunto formado pelos valores inteiros dos índices dos sensores que geraram alarmes, de forma que $T \subseteq V$. Considere d uma função que implementa a distância euclidiana entre dois elementos. São dados como entrada para o Algoritmo 1 os conjuntos V, T e a função de distância d. Sendo $t = |T|$, a saída do algoritmo é um conjunto de $t + 1$ sensores mais próximos aos listados no conjunto T e entre si.

O algoritmo inicia sua execução por meio do cálculo de um *centróide*. O centróide de um conjunto finito de pontos $p_1, p_2, \ldots, p_k \in \mathbb{R}^n$, sendo n um valor inteiro maior que 0, é um ponto c obtido por meio da equação $c = \frac{(p_1+p_2+\ldots+p_k)}{k}$ [62]. O cálculo do centróide é realizado pela função $CENTROIDE$, que tem como entrada o conjunto T e a função de distância d e tem como saída o ponto centróide entre os sensores listados em T. Tal ponto centróide é armazenado no algoritmo pela variável rc.

Algoritmo 1: Pseudocódigo da Heurística SSCCR para Diagnóstico em RSSF

Entrada: $V, T, d(x, y) \forall x, y \in V$
Saída: Conjunto de sensores para o diagnóstico

```
1  inicio
2      τ ← ∅
3      rc ← CENTROIDE(T, d)
4      raio ← max{d(rc, t_i), ∀t_i ∈ T}
5      Θ ← {x|d(rc, x) ≤ raio, ∀x ∈ V \ T}
6      se |Θ| = |T| + 1 então
7          retorna Θ
8      fim se
9      se |Θ| > |T| + 1 então
10         w ← CALCULAPESOS(Θ, T, d)
11         P ← ORDENAR(w)
12         retorna ⟨p_1, ..., p_n⟩| 1 ≤ n ≤ |T| + 1 e p_i ∈ P ∀_i = 1, ..., n
13     fim se
14     se |Θ| < |T| + 1 então
15         enquanto |Θ| + |τ| < |T| + 1 faça
16             rc ← CENTROIDE(T ∪ Θ ∪ τ, d)
17             raio ← 1.1 · max{d(rc, t_i), ∀t_i ∈ T ∪ Θ ∪ τ}
18             τ ← {x|d(rc, x) ≤ raio, ∀x ∈ V \ T \ Θ}
19             se |Θ| + |τ| = |T| + 1 então
20                 retorna Θ ∪ τ
21             fim se
22             se |Θ| + |τ| > |T| + 1 então
23                 w ← CALCULAPESOS(Θ ∪ τ, T, d)
24                 P ← ORDENAR(w)
25                 retorna ⟨p_1, ..., p_n⟩| 1 ≤ n ≤ |T| + 1 e p_i ∈ P ∀_i = 1, ..., n
26             fim se
27         fim enqto
28     fim se
29 fin
```

Em seguida, o algoritmo calcula o *raio* de um círculo centralizado em *rc*. O raio é calculado de forma que seja obtido o menor círculo que engloba todos os sensores que geraram alarmes. Desta forma, ele corresponde ao maior valor dentre as distâncias de *rc* aos sensores listados em T. Logo em seguida, é gerado um conjunto Θ que contém os índices de todos os sensores sem-falhas indicados em V menos aqueles indicados em T internos ao círculo centralizado em *rc* e de raio *raio*.

Caso o conjunto Θ de sensores venha a satisfazer as condições da t-diagnosticabilidade, ou seja, $|\Theta| = |T| + 1$, então o algoritmo devolve Θ. Caso contrário, Θ pode ter cardinali-

dade superior ou inferior a $|T|+1$. No primeiro caso, o algoritmo calcula pesos atribuídos a cada sensor, ordena os sensores com base nos pesos e devolve os $|T|+1$ sensores com menor peso. Os pesos são calculados pela função $CALCULAPESOS$ e são armazenados em w. Tal função calcula os pesos de cada sensor listado em Θ como sendo o somatório das distâncias entre ele e cada sensor listado em $T \cup \Theta$. Logo após o cálculo de w, a função $ORDENAR$ gera uma sequência ordenada de sensores com base nos pesos passados como entrada. Tal sequência é armazenada em P e os $|T|+1$ primeiros elementos de P são devolvidos.

Caso o conjunto Θ apresente um número de sensores menor que $|T|+1$ (linha 14), então é necessário selecionar mais sensores para que seja satisfeita a restrição de t-diagnosticabilidade. Tais sensores são adicionados no conjunto τ por um processo iterativo. Este processo iterativo é semelhante ao executado no início do algoritmo. Inicialmente, rc é recalculado com base no conjunto $T \cup \Theta \cup \tau$, com τ inicialmente vazio.

Logo após, $raio$ é recalculado com base nesse mesmo conjunto e com adição de 10% para aumentar a área do círculo encontrado anteriormente. O círculo é aumentado com objetivo de obter sensores externos ao conjunto Θ e que estão mais próximos de rc. Assim, τ é gerado com os índices dos sensores internos à nova circunferência que não estão listados em Θ nem em T.

Com o crescimento de τ a cada iteração, existirá um momento em que $|\Theta|+|\tau| \geq |T|+1$. Quando isso acontecer, caso $|\Theta|+|\tau| = |T|+1$ então $\Theta \cup \tau$ é devolvido; caso contrário, os pesos são recalculados, os sensores são reordenados e são devolvidos os índices dos $|T|+1$ sensores de menor peso. São recalculados os pesos dos elementos listados em $\Theta \cup \tau$ como sendo o somatório das distâncias entre eles e cada sensor listado em $T \cup \Theta \cup \tau$.

CAPÍTULO 5

ESTUDO COMPARATIVO

Neste capítulo serão apresentados os experimentos realizados neste trabalho, bem como o cenário de simulação e as ferramentas utilizadas. Posteriormente, é realizada a comparação dos resultados e a discussão dos mesmos.

5.1 Cenário da Simulação

Os experimentos realizados neste trabalho foram obtidos por meio de simulações. Encontrar um simulador para RSSF adequado para cada tipo de aplicação é uma tarefa difícil. Existem diversas pesquisas de simuladores para RSSFs [41], que cobrem apenas um número de simuladores disponíveis atualmente, incluindo os produtos comerciais.

O simulador utilizado neste trabalho foi programado em C++ com intuito de gerar pontos com coordenadas geográficas aleatórias para representar sensores sem fio em uma rede. O simulador é o mesmo utilizado nos experimentos com os algoritmos TAWR, EETA e ODTA.

São selecionados pelo simulador, de forma aleatória, t sensores para formar o conjunto T, que denota o conjunto de alarmes gerados. Posteriormente, o simulador executa os algoritmos referentes às duas heurísticas para escolha dos nós sensores que devem participar do conjunto de testes para o diagnóstico.

Para concepção das redes é inserido um conjunto de parâmetros para o simulador, tais como: tamanho em metros da área de cobertura do monitoramento; quantidade de sensores responsáveis pelo monitoramento do evento e a quantidade de sensores que reportarão mensagens de alarmes para o observador central.

O simulador é configurado, por padrão, para gerar uma rede conectada, com uma distribuição estatística uniforme, de maneira que a rede seja disposta com sensores distribuídos de forma homogênea sobre a área de monitoramento.

Para a escolha aleatória dos alarmes reportados, o simulador é configurado para definir uma região de alarmes, que limita uma área quadrada posicionada aleatoriamente sobre a rede, com um tamanho proporcional a 1% de toda a rede. A região de alarmes é responsável por compreender os sensores que são candidatos a serem alarmes na simulação e simula a localidade geográfica de um evento.

A heurística SSCCR apresentada tem o objetivo de contribuir na minimização do custo energético dos testes realizados durante o diagnóstico. Essa redução é dada pela escolha de um conjunto de sensores que estejam o mais próximos entre si, e também próximos do conjunto dos sensores que reportaram alarmes. Os experimentos realizados têm o intuito de comparar a heurística SSCCR apresentada nesse trabalho com a heurística utilizada no algoritmo do ODTA, com relação à escolha desse conjunto de sensores. Dessa forma, a propriedade avaliada para cada experimento é definida pelo somatório total das distâncias entre cada sensor para todos os outros selecionados em cada heurística. Quanto menor o valor do somatório das distâncias entre os sensores na simulação do experimento, mais eficiente é a heurística executada.

O custo total do consumo de energia para cada estratégia, dado por $Custo(D)$, é definido pelo somatório do consumo de energia consumido na comunicação entre os sensores que participam dos testes, conforme:

$$Custo(D) = \sum C_{i,j} \,|\forall (v_i, v_j) \in E_D \tag{5.1}$$

onde $C_{i,j}$ é o custo da energia gasta pelos sensores v_i e v_j que pertencem ao conjunto de arestas E_D quando o sensor v_i executa um teste sobre o sensor v_j. O custo de um teste executado pelo sensor v_i sobre o sensor v_j numa distância d é calculado como o somatório da energia gasta por v_j enviar a sequência de resultados de um auto-teste para v_i e para v_i receber tal resultado [36].

A propriedade acima foi avaliada para cada simulação executada. Para cada conjunto de parâmetros, foram geradas 100 redes diferentes, com tamanho de $512mX512m$ e considerados um conjunto total de 512 e 1024 sensores. A Tabela 5.1 apresenta o conjunto de parâmetros utilizados para execução das simulações.

Tabela 5.1: Parâmetros utilizados nas simulações

Parâmetros	**Valores**
Área de cobertura do monitoramento	512m x 512m
Distribuição dos sensores na rede	Uniforme
Número de sensores na rede	512 e 1024
Região de alarmes	1% da cobertura da rede
Número de alarmes	3, 5, 7, 9, 11, 13, 15 e 19
Configurações de rede geradas por simulação	100

Um total de amostras superior a 30 é considerado o suficiente para se obter o cálculo da média e permite utilizar uma distribuição normal [14]. Portanto, para cada conjunto de parâmetros, a execução de 100 simulações em nosso experimento é considerada relevante para utilizar os resultados.

A configuração do servidor utilizado na execução dos experimentos possui as seguintes características: Sistema Operacional Linux com Processador AMD Opteron (tm) 6136 - 2.400 MHz, 512 KB de memória cache e 10 GB de memória RAM. Para execução do experimento das duas heurísticas comparadas não foram exigidas mais que 5% da memória RAM, e do processador foi exigido 100% de seu desempenho, mas por poucos segundos para cada simulação executada. Para obter os resultados ótimos por meio do problema em programação linear, devido à alta complexidade do problema e à enorme quantidade de cálculos executados, o uso da memória RAM ficou acima dos 20% e o uso do processador em 100% durante horas para cada simulação executada. Observamos na execução das simulações que o incremento do número de alarmes na rede exigia o máximo do desempenho do processador, até que em determinado número de alarmes a simulação para se obter os resultados ótimos por meio da programação linear se tornou inviável, devido ao enorme tempo computacional exigido, de característica exponencial.

De maneira a possibilitar a comparação entre as heurísticas ODTA e SSCCR com os resultados do ótimo obtido por meio da programação linear nas simulações, os parâmetros acima mencionados foram selecionados a partir do trabalho anterior do Algoritmo ODTA [36]. Tais parâmetros de simulação permitem aproximar a utilização de uma RSSF em uma aplicação real, em uma área de cobertura com raio de 512 metros quadrados, em que os nós sensores são instalados uniformemente, que dependendo da

necessidade de precisão dos dados coletados na rede pode ser composta por um número de 512, 1024 ou mais nós sensores. Para simular as ocorrências de eventos localizados em uma região específica da área de cobertura, a região de alarmes foi definida em 1% sobre o total da área de cobertura da rede.

