UNIVERSIDADE FEDERAL DE MINAS GERAIS
Escola de Engenharia
Departamento de Engenharia Elétrica

Pós-graduação em Engenharia Elétrica

OPERADORES GEOMÉTRICOS PARA OTIMIZAÇÃO DINÂMICA COM APLICAÇÃO AO PROBLEMA DE COBERTURA E CONECTIVIDADE EM REDES DE SENSORES SEM FIO NÃO-HIERÁRQUICAS

Flávio Vinícius Cruzeiro Martins

TESE DE DOUTORADO

Belo Horizonte
30 de novembro de 2012

UNIVERSIDADE FEDERAL DE MINAS GERAIS
Escola de Engenharia
Departamento de Engenharia Elétrica

Flávio Vinícius Cruzeiro Martins

OPERADORES GEOMÉTRICOS PARA OTIMIZAÇÃO DINÂMICA COM APLICAÇÃO AO PROBLEMA DE COBERTURA E CONECTIVIDADE EM REDES DE SENSORES SEM FIO NÃO-HIERÁRQUICAS

Trabalho apresentado ao Programa de Pós-graduação em Engenharia Elétrica do Departamento de Engenharia Elétrica da UNIVERSIDADE FEDERAL DE MINAS GERAIS como requisito parcial para obtenção do grau de Doutor em Engenharia Elétrica.

Orientador: *Prof. Dr. Ricardo H. C. Takahashi*
Co-orientador: *Prof. Dr. Geraldo Robson Mateus*

Belo Horizonte
30 de novembro de 2012

Dedico esse trabalho à minha amada esposa Aline e minha linda filha Elisa.

AGRADECIMENTOS

Após uma longa jornada mais uma etapa foi vencida! Juntamente com todos meus amigos e familiares, encontrei forças para realizar mais um sonho. Entretanto, nada seria possível ou faria sentido se não dar toda honra e toda glória desta vitória a quem é o Criador e dono da minha vida, Jesus Cristo. Muito obrigado Senhor por estar comigo em toda essa caminhada, pelo cuidado e pela forma na qual me conduziu. Por me acompanhar em cada etapa, sendo o pilar da minha vida, dando todo o sustento necessário e ânimo nos momentos difíceis. Tudo que tenho, tudo que sou, e o que eu vier a ser será para honrar e glorificar a Ti.

Agradeço aos meus pais que sempre me incentivaram e me apoiaram em todas as decisões! Confesso que ficaram preocupados quando decidi seguir meus estudos e me dedicar à carreira acadêmica, mas com o tempo foram entendendo e sempre demonstrando muito orgulho ao ver um filho concluindo um doutorado. Para falar verdade, acho que a preocupação só terminou quando eu passei no concurso para ser um professor na UFOP *rs... Agradeço também às minhas irmãs que sempre estiveram ao meu lado, ajudando e apoiando em todos os momentos. Também agradeço aos meus tios que sempre se mostraram preocupados com toda essa minha trajetória, e nunca duvidaram da minha capacidade. Não posso deixar de falar dos meus avós que sempre rogam a Deus pela minha proteção, para que Ele me guarde nestas rodovias tão perigosas.

Com muito carinho agradeço aos meus irmãos da Igreja Presbiteriana Floresta, pois sempre com muito amor estiveram comigo orando e pedindo a Deus, força, inteligência e sabedoria para que eu pudesse finalizar mais esta etapa. Agradeço por permanecerem sempre ao meu lado, mesmo quando era preciso que eu abrisse mão de algumas obrigações. Sempre com muito amor buscaram compreender o quão era importante eu terminar este doutorado. Também agradeço aos irmãos da Igreja Presbiteriana Simonton, em Conselheiro Pena, que mesmo de longe sempre me acolheram em orações.

É difícil de expressar o quão grato sou pelo meu orientador, professor Ricardo Takahashi, que com toda sua paciência me acompanhou desde o início da graduação, e me ensinou os caminhos de como ser um pesquisador de excelência. Em todos os momentos me apoiou e esteve pronto a ajudar em tudo que foi necessário, sempre atencioso e disponível em todas as minhas demandas. Agradeço também ao professor Geraldo Robson, que também acompanhou alguns dos meus passos, tanto como orientador como também professor, me ajudando e incentivando a continuar nesta caminhada. Tenho muito orgulho de ter sido orientado por vocês, os terei sempre como um espelho nesta minha carreira que está apenas começando! Também tenho muito a agradecer ao professor Eduardo Carrano que sempre contribuiu ativamente para o bom andamento do meu trabalho, inclusive levantando a idéia que deu o ponta pé inicial desta tese.

Aos amigos e colegas da UFMG meu muito obrigado! Fica até difícil de citar nomes,

pois foram muitos que fizeram parte de forma direta ou indireta para que eu pudesse cumprir todas as etapas e chegar até aqui. A começar pela eficiência da Anete e da Arlete, secretárias do PPGEE. Aos colegas do GOPAC que me ajudaram durante as disciplinas. Ao meu amigo Cidiney que sempre me acompanhou nas noites, madrugadas e finais de semana na UFMG para encarar trabalhos e mais trabalhos. Lembra Rogério e Cidiney de Geometria Computacional?! Aos colegas do LaPO que direto me viam, mesmo de forma virtual, com meu login "cruzeiro" nos servidores de processo, obrigado pela paciência. Por fim, agradeço a todos os demais professores que contribuíram para minha formação!

É preciso dar mérito a uma pessoa em especial, pois ela tem sido uma verdadeira companheira, dando-me forças, e acreditando que todo sacrifício vale a pena. Nunca hesitou em depositar em mim toda sua confiança, sempre com a certeza de que no fim tudo daria certo! Foi minha auxiliadora e incentivadora, me erguendo nos momentos de desânimo e com seu colo e carinho me deu muito amor. Agradeço pela paciência e por saber entender o quão era importante terminar esta etapa. Há pouco mais de três anos eu estava na correria para terminar meu mestrado e nos casarmos, graças a Deus tudo transcorreu de forma perfeita! Agora, tínhamos novamente um motivo nobre para que eu terminasse o doutorado, este motivo tem por nome Elisa! Uma linda princesinha que chegou no dia 14 de setembro de 2012 para alegrar nossa Família, e ser um motivo de inspiração! Aos meus amores Aline e Elisa, meu muito obrigado! Amo vocês!!!

.... "Ó profundidade das riquezas, da sabedoria e da ciência de Deus!
Quão inescrutáveis são os seus juízos e quão impenetráveis os seus caminhos!
Pois quem conheceu a mente do Senhor? Ou quem se fez o seu conselheiro?
Ou quem lhe deu primeiro, para que lhe seja retribuído?
Pois dele, por ele e para ele são todas as coisas; a ele seja dada a glória para sempre. Amém."

—BÍBLIA SAGRADA (Romanos 11:33-36)

RESUMO

Recentes estudos levantaram a seguinte hipótese: a atribuição de uma geometria para o espaço das variáveis de decisão de um problema combinatório pode ser útil tanto para fornecer descrições significativas do panorama das funções (*fitness landscape*), como também para apoiar a construção sistemática de operadores evolutivos ditos *geométricos*, os quais fazem uso consistente das propriedades geométricas do espaço, em busca do ótimo do problema.

Este trabalho introduz novos operadores geométricos que constituem a concretização das entidades geométricas *direção de descida* e *subespaço* em espaços de variáveis combinatórias. Estes novos operadores geométricos são apresentados no contexto específico do Problema Dinâmico de Cobertura e Conectividade em Redes de Sensores sem Fio (PDCC-RSSF). As RSSF têm se mostrado uma área de intensa atividade de pesquisa, sendo que um importante tópico tem sido buscar uma rede com consumo energético eficiente, na tentativa de se obter redes com tempo de vida prolongado. Para estender o tempo de vida da rede, um novo modelo que captura a dinamicidade do PDCC-RSSF é proposto. Para sua solução, são desenvolvidos algoritmos genéticos em versões mono-objetivo e multiobjetivo, utilizando os operadores geométricos propostos. Uma comparação interessante é realizada em relação a uma formulação de um único objetivo definida por um problema de programação linear inteira (PLI), resolvido por métodos exatos. Esta formulação por PLI adota uma função *proxy* como objetivo, que minimiza o consumo de energia, a fim de aproximar o objetivo principal de maximização do tempo de vida da rede. Além disto, utiliza uma abordagem gulosa pra lidar com a dinâmica do sistema, resolvendo um problema estático de PLI para cada estágio, para aproximar do problema dinâmico. Até onde sabe este autor, os algoritmos aqui apresentados são os primeiros a superar o tempo de vida de uma rede sintetizada pela formulação por PLI, e ainda com um custo computacional menor.

Apesar de serem obtidas em um ambiente totalmente controlado, as soluções aqui desenvolvidas podem servir como referência para o projeto de uma RSSF. Abordagens *on-line* foram construídas com o intuito de aproximar as soluções apresentadas de situações mais reais. Tais abordagens utilizam como referência a rede projetada pela solução do novo modelo proposto para o PDCC-RSSSF. Simulações para analisar o desempenho das abordagens foram realizadas em um ambiente com falhas imprevistas. Os resultados foram comparados com a referência de projeto, sendo obtidos resultados positivos.

Além da introdução de duas entidades geométricas no contexto geométrico para problemas combinatórios, uma das contribuições deste trabalho é a apresentação de um novo modelo, verdadeiramente dinâmico, para o PDCC-RSSF, juntamente com algoritmos que conseguem resolver tal modelo. Diferente das abordagens encontradas na literatura, este trabalho propõe uma nova abordagem para o projeto de RSSF, em que todos os estágios

de funcionamento da rede são tratados de forma simultânea e não individualmente.

Palavras-chave: Redes de Sensores sem Fio, Sistemas Dinâmicos, Otimização Dinâmica, Otimização Combinatória, Otimização Multiobjetivo, Algoritmos Genéticos, Operadores Geométricos.

ABSTRACT

Some recent works raised the hypothesis that the assignment of a geometry for the decision variable space of a combinatorial problem may be useful both for providing meaningful descriptions of the fitness landscape and for supporting the systematic construction of evolutionary operators (the geometric operators) that make a consistent usage of the space geometric properties in the search for the problem optima.

This thesis introduces some new geometric operators which constitute the realization of searches along the combinatorial space versions of the geometric entities *descent directions* and *subspaces*. The new geometric operators are presented in the specific context of the *Wireless Sensor Network Dynamic Coverage and Connectivity Problem* (WSN-DCCP). The WSN is a very promising area in which several researchers seek an energy-efficient network, in an attempt to obtain a long network lifetime. In order to extend the network lifetime, a new model that captures the dynamics of the WSN-DCCP is proposed. In order to solve it, a genetic algorithm in both single-objective and multiobjective versions is developed for the WSN-DCCP, using the proposed geometric operators. An interesting comparison is performed against a single-objective formulation stated as an integer linear programming (ILP) problem, which is solved with exact methods. This ILP formulation adopts a *proxy* objective function based on the minimization of energy consumption in the network, in order to approximate the objective of network lifetime maximization, and a greedy approach for dealing with system dynamics, solving an ILP static problem for each stage, to approach the dynamic problem. Up to the authors' knowledge, the proposed GAs are the first algorithms to outperform the lifetime of resulting networks as synthesized by the ILP formulation, also running in much smaller computational times.

Although they are obtained in a fully controllable environment, the solutions developed can serve as reference for the design of a WSN. Online approaches were built with the intention of bringing the solutions presented in more real situations. Such approaches use as a reference network designed for solving the new model proposed for the WSN-DCCP. Simulations to analyze the performance of the approaches were performed in an environment with unexpected failures. The results were compared with the reference design, obtaining results very encouraging.

Besides the introduction of two geometric entities in geometric context for combinatorial problems, one of the major contributions of this thesis is to present a new truly dynamic model for the WSN-DCCP, along with algorithms that can solve such model. Differently of the approaches found in the literature, this thesis proposes a new approach for WSN design, in which all periods of the network operation are treated simultaneously rather than individually.

Keywords: Wireless Sensor Networks, Dynamical Systems, Dynamic Optimization, Combinatorial Optimization, Multiobective Optimization, Genetic Algorithms, Geometric Operators.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

CAPÍTULO 1

INTRODUÇÃO

O formalismo dos *operadores geométricos* foi apresentado por Moraglio e Poli nas referências (Moraglio & Poli, 2004; Moraglio et al., 2007; Moraglio, 2007) como uma estrutura geral para o desenvolvimento de algoritmos evolucionários. Nestas referências foi mostrado que um algoritmo evolucionário pode ser inteiramente construído com base apenas em um esquema de codificação de solução e uma função de vizinhança, que é utilizada para gerar novas soluções, a partir de uma solução viável. Se um conceito adequado de distância é empregado para controlar esta operação, tem-se uma *mutação geométrica*. Além disso, um *cruzamento geométrico* pode ser implementado como uma sequência de operações geométricas, que são realizadas de tal modo que as soluções descendentes estão presentes no menor caminho (utilizando a mesma definição de distância) que conecta os dois pais. Este tipo de abordagem pode ser utilizada como base geral para o desenvolvimento de poderosos algoritmos evolutivos para problemas bastantes diferentes, contendo codificações de soluções bem distintas. Tendo garantia que os operadores empregados são geométricos, o algoritmo pode apresentar boas propriedades que são herdadas de procedimentos de busca em espaços métricos (Carrano et al., 2010).

Conceitos geométricos, como afirmam as referências citadas, foram aplicados até agora com o papel principal de apoiar a definição de operadores que são compatíveis com a noção de distância. Esses conceitos permitem a construção mais sistemática de algoritmos, que terão comportamento mais regular. Não obstante existirem muitas utilidades em potencial, ainda não foi explorado de maneira intensiva o uso de operadores geométricos para problemas combinatórios. Algumas destas utilidades foram sugeridas na referência (Carrano et al., 2010). Uma das contribuições do trabalho aqui apresentado é a investigação do uso de duas entidades geométricas:

i) *Direção de descida*; e

ii) *Subespaço*.

A entidade *Direção de descida* já explorada preliminarmente por Carrano et al. (2010), em uma estrutura que se assemelhava a uma "busca unidimensional", é aqui estudada na forma de um operador que se assemelha a uma "busca por gradiente". Já a entidade *Subespaço* ainda não foi considerada na literatura sobre os operadores geométricos. Estas entidades dão origem a alguns operadores que são utilizados em um algoritmo genético construído nesta tese com o propósito específico de resolver um problema desafiador: o Problema Dinâmico de Cobertura e Conectividade em Redes de Sensores sem Fio (PDCC-RSSF).

O conceito de *direção de descida* em espaços de variáveis discretas é aqui concretizado pelo operador de *Cruzamento Real Polarizado* (CRP). Esse operador foi originalmente

proposto na referência (Takahashi et al., 2003), no contexto de otimização em variáveis contínuas. Tal operador funciona através da realização de um cruzamento em duas soluções "pais", avaliadas previamente (soluções provenientes do processo de seleção). Dependendo de uma escolha aleatória que é realizada a cada vez que o operador é executado, as soluções descendentes podem ser localizadas tanto no segmento de reta que liga as soluções "pais" quanto em um segmento extrapolado. A distribuição de probabilidade da geração de filhos é tal que há uma maior probabilidade de que uma solução descendente apareça perto do melhor "pai" (neste sentido o cruzamento é "polarizado"). A interpretação da busca em uma direção de descida vem do caso em que uma solução descendente é gerada ao longo do segmento extrapolado do lado do melhor "pai". Utilizando o formalismo de operadores geométricos, no Capítulo 4 é mostrada uma implementação de uma versão do operador CRP, adequada a um problema combinatório.

O conceito de *subespaço* em espaços de variáveis discretas é estabelecido teoricamente utilizando um procedimento análogo à definição de subespaços em espaços vetoriais a partir de um subconjunto de um conjunto de vetores base. Este conceito é útil para representar uma característica particular do PDCC-RSSF: a busca em um subespaço específico. Esta busca seria bastante ineficiente por meio de operações puramente estocásticas. Neste sentido, uma busca gulosa determinística pode ser mais eficaz. A busca estocástica, característica dos algoritmos evolutivos, deve ser conduzida principalmente no subespaço complementar. A estrutura específica deste problema leva à definição de dois operadores: um operador de mutação, projetado sobre um subespaço, e um operador de busca local, que trabalha no subespaço complementar.

Estas novas entidades teóricas serão aqui apresentadas no contexto de um problema de aplicação específica pelas seguintes razões:

- Em primeiro lugar, considerou-se a motivação do conceito de *subespaço*. Em particular, esta entidade necessita de um contexto a fim de poder ser totalmente apreciado. O problema específico escolhido, o PDCC-RSSF, fornece uma estrutura interessante que permite ao algoritmo tirar o máximo proveito deste conceito.

- Em segundo lugar, o PDCC-RSSF constitui um problema relevante e difícil que tem sido estudado recentemente. Os operadores aqui propostos são incorporados em um algoritmo genético mono-objetivo e também em um algoritmo genético multiobjetivo, ambos dedicados a resolver o PDCC-RSSF. Os resultados obtidos com os métodos propostos constituem, dentro do conhecido, os melhores resultados para o problema. Portanto, outra contribuição importante deste trabalho é a formulação de uma metodologia específica para lidar com o PDCC-RSSF.

É importante ressaltar que as abordagens aqui apresentadas para a solução do PDCC-RSSF operam em um nível mais abstrato dos nós sensores, não se preocupando com as características inerentes de suas camadas (física, enlace, rede, transporte e aplicação). Problemas como perda e colisão de pacotes, pacotes de controle (*overhead*), latência, confiabilidade dos dados, ciclos de trabalho (*duty cycle*), entre outros, não foram tratados nesta tese. Os protocolos das camadas física, de enlace, de rede e de transporte não foram aqui implementados. No entanto no contexto de uma implementação real, estes seriam

necessários para dar suporte às abordagens aqui apresentadas. Sendo assim, para que os dados pudessem ser disseminados pela rede, um modelo de comunicação simplificado foi sugerido, estabelecendo uma árvore de comunicação entre os sensores ativos da rede (Seção 6.1).