5.2 Experimentos Realizados e Resultados

Os experimentos realizados para as duas heurísticas abordadas neste trabalho são apresentados nesta seção, com o objetivo de obter um conjunto de sensores que apresentem o menor custo energético do ponto de vista da distância entre os sensores. Tais experimentos visam a escolha de um conjunto de sensores que participarão do assinalamento ótimo de testes executado por meio do algoritmo ODTA.

Para se obter o conjunto ótimo em termos de distâncias entre os sensores escolhidos, o problema foi formulado em programação linear inteira, e desta forma, permitir comparar os resultados ótimos obtidos com os resultados gerados pelas duas heurísticas.

Para demonstrar o comportamento das heurísticas, apresentamos nos subitens a seguir, uma simulação com a mesma configuração de rede, composta de 512 sensores, 3 alarmes e distribuição uniforme, conforme Figura 5.1, para ambas as heurísticas.

5.2.1 Escolha dos Sensores Por Meio da Heurística ODTA

No algoritmo ODTA, após o observador central da rede receber as informações de alarmes reportados, exemplificado na Figura 5.2, com base na simulação na rede apresentada na Figura 5.1, o mesmo é responsável por executar uma heurística, que tem como objetivo selecionar um conjunto de $t+1$ sensores que estejam mais próximos do conjunto de sensores que reportaram alarmes [36].

O primeiro passo da heurística é definir a menor região retangular que compreende todos os sensores que reportaram alarmes, 4, 15 e 50, conforme Figura 5.3. Em seguida é obtida a região central do retângulo, conforme a Figura 5.4. Então são obtidos os $t+1$ sensores próximos da região central e consequentemente próximos do conjunto de

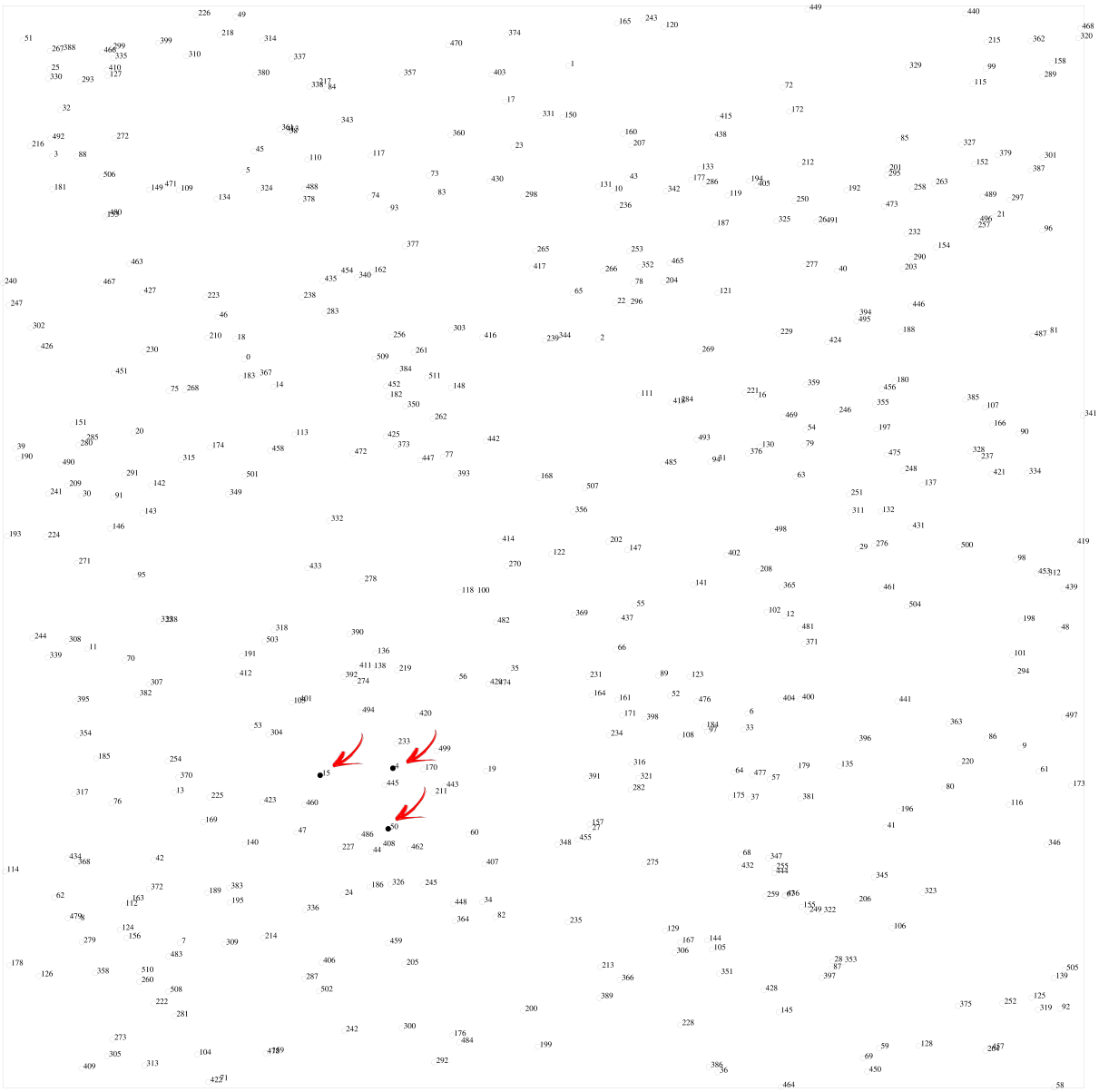


Figura 5.1: Exemplo de uma simulação para o posicionamento de 512 sensores e 3 alarmes em uma RSSF, identificados pelos números 4. 15 e 50.

sensores com alarmes, identificados pelos números 227, 445, 460 e 486, conforme mostrado na Figura 5.5.

Após a execução da heurística, os $t + 1$ sensores selecionados juntamente com o conjunto de sensores que geraram alarmes, formam um novo conjunto de $2t + 1$ sensores que participarão do assinalamento ótimo de testes para o diagnóstico empregado pelo algoritmo ODTA [36].

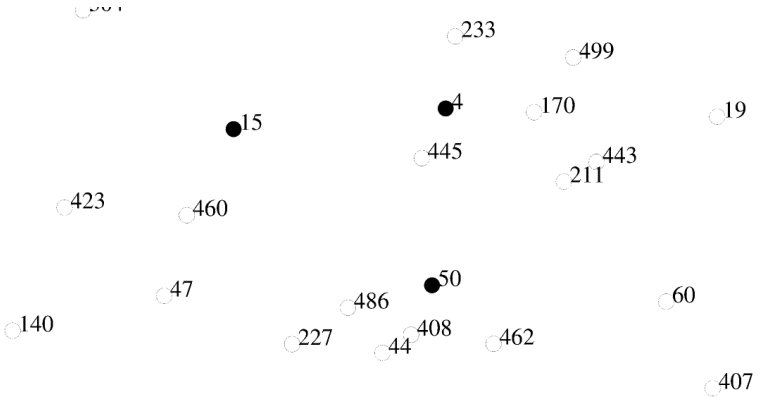


Figura 5.2: Heurística ODTA - Sensores que reportaram alarmes para o observador central da rede.

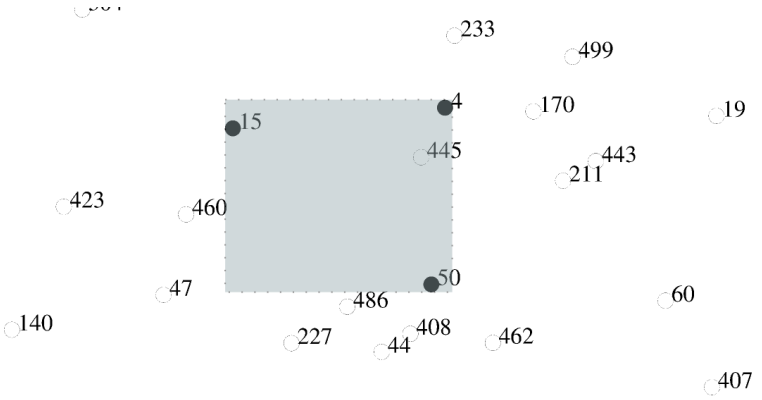


Figura 5.3: Heurística ODTA - Demonstração da menor região retangular que abrange todos os sensores que reportaram alarmes.

5.2.2 Escolha dos Sensores Por Meio da Heurística SSCCR

Para efeito de comparação do funcionamento das duas abordagens, foram utilizados os mesmos parâmetros de entrada para a simulação da heurística SSCCR, conforme a Figura 5.1 e a Figura 5.2.

Com as informações de alarmes reportadas ao observador central, o mesmo inicia a execução da heurística SSCCR, que também tem como objetivo selecionar um conjunto de $t + 1$ sensores que estejam mais próximos de si e do conjunto de sensores que reportaram alarmes, mas utilizando uma técnica mais aprimorada.

Primeiramente, a heurística SSCCR calcula o centróide, ou seja, o ponto com coordenada cartesiana (x, y) equivalente às médias das coordenadas de todos os sensores que reportaram alarmes, conforme a Figura 5.6.

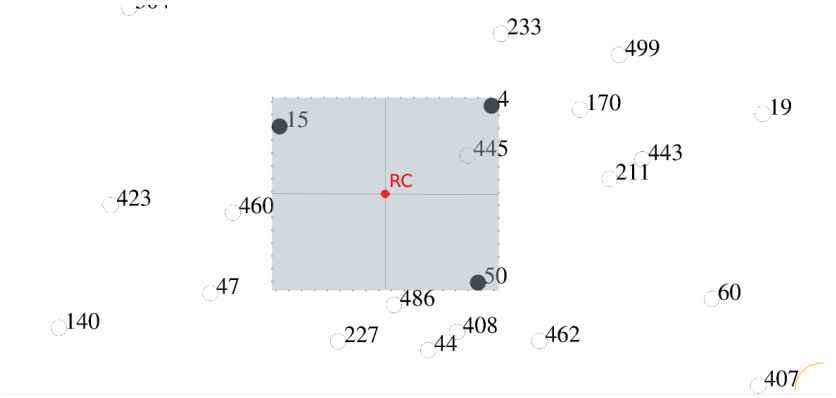


Figura 5.4: Heurística ODTA - Cálculo da região central do retângulo.

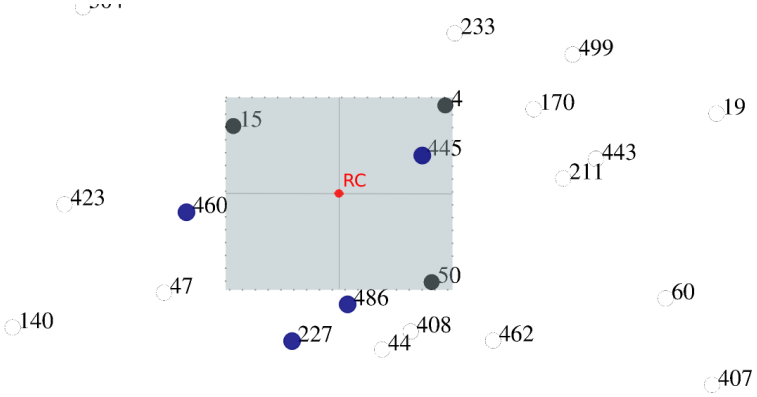


Figura 5.5: Heurística ODTA - Escolha dos $t + 1$ sensores próximos da região central.

Em seguida, é obtido o valor do raio, que é dado a partir das coordenadas geográficas do ponto centróide até o sensor mais distante com alarme, conforme Figura 5.7.

Posteriormente, a heurística SSCCR define um novo conjunto com os sensores que compreendem a circunferência com o raio calculado, subtraídos os sensores que geraram alarmes. Na Figura 5.8 pode ser observado que os sensores com identificadores 445 e 486 são selecionados, pois os mesmos estão dentro do raio. Para esse caso de simulação, o número de sensores selecionados, por meio do alcance desse raio, não satisfazem o número de $t + 1$ sensores necessários para o diagnóstico definido pelo algoritmo ODTA. Dessa maneira, a fim de se obter outros sensores próximos ao conjunto de sensores com alarmes, e a esses já selecionados, é calculado um novo centróide.

O cálculo do novo ponto centróide obtido por meio da lista de sensores que reportaram alarmes, mais os sensores mencionados acima, conforme Figura 5.9, tem o objetivo de

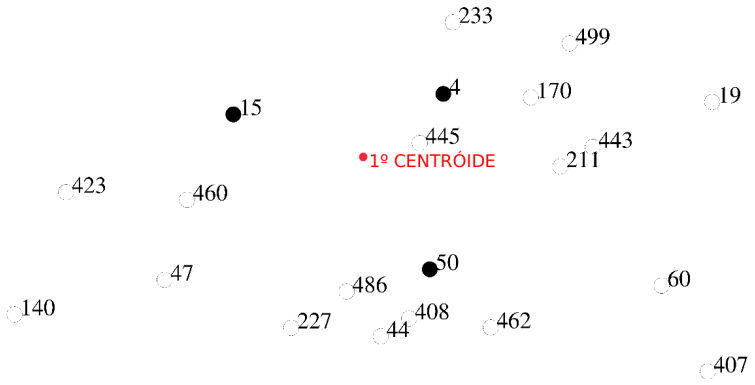


Figura 5.6: Heurística SSCCR - Cálculo do ponto centróide entre os sensores identificados por 4, 15 e 50 que reportaram alarmes.

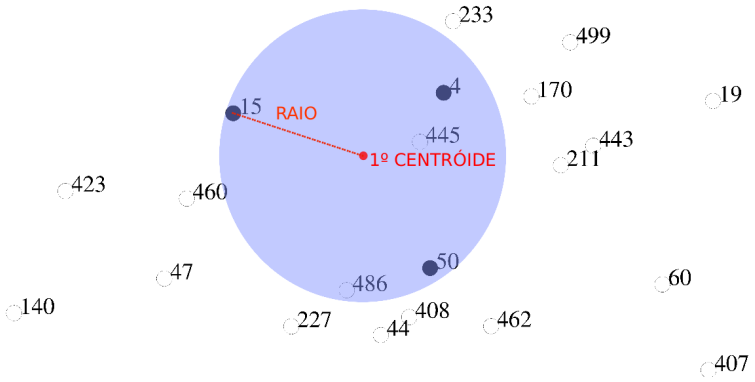


Figura 5.7: Heurística SSCCR - Valor do raio obtido entre o centróide e sensor mais distante com alarme.

reequilibrar o ponto médio entre todos esses sensores, e permitir selecionar outros sensores próximos aos alarmes e a esse conjunto de sensores escolhidos.

O valor do raio é recalculado a partir do novo ponto centróide até o sensor mais distante pertencente ao conjunto de alarmes e ao conjunto de sensores selecionados anteriormente. Então, soma-se ao raio um valor de 10% do seu tamanho, de maneira que seja possível selecionar outros sensores. Desta forma, garante-se que o novo raio recalculado tenha maior alcance que o raio anterior, conforme a Figura 5.9.

Em seguida, é verificado o número de sensores que estão estão dentro do alcance do raio recalculado, mas que não pertencem ao conjunto de alarmes e ao conjunto de sensores já selecionados anteriormente. Como mostrado na Figura 5.10, pode-se observar que existem três sensores dentro do alcance do raio recalculado, identificados pelos números: 227, 44 e 408. Somados estes três sensores, mais o número de sensores selecionados anteriormente,

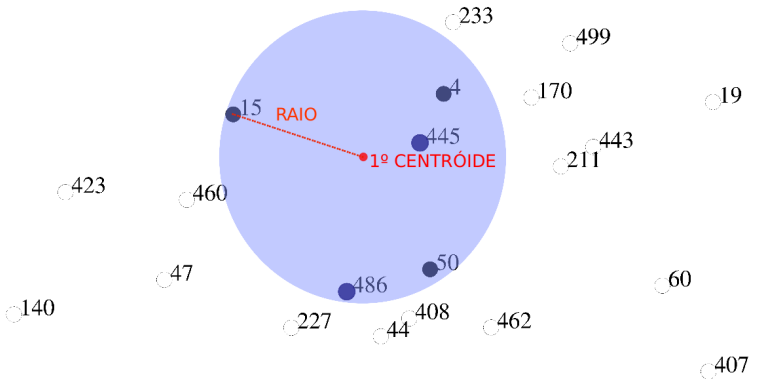


Figura 5.8: Heurística SSCCR - Sensores selecionados dentro do limite do raio

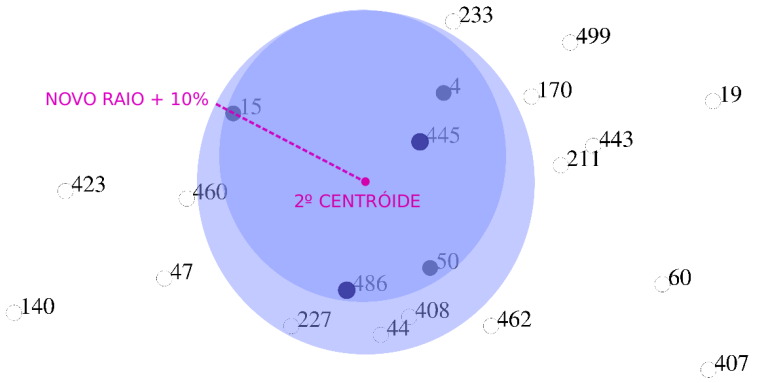


Figura 5.9: Heurística SSCCR - Cálculo do novo ponto centróide.

identificados por 445 e 486, é ultrapassado o limite de $t+1$ sensores que deverão participar do diagnóstico.

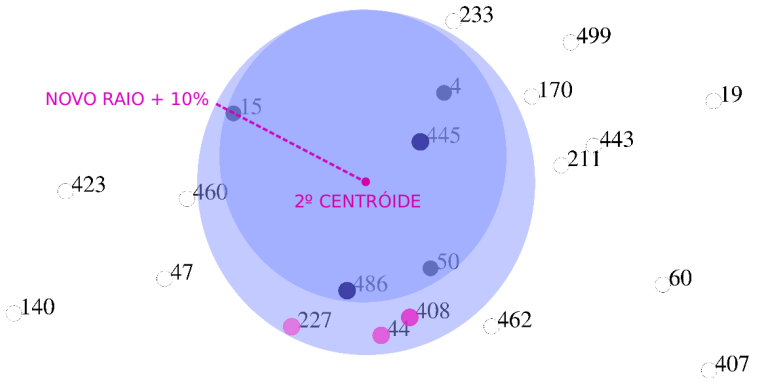


Figura 5.10: Heurística SSCCR - Verificação do número de sensores posicionados dentro do limite do novo raio calculado.

A fim de que sejam selecionados os sensores mais próximos entre si, o próximo passo da heurística SSCCR é obter um peso para cada sensor, entre os que estão envolvidos

nos passos anteriores. O peso de cada sensor é calculado pelo somatório da distância euclidiana dele para todos os outros envolvidos. A Figura 5.11, mostra as distâncias entre o sensor de número 277 até todos os demais.

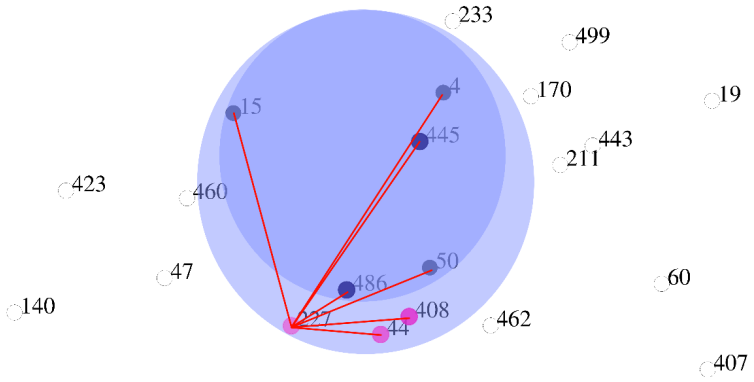


Figura 5.11: Heurística SSCCR - Cálculo do Peso para o sensor identificado por 277 dentro do maior raio.

Após obter todos os pesos para o conjunto dos sensores envolvidos, os mesmos são ordenados do menor para o maior peso. Posteriormente, os sensores que reportaram alarmes são escolhidos e removidos desse conjunto de sensores com pesos ordenados, de maneira a garantir que os mesmos participarão dos testes. Finalmente, são selecionados os primeiros $t+1$ sensores desse conjunto, identificados por 44, 408, 445, 486, conforme a Figura 5.12. Juntamente com os nós, identificados por 4, 15, 50, que reportaram alarmes são obtidos os $2t+1$ nós sensores que participarão do diagnóstico.

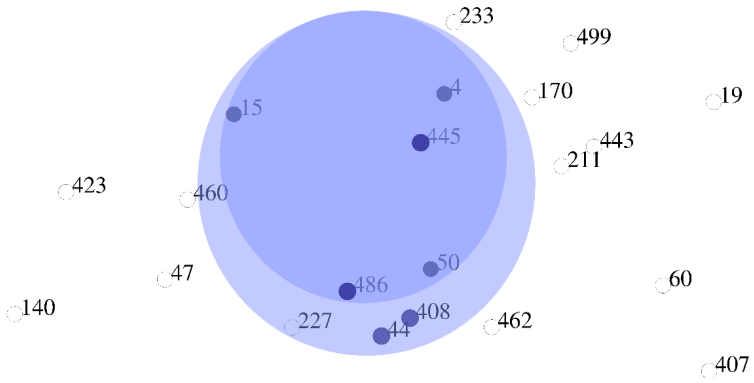


Figura 5.12: Heurística SSCCR - Seleção dos $t+1$ sensores com o menor peso calculado.

5.2.3 Inviabilidade do Fecho Convexo na Heurística SSCCR

Na heurística SSCCR, como primeiro passo, buscou-se o cálculo do envoltório convexo a partir dos alarmes reportados de maneira a permitir a seleção dos sensores internos, que não reportaram alarmes, a esse fecho convexo obtido. O algoritmo utilizado para obter o fecho convexo é conhecido como - Graham-Scan [13].

A possibilidade de utilizar o fecho convexo com o interesse de contabilizar o conjunto de sensores internos a esse envoltório foi proposta para identificar inicialmente o conjunto dos sensores mais próximos dos que geraram alarmes.

Para grande parte das simulações executadas, tanto para redes com 512 ou 1024 nós, o número de sensores que não reportaram alarmes, internos ao fecho convexo, apresentou-se insignificante, pois raramente foi obtido algum sensor que pudesse ser contabilizado no conjunto de sensores a serem escolhidos. A justificativa desta ocorrência é dada pelo parâmetro do fator da região de alarme, configurado em 1% do total da área de cobertura da rede. Procuramos apresentar nas simulações as características reais da disposição dos alarmes na rede, em função da detecção dos fenômenos monitorados. Como, por exemplo, na aplicação de monitoramento de foco de incêndio em florestas ou monitoramento de radiação em NPPs, sabe-se que a detecção inicial do evento é alertada por um número pequeno de sensores localizados em uma região específica da rede.

A Figura 5.13, apresenta a simulação de uma rede configurada com 512 sensores, com uma distribuição uniforme sobre uma área de cobertura de 512m x 512m, com 9 alarmes reportados para o observador central. É possível verificar que o conjunto de alarmes estão próximos um do outro, assim como, em um possível caso real de detecção de um determinado fenômeno.