1.1 UMA BREVE REVISÃO DA LITERATURA SOBRE PDCC-RSSF

Em muitas situações práticas, é necessário monitorar a ocorrência de alguns fenômenos em uma determinada área. Com a constante evolução da microeletrônica e dos sistemas embarcados nos últimos anos, foi possível o surgimento de uma nova tecnologia, as Redes de Sensores sem Fio (RSSFs), (Yick et al., 2008; Akyildiz et al., 2002), que têm recebido uma atenção especial, pois têm se mostrado uma importante ferramenta para executar tal tarefa. Nos dias atuais, podem-se encontrar várias aplicações reais utilizando a tecnologia de RSSF, por exemplo: monitoramento da vida selvagem e variações climáticas (Mainwaring et al., 2002), saúde (Mascarenas et al., 2009), minas em túneis (Jiang et al., 2009), avaliação de componentes tóxicos no ambiente (Tsow et al., 2009), dentre outros.

Tipicamente as RSSFs são compostas por dispositivos autônomos e compactos que são constituídos por subsistemas de sensoriamento, processamento e comunicação. Esses dispositivos são chamados nós sensores. Dentre os nós sensores podem existir alguns nós especiais chamados sorvedouros (do inglês *sink*). Esses nós geralmente possuem energia ilimitada ou renovável e têm capacidade de agregar informações recebidas por um conjunto de nós sensores presentes na rede. Os nós sorvedouros podem ser estáticos, fixos em um ponto geográfico recebendo as informações dos demais sensores da rede, ou móveis, fazendo um trajeto pela região de interesse coletando as informações monitoradas pelos demais sensores presentes na rede (Anastasi et al., 2009).

Os nós sensores são geralmente constituídos por: placa de sensoriamento, processador, rádio e bateria (Yick et al., 2008). Estes dispositivos permitem ao sensor monitorar uma área dentro de um determinado raio de cobertura e manter comunicação com a rede dentro de um raio de visibilidade. Durante o funcionamento da rede, os sensores coletam informações do ambiente e as enviam para o nó sorvedouro. Uma das formas possíveis de disseminar os dados coletados pela rede é utilizando a comunicação *multi-hop* (múltiplos saltos).

Os nós sensores normalmente possuem algumas restrições a eles inerentes, tais como: a limitação de energia, baixo desempenho no processamento e pequeno raio de comunicação. Estas características, supostamente indesejáveis, são impostas para garantir a necessidade de um baixo custo unitário e para que estes sejam bastante compactos (Mainwaring et al., 2002). Para redes instaladas em regiões de difícil acesso, não é possível ou viável trocar ou recarregar a bateria de um sensor. Esta situação impõe uma forte restrição para a construção da rede, pois o tempo de vida de uma certa configuração não pode ser estendido, ficando determinado pela duração da bateria do primeiro nó sensor que ficar fora de serviço.

Existem várias formas na literatura de tratar este problema de restrição de energia. Uma forma possível para incrementar o tempo de vida da rede é alocar um número de sensores maior que o necessário para garantir a cobertura desejada. Nesse caso, a

construção da rede pode ser vista como um problema de controle de densidade, em que o objetivo é manter a rede disponível por um maior período de tempo, fazendo novos sensores entrarem em atividade à medida em que os sensores anteriormente ativos tenham sua energia esgotada. Dado um conjunto de nós sensores, a construção da rede pode ser dividida basicamente em dois passos:

1. Escolha de um conjunto de sensores que sejam capazes de monitorar uma determinada área, garantindo a cobertura e a conectividade entre eles. Em um primeiro momento, um subconjunto dos sensores permanece ativado, enquanto outros sensores permanecem desativados [1]. A rota de comunicação entre os sensores ativos deve ser determinada neste passo.

2. Após algum tempo de funcionamento, a energia presente na bateria de alguns sensores irá acabar, fazendo com que estes sensores parem de monitorar. Esta falha de energia pode quebrar a conectividade e diminuir a cobertura da rede. Desta forma, é necessário reestabelecer de alguma maneira estes requisitos. Então, dentre o conjunto de sensores desativados, é feita uma análise para ver se existe um conjunto de sensores que ao serem ativados reestabeleça a cobertura e a conectividade da rede. Em caso afirmativo, estes sensores são colocados em funcionamento. A vida da rede chega ao fim no momento em que não for mais possível encontrar um conjunto de sensores para restaurar a cobertura e/ou a conectividade necessária.

O problema levantado desta forma torna-se intrinsecamente dinâmico, uma vez que as decisões tomadas em qualquer instante de tempo necessariamente afetam as opções de decisão posterior. Além disso, o problema de construção da rede pode ser dividido em estágios discretos, com a duração de cada estágio sendo definida pelo tempo decorrido entre eventos consecutivos de falhas de nós sensores.

É possível encontrar na literatura muitos trabalhos em que a intenção é prover mecanismos eficientes de consumo de energia para as RSSF, por exemplo: Tilak et al. (2002); Cerpa & Estrin (2002); Ye et al. (2003); Yu et al. (2006); Cardei & Wu (2006); Martins et al. (2007); Chang & Chang (2008); Podpora et al. (2008); Hu et al. (2010); Anastasi et al. (2009). Grande parte destes trabalhos busca minimizar o consumo de energia a cada unidade de tempo na tentativa de prolongar a vida da rede. Ao olhar desta forma, a cada período, um subconjunto de sensores é escolhido para permanecer ativado, de forma a garantir as restrições de cobertura e conectividade com um menor consumo de energia. Entretanto, a escolha deste subconjunto de sensores nesses trabalhos é feita olhando especificamente para o estágio corrente, não havendo a preocupação com a consequência desta escolha em estágios posteriores. Ou seja, a característica dinâmica do problema é ignorada.

Tilak et al. (2002) propõem um esquema que deve ser executado periodicamente, definindo quais nós sensores devem ser ativados ou desativados, considerando apenas as informações sobre o instante de tempo em que o algoritmo é executado. A função

[1] Em todo este trabalho um sensor que se encontra desativado estará em estado de espera, ou "dormindo" (*sleep mode*). Uma vez que o consumo de energia neste modo é extremamente baixo, este não foi considerado.

objetivo deste algoritmo é indicada como sendo a minimização do número de nós ativos, de forma a garantir a conectividade e a cobertura da rede conforme requisitado. É mostrado que este método consegue atingir uma economia significativa de energia da rede, em comparação com resultados da época. Em (Cerpa & Estrin, 2002), um modelo adaptativo autoconfigurável para os nós sensores de uma RSSF foi introduzido. Neste modelo, cada nó decide seu estado (*sleep*, passivo, teste e ativo) através da avaliação de sua conectividade. A conectividade é estimada com base nas mensagens enviadas por seus vizinhos através de uma comunicação *multi-hop* (múltiplos saltos) .

Na referência (Ye et al., 2003) os autores apresentam o PEAS (*Probing Environment and Adaptive Sleeping*), um protocolo para a realização de controle de densidade em RSSF. Nessa abordagem, os sensores têm três estados possíveis: "dormindo, sondando e trabalhando" (*sleeping, probing* e *working*), e dois algoritmos são utilizados para escolher um dos estados:

- o primeiro algoritmo é responsável por definir quais nós devem ser ativados e quais nós devem "domir";
- o segundo algoritmo é responsável por estimar o tempo médio de "sono" de cada sensor.

Com estas duas características, o algoritmo garante um crescimento linear no tempo de vida da rede em função do número de sensores dispostos. Entretanto, o algoritmo não garante a cobertura da área de sensoriamento.

Cardei & Wu (2006) propõem métodos para lidar com dois problemas relacionados com RSSF: problema de ponto de cobertura e o problema de área de cobertura. No primeiro, cada ponto de demanda deve ser coberto por pelo menos um sensor. Já no segundo, é necessário apenas cobrir uma parte dos pontos de demanda alocados em uma determinada área. Nesse trabalho, foi utilizado o escalonamento de sensores para estender o tempo de vida da rede.

Nakamura et al. (2005) propõem uma formulação de Programação Linear Inteira (PLI) para o problema de cobertura e conectividade multi-período em RSSF planas. Restrições relacionadas aos limites de energia dos nós sensores foram incluídas no modelo, em conjunto com as usuais restrições de cobertura e conectividade. Na ocorrência de uma falha em um nó sensor, um novo estágio era definido, em que novos nós eram ativados para manter a cobertura e a conectividade. Os estágios foram resolvidos por uma aproximação gulosa, estágio por estágio, sem considerar todos os estágios de forma global. O modelo resultante foi resolvido utilizando o pacote comercial CPLEX (CPLEX, 2006). Para se chegar a um modelo linear, o objetivo de maximização do tempo de vida da rede foi substituído por um objetivo *proxy*[2]: a minimização da potência consumida na rede. Apesar da introdução desta função objetivo ser uma aproximação e também pelo fato do

[2]Uma função objetivo *proxy* é uma função que se correlaciona com a função objetivo que de fato representa o problema prático em questão, sem coincidir com ela. Desta forma, a minimização de uma função *proxy* leva a uma solução que não representa o ótimo exato do problema prático. Em diversas situações, a função *proxy* é utilizada em substituição à função objetivo exata porque esta pode ser muito difícil de ser formulada, ou de difícil avaliação, ou até mesmo imprópria para ser otimizada utilizando algum método numérico específico.

método proposto não considerar a estrutura dinâmica do problema, esta abordagem tem apresentado, até o momento, um dos melhores resultados no que diz respeito ao objetivo de estender o tempo de vida da rede (PDCC-RSSF).

Os autores Podpora et al. (2008) propõem um algoritmo adaptativo que controla a energia gasta por cada nó sensor individualmente. Tal algoritmo emprega uma rede neural para estimar as alterações do sinal de detecção na vizinhança do nó. Se essas alterações não forem significativas, o nó sensor é conduzido ao estado de "sono" (*sleeping*), a fim de economizar energia. A estimativa da rede neural é baseada em dados históricos e os autores afirmam que seu uso torna possível generalizar as ações sem que se tenha conhecimento *a priori* sobre a aplicação em específico (distribuição de dados). Na referência (Leung et al., 2008), foi proposto um novo modelo para tratar informações em redes de sensores distribuídos. Nesse modelo, cada nó sensor pode tomar decisões no que diz respeito a alguns aspectos operacionais, tais como: gerenciamento de sinal e agregação dos dados coletados. Além disso, os nós são capazes de realizar uma análise inicial do cenário para apoiar suas decisões. Por fim, um procedimento que coordena as decisões tomadas pelos sensores é adotado.

Um algoritmo genético multiobjetivo híbrido para lidar com os problemas de cobertura e conectividade, um resultado desta tese, foi proposto em (Martins et al., 2011). Nesse método, a minimização da potência consumida e a maximização da cobertura são consideradas como funções objetivo do problema. Dois algoritmos são combinados:

- um algoritmo genético (global e centralizado) que é capaz de reestruturar a rede com o intuito de melhorar o desempenho da mesma; e
- um operador de busca local (local e distribuído), que funciona localmente para reestabelecer de forma rápida o funcionamento da rede depois da ocorrência de uma falha.

O algoritmo global é executado antes do início do funcionamento da rede, estabelecendo sua configuração inicial, e depois disso ele volta a ser executado em intervalos, determinado por uma variável que mede o número de nós afetados por falhas corrigidas pelo algoritmo local. Quando esta variável ultrapassa certo limiar, o algoritmo global é acionado para reestruturar a rede. Em ambos os algoritmos, o tratamento da conectividade é feito por um algoritmo determinístico de roteamento. A referência mostra que a abordagem multiobjetivo fornece informações adicionais para o projetista da rede, podendo este avaliar o *trade-off* entre a extensão da área de cobertura e a extensão do tempo de vida da rede.

Todos os métodos descritos acima compartilham uma característica: embora manipulem um problema dinâmico, a estrutura dinâmica do problema não é totalmente considerada. Mesmo nos trabalhos em que o problema é modelado com múltiplos estágios, cada estágio é resolvido de forma independente, não havendo a preocupação de tratar a interdependência entre estágios. Essa forma de abordar o problema acaba se tornando em certo sentido "gulosa", uma vez que o ótimo em cada estágio é obtido ao preço de comprometer a possibilidade de aumento de desempenho nos estágios subsequentes. Assim, este tipo de abordagem muitas vezes impede o algoritmo de alcançar o ótimo global do problema

(Bertsekas, 1995). Três trabalhos recentes buscam modelar o Problema Dinâmico de Cobertura e Conectividade (PDCC-RSSF) como um problema verdadeiramente dinâmico. Uma breve descrição desses trabalhos pode ser vista a seguir.

Nakamura (2010) estendeu o modelo de Programação Linear Inteira (PLI) modelado em Nakamura et al. (2005), de forma a considerar a interação entre os estágios do problema dinâmico. Esta nova abordagem foi capaz de alcançar resultados melhores aos obtidos por Nakamura et al. (2005), mas apenas para casos em que a instância do problema é pequena. A implementação desta formulação, utilizando o pacote comercial CPLEX, não foi capaz de lidar com instâncias que possuam uma quantidade média ou grande de sensores.

Hu et al. (2010) modela o escalonamento dos sensores como um problema de conjuntos disjuntos. Na mesma referência o algoritmo *Schedule Transition Hybrid Genetic Algorithm* (STHGA) foi proposto para lidar com este problema. Com base em (R.Williams, 1979), é avaliado o número mínimo de sensores que devem permanecer ativos para garantir a cobertura total da área. Em seguida, o STHGA busca por um conjunto de tamanho mínimo que satisfaça as restrições de cobertura, de tal maneira que o número de conjuntos disjuntos seja maximizado. Os resultados apresentados nessa referência superaram algumas abordagens presentes na literatura. No entanto, é importante notar que tal abordagem não lida com o problema de conectividade, e que esta trata apenas de problemas nos quais a cobertura completa da área se faz necessária.

De uma forma diferente, em Martins et al. (2010) é proposto um Algoritmo Genético (AG) para construção de uma RSSF, considerando de forma completa a natureza dinâmica do problema e os requisitos de cobertura e conectividade, também um resultado desta tese. Nesse trabalho, todos os estágios da rede são considerados de forma conjunta. Como resultado, o algoritmo apresenta uma solução pré-processada em que se define exatamente como a rede irá operar durante seu tempo de vida. É informado o estágio, ou instante de tempo exato em que cada nó sensor poderá ser ativado de forma a assegurar os requisitos pré-estabelecidos. Cada indivíduo do AG é uma solução candidata, sua decodificação representa a simulação total da RSSF, possibilitando assim medir todo seu tempo de vida. A função objetivo a ser maximizada é justamente o tempo de vida da rede. Nesse algoritmo, a cobertura é definida como um parâmetro de projeto, assim tornando a abordagem capaz de lidar com quaisquer requisitos de cobertura. Os resultados apresentados em tal publicação constituem, de fato, um formato preliminar dos resultados reportados nesta tese.

Este trabalho de tese tem por objetivo construir um algoritmo para escalonamento dos nós de RSSF que trate o problema de forma realmente dinâmica, assim respeitando a natureza do problema. Será aqui mostrado que a abordagem proposta supera as outras abordagens, que não tratam o problema dessa forma.

Além dos trabalhos citados, nesta tese serão propostas, no Capítulo 4, duas variações de um algoritmo genético para projetar uma RSSF:

- Na primeira variação, o AG é construído para maximizar a vida útil de uma rede, dada uma cobertura mínima exigida (definida pelo usuário). Esta variação é desenvolvida a partir da proposta feita por Martins et al. (2010), também construída

sobre o conceito de *operadores geométricos*. A principal novidade do algoritmo aqui sugerido está relacionado com o uso dos conceitos de *subespaço* e *direção de descida* na construção de novos operadores. A aplicação destes conceitos pode ser considerada como um passo decisivo para se atingir os níveis de desempenho que serão apresentados.

- A segunda variação é uma extensão multiobjetivo do algoritmo mono-objetivo sugerido na primeira versão. Nesta abordagem, a restrição de cobertura é relaxada transformando-se em uma função objetivo que é considerada em conjunto com a função de maximização do tempo de vida da rede. A abordagem multiobjetivo trata o problema de geração de estimativas do conjunto Pareto-ótimo com base no algoritmo *Non Dominated Sorting Genetic Algorithm II* (NSGA II) (Deb et al., 2002), utilizando praticamente a mesma codificação e operadores muito semelhantes aos adotados na versão mono-objetivo do algoritmo genético.

1.1.1 Objetivos

Esta tese possui objetivos de duas naturezas, relacionadas: (i) ao tratamento de um problema com motivação prática relevante e de elevado grau de dificuldade; e (ii) ao estudo de algumas questões de fundo teórico, que se situam nos fundamentos da teoria dos algoritmos evolutivos. A construção deste trabalho foi feita de maneira tal que os objetivos fossem tratados de maneira complementar: o problema prático foi resolvido de maneira satisfatória em virtude da inovação de ordem teórica introduzida, e a questão teórica somente pôde ser apresentada adequadamente a partir da motivação prática.

O objetivo de ordem teórica deste trabalho consiste especificamente em explorar o conceito de *operadores geométricos*, recentemente proposto na literatura, empregando esse quadro teórico geral para propor duas entidades geométricas que são novas no contexto da otimização combinatória: as *direções de descida* e os *subespaços*. Essas entidades dão origem a operadores que são utilizados em algoritmos genéticos especialmente construídos para tratar um problema relacionado com uma aplicação prática.