A partir desta simulação, demonstramos na Figura 5.14 o comportamento da execução do algoritmo Graham-Scan na obtenção do fecho convexo a partir do conjunto de sensores que reportaram alarmes.

Após obter o fecho convexo, foram contabilizados os sensores internos a este envoltório obtido. Porém, para a maioria das simulações executadas não foram obtidos, por meio desse algoritmo, sensores que pudessem ser selecionados. Portanto, a implementação deste

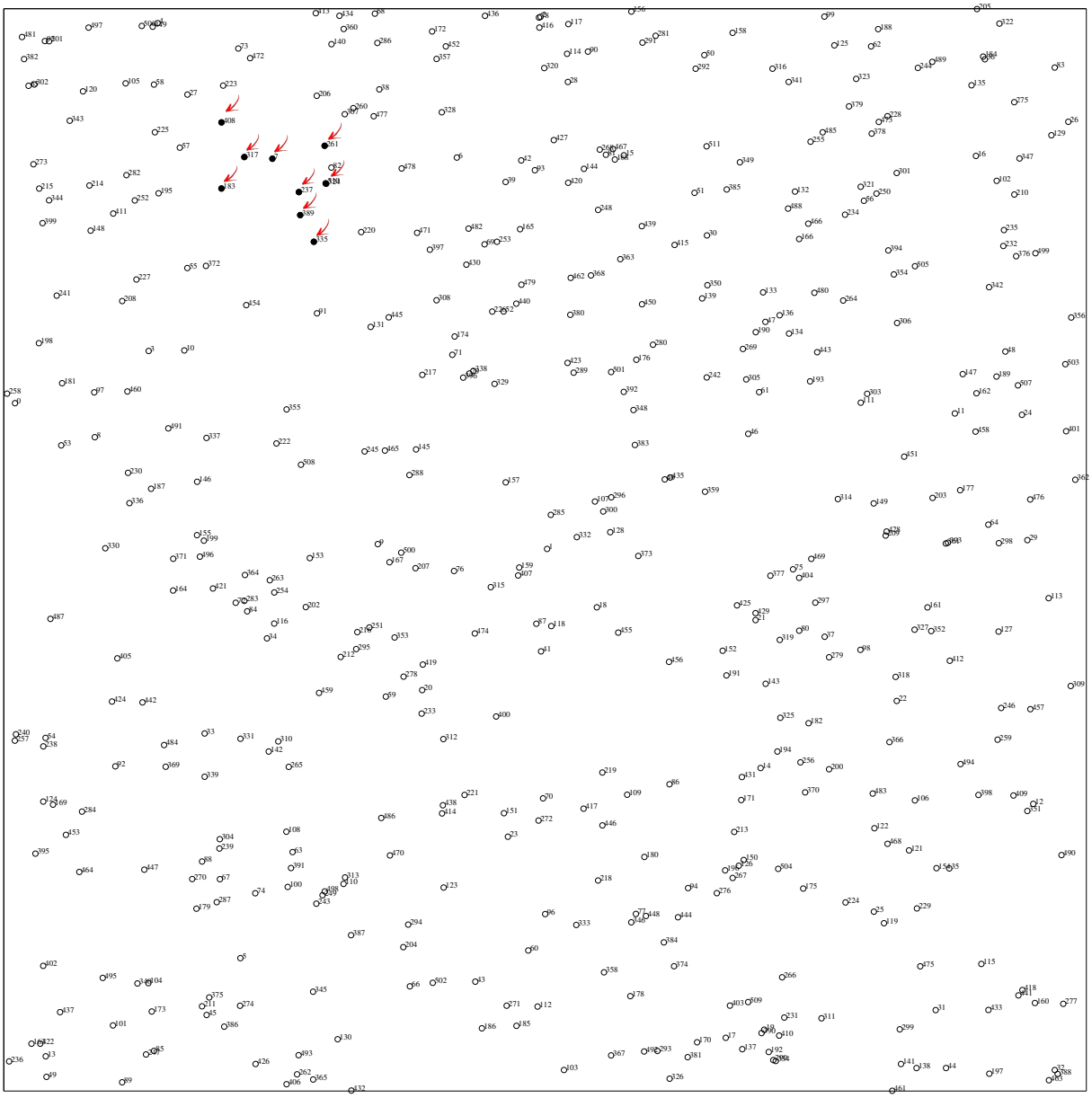


Figura 5.13: Exemplo de uma simulação para o posicionamento de 512 sensores e 9 alarmes em uma RSSF.

algoritmo como passo que antecede o cálculo do centroide e raio na heurística SSCCR apresentada mostrou-se ineficiente. Assim, optou-se por calcular diretamente o centroide, pois essa abordagem permitiu obter vários sensores próximos do conjunto de alarmes.

5.2.4 Ótimo Obtido Pela Programação Linear Inteira

Foi utilizada uma interface de alto nível, conhecida como LEMON [37], juntamente com o GLPK [25] como solucionador, para resolver a formulação do problema em pro-

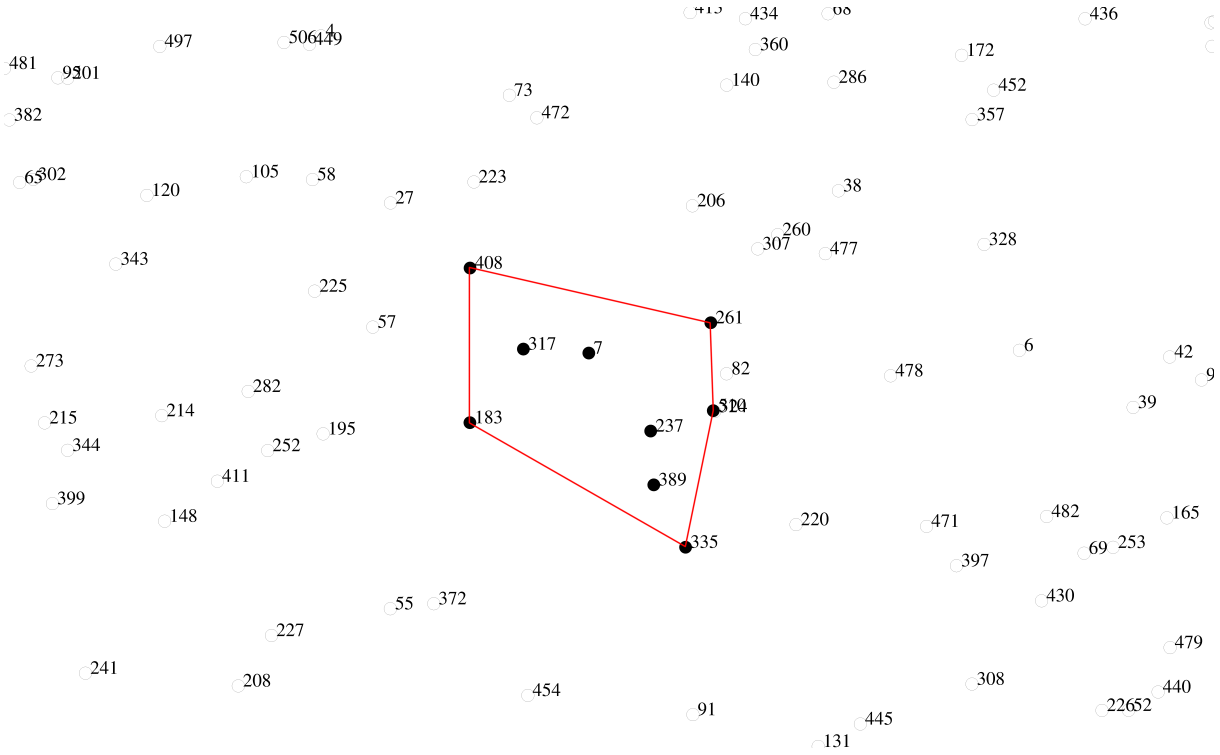


Figura 5.14: Fecho convexo para um conjunto de 9 alarmes.

gramação linear e possibilitar a comparação entre a solução ótima, assim obtida, e o conjunto de sensores selecionados para cada uma das heurísticas abordadas.

Foram aplicados os mesmos parâmetros de simulação das heurísticas apresentadas, de maneira a permitir a comparação da solução ótima obtida para cada conjunto de parâmetros.

A execução do algoritmo para solucionar a formulação do problema em programação linear inteira é intratável em termos computacionais. Conforme o número de alarmes aumentam nas simulações, maior o tempo de processamento da solução. Foi observado por meio dos resultados obtidos, que a partir de um número de 11 alarmes, a execução do experimento se tornou muito onerosa com relação ao tempo de processamento, o que acarretou na diminuição do número de amostras obtidas.

LEMON

O *Library for Efficient Modeling and Optimization in Networks* - LEMON é uma biblioteca de grafos de código aberto escrito em C++, que fornece implementações efi-

cientes de estruturas de dados e algoritmos comuns com foco em tarefas de otimização combinatória, relacionadas principalmente com grafos e redes. É mantido pelo grupo de pesquisa EGRES da *Eötvös Loránd University*, em Budapeste, Hungria [17, 37].

As ferramentas da biblioteca são concebidas para serem versáteis, convenientes e altamente eficientes. Podem ser combinadas facilmente para resolver problemas complexos de otimização da vida real. O LEMON também contém algumas ferramentas de otimização metaheurísticas e fornece uma interface geral de alto nível para vários Programas Lineares e de Programação Inteira Mista. O LEMON não implementa um solucionador, mas provê uma interface de alto nível para diversos solucionadores, como GLPK, ILOG CPLEX, CLP, CBC, SoPlex [17, 37].

A Programação Linear é um dos métodos mais importantes da pesquisa operacional. Inúmeros problemas de otimização podem ser formulados e resolvidos usando técnicas de Programação Linear. O objetivo principal do LEMON é dar suporte a pesquisadores e profissionais que trabalham na área da teoria dos grafos e de otimização de rede, por meio do estabelecimento de uma biblioteca de código aberto que seja mais adequada do que outras alternativas no mercado. LEMON apresenta uma interface simples e uma variedade de algoritmos complexos disponíveis para serem utilizados [17, 37].

LEMON também inclui várias ferramentas simples para medir o desempenho de algoritmos, que podem ser utilizadas para comparar diferentes implementações do mesmo problema [17, 37].

Foi concebido para ser multi-plataforma com suporte a uma ampla gama de sistemas operacionais e compiladores. Testado em sistemas Linux, Windows, OSX e AIX com os seguintes compiladores: GCC, Intel C++, IBM xlC, Visual C++ e MinGw. Tem como base o CMake como ambiente de compilação, o que permite integrar com várias IDEs, como Visual Studio, Eclipse ou CodeBlocks [17, 37].

GLPK

O *GNU Linear Programming Kit* - GLPK [25] é um pacote de software com um conjunto de rotinas escritas na linguagem de programação em C, disponível em código aberto

e livre de custo. É organizado na forma de bibliotecas que são compostas por algoritmos de pesquisa operacional bem conhecidos para solucionar problemas de otimização. O pacote GLPK inclui os seguintes componentes principais: métodos simplex primal e dual, método primal-dual de pontos interiores, método *branch-and-cut* e *branch-and-bound*, tradutor para GNU MathProg, interface de programação de aplicativo (API), *Stand-alone* LP e MIP solucionador. Tais métodos e algoritmos permitem resolver em grande escala Programas Lineares, Programação Inteira Mista e outros problemas relacionados [25].

GLPK não é uma aplicação, pois o mesmo não possui uma função *main*( ) e consequentemente não pode ser executado. Os usuários devem alimentar as informações do problema nas rotinas do algoritmo por meio de uma interface do solucionador, conhecida como API do GLPK. O programa responsável por realizar essa interface com a API é o programa *glpsol*, conhecido também como o solucionador [25].