O objetivo de ordem prática consiste em resolver o Problema Dinâmico de Cobertura e Conectividade em Redes de Sensores Sem Fio (PDCC-RSSF), visando manter uma rede em funcionamento pelo maior tempo possível, obedecendo a requisitos mínimos de cobertura e conectividade. Esse problema deve ser resolvido em duas versões, uma mono-objetivo, que trata a área coberta como restrição, e a outra considerando como objetivo adicional a maximização da área coberta. Como objetivo metodológico, estabelece-se ainda a meta de realizar o tratamento do PDCC-RSSF utilizando uma formulação dinâmica.

1.1.2 Organização do Trabalho e Resultados

Nesta tese, as definições e conceitos necessários para a elaboração do trabalho serão primeiramente apresentados nos Capítulos 2 e 3. O Problema Dinâmico de Cobertura e Conectividade em Redes de Sensores Sem Fio (PDCC-RSSF) é modelado no Capítulo 3 como um Problema Dinâmico (PD). Com base nesta modelagem e nos conceitos das

entidades geométricas *direção de descida* e *subespaço*, no Capítulo 4 é construída uma versão mono-objetivo e outra multiobjetivo de um Algoritmo Genético (AG) que busca maximizar o tempo de vida da rede. Os resultados destes algoritmos possibilitam criar um escalonamento dos sensores presentes na rede, informando as sequências de ativação e de desativação dos sensores a partir de eventos ocorridos.

Uma versão mono-objetivo utilizando uma técnica de programação dinâmica aproximada é modelada e resolvida pelo algoritmo WSNdsGA (**W***ireless* **S***ensor* **N***etwork* **d***ynamic* **s***cheduling* **G***enetic* **A***lgorithm*). O algoritmo apresentado no Capítulo 4 é uma primeira proposta de solução para o PDCC-RSSF. Conforme as características do AG presente no WSNdsGA, cada indivíduo é um candidato à solução do problema e ao final de algumas gerações, a melhor solução é escolhida. Esta solução pode ser decodificada de forma a fornecer todo o escalonamento dos nós sensores presente na rede. Esta nova modelagem possibilitou atingir resultados significativamente melhores que os anteriormente reportados na literatura. Estes resultados são apresentados na Seção 6.2 do Capítulo 6.

Grande parte das soluções propostas para PDCC-RSSF, cujo objetivo principal é estender o tempo de vida da rede, faz uso de um objetivo *proxy* que minimiza a potência consumida a cada período de tempo. Tendo em vista esta lógica de economia de energia, o problema pode ser modelado por Programação Linear Inteira (PLI) e resolvido de forma exata por pacotes tais como o CPLEX (Nakamura, 2010; Martins et al., 2007). A solução da nova modelagem apresentada nesta tese, em uma instância contendo 100 nós sensores, supera em pelo menos 40,8% a modelagem por PLI, que é aplicada estágio a estágio. Os resultados apresentados também superam de forma significativa as soluções presentes em Martins et al. (2011).

A versão mono-objetivo é adaptada de forma a possibilitar construir um AG multiobjetivo (MOWSNdsGA). Nesta nova abordagem também apresentada ao longo do Capítulo 4, a restrição de cobertura é relaxada e transformada em um novo objetivo. Desta forma, é possível traçar uma fronteira Pareto-ótima mostrando o *tradeoff* entre cobertura e tempo de vida da rede. Através dessa curva Pareto-ótima, o usuário pode escolher as melhores soluções para diferentes taxas de coberturas. Os resultados para esta versão são apresentados na Seção 6.3 do Capítulo 6. Nessa seção os resultados são comparados com outro algoritmo multiobjetivo apresentado na literatura (Martins et al., 2011), atingindo melhores resultados para todas as instâncias testadas. Os resultados apresentados são validados pelas soluções mono-objetivo para cada faixa de cobertura. A grande vantagem da abordagem proposta é poder disponibilizar em uma única execução do algoritmo diferentes possibilidades de configurações da rede. Desta forma o usuário pode escolher a configuração que melhor se adequar à situação desejada no momento, priorizando a taxa de cobertura ou o tempo de vida da rede.

As soluções apresentadas no Capítulo 4 foram construídas com o intuito de serem utilizadas de forma pré-processada. De posse dos dados da rede o algoritmo é executado, e com sua resposta é possível configurar previamente o escalonamento dos nós sensores presentes na rede. Tal solução serve como referência para o funcionamento real das RSSFs, no sentido de representar uma solução que constitua um limitante superior para o desempenho possível de ser atingido pela rede. Essa solução, claro, não pode ser implementada diretamente em redes reais, pois seu cômputo requer informações não disponíveis

de maneira exata, que incluem por exemplo as durações exatas das baterias de todos os nós sensores.

Para tentar aproximar o que realmente ocorre em uma RSSF real, no Capítulo 5 são apresentadas quatro diferentes abordagens para funcionamento *online*. Todas as abordagens utilizam como solução inicial o escalonamento fornecida pelo algoritmo pré-processado WSNdsGA. Para simular a rede em um ambiente que mais aproximadamente corresponda a situações reais, falhas inesperadas foram inseridas nos nós sensores. Como primeira tentativa de solução para esta situação, foi tentada a utilização da própria solução fornecida pelo WSNdsGA, ou seja, na presença de uma falha, inesperada ou não, prossegue-se no processo de decodificação do escalonamento previamente calculado até que a falha seja sanada.

Outras três maneiras de resolver o problema de forma online foram propostas no mesmo capítulo. Ambas utilizam inicialmente a sequência previamente calculada. Em uma das formas de solução, ao surgir uma falha os próprios sensores se auto-reconfiguram de forma a tentar restaurar os requisitos mínimos de cobertura e conectividade. Após essa reconfiguração, ajustes necessários podem ser realizados pelo nó sorvedouro. Essa solução é parcialmente centralizada e parcialmente distribuída, e seus resultados se mostraram um pouco superiores aos observados no escalonamento sequencial da solução inicialmente estabelecida. As outras duas abordagens tratam a informação sobre as falhas de maneira apenas centralizada, sendo que um algoritmo de reconfiguração é executado pelo sorvedouro de forma a responder às falhas. Os resultados destas duas abordagens se mostraram superiores aos dois anteriores; em contrapartida os tempos de resposta de ambas foram superiores. As análises dos resultados destas quatro abordagens podem ser vistas na Seção 6.4.

Para cada uma das abordagens foram realizados testes com instâncias de 36, 49, 64, 81 e 100 nós sensores sem fio baseados no padrão MICA2DOT (XBOW, 2006a) (Figura 1.1), distribuídos de forma aleatória em uma área de $2500m^2$. Para cada instância um nó sorvedouro está presente para coletar e processar todos os dados monitorados pelos sensores, ficando ele também responsável por garantir, em alguns casos, o controle da rede. Os algoritmos mais dispendiosos são executados no sorvedouro, que possui maior poder de processamento e energia ilimitada.

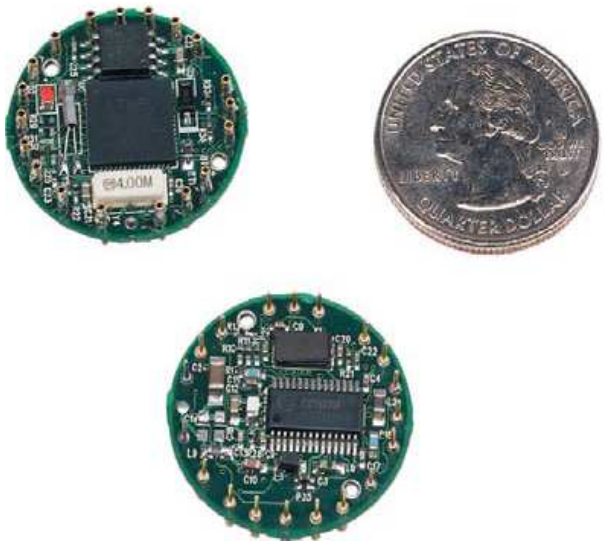


Figura 1.1 Nó sensor Mica2dot

CAPÍTULO 2

REDES DE SENSORES SEM FIO

As Redes de Sensores sem Fio (RSSFs) têm sido objeto de grande interesse de pesquisa nos últimos anos. Devido às suas características intrínsecas, esse tipo de rede possibilita monitorar áreas inóspitas e de difícil acesso. Compostas geralmente por uma grande quantidade de nós sensores e por pelo menos um nó sorvedouro (nó *sink*), as RSSFs realizam comunicação por "múltiplos saltos" (do inglês *multi-hop*). Na Figura 2.1, é possível ver um exemplo de como funciona este tipo de rede. Cada nó sensor é responsável por monitorar uma determinada área e enviar as informações coletadas para um vizinho próximo, que propaga seus dados para outro vizinho, e assim por diante, até que as informações cheguem ao nó sorvedouro. Este é o responsável por processar as informações, podendo tomar algumas decisões quanto ao funcionamento da rede ou apenas enviar as informações ao usuário. Esta forma de comunicação evita que todos os nós sensores necessitem se comunicar diretamente com o nó sorvedouro, evitando um gasto excessivo de energia, pois a potência necessária para a transmissão é reduzida em virtude da menor distância de comunicação.

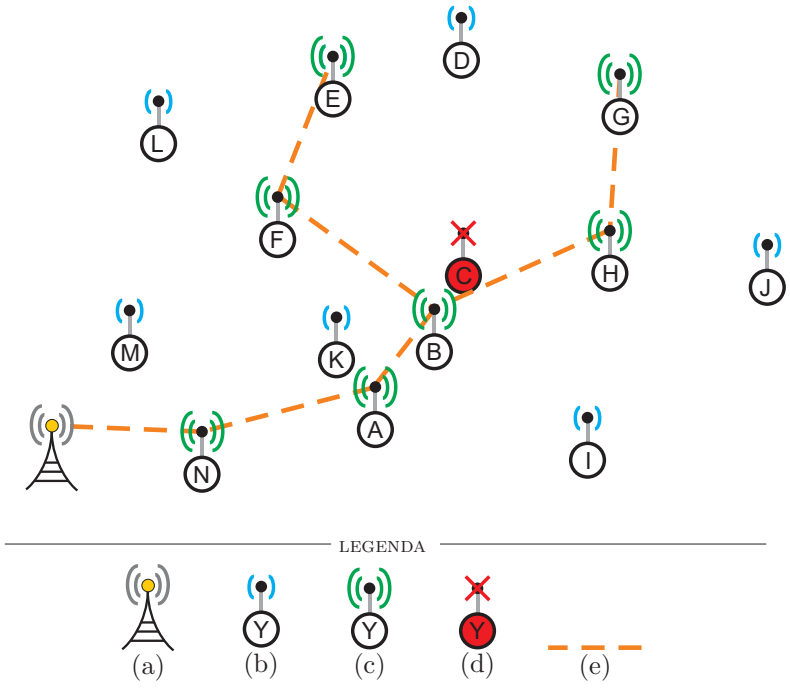


Figura 2.1 Exemplo de comunicação *multi-hop*. Legenda: (a) sorvedouro, (b) sensor Y desativado, (c) sensor Y ativado, (d) sensor Y falho e (e) link de comunicação.

Consideradas como casos especiais de redes móveis *ad hoc* (MANET - *Mobile Ad hoc Network*) (Loureiro et al., 2003) as RSSFs são constituídas por nós sensores que são

projetados para coletar informações diversas e para propagar essas informações. Como pode ser visto na Figura 2.2, encontrada também em (Vieira et al., 2003), os nós sensores são compostos basicamente por:

- unidade de comunicação sem fio;
- unidade de energia;
- unidade de sensoriamento;
- unidade de computação.

A grande maioria dos nós sensores foram projetados para ter dimensões pequenas e um baixo custo, o que trouxe como consequência uma baixa capacidade de recursos como: energia, processador e transceptor. Apesar dos recursos individuais dos nós sensores serem limitados, um esforço colaborativo entre eles permite a realização de tarefas de maior complexidade, o que não seria possível por meio de apenas um componente (Ruiz et al., 2004b).

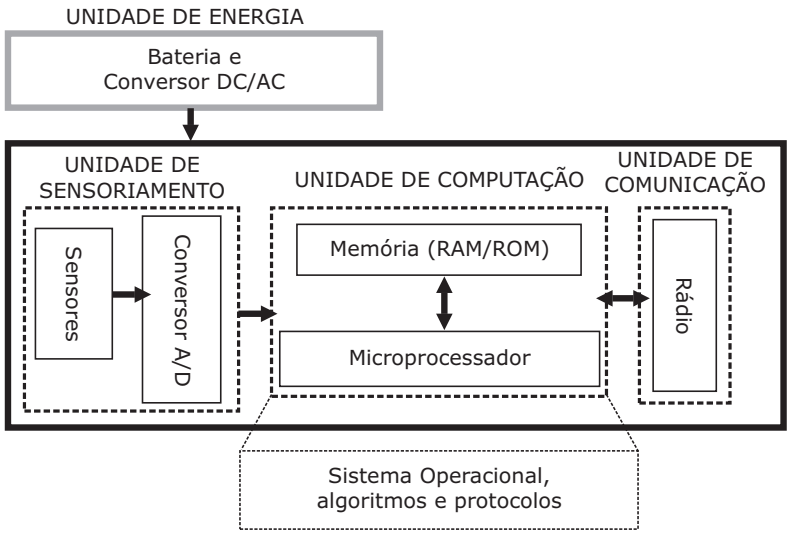


Figura 2.2 Principais componentes de um nó sensor.

Cada nó sensor é projetado para monitorar e enviar informações a uma certa distância. Este limite está relacionado à potência despendida pelo nó sensor para efetuar cada uma destas tarefas. A estes limites dão-se o nome de Raio de Sensoriamento (RS) e Raio de Comunicação (RC) (Figura 2.3), que podem ser definidos da seguinte maneira:

Definição 1. ***Raio de Sensoriamento - RS:*** *delimita a região que um nó sensor é capaz de monitorar. A região corresponde a uma circunferência de raio RS com o nó sensor no centro. Não considerando possíveis obstáculos e interferências, todo e qualquer evento que o nó sensor estiver preparado para coletar e que ocorrer nesta região será registrado.*

□

Definição 2. ***Raio de Comunicação - RC:*** *delimita a região para a qual um nó sensor é capaz de transmitir uma informação. A região corresponde a uma circunferência de raio RC com o nó sensor no centro. Não considerando possíveis obstáculos e interferências, um nó sensor é capaz de transmitir informações para outro nó sensor ou para o sorvedouro se estes estiverem a uma distância máxima de RC.* □

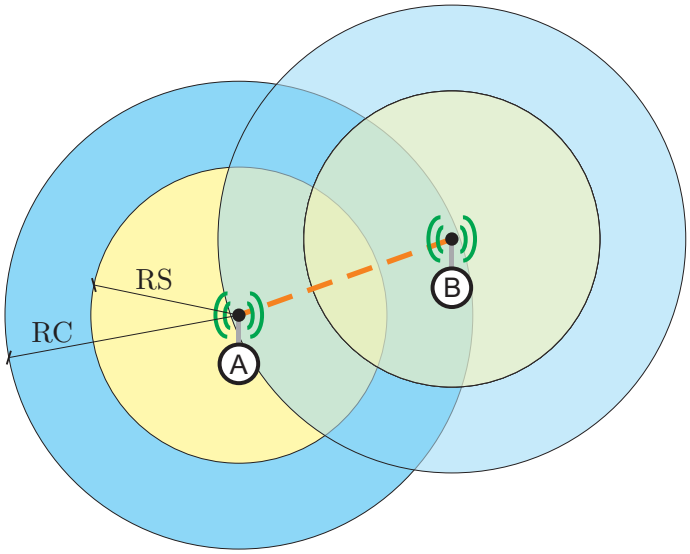


Figura 2.3 Exemplo para identificar os Raios de Sensoriamento (RS) e Comunicação (RC)

Geralmente nas RSSFs o RC é maior que o RS. No entanto existem possibilidades de variações destes limites de forma dinâmica, de acordo com a demanda, potência ou estado de energia do nó. Sabe-se que quanto maior for a distância para a qual um nó sensor enviar uma informação, mais energia será gasta com essa operação (Akyildiz et al., 2002; Yick et al., 2008).

Existem vários projetos para o desenvolvimento de nós sensores com boa capacidade de monitoramento a um baixo custo de energia. As tecnologias utilizadas procuram um menor consumo nas tarefas a serem executadas pelo sensor. As formas de comunicação exploradas nas RSSF são: ótica, infravermelho e rádio frequência (RF), sendo a mais utilizada a RF. Alguns aspectos afetam o consumo de energia gasta com a transmissão por RF em um nó sensor, sendo eles: tipo de modulação, taxa de dados e raio de transmissão. Normalmente os dispositivos de RF presentes nos nós sensores podem operar em diferentes modos: transmitindo, recebendo, "escuta" (*idle*) e "dormindo" (*sleep*) (Vieira et al., 2003).

Loureiro et al. (2003) separam as funcionalidades de uma RSSF em cinco grupos de atividades: estabelecimento da rede, manutenção, sensoriamento, processamento e comunicação. Segundo os autores, as funções podem ser empregadas de forma simultânea e podem estar presentes em diferentes momentos do tempo de vida das redes.

- **Estabelecimento:** Esse grupo de funções dá suporte ao processo de como os sensores são dispostos no ambiente e se organizam inicialmente. Em determinadas

aplicações é possível fazer um estudo da localização dos sensores, ou seja, previamente fazer uma análise da melhor disposição dos nós presentes na rede. Em contrapartida existem aplicações, como monitorar áreas inóspitas ou de grande risco, nas quais não é possível inserir os sensores de maneira previamente estudada. Nesses casos é possível lançar os sensores através de um veículo aéreo na região a ser monitorada. Como uma RSSF é um sistema auto-organizado, a própria rede pode buscar meios para que seja descoberta a localização de cada sensor. Devido a esta característica de auto-organização, os sensores podem comunicar entre si, podendo formar grupos (*clusters*) ou até mesmo se adaptarem para acomodar a ocorrência de falhas.

- **Manutenção:** Essas funções são responsáveis por fazer com que a RSSF fique em funcionamento pelo maior tempo possível. Com o passar do tempo, alguns sensores podem perder sua capacidade de sensoriamento e/ou comunicação, sendo necessário algum procedimento para que os requisitos da aplicação continuem sendo atendidos. Como na maioria das vezes os nós sensores são estacionários, na ocorrência de falhas se faz necessária normalmente uma mudança de topologia. As funções de manutenção podem ser utilizadas concomitantemente com funções de outros grupos (estabelecimento, sensoriamento, processamento e comunicação), sempre de forma a buscar maneiras para prolongar o tempo de vida da rede.

- **Sensoriamento:** Como visto na Figura 2.2 cada nó sensor é composto por uma unidade de sensoriamento. Através das funções deste grupo, os sensores têm habilidade para perceber o ambiente ao seu redor (até um determinado limite, segundo a Definição 1) e coletar informações deste ambiente. Conforme o tipo de nó sensor utilizado e o tipo de aplicação, a tarefa de sensoriamento pode prover algumas informações como: tipo do dado coletado (exemplo: temperatura, pressão, imagem, etc.), frequência da amostragem, distância do alvo monitorado. Nesta tarefa pode ser possível identificar regiões que mais de um nó sensor esteja monitorando. Este fator pode ser importante para a manutenção da rede, pois havendo sobreposição de áreas monitoradas os sensores podem mudar seus estados, assim economizando energia.

- **Processamento:** Loureiro et al. (2003) dividem o processamento na rede de sensores em duas categorias:

  - Processamento de suporte: "Diz respeito a todo processamento funcional dos sensores, ou seja, o processamento envolvido com o gerenciamento, comunicação e manutenção da rede, como por exemplo, as atividades envolvidas com os protocolos."
  - Processamento da informação: "Os dados coletados pelo nodo sensor podem ser processados em função da aplicação e/ou do envolvimento do nodo sensor em relações de colaboração. Os dados poderão estar sujeitos a compressão, correlação, criptografia, assinatura digital, etc. Um outro processamento importante diz respeito aos gatilhos que definem os estímulos para a coleta dos

dados. Por exemplo, os nodos sensores de temperatura podem ter seu processamento estimulado em função de uma variação ou rompimento dos limites estabelecidos."

- **Comunicação:** Cada nó sensor presente na rede tem uma certa capacidade para comunicação, conforme Definição 2. Normalmente a potência despendida na comunicação não deve ser muito grande para que a bateria do nó sensor dure por mais tempo. Devido a este motivo e às características físicas do próprio sensor, a distância de comunicação é normalmente reduzida. Os dados coletados utilizando as funções de sensoriamento precisam ser enviados à estação rádio-base. No entanto, devido à limitação da comunicação, nem sempre isto pode ser feito de forma direta. Assim, a RSSF faz uso de envio de dados por "multi-saltos" (*multi-hop*). Detalhes de protocolos de comunicação em RSSF pode ser vistos em (Demirkol et al., 2006).

2.1 CARACTERIZAÇÃO DAS RSSFS

Uma classificação de RSSF pode ser feita de acordo com o seu objetivo e sua área de aplicação. Segundo Ruiz et al. (2004a), a aplicação para qual uma RSSF foi construída influenciará diretamente nos seguintes quesitos:

- funções exercidas pelos nós da rede;
- arquitetura dos nós sensores presentes;
- quantidade de nós presentes na rede;
- distribuição inicialmente planejada para a rede;
- forma pela qual os nós são inseridos no ambiente;
- escolha dos protocolos da pilha de comunicação;
- tipo de dado que será tratado;
- tipo de serviço que será provido pela rede;
- tempo de vida da rede.

Conforme definido por Ruiz (2003), as RSSFs podem ser classificadas conforme a configuração (Tabela: 2.1), o sensoriamento (Tabela: 2.2) e o tipo de comunicação (Tabelas: 2.3 e 2.4).

Tabela 2.1 Caracterização das Redes de Sensores sem Fio segundo a Configuração. (Ruiz, 2003)

Configuração		
Composição	Homogênea	Rede composta de nós que apresentam a mesma capacidade de hardware. Eventualmente os nós podem executar software diferente.
	Heterogênea	Rede composta por nós com diferentes capacidades de hardware.
Organização	Hierárquica	RSSF em que os nós estão organizados em grupos (*clusters*). Cada grupo terá um líder (*cluster-head*) que poderá ser eleito pelos nós comuns. Os grupos podem organizar hierarquias entre si.
	Plana	Rede em que os nós não estão organizados em grupos.
Mobilidade	Estacionária	Todos os nós sensores permanecem no local onde foram depositados durante todo o tempo de vida da rede.
	Móvel	Rede em que os nós sensores podem ser deslocados do local onde inicialmente foram depositados.
Densidade	Balanceada	Rede que apresenta uma concentração e distribuição de nós por unidade de área considerada ideal segundo a função objetivo da rede.
	Densa	Rede que apresenta uma alta concentração de nós por unidade de área.
	Esparsa	Rede que apresenta uma baixa concentração de nós por unidade de área.
Distribuição	Irregular	Rede que apresenta uma distribuição não uniforme dos nós na área monitorada.
	Regular	Rede que apresenta uma distribuição uniforme dos nós na área monitorada.

Tabela 2.2 Caracterização das Redes de Sensores sem Fio segundo o Sensoriamento. (Ruiz, 2003)

Sensoriamento		
Coleta	Periódica	Os nós sensores coletam dados sobre o(s) fenômeno(s) em intervalos regulares. Um exemplo são as aplicações que monitoram o canto dos pássaros. Os sensores farão a coleta durante o dia e permanecerão desligados durante a noite.
	Contínua	Os nós sensores coletam os dados continuamente. Um exemplo são as aplicações de exploração interplanetária que coletam dados continuamente para a formação de base de dados para pesquisas.
	Reativa	Os nós sensores coletam dados quando ocorrem eventos de interesse ou quando solicitado pelo observador. Um exemplo são as aplicações que detectam a presença de objetos na área monitorada.
	Tempo Real	Os nós sensores coletam a maior quantidade de dados possível no menor intervalo de tempo. Um exemplo são aplicações que envolvem risco para vidas humanas tais como aplicações em escombros ou áreas de desastres. Outro exemplo são as aplicações militares onde o dado coletado é importante na tomada de decisão e definição de estratégias.

Tabela 2.3 Caracterização das Redes de Sensores sem Fio segundo a Comunicação (Parte A). (Ruiz, 2003)

Classificação segundo a Comunicação		
Disseminação	Programada	Os nós disseminam dados em intervalos regulares.
	Contínua	Os nós disseminam os dados continuamente.
	Sob Demanda	Os nós disseminam os dados em resposta à consulta do observador e à ocorrência de eventos.
Tipo Conexão	Simétrica	Todas as conexões existentes entre os nós sensores, com exceção do nó sorvedouro têm o mesmo alcance.
	Assimétrica	As conexões entre os nós comuns têm alcances diferentes.
Transmissão	Simplex	Os nós sensores possuem transceptor que permite apenas transmissão da informação.
	Half-duplex	Os nós sensores possuem transceptor que permite transmitir ou receber em um determinado instante.
	Full-duplex	Os nós sensores possuem transceptor que permite transmitir e receber dados ao mesmo tempo.

Tabela 2.4 Caracterização das Redes de Sensores sem Fio segundo a Comunicação (Parte B). (Ruiz, 2003)

Classificação segundo a comunicação		
Alocação de Canal	Estática	Neste tipo de rede se existirem "n" nós, a largura de banda é dividida em "n" partes iguais na frequência (FDMA - *Frequency Division Multiple Access*), no tempo (TDMA - *Time Division Multiple Access*), no código (CDMA - *Code Division Multiple Access*), no espaço (SDMA - *Space Division Multiple Access*) ou ortogonal (OFDM - *Orthogonal Frequency Division Multiplexing*). A cada nó é atribuída uma parte privada da comunicação, minimizando a interferência.
	Dinâmica	Neste tipo de rede não existe atribuição fixa de largura de banda. Os nós disputam o canal para comunicação dos dados.
Fluxo de Informação	*Flooding*	Neste tipo de rede, os nós sensores fazem *broadcast* de seus dados para seus vizinhos que fazem *broadcast* desses dados para outros até que a informação alcance o ponto de acesso. Esta abordagem promove um alto *overhead* mas está imune às mudanças dinâmicas de topologia e a alguns ataques de impedimento de serviço (DoS - *Denial of Service*).
	Multicast	Neste tipo de rede os nós formam grupos e usam o *multicast* para comunicação entre os membros do grupo.
	Unicast	Neste tipo de rede, os nós sensores podem se comunicar diretamente com o ponto de acesso usando protocolos de roteamento "multi-saltos".
	Gossiping	Neste tipo de rede, os nós sensores selecionam os nós para os quais enviam os dados.
	Bargaining	Neste tipo de rede, os nós enviam os dados somente se o nó destino manifestar interesse, isto é, existe um processo de negociação.

De acordo com Yick et al. (2008), em uma aplicação em RSSF a gama de tarefas é classificada em três diferentes grupos, conforme mostrado na Figura 2.4. Esses grupos compreendem:

- **Sistema:** neste grupo, cada nó sensor é considerando um sistema individual e para suportar diferentes aplicações de software, o desenvolvimento de novas plataformas, sistemas operacionais e sistemas de armazenamento são necessários.
- **Protocolos de Comunicação:** é o grupo responsável por permitir a comunicação entre a aplicação e os nós sensores. Além disso, também é o responsável pela comunicação entre os próprios nós sensores da rede.
- **Serviços:** neste grupo são desenvolvidos os serviços para dar suporte à aplicação, considerando o desempenho do sistema e a eficiência da rede.

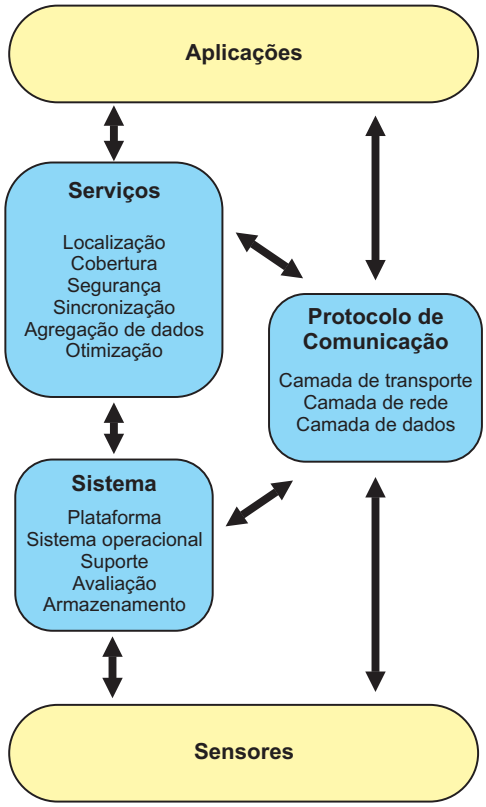


Figura 2.4 Tarefas nas RSSF (Yick et al., 2008).

2.2 ARQUITETURA DE RSSFS

Conforme já mencionado, as RSSFs são consideradas um caso especial de rede móvel *ad hoc* (MANET - *Mobile Ad hoc Network*) (Loureiro et al., 2003). Devido às suas características particulares não é possível aplicar diretamente os protocolos utilizados nas redes *ad hoc* convencionais.

Na Figura 2.5 é possível visualizar a pilha de protocolos utilizada pelos nós sensores de uma RSSF. Os protocolos de comunicação consistem nas camadas de aplicação, transporte, rede, enlace e física (Yick et al., 2008). A camada de aplicação é composta por protocolos de gerenciamento e consulta. Com base no tipo de aplicação, diferentes tipos de softwares podem ser construídos para utilização nesta camada.

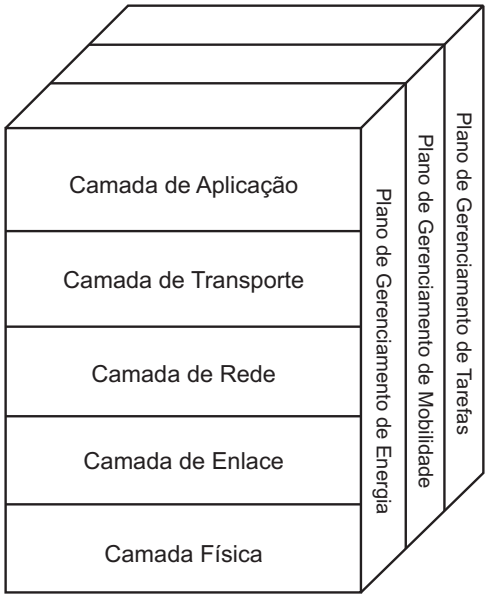


Figura 2.5 Pilha de protocolos das RSSF (Akyildiz et al., 2002).

A camada de transporte é a responsável por garantir a qualidade e a confiabilidade dos dados tanto no sorvedouro como nos nós sensores. Esta camada deve suportar múltiplas aplicações, recuperação de pacotes perdidos e mecanismo de controle de congestionamento (Yick et al., 2008). Exemplo destes protocolos podem ser vistos em (Iyer, 2005; Zhou et al., 2005; Park et al., 2004; Gungor & Özgür B. Akan, 2006). No entanto, ao contrário das redes tradicionais, a utilização deste protocolo nem sempre é necessária, pois a maioria das aplicações em RSSF admitem perda de dados (Ruiz et al., 2004a).

A principal função da camada de rede é fornecer o roteamento dos dados através da rede, da fonte ao destino. Como as características das RSSF diferem das redes físicas, ao se projetar um protocolo de rede este deve ser escalável, e deve gerenciar facilmente a comunicação entre os vários nós sensores presentes na rede, garantindo que os dados coletados por eles consigam facilmente chegar ao nó sorvedouro. Exemplos destes protocolos podem ser vistos em (Zhang et al., 2006; Yin & Madria, 2006; Du et al., 2006).

A camada de enlace está relacionada com a transferência de dados entre nós sensores que compartilham o mesmo link. Como mencionado em (Ruiz et al., 2004a), os requisitos da camada de enlace são diferentes para os diferentes tipos de RSSF apresentados nas Tabelas 2.1, 2.2, 2.3 e 2.4. Devido à sua dependência da aplicação e à forma de comunicação sem fio, para uma transferência de dados eficaz é necessário um controle de acesso ao meio (MAC - *Medium Access Control*). Conforme mostrado em (Yick et al., 2008) o projeto de um protocolo MAC deve ter os seguintes atributos: eficiência energética, es-

calabilidade, sincronização de quadro, utilização da largura de banda, controle de fluxo e controle de erro para comunicação de dados. Exemplos deste protocolo podem ser vistos em (Rajendran et al., 2003; Firoze et al., 2007; Saxena et al., 2008; Yigitel et al., 2010). O MAC em uma RSSF deve procurar atingir dois objetivos: criar uma infraestrutura e fazer uma divisão justa e eficiente dos meios de comunicação entre os nós sensores. Através do MAC é possível estabelecer a comunicação *multi-hop* e também a habilidade de auto organização das RSSF. Outra função importante desta camada é a técnica de recuperação, ou controle de erro. Algumas dessas técnicas são: *automatic repeat request* (ARQ) (Ganti et al., 2006), *forward error correction* (FEC) (Blahut, 1983), *hybrid ARQ (HARQ)* (Liu et al., 1997), *simple packet combining* (SPaC) (Dubois-ferrière et al., 2005), e *multi-radio diversity* (MRD) (Miu et al., 2005).

Por fim, há a camada física, que é responsável por interagir com a camada MAC realizando transmissão, recepção e modulação. Esta interação também é responsável por detectar e corrigir prováveis erros na rede. É nesta camada que é construída uma interface para transmissão de fluxos de bits sobre o meio de comunicação físico (Yick et al., 2008). Segundo essa mesma referência, a minimização do consumo de energia com o intuito de maximizar a vida útil da rede pode começar na camada física. A energia gasta para o funcionamento do circuito de rádio é fixa, no entanto a energia despendida para transmitir os dados pode variar com base na interferência, perda de canal e na distância de transmissão. Desta forma, a seleção adequada da potência de transmissão é necessária para minimizar a perda de energia e para um melhor funcionamento da rede.

Além das camadas de aplicação, de transporte, de rede, de enlace e física, na Figura 2.5 é possível visualizar 3 planos de gerenciamento, que são: plano de gerenciamento de energia, de mobilidade e de tarefas. Estes planos ajudam os nós sensores a coordenarem as tarefas de sensoriamento e a reduzirem o consumo total de energia. Através destes planos de gerenciamento torna-se possível fazer com que os sensores façam um trabalho colaborativo na rede (Akyildiz et al., 2002).

Devido ao consumo de energia ser um fator muito importante em uma RSSF, a gerência do gasto de energia de cada nó sensor fica a cargo do plano de gerenciamento de energia. Por exemplo, quando o nível de energia de um nó sensor está muito baixo, este pode enviar uma mensagem em *broadcast* para informar sua situação aos demais. Desta forma seus vizinhos passam a saber da sua limitação e podem decidir não utilizá-lo mais para um possível roteamento dos dados via *multi-hop*. No entanto este nó sensor pode continuar realizando o sensoriamento, enviando mensagens apenas quando detectar algum evento.

O plano de gerenciamento de mobilidade é utilizando quando a rede possui algum nó sensor com possibilidade de locomoção, veja Tabela 2.1. Neste plano de gestão é detectado e registrado o movimento dos nós sensores de forma a garantir um caminho de volta dos dados coletados para o usuário.