O GLPK assume a seguinte formulação de um problema em programação linear:

- minimizar ou maximizar uma função objetivo:

$$z = c_1 x_{m+1} + c_2 x_{m+2} + \ldots + c_n x_{m+n} + c_0 \tag{5.2}$$

- sujeita às restrições lineares:

$$\begin{gathered} x_1 = a_{11} x_{m+1} + a_{12} x_{m+2} + \ldots + a_{1n} x_{m+n} \\ x_1 = a_{21} x_{m+1} + a_{22} x_{m+2} + \ldots + a_{2n} x_{m+n} \\ \ldots \\ x_m = a_{m1} x_{m+1} + a_{m2} x_{m+2} + \ldots + a_{mn} x_{m+n} \end{gathered} \tag{5.3}$$

- e limites de variáveis:

$$\begin{gathered} l_1 \leq x_1 \leq u_1 \\ l_2 \leq x_2 \leq u_2 \\ \ldots \\ l_{m+n} \leq x_{m+n} \leq u_{m+n} \end{gathered} \tag{5.4}$$

onde: $x_1, x_2, \ldots, x_m$ são variáveis auxiliares; $x_{m+1}, x_{m+2}, \ldots, x_{m+n}$ são variáveis estruturais; z é a função objetivo; $c_1, c_2, \ldots, c_n$ são coeficientes objetivos; c_0 é o termo constante da função objetivo; $a_{11}, a_{12}, \ldots, a_{mn}$ são os coeficientes das restrições; $l_1, l_2, \ldots, l_{m+n}$ são os limites inferiores das variáveis; $u_1, u_2, \ldots, u_{m+n}$ são os limites superiores das variáveis [39].

Para resolver um problema de programação linear é necessário encontrar os valores de todas as variáveis estruturais e auxiliares, os quais: (i) satisfazem todas as restrições lineares; (ii) estão dentro de seus limites; e (iii) fornecem o menor ou o maior valor para minimizar ou maximizar a função objetivo [39].

Essa formulação em LP resolve problemas de programação linear em sua forma comum, em que se permite obter soluções relaxadas do problema. Para problemas que exigem variáveis inteiras, o GLPK considera a formulação do problema em Programação Linear Inteira Mista ou também conhecida como MIP [39]. A diferença é que na formulação em MIP as variáveis estruturais só podem assumir valores inteiros. Nosso trabalho foi implementado por meio da formulação em MIP, pois as variáveis referentes aos sensores na rede devem assumir valores inteiros, impedindo que os nós sensores sejam decompostos na solução.

No Apêndice deste trabalho, é apresentado o algoritmo da formulação do problema em programação linear inteira, programado na linguagem C/C++ por meio da interface do LEMON e do solucionador GLPK.

5.2.5 Comparação e Discussão dos Resultados

Para compararmos o desempenho da heurística SSCCR com a heurística utilizada pelo algoritmo ODTA, bem como para obter a solução ótima por meio da programação linear inteira, foram utilizados os mesmos parâmetros de simulação, conforme especificado na Tabela 5.1.

O gráfico apresentado na Figura 5.15 demonstra o comportamento das duas heurísticas em relação à proporção da amostra que atingiu o ótimo, em uma configuração de rede com 512 e 1024 sensores, e também para diferentes números de alarmes reportados, dentro do seguinte conjunto: $\{3, 5, 7, 9, 11, 13, 15, 17, 19\}$.

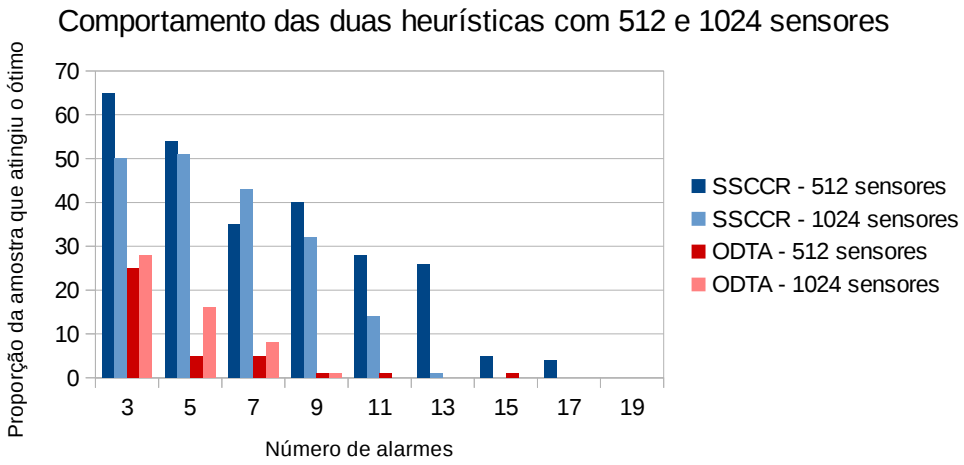


Figura 5.15: Simulação com 512 e 1024 sensores

O gráfico demonstra que a heurística SSCCR obteve uma melhora considerável na escolha do conjunto de $t + 1$ sensores com relação à heurística ODTA. Para simulações executadas em uma rede com 512 sensores e 3 alarmes, a proporção da amostra que atingiu o ótimo por meio da heurística SSCCR, totalizou 65%, sendo que a heurística ODTA apresentou 25%. Para as configurações da rede com 1024 sensores e 3 alarmes, a heurística SSCCR totalizou 50%, sendo que a heurística ODTA 28%.

Nos resultados apresentados para as simulações com 512 e 1024 sensores, foi observado que, o aumento do número de alarmes na rede degrada o desempenho das heurísticas. Mas é possível constatar, que no eixo da proporção da amostra que atinge o ótimo, a heurística SSCCR apresenta melhores resultados para todos os parâmetros simulados.

Nas Figuras 5.16 e 5.17 são apresentadas graficamente os resultados obtidos com intervalo de confiança de 95% para uma proporção populacional [14], com uma amostra de 100 configurações de redes, para simulações com 512 e 1024 sensores, e com diferentes números de alarmes. Os gráficos apresentam o comportamento das heurísticas na escolha do conjunto de sensores para o diagnóstico com relação à solução ótima encontrada.

Nota-se que a margem de erro aumenta consideravelmente para os resultados da heurística SSCCR com parâmetros de 512 sensores e 17 alarmes e 1024 sensores com 13 alarmes. Isso se deve em função da redução do número de amostras dos valores ótimos obtidos, visto que o custo do tempo computacional para processar tais resultados é one-

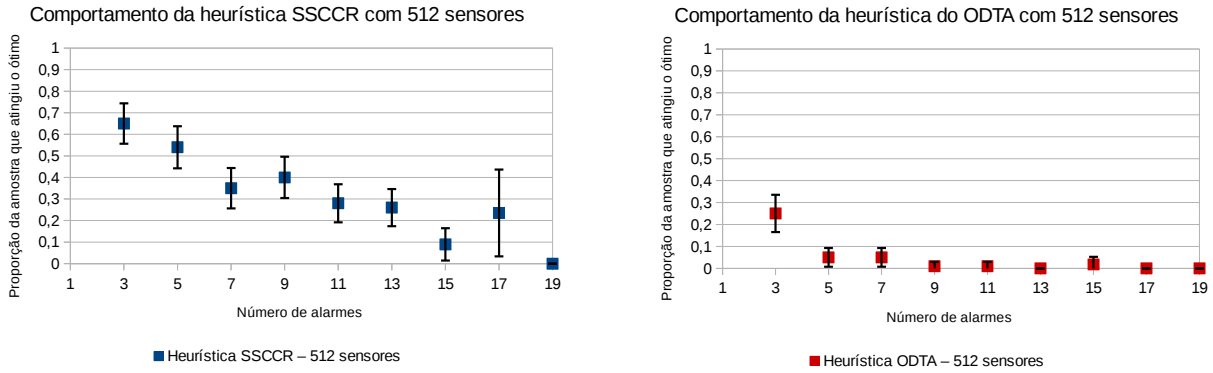


Figura 5.16: Resultados obtidos com intervalo de confiança de 95% na simulação com 512 sensores

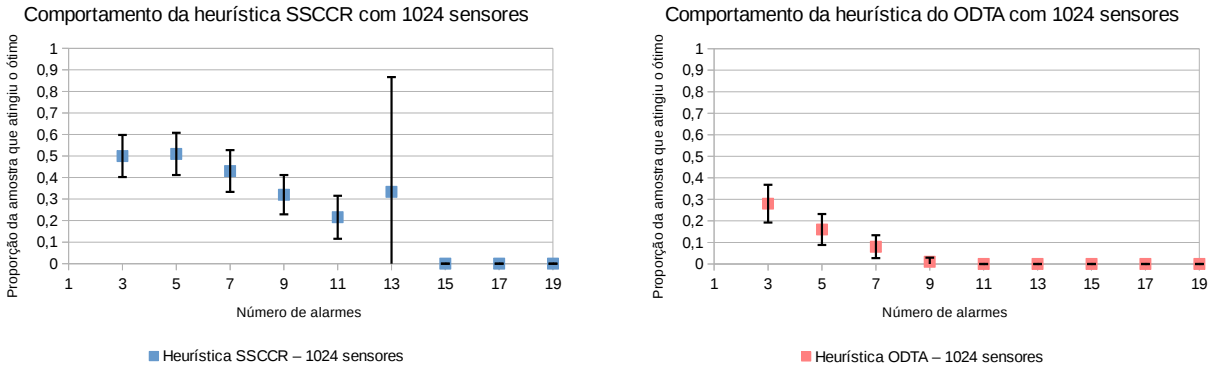


Figura 5.17: Resultados obtidos com intervalo de confiança de 95% na simulação com 1024 sensores

roso. Por outro lado, a heurística empregada pelo algoritmo do ODTA não apresenta esse aumento na margem de erro, pois nos parâmetros de rede com um número de alarmes acima de 9 sensores, esta heurística tem uma alta degradação no desempenho das amostras simuladas, logo ocasionalmente alcança o ótimo para fins de comparação.

Nos gráficos da Figura 5.18 são apresentados os resultados das simulações agrupados por três classes, 100%, 99,99% até 80% e menor que 80%, com relação a proporção da amostra que atingiu os resultados entre os intervalos de frequência definidos. Gráficos do lado esquerdo da figura apresentam as simulações com 512 nós sensores e do lado direito as simulações com 1024, com diferentes números de alarmes. Por meio destes gráficos podemos observar que a heurística SSCCR apresenta melhores resultados que a heurística do ODTA, em termos de distâncias entre o conjuntos de nós sensores selecionados, indiferentemente do número de nós sensores, ou seja, a escalabilidade de nós sensores na rede não representa degradação do desempenho.