O plano de gerenciamento de tarefas é responsável por balancear e escalonar as tarefas de sensoriamento em uma determinada região. Grande parte das RSSFs possuem uma alta densidade de nós sensores, de forma que nem todos os sensores presentes em uma mesma região precisam executar a tarefa de detecção ao mesmo tempo. Com base neste levantamento é possível balancear e escalonar as tarefas entre os sensores de forma a

equilibrar o gasto de energia da rede. Como resultado, é possível que em certos momentos sensores que possuem mais energia fiquem responsáveis por executar mais tarefas.

2.3 TIPOS DE PROBLEMAS RELACIONADOS ÀS RSSF

O estudo das RSSFs pode ser dividido em vários subproblemas. Nesta seção são apresentados alguns destes subproblemas:

- distribuição e organização dos nós sensores na rede;
- localização;
- agregação e fusão de dados;
- roteamento e clusterização visando economia de energia;
- escalonamento;
- gerência de qualidade de serviço.

Além de mostrar os problemas que envolvem as RSSFs, o objetivo desta Seção 2.3 é também mostrar a influência das técnicas de Inteligência Computacional (IC) como ferramenta para resolver tais problemas.

Técnicas de IC vêm sendo aplicadas em muitas áreas da ciência, como: biometria, controle, robótica e para resolução de problemas combinatórios em gerais. A IC pode combinar elementos de aprendizagem, adaptação, evolução e lógica difusa para criar máquinas inteligentes. É possível encontrar na literatura trabalhos que têm mostrado a eficiência da aplicação de técnicas de IC em RSSFs.

Kulkarni et al. (2011) fazem uma revisão na literatura das técnicas de IC aplicadas a RSSF. Tal artigo procura fazer uma ponte com os temas abordados nas conferências e revistas que não têm como foco principal a área de RSSF.

2.3.1 Distribuição e Organização

A Tabela 2.1 apresenta a classificação de RSSF segundo a distribuição dos nós sensores, podendo esta ser irregular ou regular. Os nós sensores utilizados para monitorar áreas inóspitas frequentemente são arremessados por um veículo aéreo, o que faz com que a sua distribuição fique não uniforme. Por outro lado, exemplos de redes com distribuição regular podem ser encontrados em sistemas de monitoramento biológico e sistemas industriais, onde os sensores podem ser colocados manualmente em pontos estratégicos para coleta de dados.

Técnicas de IC têm ajudado muito no processo de construção e no planejamento da organização dos nós sensores de uma RSSF. Em (Zhao & Liang, 2005) é mostrado uma algoritmo para organizar a disposição dos sensores através de lógica difusa. O algoritmo divide a área em um *grid*, formando sub-áreas, e faz uso da lógica difusa para determinar o número de sensores que devem ser postos em uma determinada sub-área. Os autores

mostram que a "distribuição difusa" representa uma melhoria significativa em comparação com o pior caso de cobertura em uma distribuição uniforme.

Uma outra técnica que pode ser utilizada são os algoritmos evolucionários. Um exemplo pode ser visto em (Carballido et al., 2007). Neste artigo os autores desenvolvem um Algoritmo Genético (AG) para ser utilizado como um sistema de apoio a decisão. Um engenheiro de processos introduz no sistema informações relativas ao processo. Este é modelado e resolvido por um AG que retorna como resultado um conjunto de soluções. A solução escolhida passa por uma validação para saber se todos os critérios foram satisfeitos; caso não tenham sido, a rede passa por ajustes para ser testada novamente. Segundo os autores, o uso do AG como sistema de apoio à decisão sugeriu configurações iniciais de rede com custo 60% menor e um menor tempo de validação, em comparação com projetos baseados apenas na experiência e habilidade do projetista.

2.3.2 Localização

Para monitorar ambientes inóspitos e de difícil acesso, os nós sensores frequentemente são lançados de um veículo aéreo, não sendo possível determinar de antemão a localização em que irão ficar. No entanto, saber a localização destes sensores é importante em muitas situações. Para determinadas aplicações não faz sentido detectar apenas a presença de um evento sem saber o local onde ele ocorreu. Outro aspecto importante é a necessidade de se saber a localização dos sensores para programar a forma de fazer o roteamento dos dados coletados. Uma revisão sobre sistemas de localização de nós sensores em uma RSSF pode ser encontrado em (Boukerche et al., 2007).

Um algoritmo de localização utilizando AG é proposto em (Nan et al., 2007). No artigo os autores assumem que os nós sensores têm capacidade de medir a distância entre eles e seus vizinhos de "um salto". Os nós sensores são divididos em dois tipos, os chamados nós âncoras, cuja localização é conhecida, e os nós que não sabem sua localização. Os sensores cuja localização é desconhecida utilizam as coordenadas dos demais e algoritmos de localização para estimar suas posições. O AG é utilizado para otimizar uma função objetivo que representa a medida quantitativa do valor estimado das coordenadas. Os resultados apresentam erros de localização abaixo de 1% em todas as simulações. No entanto todo resultado produzido pelo AG necessita ser propagado para todos os nós sensores.

Um esquema de localização centralizado de duas fases é desenvolvido em (Marks & Niewiadomska Szynkiewicz, 2007). Na primeira fase do algoritmo apenas os nós sensores que possuem como vizinho (sensores que estão dentro do raio de comunicação) pelo menos três nós sensores âncoras são localizados. O sensores são separados em dois grupos, A: os que têm sua localização conhecida e B: os que a localização ainda não foi descoberta. Então é selecionado um nó sensor i do grupo B e de forma aleatória são selecionados, caso existam, três nós vizinhos de B pertencentes ao grupo A. A partir destes nós é estimada a localização de i. Uma vez conhecida a localização de i este é removido de B e colocado em A. Este passo é repetido até que não seja mais possível a localização de mais sensores. A segunda fase do algoritmo aplica técnicas de IC para melhorar a estimativa da localização realizada na primeira fase descrita acima. Foram realizados testes utilizando *Simulated*

Annealing (SA) e AG, ambos obtiveram erros abaixo de 1%, no entanto, segundo os autores, o SA obteve menores erros de localização e melhor performance computacional, em comparação com o AG.

2.3.3 Agregação e Fusão de dados

A fusão de dados é uma técnica utilizada para combinar os dados coletados por múltiplas fontes, de forma que a informação resultante possa ser mais precisa do que quando coletada por uma única fonte. Outro objetivo da fusão de dados é o de agregar os dados de forma a diminuir a sobrecarga da comunicação individual entre os nós sensores presentes na rede e o sorvedouro. Mais detalhes sobre fusão e agregação de dados em RSSF podem ser encontrados em (Nakamura, 2007) e (Rajagopalan & Varshney, 2006).

Soluções utilizando AG, lógica difusa, aprendizado por reforço e redes neurais são encontradas na literatura (Kulkarni et al., 2011). Em (Lazzerini et al., 2006) é proposta uma abordagem distribuída baseado em números difusos e operadores de média ponderada para um consumo eficiente de energia, em uma rede com fluxo de informação por *flooding* (ver Tabela 2.4). Neste estudo cada nó sensor mantém uma estimativa global de valor de agregação. Os autores representam este valor de agregação por um número que eles chamam de "número triangular simétrico difuso" (*symmetric triangular fuzzy number*). A agregação é realizada localmente por cada nó quando este coleta novos dados ou recebe dados de seu vizinho. Cada nó sensor mantém uma tabela com os valores recebidos de seus vizinhos, e com base na estimativa o nó sensor decide se um dado recém-medido deve ser propagado ou não pela rede. Utilizando esta técnica, o número de mensagens transmitidas foi reduzido, e como consequência houve um menor consumo de energia da rede.

Já em (Wu et al., 2004) é mostrada uma aplicação utilizando um agente móvel em uma RSSF distribuída. Este agente seleciona os nós sensores a serem visitados de forma a fazer a fusão dos dados coletados de forma apropriada. Um AG é utilizado para determinar a melhor rota para este agente. O número de nós na rota e a sequência em que estes são visitados pelo agente móvel têm um impacto na energia consumida, perda de caminho e precisão na detecção. Desta forma é elaborada uma função objetivo com base nestes três aspectos.

2.3.4 Roteamento e Clusterização Visando Economia de Energia

Rotear certo pacote de dados é encontrar um caminho para que a mensagem vá da origem em que o dado foi coletado até o destino desejado. Segundo os autores Kulkarni et al. (2011), em métodos proativos de roteamento, tabelas de roteamento são criadas e armazenadas independentemente de quando as rotas são utilizadas. Já em métodos reativos, as rotas são calculadas quando necessário. Em redes com grande quantidade de sensores, armazenar tabelas para roteamento pode acarretar enorme gasto com memória, o que torna necessário o uso de um método híbrido, proativo e reativo. Os mesmos autores afirmam que outra solução possível é fazer uso de agrupamentos (*clusters*), criando hierarquias.

Um algoritmo distribuído utilizando Redes Neurais (RNs) é utilizado por Barbancho

et al. (2006) para roteamento de dados. É utilizada uma RN em cada nó sensor da rede para gerenciar o fluxo dos dados. Com base em uma modificação do algoritmo de *Dijkstra* os autores propõem encontrar o caminho mínimo entre a estação base e cada nó sensor presente na rede. O peso das arestas utilizado no *Dijkstra* entre um nó sensor e seus vizinhos é calculado pelo próprio sensor. Para este cálculo é utilizado um mapa auto-organizado (SOM - *self organizing map*) baseado nos seguintes parâmetros de qualidade de serviço (QoS): latência, taxa de erro, ciclos de trabalho e transferência. Cada nó sensor faz um "ping" em seus vizinhos para descobrir a qualidade de cada "link".

Utilizando uma arquitetura de rede hierárquica, os autores Wazed et al. (2007) constroem um AG centralizado para fazer o roteamento em uma RSSF que visa maximizar o tempo de vida da rede. O AG retorna como resultado a rota de cada *cluster head* para o sorvedouro. O algoritmo tem como função objetivo maximizar o tempo de vida da rede, que é determinado pela falha do primeiro *cluster head*. Os resultados atingidos superam em média 200% em relação ao tempo de vida de dois algoritmos da literatura.

2.3.5 Escalonamento

Normalmente uma RSSF possui uma alta densidade de nós, o que possibilita dividir a tarefa de sensoriamento e roteamento entre os sensores presentes na rede. Sabe-se que um dos fatores críticos de uma RSSF é a energia, pois na maioria dos casos não é possível repor a bateria dos nós sensores. Uma das formas de economizar energia de uma rede é fazer um escalonamento entre os sensores. Mantendo os requisitos desejados, é possível fazer o revezamento das tarefas, equilibrando o consumo de energia da rede. Este escalonamento possibilita deixar alguns nós em modo *sleep* enquanto outros permanecem ativos executando as tarefas necessárias.

Em (Martins et al., 2010), os autores propõem um algoritmo pré-processado baseado em um AG. O problema é modelado de forma dinâmica tendo como objetivo maximizar o tempo de vida da rede. A decodificação da solução do algoritmo possibilita saber a sequência de ativação e desativação dos nós sensores de forma a maximizar o tempo de vida. Os resultados se mostraram superiores quando comparados a uma modelagem por programação linear Inteira (PLI) que busca minimizar o consumo de energia da rede, e que não trata a interação entre estágios, com o mesmo propósito de maximizar o tempo de vida.

Uma técnica popular utilizada para alocação de canal estático em rede de sensores é o TDMA (*time division multiple access*). Esta técnica é livre de colisões e de *overhead* de pacotes de controle. Utiliza o conceito de ciclo de operação reduzido com tempos de atividade (*listen*) e de repouso (*sleep*) para evitar o desperdício de energia com a escuta de pacotes destinados a outros nós (*overhearing*) e com a escuta do meio sem tráfego (*idle listening*) (Ruiz et al., 2004a).

Um método TDMA multiobjetivo de escalonamento é apresentado em (Mao et al., 2007). Os autores sugerem um algoritmo híbrido utilizando o método de Otimização por Enxame de Partículas (PSO - *Particle Swarm Optimization*) e AG. No problema de escalonamento TDMA o tempo é dividido em intervalos iguais chamados "*slots* de tempo". O desafio é atribuir os "*slots* de tempo" para os nós sensores de forma que não

ocorram colisões. Uma função objetivo do problema pondera os dois objetivos que são: energia consumida pela rede e o número total de "*slots*" no escalonamento. A função é otimizada inicialmente pelo método PSO, e a população final é utilizada como população inicial para o AG, sendo assim o AG fica responsável por refinar a solução. Os resultados obtidos pelos autores atingiram um pequeno ganho em relação à técnica aplicada em (Ergen & Varaiya, 2005).

2.3.6 Gerência de Qualidade de Serviço

Qualidade de Serviso (QoS - *Quality of Service*) geralmente se refere à qualidade tal como percebida pelo usuário e/ou aplicação. No universo de redes, QoS é entendida como uma medida de qualidade do serviço que a rede oferece para as aplicações e os utilizadores (Kulkarni et al., 2011). Um dos fatores mais críticos para garantir a qualidade de serviço (QoS - *Quality of Service*) em uma RSSF é o consumo de energia. Outros parâmetros como quantidade de nós sensores, cobertura e conectividade na rede, são fatores que devem ser abrangidos pela gerência de QoS. Estudos mais aprofundados da relação de QoS nas RSSFs podem ser vistos em (Martínez et al., 2007) e (Chen et al., 2011).

Com características centralizadas e distribuídas, os autores Munir et al. (2007) apresentam a abordagem *Fuzzy-QoS*, um método de estimativa de congestionamento da rede baseado em lógica difusa. O método faz uso da taxa de chegada de pacotes e do tamanho do *buffer* (variáveis *fuzzy*) para determinar e também manter uma tabela *fuzzy* da situação do congestionamento do *buffer*. Essa tabela mostra a estimativa do atual nível de congestionamento, e serve como apoio para cada nó sensor na decisão de descartar ou não um pacote. Os autores afirmam que tal abordagem reduziu o número de descartes de pacotes importantes.

Em (Khoukhi & Cherkaoui, 2008) é apresentado o *FuzzyMARS*, baseado também na teoria de lógica difusa. Os autores propõem uma solução para um menor esforço na regulação do tráfego em tempo real e no controle de admissão. O sistema *fuzzy* recebe como entrada o atraso da camada MAC e retorna como saída a taxa de regulação do tráfego. Os resultados apresentados mostram que o *FuzzyMARS* apresenta baixos e estáveis atrasos em diferentes canais de tráfego, escalabilidade e mobilidade da rede, preservando o rendimento.

2.4 CONSIDERAÇÕES SOBRE A RSSF ADOTADA NESTA TESE

Para a elaboração do modelo e dos algoritmos que serão propostos nesta tese, algumas considerações devem ser feitas:

- Os nós sensores possuem um identificador único e "sabem" sua localização.
- A aplicação requer monitoramento contínuo e coleta de dados periódica.
- O tráfego na RSSF é relativo apenas a dados da aplicação.
- Um conjunto de nós sensores pode cobrir uma determinada área com certa redundância.

- O protocolo de roteamento estabelece um caminho mínimo entre o nó sensor ativo e o nó sorvedouro, sendo que apenas sensores ativos podem pertencer à rota de comunicação. O gasto com a energia de transmissão (proporcional com a distância) foi utilizado como peso na aresta do grafo para efeitos de cálculo do caminho mínimo.

De acordo com as definições apresentadas neste Capítulo, a RSSF considerada neste trabalho possui as seguintes características: composição homogênea, organização plana, é estacionária e possui uma rede densa com distribuição irregular dos nós sensores. O sensoriamento possui monitoramento contínuo, no entanto, a disseminação de dados é programada. Possui um tipo de conexão simétrica e transmissão half-duplex. Por fim, a alocação de canal é estática e o fluxo de informação é *unicast*.

Como já mensionado no Capítulo 1, as abordagens aqui apresentadas para a solução do PDCC-RSSF operam em um nível mais abstrato dos nós sensores, não se preocupando com as características inerentes de suas camadas (física, enlace, rede, transporte e aplicação). Problemas como perda e colisão de pacotes, pacotes de controle (*overhead*), latência, confiabilidade dos dados, ciclos de trabalho (*duty cycle*), entre outros, não foram tratados nesta tese. Um modelo de comunicação simplificado foi aqui adotado, estabelecendo uma árvore de comunicação entre os sensores ativos da rede (Seção 6.1).

CAPÍTULO 3

DEFINIÇÕES E MODELOS

As Redes de Sensores sem Fio (RSSFs) são projetadas para monitorar uma determinada região de interesse. Tal região é uma área contínua que é discretizada por conveniência: isso permite o uso de algoritmos discretos para avaliar a cobertura. Desta forma, são atribuídos "pontos de demanda" aos elementos discretos que descrevem a área desejada. Nos exemplos aqui tratados, cada ponto de demanda irá corresponder a uma área de $1m^2$. Por exemplo, ao se monitorar uma área de $10m \times 10m$, esta área é dividida em 100 pontos de demanda, igualmente espaçados e distribuídos em um *grid*, como se pode ver na Figura 3.1. Analogamente, quando se considera uma área de $50m \times 50m$, há 2500 pontos de demanda, correspondentes às células do *grid* (Figura 3.2).