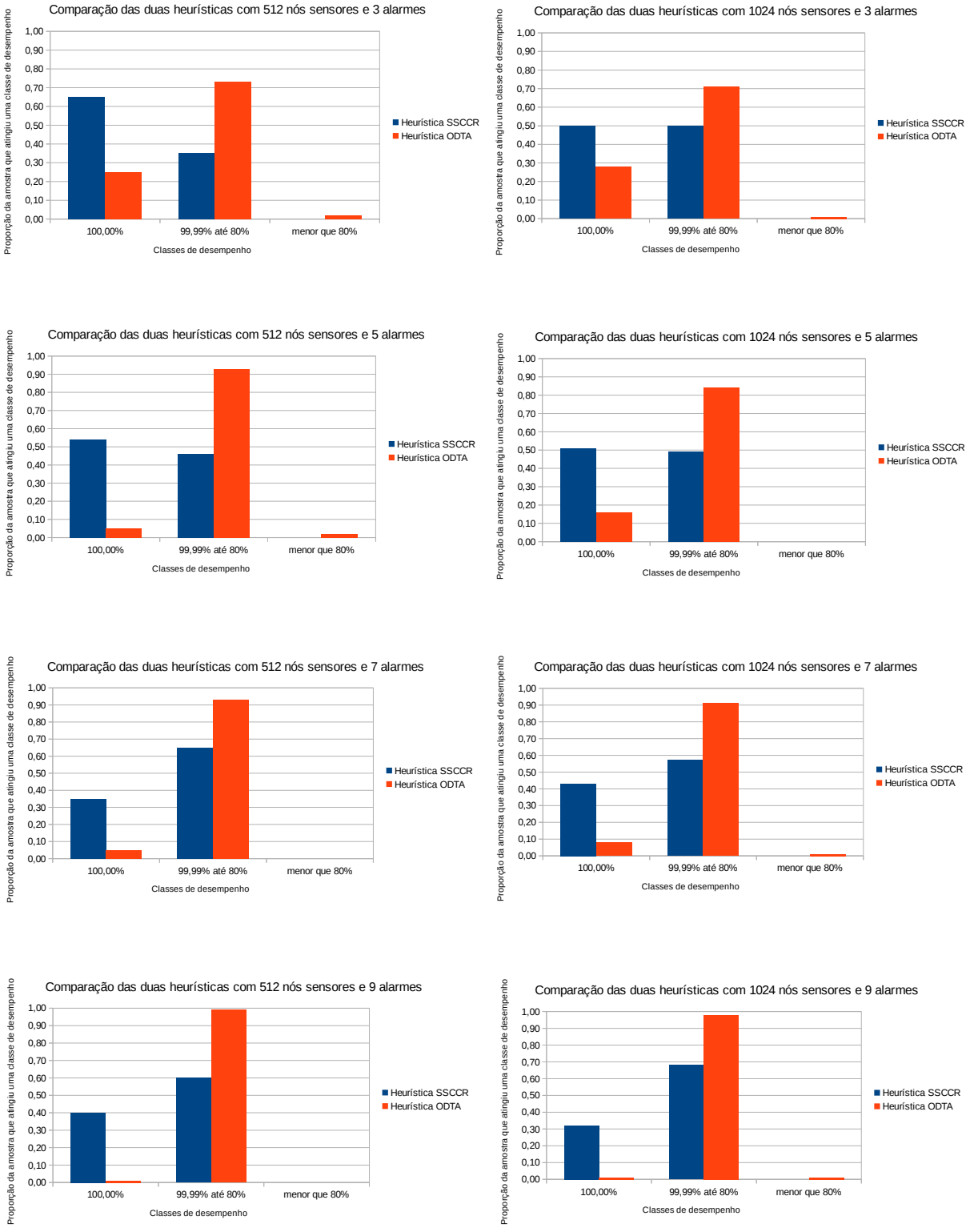


Figura 5.18: Resultados das simulações agrupados por três classes de desempenho com 512 e 1024 nós sensores e diferentes números de alarmes

Para comparar o desempenho das heurísticas, por meio das amostras simuladas, em relação a frequência em que elas atingiram um resultado ótimo, bem como avaliar o

desempenho mesmo quando não atingiram o ótimo esperado, dividimos arbitrariamente a apresentação desses resultados em três classes de desempenho. A primeira classe de desempenho, denominada de 100%, representam os resultados em que as heurísticas atingiram o ótimo esperado. A segunda classe de desempenho, representa os resultados em que ambas heurísticas não atingiram o ótimo esperado, mas que apresentaram resultados da proporção da amosta que atingiu o intervalo de frequência defino entre 99,99% até 80%. Essa classe tem objetivo de apresentar os resultados com valores da proporção em que a amostra atingiu acima de 80%. Na terceira classe de desempenho, definida pelo intervalo de frequência de 80% a 0%, são apresentados os resultados que também não atingiram o ótimo esperado, pois representam os resultados da proporção da amostra em que foi atingido valores inferiores que 80%, no qual consideramos ruins.

No primeiro gráfico do lado esquerdo, apresenta-se a comparação das duas heurísticas nas simulações com 512 nós sensores e 3 alarmes. Para a proporção da amostra que atingiu a classe de desempenho de 100%, ou seja, proporção da amostra que atingiu o ótimo, a heurística SSCCR totalizou 65% dos casos, sendo que a heurística ODTA apresentou 25% dos casos. Na classe de desempenho de 99,99% até 80%, a heurística SSCCR totalizou 35% dos casos e a heurística ODTA resultou em 73% dos casos. Na classe menor que 80% somente a heurística ODTA apresentou 2% dos casos.

Na classe de desempenho de 100%, que representa a proporção da amostra que atingiu o ótimo, observamos que em nenhum dos gráficos a heurística SSCCR apresentou resultados com desempenho inferior a heurística do ODTA.

Resultados em que a proporção da amostra não atingiu o ótimo na execução da heurística SSCCR, os valores ficaram em média acima dos 90% e nunca inferior a 80%. Ao contrário da heurística do ODTA que em algumas situações apresenta resultados com desempenho inferior, representados pela classe menor que 80% nos gráficos.

As Figuras 5.19 e 5.20 apresentam a comparação do tempo médio de execução das simulações para ambas heurísticas com o tempo médio de execução para obter-se os resultados ótimos através da programação linear. As heurísticas ODTA e SSCCR levam poucos segundos em tempo computacional para executar as simulações, mesmo com a

presença de um número maior de alarmes na rede.

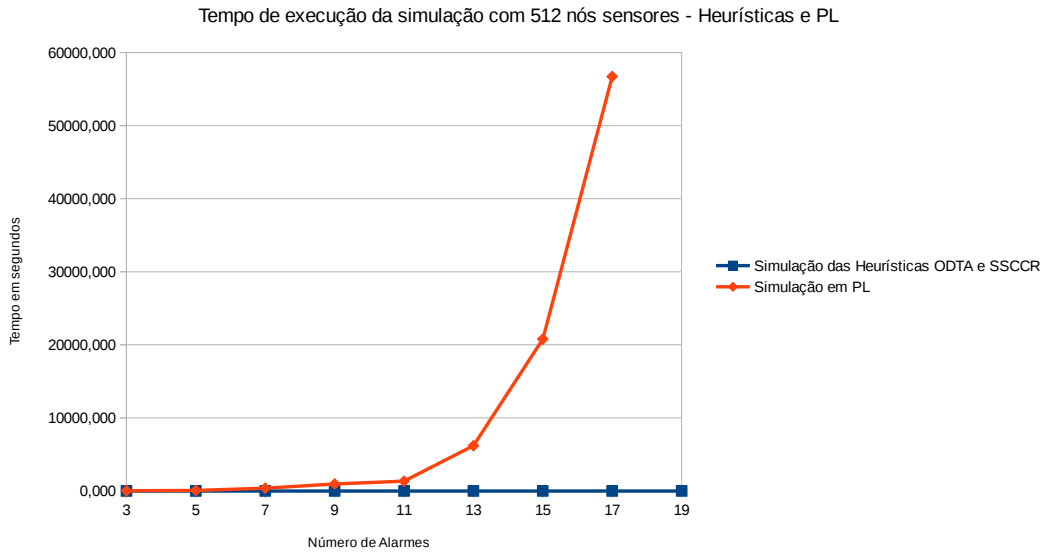


Figura 5.19: Comparação do tempo de execução das simulações para ambas heurísticas comparado a simulação da PL - 512 nós sensores

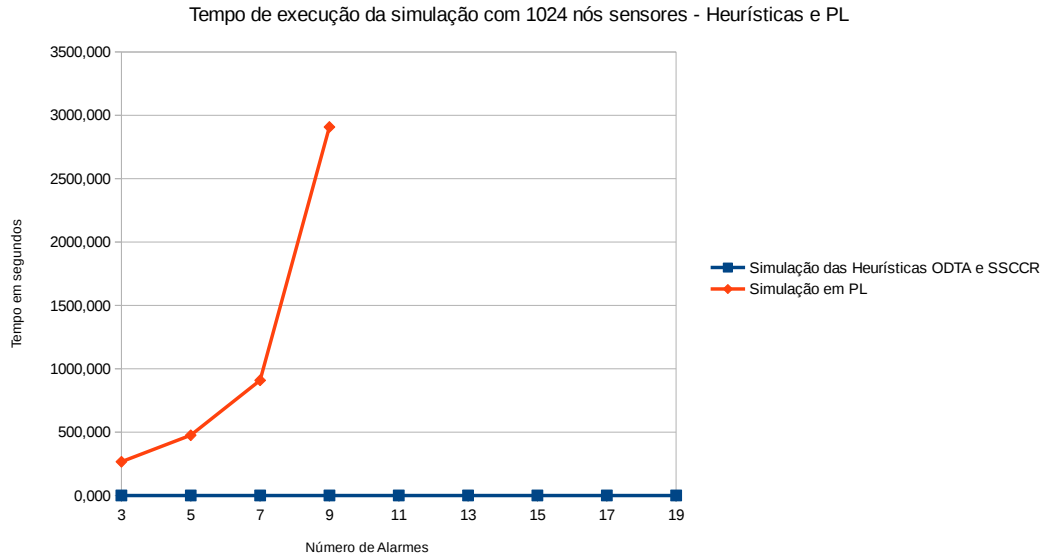


Figura 5.20: Comparação do tempo de execução das simulações para ambas heurísticas comparado a simulação da PL - 1024 nós sensores

Por outro lado, o tempo computacional exigido para executar as simulações por meio da programação linear é muito superior, como pode ser observado nos gráficos, tanto para simulações com rede de 512 ou 1024 sensores. Conforme os alarmes nas simulações aumentam, maior o tempo computacional para se obter os resultados. Nas simulações

de rede com 512 sensores, a partir de 17 alarmes reportados a execução do experimento tornou-se inviável, pois o tempo computacional exigido aumentou consideravelmente. Da mesma forma para simulações com 1024 sensores, a partir de 9 alarmes a execução do experimento poderia demorar meses, devido à alta complexidade e número de cálculos a serem realizados por meio da programação linear na obtenção dos resultados ótimos.

Os resultados apresentados por meio dos gráficos demonstram que a escolha do conjunto de $t + 1$ nós sensores, juntamente com os t alarmes para o assinalamento de testes, a ser empregado pelo algoritmo ODTA, pode ser aprimorado em termos de distâncias entre os nós sensores com a utilização da heurística SSCCR, e contribuir para eficiência energética na execução do diagnóstico de falhas em RSSFs, visto que a distância entre os nós sensores é um fator determinante no consumo de energia.

CAPÍTULO 6

CONCLUSÃO

Neste trabalho, foram apresentados o conceito e o objetivo do Diagnóstico em Nível de Sistema com ênfase no primeiro modelo, denominado PMC, com aplicação em Redes de Sensores Sem Fio especificamente na área de tolerância a falhas.

Foi considerada uma RSSF composta por nós sensores, distribuídos de forma homogênea sobre um campo de sensoriamento, com finalidade de monitorar dado fenômeno físico, capaz de reportar mensagens de alarme para uma estação base. Esta estação base tem a responsabilidade de coletar tais informações e prover o diagnóstico do sistema, com base nas características do modelo PMC. Com objetivo de garantir que os alarmes não são falsos positivos, a estação base solicita um conjunto de testes entre os sensores presentes na região onde o alarme foi gerado, e então um algoritmo de diagnóstico em nível de sistema obtém e analisa a síndrome do sistema, e por fim identifica os sensores falhos.

O problema de selecionar um número mínimo de sensores, com a melhor posição geográfica para o assinalamento ótimo de testes, e que satisfaçam as condições de diagnosticabilidade do sistema, tem características de ser um problema computacionalmente intratável. Devido à ausência de uma prova formal neste trabalho de que o problema abordado é NP-Difícil, foi apresentada a heurística SSCCR.