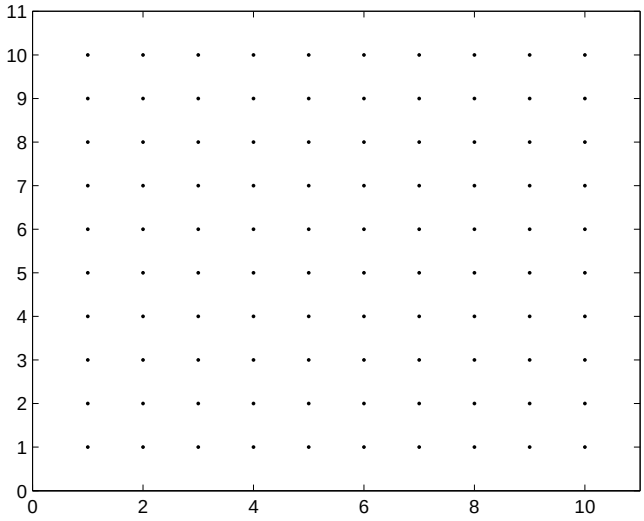


Figura 3.1 Exemplo de pontos de demanda em uma área de $100m^2$

3.1 DEFINIÇÃO DO PROBLEMA

A área a ser monitorada, na formulação aqui tratada, será descrita por pontos uniformemente espaçados, utilizados para avaliar a cobertura da rede. Cada um desses pontos d_i é referido como um ponto de demanda, que pode ser coberto por um nó sensor ativo s_j se a distância entre d_i e s_j for menor do que o raio de sensoriamento (RS - Definição 1) do sensor.

Sendo:

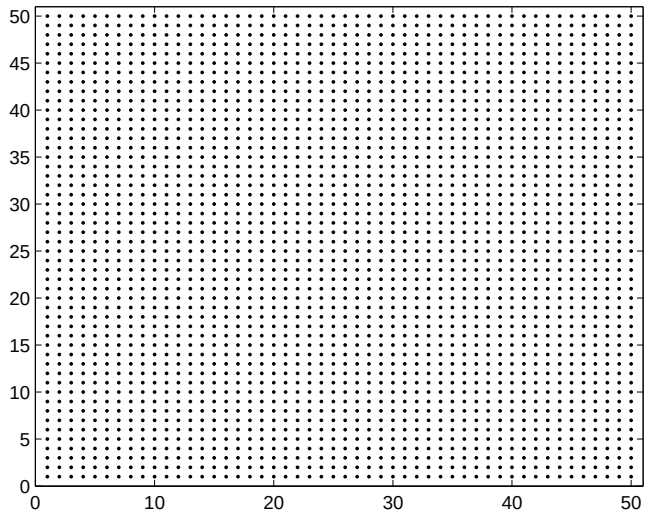


Figura 3.2 Exemplo de pontos de demanda em uma área de $2500m^2$

- $\mathcal{S}$ um conjunto de sensores;
- m o nó sorvedouro;
- $\mathcal{D}$ o conjunto de pontos de demanda d_i que descrevem a área a ser monitorada;
- C um número entre 0 e 1 que representa uma fração dos pontos de demanda $d \in \mathcal{D}$ que se requer que estejam cobertos.

A seguir é definido o problema de cobertura, o problema de conectividade e consequentemente o problema dinâmico de cobertura e conectividade em RSSF.

Definição 3. ***Problema de Cobertura:*** *O problema de cobertura em RSSF consiste em garantir que um conjunto de nós sensores $s \in \mathcal{S}$ cubra pelo menos uma fração C dos pontos de demanda $d \in \mathcal{D}$.* □

Definição 4. ***Problema de Conectividade:*** *O prolema de conectividade consiste em garantir que haja pelos menos um caminho entre cada nó sensor ativo $s \in \mathcal{S}$ e o nó sorvedouro m.* □

Definição 5. ***Problema Dinâmico de Cobertura e Conectividade em Redes de Sensores sem Fio (PDCC-RSSF):*** *O PDCC-RSSF busca uma sequência de configurações da rede que maximize seu tempo de vida, garantindo o atendimento às restrições de cobertura e conectividade em cada configuração.* □

Na elaboração do modelo e dos algoritmos algumas considerações devem ser feitas:

- Os nós sensores possuem um identificador único e "sabem" sua localização.

- A aplicação requer monitoramento contínuo e coleta de dados periódica.
- O tráfego na RSSF é relativo apenas a dados da aplicação.
- Um conjunto de nós sensores pode cobrir uma determinada área com certa redundância.
- O protocolo de roteamento estabelece um caminho mínimo entre o nó sensor ativo e o nó sorvedouro, sendo que apenas sensores ativos podem pertencer à rota de comunicação. Para o cálculo do caminho mínimo, o gasto com a energia de transmissão (proporcional com a distância) foi utilizado como peso na aresta do grafo.

De acordo com as definições apresentadas no Capítulo 2, a RSSF considerada neste trabalho possui as seguintes características:

- Segundo a configuração (Tabela 2.1): possui composição homogênea, organização plana, é estacionária e possui uma rede densa com distribuição irregular dos nós sensores.
- Segundo o sensoriamento (Tabela 2.2): possui um monitoramento contínuo.
- Segundo a comunicação (Tabelas 2.3 e 2.4): a disseminação de dados é programada, possui um tipo de conexão simétrica e transmissão half-duplex. Por fim, a alocação de canal é estática e o fluxo de informação é *unicast*.

Sob tais pressupostos, torna-se possível modelar a RSSF como um sistema dinâmico (Martins et al., 2010), como representado na Equação (3.1):

$$Y(k+1) = Y(k) + U(k) + Z(k) \tag{3.1}$$

Sujeito a:

$$u_i(k) \in \{0,1\} \quad \text{se} \quad y_i(k) = 0 \quad \forall i \in \{1,\ldots,|\mathcal{S}|\} \quad \forall k \in \{1,\ldots,n_{stg}\} \tag{3.2}$$

$$u_i(k) \in \{0,-1\} \quad \text{se} \quad y_i(k) = 1 \quad \forall i \in \{1,\ldots,|\mathcal{S}|\} \quad \forall k \in \{1,\ldots,n_{stg}\} \tag{3.3}$$

$$u_i(k) = 0 \quad \text{se} \quad \sum_{l=1}^{k} z_i(l) \neq 0 \quad \forall i \in \{1,\ldots,|\mathcal{S}|\} \quad \forall k \in \{1,\ldots,n_{stg}\} \tag{3.4}$$

$$z_i(k) = 0 \quad \text{se} \quad y_i(k) = 0 \quad \forall i \in \{1,\ldots,|\mathcal{S}|\} \quad \forall k \in \{1,\ldots,n_{stg}\} \tag{3.5}$$

$$Y(0) = [\ 0\ 0\ 0\ \ldots\ 0\ ] \tag{3.6}$$

em que:

- $Y(k)$ é um vetor $|\mathcal{S}|\times 1$ que representa o estado dos sensores $\mathcal{S} = \{s_1, s_2, \ldots, s_{|\mathcal{S}|}\}$ no estágio k. $y_i(k) = 1$ quando o nó sensor i está ativado no estágio k, caso contrário, $y_i(k) = 0$ (o nó sensor está desativado).
- $U(k)$ é um vetor $|\mathcal{S}| \times 1$ que representa as ações de controle (ativação e desativação dos sensores) que são realizadas no final do estágio k. A variável $u_i(k)$ pode assumir três diferentes valores:
  - $u_i(k) = 1$: o nó sensor i é ativado na transição do estágio k para o estágio $k + 1$;
  - $u_i(k) = -1$: o nó sensor i é desativado na transição do estágio k para o estágio $k + 1$;
  - $u_i(k) = 0$: o estado do nó sensor i permanece inalterado do estágio k para o estágio $k + 1$.
- $Z(k)$ é um vetor $|\mathcal{S}| \times 1$ que representa o evento não controlado de bateria descarregada em um nó sensor, no final do estágio k. Se $z_i(k) = -1$, então o nó sensor i é desativado na transição do estágio k para o estágio $k + 1$, não podendo mais ser ativado. Se $z_i(k) = 0$, então a bateria do nó sensor mantém o estado anterior no estágio $k + 1$ (carregada ou não carregada).
- $Y(0)$ é a condição inicial do sistema dinâmico.

Pressuposto 1. *No sistema dinâmico apresentado para as RSSF é assumido que, um estágio k termina após o primeiro nó sensor ativo falhar devido a um evento de bateria descarregada.* □

O estado de cada nó sensor no estágio k é definido: pelo seu estágio anterior, pela a ação de controle que é realizada no fim do estágio $k - 1$, e por uma eventual falha de bateria que pode ocorrer no nó sensor. Neste sistema, conforme o Pressuposto 1, o estágio k termina quando o primeiro nó sensor ativo falhar devido a um evento de falha de energia. O propósito deste trabalho é o de encontrar a sequência (*scheduling*) ótima de ativação de nós sensores e controles de desativação com o objetivo de maximizar o tempo de vida da rede, garantindo que as restrições de cobertura e conectividade sejam atendidas.

O problema proposto pode ser tratado como um problema de controle ótimo, em que as ações de controle ($U(k)$) são escolhidas de forma que a cobertura verificada na rede em cada instante siga o sinal de referência (limite de cobertura C), ao mesmo tempo assegurando que a conectividade seja mantida e o tempo de vida da rede seja maximizado. Neste modelo, as variáveis de decisão são as ações de controle, que definem quando cada nó sensor será ativado ou desativado. Existem, além disso, eventos não controlados relacionados com a perda da carga da bateria $Z(k)$. Tal como discutido em (Martins et al., 2010), o tempo de vida de rede pode ser modelado como se segue:

$$f_{lifetime}[\mathcal{U}, Y(0), E(0)] = \sum_{k=1}^{n_{stg}} F_{time}[Y(k-1) + U(k-1) + Z(k-1), E(k)] \qquad (3.7)$$

em que:

- $E(k)$ é um vetor $|\mathcal{S}| \times 1$ que representa a energia residual dos nós sensores no início do estágio k.
- $\mathcal{Y} = \{Y(0), Y(1), \ldots, Y(n_{stg})\}$.
- $\mathcal{U} = \{U(0), U(1), \ldots, U(n_{stg} - 1)\}$.
- $\mathcal{E} = \{E(0), E(1), \ldots, E(n_{stg})\}$.
- $n_{stg} \in \mathbb{N}$ é o número de estágios ocorridos até que a rede fique fora de serviço[1]. Este número varia de solução para solução, em função das variáveis $\mathcal{Y}$, $\mathcal{U}$ e $\mathcal{E}$
- $F_{time}[Y(k-1) + U(k-1) + Z(k-1), E(k)]$ é uma função que retorna o intervalo de tempo em que o sistema permanece no estágio k (em unidades de tempo – $u.t.$).
- $Y(k-1) + U(k-1) + Z(k-1)$ fornece a informação sobre o conjunto de nós que permanecerá ativo ao fim do estágio k.

O número de estágios pode variar dependendo da sequência de ativação dos nós sensores. Além disso, o último estágio termina quando o conjunto de nós sensores já não for capaz de restabelecer o nível de cobertura C, garantindo também a conectividade. Neste caso a rede fica fora de serviço.

É assumido que os vetores $Y(0)$ e $E(0)$ são conhecidos a priori, uma vez que todos os nós estão inicialmente desativados e a energia residual no início do primeiro estágio é a capacidade total das baterias. Os valores de $Y(k)$ e $E(k)$ em qualquer estágio k pode ser calculado utilizando $\mathcal{U}$, $Y(0)$ e $E(0)$.

Outra questão importante é a modelagem do gasto de energia de cada nó sensor. Deve ser notado que a energia de cada nó sensor $e_i(k) \in E(k)$ pode diminuir tanto quanto for a duração do estágio k. A energia residual $e_i(k+1)$ de cada nó sensor ativo i, pouco antes do início do estágio $k+1$, pode ser modelada como:

$$e_i(k+1) \;=\; e_i(k) - e_i^{act}(k) - e_i^{cons}(k) \;\cdot\; F_{time}\left[Y(k) + U(k) + Z(k), E(k)\right] \qquad (3.8)$$

em que:

- $e_i^{act}(k)$ é a energia gasta com a operação de ativação, se o nó sensor i é ativado no início do estágio k.
- $e_i^{cons}(k)$ é a energia consumida pelo nó sensor i ao longo do estágio k com as tarefas de monitoramento e comunicação a cada $u.t.$.

[1] A rede é considerada fora de serviço quando ela não é capaz de garantir a cobertura pré-estabelecida (C) e os requisitos de conectividade.

As expressões que representam a energia de ativação e energia consumida são apresentadas respectivamente em (3.9) e (3.10).

$$e_i^{act}(k) = \alpha_i(k) \cdot AE_i \tag{3.9}$$

onde:

$$\alpha_i(k) = \begin{cases} 1 \text{ se } u_i(k) = 1 \\ 0 \text{ caso contrário} \end{cases}$$

$$e_i^{cons}(k) = y_i(k) \cdot \left(MP_i + \sum_{l \in (\mathcal{S}-i)} \sum_{(p,i) \in \mathcal{I}^i} RP_i \cdot w_{l,p,i}(k) + \sum_{l \in \mathcal{S}} \sum_{(i,j) \in \mathcal{O}^i} TP_{i,j} \cdot w_{l,i,j}(k) \right) \tag{3.10}$$

Em que:

- $\mathcal{A}^s$ é o conjunto de arcos que conectam o nó sensor em outros nós sensores.
- $\mathcal{A}^m$ é o conjunto de arcos que conectam os nós sensores no nó sorvedouro.
- AE_i é a energia gasta na ativação do nó sensor i.
- MP_i é a potência de manutenção do nó sensor i (por $u.t.$).
- RP_i é a potência de recepção do nó sensor i (por $u.t.$).
- $TP_{i,j}$ é a potência de transmissão entre os sensores i e j, ou sensor i e o sorvedouro, $(i,j) \in \{\mathcal{A}^s \cup \mathcal{A}^m\}$ (por $u.t.$).
- $\mathcal{I}^i \subset \mathcal{A}^s$ é o conjunto de arcos do conjunto $\mathcal{A}^s$ que chegam no nó sensor $i \in \mathcal{S}$.
- $\mathcal{O}^i \subset \{\mathcal{A}^s \cup \mathcal{A}^m\}$ é o conjunto de arcos do conjunto $\mathcal{A}$ que saem do nó sensor $i \in \mathcal{S}$.
- $w_{l,i,j}(k)$ é a variável de decisão que assume 1 se o arco $(i,j) \in \{\mathcal{A}^s \cup \mathcal{A}^m\}$ está presente no caminho que conecta o nó sensor l ao nó sorvedouro m no estágio k, ou 0 caso contrário.

Por convenção assume-se, sem perda de generalidade, que o grafo estruturado em árvore tenha o nó sorvedouro como sua raiz, e que os arcos sejam orientados a partir das folhas para a raiz, de forma a definir adequadamente $\mathcal{I}^i$ e $\mathcal{O}^i$.

A seguir serão apresentadas as restrições do problema. Para assegurar que a cobertura da rede, em qualquer fase, não seja menor do que C, a primeira restrição (3.11) é definida:

$$g_1 \ : \quad G_{cov}[Y(k) + U(k) + Z(k)] \geq C \cdot |\mathcal{D}| \ , \forall k \in \{1, \ldots, n_{stg}\} \tag{3.11}$$

Nessa expressão:

- $G_{cov}[P(k)]$ é a função que retorna o número de pontos de demanda que são cobertos pelos nós sensores que estão ativos no estágio k.

Além desta restrição, outras seis devem ser satisfeitas. Elas são mostradas em (3.12), (3.13), (3.14), (3.15), (3.16) and (3.17).

$$g_2 \ : \ \mathcal{B}_{l,j} \cdot r_{l,j}(k) \leq y_l(k) \ \forall l \in \mathcal{S} \ , \ \forall j \in \mathcal{D} \ , \ \forall k \in \{1, n_{stg}\} \tag{3.12}$$

$$g_3 \ : \ r_{l,j}(k) \in \{0, 1\} \ \forall l \in \mathcal{S} \ , \ \forall j \in \mathcal{D} \ , \ \forall k \in \{1, n_{stg}\} \tag{3.13}$$

$$g_4 \ : \ \sum_{(i,p) \in \mathcal{I}^p} w_{l,i,p}(k) - \sum_{(p,j) \in \mathcal{O}^p} w_{l,p,j}(k) = 0 \ , \ \ \forall p \in (\mathcal{S} - l) \ , \ \forall l \in \mathcal{S} \ , \ \forall k \in \{1, n_{stg}\} \tag{3.14}$$

$$g_5 \ : \ \sum_{(l,j) \in \mathcal{O}^l} w_{l,l,j}(k) = y_l(k) \ , \ \ \forall l \in \mathcal{S} \ , \ \forall k \in \{1, n_{stg}\} \tag{3.15}$$

$$g_6 \ : \ w_{l,i,p}(k) \leq y_i(k) \ , \ \forall i \in \mathcal{S} \ , \ \forall l \in (\mathcal{S} - p) \ , \ \forall (i,p) \in (\mathcal{A}^s \cup \mathcal{A}^m) \ , \ \forall k \in \{1, n_{stg}\} \tag{3.16}$$

$$g_7 \ : \ w_{l,i,p}(k) \leq y_p(k) \ , \ \ \forall p \in \mathcal{S} \ , \ \forall l \in (\mathcal{S} - p) \ , \ \forall (i,p) \in (\mathcal{A}^s \cup \mathcal{A}^m) \ , \ \forall k \in \{1, n_{stg}\} \tag{3.17}$$

Nestas expressões:

- $\mathcal{B}$ é a matriz de conectividade em que uma célula $b_{l,j}$ assume valor 1 se o nó sensor l alcança o ponto de demanda j, ou 0 em caso contrário;
- $r_{l,j}(k)$ é uma variável binária que assume valor 1 se o nó sensor l cobre o ponto de demanda j no estágio k, ou 0 em caso contrário.

As restrições (3.12) e (3.13) são relativas ao problema de cobertura (Definição 3). A restrição (3.12) impõe que um nó sensor somente possa cobrir um determinado ponto de demanda se estiver ativo, e a constante em (3.13) assegura que a variável r seja binária. As restrições (3.14), (3.15), (3.16) e (3.17) são relativas ao problema de conectividade (Definição 4). As constantes (3.14) e (3.15) asseguram a existência de um caminho entre cada nó sensor ativo e o nó sorvedouro. As restrições (3.16) e (3.17) garantem que haja apenas nós sensores ativos no caminho.

Portanto, o PDCC-RSSF pode ser descrito como:

$$\mathcal{U}^* = \arg\max_{\mathcal{U}} f_{lifetime}[\mathcal{U}, Y(0), E(0)] \tag{3.18}$$

$$\text{sujeito a:} \begin{cases} g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \\ g_6 \\ g_7 \\ Y(0) = [0\ 0\ \ldots\ 0]' \\ E(0) = [E_{max}\ E_{max}\ \ldots\ E_{max}]' \end{cases} \tag{3.19}$$

em que: E_{max} é a capacidade de energia armazenada em cada nó.

Toda a modelagem do consumo de energia de cada nó sensor presente na rede, bem como as restrições de cobertura e conectividade, foram baseadas em um uma modelagem por Programação Linear Inteira (PLI) apresentada em (Nakamura et al., 2005) e (Nakamura, 2010). Variações destas modelagens podem ser vistas em (Martins et al., 2007), (Nakamura et al., 2007), (Martins et al., 2008), (Quintao et al., 2004) e (Quintao et al., 2005).

3.2 MODELAGEM MULTIOBJETIVO

Em alguns contextos de projeto de sistemas, a abordagem multiobjetivo pode apresentar uma vantagem ao ser comparada com a mono-objetivo, em virtude de produzir a informação referente ao *trade-off* entre diferentes objetivos de projeto. A abordagem multiobjetivo produz não apenas uma solução ótima, mas um conjunto de soluções chamadas *Pareto-ótimas*, que representam soluções ótimas para diferentes compromissos entre os objetivos.

Passa-se agora à formulação do Problema de Otimização Multiobjetivo (POM), com a definição do que seriam as suas soluções. Define-se preliminarmente as seguinte notação:

x **:** Vetor de variáveis de decisão. Esse vetor possui n coordenadas.

χ **:** Conjunto contendo todas as possíveis instâncias dos vetores de variáveis de decisão x.

$\chi*$ **:** Conjunto de *soluções eficientes*, ou *conjunto Pareto-ótimo* de um Problema de Otimização Multiobjetivo (POM). O conjunto $\chi*$ é um subconjunto do espaço χ.

Υ **:** Espaço dos objetivos, isto é, o espaço no qual se representa a imagem da função $\mathbf{f}(\cdot) : \chi \mapsto \mathbb{R}^m$. Esse espaço, no caso dos problemas aqui considerados, será sempre um espaço de variáveis reais $\mathbb{R}^m$.

x^* **:** Um ponto x pertencente ao conjunto $\chi*$.

η_x **:** O conjunto dos pontos x factíveis, subconjunto do conjunto χ.

Sejam os vetores $a \in \mathbb{R}^m$ e $b \in \mathbb{R}^m$. Definem-se da seguinte forma as operações de comparação entre esses vetores:

- $a \leq b$ se $a_i \leq b_i$ para todo $i = 1, \ldots, n$
- $a \geq b$ se $a_i \geq b_i$ para todo $i = 1, \ldots, n$
- $a < b$ se $a_i < b_i$ para todo $i = 1, \ldots, n$
- $a > b$ se $a_i > b_i$ para todo $i = 1, \ldots, n$
- $a \neq b$ se $a_i \neq b_i$ para algum $i = 1, \ldots, n$

O conceito de *dominância* encontra-se no centro da formulação do Problema de Otimização Multiobjetivo:

Definição 6. ***Dominância:*** *Em um problema de minimização diz-se que o ponto $x_1 \in \chi$ domina o ponto $x_2 \in \chi$ se $\mathbf{f}(x_1) \leq \mathbf{f}(x_2)$ e $\mathbf{f}(x_1) \neq \mathbf{f}(x_2)$. Equivalentemente, diz-se que $\mathbf{f}(x_1) \in \Upsilon$ domina $\mathbf{f}(x_2) \in \Upsilon$, nessas mesmas condições.* □

A partir desse conceito, é possível definir as soluções do Problema de Otimização Multiobjetivo (POM):

Definição 7. ***Solução Pareto-ótima:*** *Diz-se que $x^* \in \eta_x$ é uma solução Pareto-ótima de um POM se não existe qualquer outra solução $x \in \eta_x$ tal que $\mathbf{f}(x) \leq \mathbf{f}(x^*)$ e $\mathbf{f}(x) \neq \mathbf{f}(x^*)$, ou seja, se x^* não é dominado por nenhum outro ponto factível.* □

Desta forma, é possível formular o POM da seguinte forma:

$$(POM) : \begin{cases} \chi^* & = & \arg\min\limits_{x} \mathbf{f}(x) \\ & & \text{sujeito a: } x \in \eta_x \end{cases} \tag{3.20}$$

A minimização representada nessa expressão diz respeito à determinação das soluções *minimais* no conjunto η_x, com respeito à ordem parcial induzida pelo conceito de dominância. É possível definir um problema de maximização com a simples substituição, na definição de dominância, do operador $\leq$ pelo operador $\geq$.

Segundo Takahashi (2004), dado um POM, as soluções multiobjetivo (ou soluções Pareto-ótimas) são soluções que, entre si, não apresentam um ordenamento, ou seja, não há como definir, a partir da avaliação das funções objetivo, que uma solução seja melhor que a outra. Então, um ponto pertencente ao conjunto Pareto-ótimo tem as seguintes propriedades:

- É melhor (ou pelo menos equivalente) que qualquer outro ponto em pelo menos um objetivo.
- Não é dominado por nenhum outro ponto, embora não necessariamente domine todos os pontos que não sejam Pareto-ótimos.

Quando se trata de um problema de projeto, uma vez determinado o conjunto Pareto-ótimo do problema, normalmente é necessário escolher apenas uma solução para ser implementada. A informação contida em todo o conjunto Pareto-ótimo é importante para

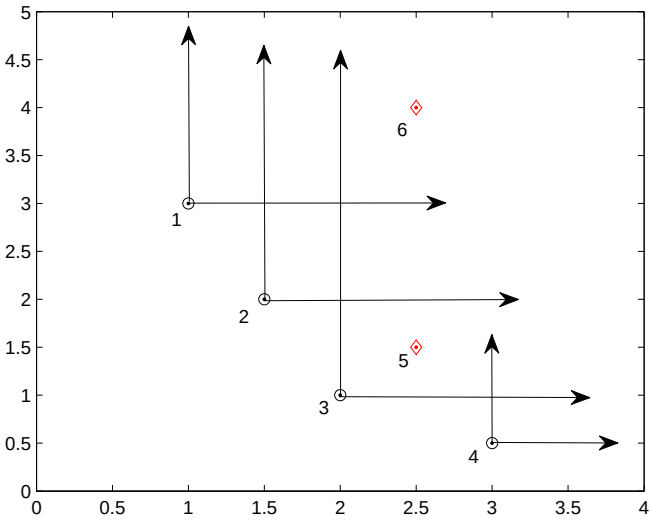


Figura 3.3 Exemplo de um conjunto Pareto-ótimo e de dominância

guiar a escolha dessa solução a ser implementada, pois revela os *trade-offs* envolvidos na opção por uma solução em detrimento de outras.

Na Figura 3.3 pode-se ver um exemplo em que os pontos 1, 2, 3 e 4, fazem parte do conjunto Pareto-ótimo da solução apresentada. Têm-se também nesta figura que as soluções 1 e 2 dominam a solução 6, e a solução 3 domina a 5 e a 6.

3.2.1 Modelagem Multiobjetivo para o PDCC-RSSF

Na abordagem original de apenas um objetivo, a cobertura é modelada como uma restrição que tem o papel de garantir que a cobertura da rede em cada instante de tempo seja maior ou igual a C. No entanto, a restrição de cobertura pode ser relaxada como um objetivo, de forma a permitir modos de operação para estender o tempo de vida da rede ao custo de reduzir a taxa de cobertura. Neste sentido, a cobertura da rede é considerada como um objetivo para ser maximizado, como mostrado em (3.21).

$$f_{coverage}[\mathcal{U}, Y(0), E(0)] = \frac{1}{n_{stg}} \sum_{i=1}^{n_{stg}} G_{cov}[Y(k-1) + U(k-1) + Z(k-1), E(k)] \quad (3.21)$$

A função (3.21) avalia a média da cobertura da rede durante seu tempo de vida. Uma vez que o valor da média tenha sido estabelecido, é interessante conhecer a cobertura mínima C_{min} de uma solução, pois podem existir aplicações em que taxas de cobertura abaixo de um certo limiar sejam inaceitáveis em qualquer instante de tempo. Desta forma, a restrição g_1 deve ser adaptada da seguinte forma:

$$g_1' \ : \quad G_{cov}[Y(k) + U(k) + Z(k)] \geq C_{min} \cdot |\mathcal{D}| \ , \forall k \in \{1, \ldots, n_{stg}\} \quad (3.22)$$

As demais restrições presentes no problema mono-objetivo também fazem parte da formulação multiobjetivo. Portanto, a modelagem multiobjetivo para o PDCC-RSSF pode

ser vista em (3.23) e (3.24).

$$\mathbb{U}^* = \arg\max_{\mathcal{U}} \begin{pmatrix} f_{lifetime}(\mathcal{U}, Y(0), E(0)) \\ f_{coverage}(\mathcal{U}, Y(0), E(0)) \end{pmatrix} \tag{3.23}$$

$$\text{sujeito a:} \begin{cases} g_1' \\ g_2 \\ g_3 \\ g_4 \\ g_5 \\ g_6 \\ g_7 \\ Y(0) = [0\ 0\ \ldots\ 0]' \\ E(0) = [E_{max}\ E_{max}\ \ldots\ E_{max}]' \end{cases} \tag{3.24}$$

em que:

$\mathbb{U}^* = \{\mathcal{U}_1, \ldots, \mathcal{U}_{|\mathbb{U}|}\}$ representa o conjunto Pareto-ótimo em relação às funções de tempo de vida e cobertura da rede.

O objetivo do algoritmo de otimização multiobjetivo que será construído para tratar esse problema será gerar amostras "bem espaçadas" do conjunto Pareto-ótimo $\mathbb{U}^*$, capazes de fornecer para o usuário uma boa representação do *trade-off* entre os objetivos de projeto.

CAPÍTULO 4

ALGORITMO GENÉTICO GEOMÉTRICO PARA RSSF

A partir dos modelos e conceitos apresentados no Capítulo 3, será construído, ao longo deste capítulo, um algoritmo intrinsecamente dinâmico para resolver o Problema Dinâmico de Cobertura e Conectividade em Redes de Sensores Sem Fio (PDCC-RSSF). A cada passo desta construção, será apresentada também uma extensão multiobjetivo do algoritmo. As soluções apresentadas aqui servem como referência para o projeto e gerenciamento de uma RSSF.

4.1 ALGORITMO GENÉTICO

Os algoritmos genéticos (AGs) vêm sendo estudados intensamente nos últimos anos, tendo se constituído uma boa ferramenta para tratar, de maneira sistemática, a otimização global (pelo menos aproximada) de funções genéricas. Desta forma, é possível tratar funções não diferenciáveis, descontínuas e/ou multimodais, ou seja, funções que não exibem nenhuma das propriedades usualmente necessárias para permitir a aplicabilidade de métodos convencionais de programação matemática. No caso de problemas de variáveis discretas, os AGs são particularmente importantes no caso de problemas que se enquadram na classe dos problemas $\mathcal{NP}$-Difíceis.

Através de um conjunto de soluções-candidatas consideradas simultaneamente, denominado de população, o algoritmo genético trabalha em busca de uma solução ótima. A cada solução-candidata se atribui o nome de indivíduo. Por fazer uso de populações e estas evoluírem em busca de uma solução, este tipo de algoritmo é conhecido na literatura como algoritmo evolucionário. Pode ser considerado um AG o algoritmo que emprega, dentre outras, pelo menos alguma versão de três operações básicas correspondentes aos elementos constituintes da evolução dos seres biológicos, definidas por:

Cruzamento: operação que combina a informação de dois indivíduos, gerando novos indivíduos.

Mutação: operação que "perturba" um indivíduo, gerando um novo indivíduo com alguma semelhança com o indivíduo que o originou.

Seleção: operação que gera uma nova população a partir da população corrente, com maior probabilidade de inserção, na nova população, dos indivíduos de melhor valor de função-objetivo e, consequentemente, menor probabilidade para os indivíduos com pior valor de função-objetivo.

Esses operadores são aplicados repetindo-se de forma sequencial/iterativa até a convergência para um ponto que atenda algum critério de parada. Algoritmos constituídos segundo esse esquema são genericamente denominados AGs. Algumas referências que

abordam o estudo desses operadores são (Belmont-Moreno., 2001; Choi & Oh, 2000; Hasancebi & Erbatur, 2000; Takahashi et al., 2003; Vasconcelos et al., 2001). A maioria dos AG's atualmente utilizados emprega, além desses operadores básicos, também outros operadores adicionais, com o intuito de aumentar a eficiência do algoritmo. Alguns desses operadores possuem aplicabilidade genérica, mantendo o grau de generalidade do AG básico, tais como:

Elitismo: causa a seleção determinística de parte da população corrente, usualmente os melhores indivíduos, para integrarem a nova população;

Nicho: evita a concentração excessiva de indivíduos explorando a mesma região de um espaço de busca.

Vários operadores não-básicos como esses, ou diversos outros, foram estudados em referências como (Takahashi et al., 2003; Vasconcelos et al., 2001; Fan et al., 2000; Potts et al., 1994; Sareni & Krahenbuhl, 1998).

4.1.1 Algoritmos Genéticos Multiobjetivo

Segundo Goldberg (1989a), um AG mono-objetivo pode ser adaptado para obtenção de um AG multiobjetivo. As adaptações necessárias são:

- Selecionar dentro de uma população inicial $\mathcal{Q}_0$, o grupo de indivíduos que forma a estimativa corrente do conjunto Pareto-ótimo; a esse grupo é dado o nome de $\bar{\mathcal{P}}_0$.

- A cada iteração, uma nova população $\mathcal{Q}_i$ é gerada pela aplicação dos operadores genéticos. Deve-se recalcular a estimativa do conjunto Pareto-ótimo $\bar{\mathcal{P}}_i$, eventualmente excluindo alguns indivíduos e incluindo outros. Todos os indivíduos dominados devem ser excluídos e, na eventualidade de se excluírem indivíduos não-dominados, deve-se escolher preservar os indivíduos de regiões menos densamente amostradas do conjunto $\bar{\mathcal{P}}_i$, eliminando-se indivíduos das regiões mais densamente amostradas.

- O conjunto $\bar{\mathcal{P}}_i$ é empregado como o "conjunto-elite" na operação de elitismo.

- O funcional que guia o operador de seleção deve ser composto com os funcionais individuais que compõem o vetor de objetivos.

Desta maneira, ao invés de procurar por apenas uma solução, como era feito no caso mono-objetivo, o AG multiobjetivo pode encontrar um conjunto de soluções Pareto-ótimas pertencentes a $\bar{\mathcal{P}}_i$. A estrutura básica da maioria dos AGs multiobjetivo é constituída dessa forma.

Em seguida, será discutido um dos algoritmos genéticos multiobjetivo mais conhecidos na literatura, o NSGA II (*Non-dominated Sorting Genetic Algorithm*), introduzido em (Deb et al., 2002).

4.1.1.1 NSGA-II Deb et al. (2002) apresentam um procedimento baseado em ordenação elitista por dominância (*Pareto ranking*). Este procedimento busca classificar as soluções de um conjunto M em diversas fronteiras $\mathcal{F}_1$, $\mathcal{F}_2$, ... $\mathcal{F}_k$, conforme o grau de dominância de cada solução. Desta forma, a fronteira $\mathcal{F}_1$ possui soluções que não são dominadas por nenhuma outra de M. Já a fronteira $\mathcal{F}_2$ possui soluções não dominadas por nenhuma solução do conjunto $M - \mathcal{F}_1$; da mesma forma $\mathcal{F}_3$ contém soluções não dominadas de $M - (\mathcal{F}_1 \cup \mathcal{F}_2)$ e assim sucessivamente. Este procedimento pode ser visto no Algoritmo 1, no qual:

- nd_i representa o número de soluções que dominam a solução i;
- U_i é o conjunto de soluções que são dominadas pela solução i.

Algoritmo 1 Pseudocódigo para Ordenação por dominância

```
1: função NdSorting(M)
2:     para cada solução i ∈ M faça
3:         nd_i = 0
4:         U_i = ⊘
5:         para cada solução j ≠ i e j ∈ M faça
6:             se i ≤ j então U_i = U_i ∪ j
7:             se j ≤ i então nd_i = nd_i + 1
8:         fim para
9:         se nd_i = 0 então F_1 = F_1 ∪ i
10:    fim para
11:    k = 1
12:    enquanto F_k ≠ ⊘ faça
13:        temp = ⊘
14:        para solução i ∈ F_k faça
15:            para solução j ∈ U_i faça
16:                nd_j = nd_j − 1
17:                se nd_j = 0 então temp = temp ∪ j
18:            fim para
19:        fim para
20:        k = k + 1
21:        F_k = temp
22:    fim enquanto
23:    retorno F_1, F_2, ... F_k
24: fim função
```

O NSGA-II trabalha com duas populações, P_t e Q_t, de mesmo tamanho N, onde t representa o índice da geração corrente do algoritmo (Figura 4.1). Na primeira geração, os indivíduos iniciais, contidos em P_t geram as soluções em Q_t aplicando os operadores de cruzamento, mutação e seleção por torneio. A seguir, é formada a população P_{t+1} também de tamanho N. P_{t+1} é preenchida a partir de uma seleção entre os indivíduos

contidos em $R_t = P_t \cup Q_t$. Utilizando o Algoritmo 1, esta seleção é feita de forma elitista, permanecendo na próxima população os N indivíduos contidos nas primeiras fronteiras.