A heurística SSCCR foi concebida com o intuito de aprimorar a heurística empregada no algoritmo ODTA para a escolha do conjunto de $t+1$ sensores, com as menores distâncias entre si e do conjunto dos t nós sensores que reportaram alarmes. Como a distância entre nós sensores é um fator determinante do consumo de energia no processo de comunicação, a escolha do conjunto dos nós sensores mais próximos geograficamente entre si pode representar eficiência energética no processo de diagnóstico de falhas.

Conformes os resultados apresentados neste trabalho, é possível observar o comportamento das heurísticas SSCCR e ODTA com os resultados do ótimo obtido pela formulação

do problema em programação linear inteira. A heurística SSCCR apresentou melhora no desempenho em relação a heurística do ODTA, assim como o alcance da solução ótima em tempo polinomial em alguns casos. É importante ressaltar, que na execução das simulações para se obter o conjunto ótimo em termos de distâncias, por meio da programação linear, demandou-se um tempo maior (exponencial) e dependendo do número de alarmes na simulação o tempo computacional de execução tornou-se impraticável.

Neste trabalho buscou-se o aprimoramento da heurística ODTA com a utilização do fecho convexo como primeira tentativa de se obter o conjunto com os $t + 1$ nós sensores geograficamente próximos de si e do conjunto de alarmes interno ao envoltório convexo. Devido as particularidades de algumas aplicações, não foi possível obter sucesso no uso do fecho convexo, em consequência de que a detecção inicial de eventos é alertada por um pequeno conjunto de nós sensores em uma região específica da rede, o que não permitiu selecionar na maioria das simulações executadas nenhum nó sensor para o assinalamento de testes.

Na descrição da heurística SSCCR inicia-se com o cálculo do ponto centróide, a utilização do raio juntamente com a circunferência, e as iterações necessárias para obter os nós sensores mais próximos de si e dos alarmes, bem como o recálculo do ponto centróide para reequilibrar o conjunto de sensores selecionados e até obter os $t + 1$ nós sensores. Desta forma, a heurística apresenta melhores resultados do que a heurística do ODTA, que por outro lado apenas obtém o ponto central do menor retângulo que envolve o conjunto de alarmes, e a partir dele são obtidos os $t+1$ sensores mais próximos a partir desse ponto calculado.

Para que a heurística SSCCR possa ser executada, há uma restrição em relação ao número mínimo de três alarmes que devem ser passados como entrada para o algoritmo, pois o cálculo inicial do centróide, raio e circunferência só possui efetividade para poder selecionar os sensores mais próximos, se existir uma lista de no mínimo três pontos com coordenadas cartesianas, de maneira que a circunferência calculada seja do tamanho suficiente para alcançar os $t + 1$ nós sensores próximos.

A definição dos parâmetros de 512 e 1024 nós sensores distribuídos na área de co-

bertura da rede, tem objetivo de mostrar o comportamento da heurística em simulações em que a densidade dos nós sensores é aumentada. Ambas heurísticas não apresentam nenhuma degradação em tempo computacional para devolver os resultados em função da escalabilidade da rede, em virtude de que a execução dos cálculos é centralizada em uma região específica da rede. Por outro lado, em redes mais densas, por exemplo: 1024, 2048 e 4096, onde os nós sensores estão praticamente muito próximo um dos outros, o consumo de energia que será demandado para realização do diagnóstico é inferior do que em redes menos densas.

A solução apresentada neste trabalho não conseguiu atingir o ótimo esperado em todas as simulações executadas, devido à complexidade do problema na escolha de um conjunto de sensores que estejam próximos entre si, em termos de distância geográfica, e dos sensores que reportaram alarmes. Mas foi possível alcançar resultados iguais e na maioria melhores que a heurística do ODTA em todas as simulações executadas.

Em todas as simulações foram utilizadas apenas a distribuição homogênea dos nós sensores na área de cobertura. Não foram utilizadas outras distribuições, mas supõe-se que o desempenho das heurísticas pode sofrer degradação.

Como trabalhos futuros, pretende-se implementar a heurística SSCCR for ODTA por meio de simuladores específicos para RSSF presentes na literatura, e compará-la com outras heurísticas existentes. Podem ser utilizados diferentes parâmetros de distribuição dos nós sensores na área de cobertura e observar se a heurística SSCCR sofrerá degradação no desempenho. Realizar um estudo mais aprofundado sobre a possibilidade de agregar a heurística SSCCR juntamente com o algoritmo ODTA na arquiteura MANNA dentro da área funcional de configuração, especificamente no tratamento de falhas na rede, de maneira a fornecer uma solução viável para execução do diagnóstico com eficiência energética. Como abordagens futuras pode ser realizada uma nova busca pela solução ótima do problema abortado neste trabalho ou apresentar uma prova de que o problema é NP-Difícil.

BIBLIOGRAFIA

[1] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. Wireless sensor networks: a survey. *Computer Networks – Elsevier vol. 38* (2002), 393–422.

[2] Akyildiz, I. F., and Vuran, M. C. *Wireless sensor networks*, 1 ed. John Wiley e Sons Ltd, Singapore by Markono, 2010.

[3] Alrajei, N., and Fu, H. A survey on fault tolerance in wireless sensor networks.

[4] Anastasi, G., Conti, M., Di Francesco, M., and Passarella, A. Energy conservation in wireless sensor networks: A survey. *Ad hoc networks 7*, 3 (2009), 537–568.

[5] Avizienis, A., Laprie, J. C., Randell, B., and Landwehr, C. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing vol. 1* (2004), 11–33.

[6] Bagchi, A., and Hakimi, S. L. An optimal algorithm for distributed system level diagnosis. *IEEE* (1991), 214–221.

[7] Baronti, P., Pillai, P., Chook, V. W., Chessa, S., Gotta, A., and Hu, Y. F. Wireless sensor networks: A survey on the state of the art and the 802.15.4 and zigbee standards. *Computer Communications – Elsevier vol. 30* (2007), 165–1695.

[8] Barsi, F., Grandoni, F., and Maestrini, P. A theory of diagnosability of digital systems. *IEEE Transactions On Computers vol. C-25* (1976), 585–593. n. 06.

[9] Bianchini, R., and Buskens Jr, R. W. Implementation of online distributed system-level diagnosis theory. *IEEE Transactions on Computers vol. 41*, 5 (1992), 616–626.

[10] Bianchini, R., and Jr, R. B. An adaptive distributed system-level diagnosis algorithm and its implementation. *IEEE Conference Publications - Fault-Tolerant Computing - FTCS-21* (1991), 222–229.

[11] Blough, D. M., and Brown, H. W. The broadcast comparison model for on-line fault diagnosis in multicomputer systems: Theory and implementation. *IEEE Transactions on Computers vol. 48*, 5 (maio 1999), 470–493.

[12] Chen, G., Hanson, S., Blaauw, D., and Sylvester, D. Circuit design advances for wireless sensing applications. *Proceedings of the IEEE vol. 8* (2010), 1808 – 1827.

[13] Cormen, T. H. *Introduction to algorithms*. MIT press, 2009.

[14] da Cunha, S., and Carvajal, S. *Estatistica Basica - a Arte de Trabalhar com Dados*. Campus, 2009.

[15] Dahbura, A. T., and Masson, M. An $\theta(n^{2.5})$ fault identification algorithm for diagnosable systems. *IEEE Transactions on Computers vol. 100*, n. 6 (1984), 486–492.

[16] Dargie, W., and Poellabauer, C. *Fundamentals of wireless sensor networks: theory and practice*, 1 ed. John Wiley e Sons Ltd, Singapore by Markono, 2010.

[17] Dezső, B., Jüttner, A., and Kovács, P. Lemon–an open source c++ graph template library. *Electronic Notes in Theoretical Computer Science 264*, 5 (2011), 23–45.

[18] Duarte Jr., and Nanya, T. A hierarchical adaptive distributed system-level diagnosis algorithm. *IEEE Transactions on Computers vol. 47*, n. 1 (1998), 34–45.

[19] Duarte Jr., Ziwich, R. P., and Albini, L. C. P. A survey of comparison-based system-level diagnosis. *ACM Computing Surveys vol. 43* (2011), 01–56.

[20] Duarte Jr, E. P., Nanya, T., Noguchi, S., and Mansfield, G. Non-broadcast network fault-monitoring based on system-level diagnosis. In *Integrated Network Management V* (1997), Springer, pp. 597–609.

[21] Duarte Jr., E. P., Weber, A., and Fonseca, K. Distributed diagnosis of dynamic events in partitionable arbitrary topology networks. *Parallel and Distributed Systems, IEEE Transactions on 23*, 8 (Aug 2012), 1415–1426.

[22] Gao, S., Tang, Y., and Qu, X. Lssvm based missing data imputation in nuclear power plant's environmental radiation monitor sensor network. *fifth International Conference on Advanced Computational Intelligence(ICACI)* (2012), 479–484.

[23] Garber, L. News briefs: Google plans smart contact lenses, 2014. Disponível em: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6766139. Acesso em jul/2014.

[24] Garey, M. R., and Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, 1 ed. W. H. Freeman e Co., New York, NY, USA, 1979.

[25] GLPK. Glpk - gnu linear programming kit. Disponível em: http://www.gnu.org/software/glpk/. Acesso em Nov/2014.

[26] Gomaa, R., Adly, I., Sharsher, K., Safwat, A., and Ragai, H. Zigbee wireless sensor network for radiation monitoring at nuclear facilities. *IEEE - IFIP WMNC2013* (2013), 01–04.

[27] Gomaa, R., Adly, I., Sharsher, K., Safwat, A., and Ragai, H. Real time radiological monitoring of nuclear facilities using zigbee technology. *IEEE Sensors Journal* (2014), 01–06.

[28] Hakimi, S. L., and Amin, A. T. Characterization of connection assignment of diagnosable systems. *IEEE Transactions On Eletronic Computers* (1974), 86–88.

[29] Hakimi, S. L., and Nakajima, K. On adaptive system diagnosis. *IEEE Transactions On Eletronic Computers vol. c-33* (Mar. 1984), 234–240. n. 03.

[30] Hashemian, H., Kiger, C. J., Riggsbee, E. T., Phipps, K. O., and O'Hagan, R. Wireless sensors for equipment health and condition monitoring in nuclear power plants. *IEEE - Future of Instrumentation International Workshop (FIIW)* (2011), 83–86.

[31] Hosseini, S. H., Kuhl, J. G., and Reddy, S. M. A diagnosis algorithm for distributed computing systems with dynamic failure and repair. *IEEE Trans. Comput. vol. 33*, 3 (Mar. 1984), 223–233.

[32] Huang, F., and Sun, T. Design and implementation of radiation dose monitoring system based on wireless sensor network. *2011 International Conference on Future Information Technology vol. 13* (2011), 430–434.

[33] Ilyas, M., and Mahgoub, I. *Handbook of sensor networks : compact wireless and wired sensing systems*, 1 ed. CRC Press LCC, USA, 2004.

[34] Krishnamachari, B. *Networking Wireless Sensors*, 1 ed. Cambridge University Press, New York, EUA, 2005.

[35] Kuorilehto, M., and et al. *Ultra-low energy wireless sensor networks in practice: Networks in Practice Theory, Realization and Deployment*, 1 ed. John Wiley e Sons Ltd, The Atrium, Southern Gate, Chichester, 2007.

[36] Kutzke, A., Weber, A., and Chessa, S. An optimal design testing assignment for wireless sensor networks. *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos vol. 18* (2012), 145–158.

[37] LEMON. The lemon graph library project by egervary research group on combinatorial optimization. Disponível em: https://lemon.cs.elte.hu/trac/lemon. Acesso em Nov/2014.

[38] Ma, J., and Jiang, J. Applications of fault detection and diagnosis methods in nuclear power plants: A review. *Progress in Nuclear Energy* (2011), 255–266.

[39] Makhorin, A. Reference manual for glpk - version 4.45. DRAFT, December 2010. Disponível em: http://kam.mff.cuni.cz/ elias/glpk.pdf. Acesso em Jan/2014.

[40] Matin, M. A. Overview of wireless sensor network - technology and protocols, 2012. Disponível em: http://www.intechopen.com/books/wireless-sensor-networks-technology-and-protocols/overview-of-wireless-sensor-network. Acesso em abr/2014.

[41] Musznicki, B., and Zwierzykowski, P. Survey of simulators for wireless sensor networks. *International Journal of Grid and Distributed Computing 5*, 3 (2012), 23–50.

[42] Oh, S. A radiation source detection around atomic power station. *Advanced Science and Tecnology Letters vol. 62* (2014), 29–32. Sensor 2014.

[43] Pantazis, N. A., Nikolidakis, S. A., and Vergados, D. D. Energy-efficient routing protocols in wireless sensor networks: A survey. *IEEE COMMUNICATIONS SURVEYS & TUTORIALS 15*, 2 (Second Quarter 2013), 551–591.

[44] Patwari, N., III, A. O. H., Perkins, M., Correal, N. S., and O'dea, R. Relative location estimation in wireless sensor networks. *IEEE Transactions on Signal Processing vol. 51*, 8 (2008), 2137–2148.

[45] Potdar, V., Sharif, A., and Chang, E. Wireless sensor networks: a survey. *International Conference on Advanced Information Networking and Applications Workshops – IEEE - Computer Society* (2009), 636 – 641.

[46] Preparata, F. P., Metze, G., and Chien, R. T. On the connection assignment problem of diagnosable systems. *IEEE Transactions On Eletronic Computers vol. EC-16* (1967), 848–854. n. 06.

[47] Pughat, A., and Sharma, V. A survey on dynamic power management approach in wireless sensor networks. In *Power India International Conference (PIICON), 2014 6th IEEE* (2014), IEEE, pp. 1–6.

[48] Rangarajan, S., Dahbura, A. T., and Ziegler, E. A. A distributed system-level diagnosis algorithm for arbitrary network topologies. *IEEE Transactions on Computers vol. 44*, n. 2 (1995), 312–334.

[49] Rappaport, T. S. Wireless communications principles e p practice, 2002.

[50] Rawat, P., Singh, K. D., Chaouchi, H., and Bonnin, J. M. Wireless sensor networks: a survey on recent developments and potential synergies. *J Supercomput – Springer vol. 68* (2013), 01 – 48.

[51] Ruiz, L. B., Nogueira, J. M., and Loureiro, A. A. Manna: A management architecture for wireless sensor networks. *Communications Magazine, IEEE 41*, 2 (2003), 116–125.

[52] Santi, P., and Chessa, S. Crash faults identification in wireless sensor networks. *Comput Commun n. 25 vol. 14* (2002), 1273–1282.

[53] Seidel, S. Y., and Rappaport, T. S. 914 mhz path loss prediction models for indoor wireless communications in multifloored buildings. *Antennas and Propagation, IEEE Transactions on 40*, 2 (1992), 207–217.

[54] Siqueira, I. G., Ruiz, L. B., Loureiro, A. A., and Nogueira, J. M. Coverage area management for wireless sensor networks. *International Journal of Network Management 17*, 1 (2007), 17–31.

[55] Souza, L. M. S., Vogt, H., and Beigl, M. A survey on fault tolerance in wireless sensor networks. *Universitat Karlsruhe* (2007), 01–11.

[56] Stahl, M., Buskens, R., and Bianchini Jr., R. Simulation of the adapt on-line diagnosis algorithm for general topology networks. *IEEE* (1992), 180–187.

[57] Subbiah, A., and Blough, D. M. Distributed diagnosis in dynamic fault environments. *IEEE Transactions on Parallel and Distributed Systems vol. 15*, 5 (2004), 453–467.

[58] Toh, C. K. *Ad Hoc Mobile Wireless Networks: Protocols and Systems*, 1 ed. Prentice Hall, Singapore by Markono, 2001.

[59] Venkataraman, G., S, S. E., and Thambipillai, S. Energy-efficient cluster-based scheme for failure management in sensor networks. *The Institution of Engineering and Technology - IET Comun vol. 2* (2008), 528–537.

[60] Weber, A., Kutzke, A., and Chessa, S. Diagnosability evaluation for a system-level diagnosis algorithm for wireless sensor networks. *Fifteenth IEEE Symposium on Computers and Communications* (2010), 241–244.

[61] Weber, A., Kutzke, A., and Chessa, S. Energy-aware test connection assignment for the self-diagnosis of a wireless sensor network. *Journal of the Brazilian Computer Society (Impresso) vol. 18* (2012), 19–27.

[62] Weisstein, E. W. Mathworld wolfram research. geometric centroid, 2002. Disponível em: http://mathworld.wolfram.com/GeometricCentroid.html. Acesso em jun/2014.

[63] Williamson, D. P., and Shmoys, D. B. *The Design of Approximation Algorithms*, 1 ed. Cambribge University Press, Electronic web edition, 2010. http://www.designofapproxalgs.com/book.pdf.

[64] Yick, J., Biswanath, Mukherjee, and Ghosal, D. Wireless sensor networks survey. *Computer Networks – Elsevier vol. 52* (2008), 2292 – 2330.

[65] Z., Taghikhaki, and Sharifi, M. A trust-based distributed data fault detection algorithm for wireless sensor networks. *Proceedings of International Workshop on Internet and Distributed Computing Systems - IDCS'08 vol. 2* (2008), 1–6.

APÊNDICE

Neste apêndice, apresentamos o algoritmo da formulação do problema em programação linear inteira, programado na linguagem C/C++ por meio da interface do LEMON e solucionador GLPK, que tem como objetivo selecionar um conjunto ótimo de sensores que apresentem o menor custo energético do ponto de vista da distância entre os sensores.

```
1  //>>>>>>>>>>>>>>>>>>>>>>>>> Programa Linear Inteiro  <<<<<<<<<<<<<<<<<<<<<<<<<
2
3  #include <lemon/mip_glpk.h>
4
5  using namespace lemon;
6
7  vector<double> * Network::otimal_PLI() {
8
9  Mip mip;                  //typedef MipGlpk Mip - Solucionador PLI padrão
10
11 /* --- CONFIGURAÇÃO INICIAL --- */
12 int V;                    // Número de nós
13 V = n;                    // V recebe o número de nós do parâmetro da simulação
14 int E = V*(V-1)/2;        // Número de arestas
15 double d[E];              // Valor da distância de cada aresta
16 int T[t];                 // Conjunto dos nós que geraram o alarme
17 edge_t S[(2*t)+1];        // S contém T + os nós que realizarão o diagnóstico
18
19 /* --- Variaveis diversas --- */
20 int k=0;
21 int i, j;
22 Pos p, q;
23 Node *a, *b;
24
25 /* --- NÓS (xi)--- */
26 Mip::Col x[V];
27 for (i = 0; i < V; i++){
28   x[i] = mip.addCol();
29 }
30
31 /* --- Arestas ligadas aos Nós com Alarme - Valores gerados por simulador --- */
32 Mip::Col y[V][V];
33 for (i=0; i<V-1; ++i){
34   for (j=i+1; j<V; ++j){
35     y[i][j] = mip.addCol();
36   }
37 }
38
39 /* --- Atribuição do conjunto T --- *
40  * O conjunto T é formado pelos índices dos nós que geraram alarmes *
41  * --- Valores gerados por simulador --- */
42
43 for (i = 0; i < t; i++) {
44   a = nodes.at(alarmNodes->at(i));
45   p = a->getPos();
46   T[i] = alarmNodes->at(i);
47 }
48
49 /* --- Matriz com os valores das distâncias entre os vértices --- */
50
51 for (i = 0; i < V-1; i++) {
52   a = nodes.at(i);
53   p = a->getPos();
54   for (j = i + 1; j < V; j++) {
55     b = nodes.at(j);
56     q = b->getPos();
57     distancias[i][j] = getPDist(p, q);
58     }
59 }
60
61 /* --- Restrição 0 --- */
62 /* --- Configurando o tipo de variável para NÓS (xi) - INTEGER --- */
63 /* --- Configurando o tipo de variável para Arestas ligadas aos Nós com Alarme ---*/
64 /* --- Valores gerados por simulador - REAL --- */
65
66 for (i=0; i<V; i++) {
67   mip.colType(x[i], Mip::INTEGER);
68   for (j = i+1; j<V; j++) {
69   mip.colUpperBound(y[i][j], Mip::INTEGER);
70   }
71 }
72
73 /* Inteiro */
74 for (i=0; i<V; i++) {
75   mip.colLowerBound(x[i],0);
76   mip.colUpperBound(x[i],1);
77   for (j = i+1; j<V; j++) {
78     mip.colLowerBound(y[i][j],0);
79     mip.colUpperBound(y[i][j],1);
80   }
81 }
82
83 /* --- Restrição 1 ---
```

```
84   * Esta restrição é referente ao somatório dos xi participantes da       *
85   * solução, que deve ser igual a 2t+1 nós. Caso existam mais nós que     *
86   * esta restrição, devem ser ignorados, pois o sistema deve trabalhar    *
87   *apenas com t+1 nós                                                     */
88  {
89  Mip::Expr expr;
90  for (i = 0; i < V; i++){
91  expr += x[i];
92  }
93  mip.addRow( expr == 2*t + 1 );
94  }
95
96  /* --- Restrição 2 ---
97   * Esta restrição garante que se o nó pertence ao conjunto de alarmes,*
98   * obrigatoriamente xi deve fazer parte da solução                     */
99
100 for(i=0; i<t; i++){
101   mip.addRow(((Mip::Expr) x[T[i]]) == 1);
102 }
103
104 /* As restricoes 3 e 4 garantem que, se um enlace esta presente, os    *
105  * nós adjacentes fazem parte da solucão.                              */
106 /* --- Restrição 3 --- */
107 for (i=0; i<V; i++){
108   for (j = i+1; j<V; j++) {
109     Mip::Expr expr1 = x[i] + x[j];
110     Mip::Expr expr2 = y[i][j] + 1;
111     mip.addRow( expr1 <= expr2);
112   }
113 }
114
115 /* --- Restrição 4 --- */
116 for (i=0; i<V; i++) {
117   for (j = i+1; j<V; j++) {
118     Mip::Expr expr1 = x[i];
119     Mip::Expr expr2 = y[i][j];
120     mip.addRow( expr1 >= expr2 );
121   }
122 }
123
124 //Função objetivo:
125 Mip::Expr obj;
126 for (i = 0; i < V-1; i++)
127   for (j = i+1; j < V; j++)
128     obj += distancias[i][j] * y[i][j];
129
130 mip.min();
131 mip.obj(obj);
132
133 /* --- Solucionando o MIP --- */
134 mip.solve();
135
136 /* --- Imprimindo a solução ---*/
137 if (mip.type() == Mip::OPTIMAL){
138   somatorio_distancia_nodos_heuristica = mip.solValue();
139   printf("\nValor da funcao objetivo: %f \n\n", mip.solValue());
140     for (i = 0; i < V; i++){
141       printf("x[%d] = %f\n", i, mip.sol(x[i]));
142     }
143     for (i = 0; i < V-1; i++){
144       for (j=i+1; j<V; j++){
145         printf("y[%d][%d] = %f\n", i, j, mip.sol(y[i][j]));
146       }
147     }
148 }
149 else{
150   printf("Solucao otima nao foi encontrada.");
151 }
```