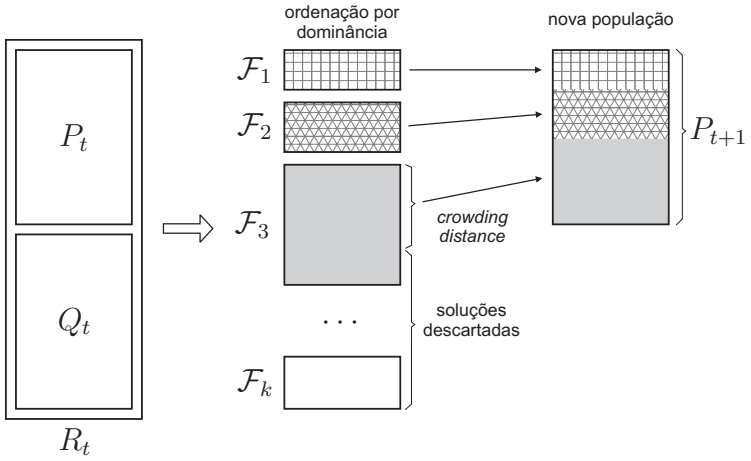


Figura 4.1 Exemplo do funcionamento do NSGA-II. (Deb, 2001)

Como visto na Figura 4.1, ao preencher as últimas vagas da nova população, é bem provável que os indivíduos presentes na última fronteira a ser transportada para a nova população ultrapassem o número N de posições na população total. Para produzir a escolha de quais indivíduos serão selecionados e quais serão descartados nessa última fronteira, é utilizado o índice de *crowding-distance*. A *crowding-distance* é uma estimativa de densidade de soluções que estão ao redor de cada indivíduo da população. Selecionando-se para passar para a próxima população as soluções cujas vizinhanças sejam menos densamente ocupadas por outras soluções, promove-se o aumento da diversidade das soluções na população. O pseudocódigo deste procedimento pode ser visto no Algoritmo 2, no qual:

- $crDist_i$ é o valor de *crowding-distance* da solução i;
- $numObj$ representa o número de objetivos do problema;
- $f_m(.)$ é o valor da função objetivo m.

4.2 CODIFICAÇÃO E DECODIFICAÇÃO

Nesta seção serão apresentados os detalhes da codificação e decodificação utilizados no algoritmo a ser apresentado para resolver o PDCC-RSSF.

4.2.1 Codificação

Para que a codificação de um indivíduo no algoritmo de otimização seja compatível com a formulação do problema estabelecida no Capítulo 3, cada nó sensor será identificado por uma "marca" presente no conjunto $\{1, \ldots, |\mathcal{S}|\}$. As variáveis de decisão, que representam as ações de ativação ou desativação dos nós sensores.

Algoritmo 2 Pseudocódigo para o cálculo da *crowding-distance*

```
1: função crowDist(M)
2:     para cada solução i ∈ M faça
3:         crDist_i = 0
4:     fim para
5:     para m de 1 até numObj faça
6:         Classificar M por f_m
7:         crDist_1 = crDist_|M| = ∞
8:         para n de 2 até |M| − 1 faça
9:             crDist_n = crDist_n + f_m(M_{n+1}) − f_m(M_{n−1})
10:        fim para
11:    fim para
12: fim função
```

No algoritmo proposto cada solução é codificada como uma permutação dos elementos de $\{1, \ldots, |\mathcal{S}|\}$. Esta permutação representa a sequência em que os nós sensores devem ser ativados durante a operação da rede. A Figura 4.2 mostra um exemplo desta codificação para uma instância de 8 nós sensores. Neste exemplo, os nós sensores são ativados na seguinte ordem: 7, 5, 3, 4, 2, 6, 8, 1. Isto significa que, neste *scheduling*, o nó sensor com a "etiqueta numérica" 7 será associado ao estado da variável x_1, e assim por diante.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
7	5	3	4	2	6	8	1

Figura 4.2 Exemplo da codificação de uma solução em um problema mono-objetivo.

4.2.2 Decodificação

Cada indivíduo é decodificado utilizando as sequências apresentadas no Algoritmo 3. Basicamente, o algoritmo tem o seguinte comportamento:

- H_{x_j} representa o estágio em que o sensor x_j é ativado.
- Os nós sensores presentes na codificação são ativados da esquerda para direita.
- Os nós sensores são ativados na sequência até que a cobertura C seja atingida.
- No surgimento de falhas, novos nós sensores são ativados seguindo a mesma lógica do passo anterior.
- O processo se encerra quando não é mais possível reestabelecer a cobertura desejada ativando o restante dos nós sensores.

Cada operação mostrada no Algoritmo 3 pode ser visualizada no exemplo mostrado nas Figuras 4.3 e 4.4. Neste exemplo, os nós sensores 7, 5 e 3 são ativados no início do

Algoritmo 3 Pseudocodigo para Decodificação do Indivíduo

função DECODEIND($\mathbf{x} = \{x_1, \ldots, x_{|\mathcal{S}|}\}, E_{max}, C$)
  faça cada sensor como desativado;
  $j \leftarrow 1$ ▷ sensor corrente
  $c_{stg} \leftarrow 0$; ▷ estágio corrente
  $E(0) = [E_{max}\ E_{max}\ \ldots\ E_{max}]'$;
  $\mathcal{A}_s \leftarrow \emptyset$; ▷ conjunto dos sensores ativos
  enquanto $j < |\mathcal{S}|$ **faça**
    enquanto $G_{cov}[\mathcal{A}_s] < C \cdot |\mathcal{D}|$ e $j < |\mathcal{S}|$ **faça**
      ative o sensor x_j e o **inclua** em $\mathcal{A}_s$;
      $H_{x_j} \leftarrow c_{stg} + 1$
      $j \leftarrow j + 1$;
    fim enquanto
    se $G_{cov}[\mathcal{A}_s] \geq C \cdot |\mathcal{D}|$ **então**
      simule a rede até que ocorra a primeira falha em $\mathcal{A}_s$;
      remova os nós sensores falhos de $\mathcal{A}_s$;
      $c_{stg} \leftarrow c_{stg} + 1$;
      atualize $E(c_{stg})$;
    fim se
  fim enquanto
  retorno H.
fim função

estágio 1, pois pelo exemplo eles são suficientes para garantir a fração C de cobertura desejada. Após um tempo t_1, o nó sensor 5 falha, reduzindo a cobertura para um nível indesejado. Para que a cobertura possa ser reestabelecida os nós sensores 4 e 2 são ativados, configurando o início do estágio 2, que tem duração até a falha do nó sensor 3. Ao falhar o nó sensor 3 após um tempo de $t_1 + t_2$, é iniciado o estágio 3. No início deste estágio, foi necessário apenas a ativação do nó sensor 6 para que a cobertura fosse reestabelecida. O processo é repetido até que não existam nós sensores disponíveis para que seja possível restaurar a fração C da cobertura. Isto é caracterizado na Figura 4.3 ao falhar o nó sensor 7, após o tempo $t_1 + t_2 + t_3$.

Vale a pena enfatizar que com o mecanismo proposto de codificação/decodificação, torna-se possível modelar a natureza dinâmica do problema. Com base nos pressupostos demonstrados no Capítulo 3, é possível perceber que a solução aqui apresentada sempre respeita as restrições de cobertura (g_1 a g_3) e conectividade (g_4 a g_7). Isto é assegurado pelo processo de decodificação, pois este sempre constrói um estágio de maneira a garantir a conectividade e a cobertura no nível C requerido.

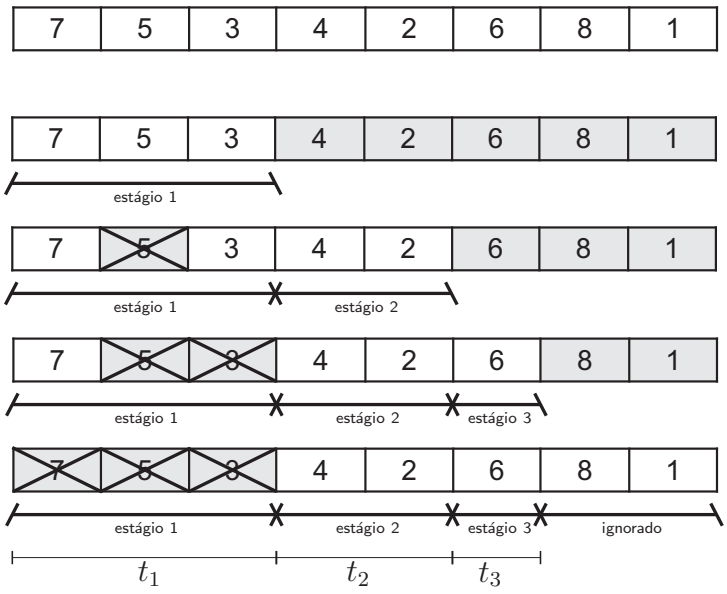


Figura 4.3 Exemplo de decodificação da solução.

4.2.3 Codificação e Decodificação Multiobjetivo

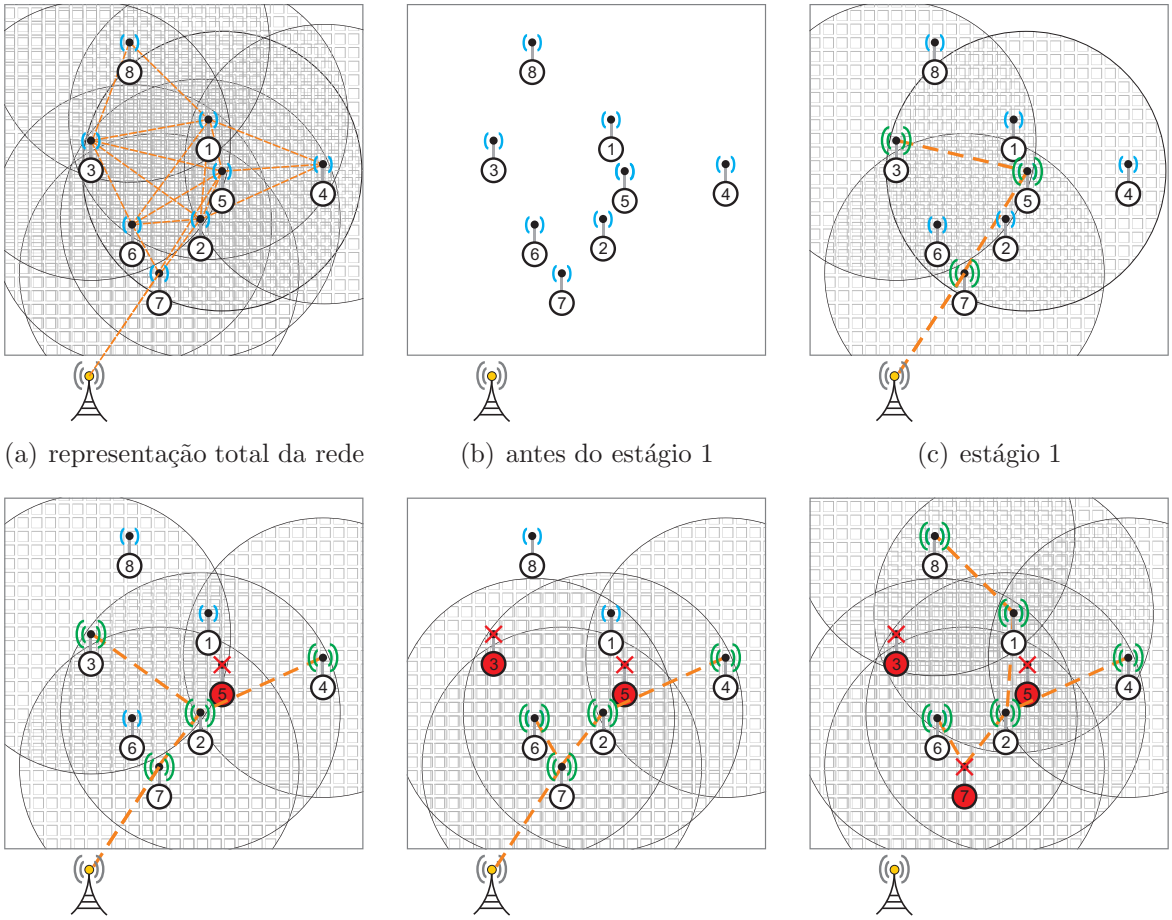
A codificação proposta na subseção anterior foi elaborada para ser utilizada em um algoritmo mono-objetivo. Para ser usada em um algoritmo multiobjetivo, fazem-se necessárias adaptações. Como visto na modelagem do problema multiobjetivo a restrição de cobertura foi relaxada e um novo objetivo foi criado. Como consequência, é incluída na codificação uma variável real $C_{min} \leq C_i \leq 1.00$ na representação de cada indivíduo. Dado um indivíduo i, a variável C_i indica a cobertura mínima a ser considerada na decodificação de tal solução. Esta variável adicional substitui o parâmetro C (fração desejada de cobertura) que aparece na versão mono-objetivo. Uma vez que cada solução-candidata tem o seu próprio parâmetro C_i, é possível avaliar diferentes soluções de cobertura simultaneamente na população do algoritmo.

Um exemplo da proposta de codificação multiobjetivo pode ser visto na Figura 4.6. Neste exemplo, a sequência de ativação dos nós é a mesma da Figura 4.2, e a cobertura mínima requerida para a solução é de 75%.

O processo de decodificação utilizado pelo algoritmo mono-objetivo pode ser utilizado diretamente na versão multiobjetivo. Para isto, basta substituir o parâmetro C pela variável C_i antes de decodificar cada solução.

4.3 ESTRUTURA GEOMÉTRICA PARA O PDCC-RSSF

Uma estrutura geométrica é aqui definida para o Problema Dinâmico de Cobertura e Conectividade em Redes de Sensores sem Fio (PDCC-RSSF). O elemento constitutivo dessa geometria é a operação de ***edit move***, que consiste de uma simples troca entre duas coordenadas. Duas soluções são **adjacentes** se é possível mover de uma para outra através de um *edit move*. A **vizinhança** de uma solução $\mathbf{x}^a$ é representada pelo



(a) representação total da rede (b) antes do estágio 1 (c) estágio 1

(d) estágio 2 (e) estágio 3 (f) após o estágio 3

Figura 4.4 RSSF segundo a decodificação mostrada na Figura 4.3. A área hachurada representa o raio de comunicação, demais representações são mostradas na Figura 4.5

(a) (b) (c) (d) (e)

Figura 4.5 Legenda da RSSF. (a) sorvedouro, (b) sensor Y desativado, (c) sensor Y ativado, (d) sensor Y falho e (e) link de comunicação.

conjunto de soluções que são adjacentes a $\mathbf{x}^a$. A **distância** entre duas soluções $\mathbf{x}^a$ e $\mathbf{x}^b$ é representada pelo número de *edit moves* que são necessários para transformar $\mathbf{x}^a$ em $\mathbf{x}^b$. Um **caminho** entre duas soluções $\mathbf{x}^a$ e $\mathbf{x}^b$ engloba as soluções que são encontradas ao realizar os *edit moves* durante a transformação de $\mathbf{x}^a$ em $\mathbf{x}^b$. Se este caminho tem o número mínimo de movimentos necessários para a transformação, então seu comprimento corresponde à **distância** entre $\mathbf{x}^a$ e $\mathbf{x}^b$. Na referência Moraglio (2007) é mostrado que estas definições são adequadas para a definição de operadores geométricos, no caso de problemas de permutação.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
7	5	3	4	2	6	8	1
C_i							
0.75							

Figura 4.6 Exemplo de codificação para o algoritmo multiobjetivo.

Seja $n = |\mathcal{S}|$ o número de nós sensores. As seguintes entidades geométricas são formalmente definidas:

Definição 8. Vetor de variáveis de decisão: *Seja* $\mathbf{x}$ *um vetor n-dimensional:*

$$\mathbf{x} = \left[\begin{array}{cccc} x_1 & x_2 & \ldots & x_n \end{array} \right]$$

que pertence ao conjunto $\mathcal{X}$ *definido por:*

$$\mathcal{X} = \{\mathbf{x} \mid x_i \in \{1, 2, \ldots, n\}\,,\ e\ x_i \neq x_j \forall i \neq j\}$$

O vetor $\mathbf{x}$ *é chamado de* vetor de variáveis de decisão *do PDCC-RSSF, e o conjunto* $\mathcal{X}$ *é chamado de* variedade das variáveis de decisão *do PDCC-RSSF.*

□

Deve-se notar que o conjunto $\mathcal{X}$, tal como definido, corresponde ao conjunto de vetores n-dimensionais, cujas componentes são permutações do conjunto dos n primeiros números inteiros. Cada coordenada x_i de um vetor de variáveis de decisão $\mathbf{x} \in \mathcal{X}$ armazena um número inteiro no intervalo $\{1, 2, \ldots, n\}$ que representa o rótulo de um nó sensor do PDCC-RSSF. A sequência de coordenadas, de x_1 a x_n, representa a sequência de ativação dos nós do PDCC-RSSF. Desta forma, o nó cujo rótulo é armazenado em x_1 é o primeiro a ser ativado, e assim por diante, até que o nó cujo rótulo armazenado é x_n seja o último a ser ativado.

No contexto da geometria em variáveis contínuas, uma superfície em um espaço vetorial é uma variedade. Em particular, uma variedade pode não ser um conjunto que satisfaça as propriedades de um espaço vetorial, embora constitua um subconjunto de um espaço vetorial. Em alguns casos, uma variedade pode localmente apresentar propriedades de um espaço vetorial que não são válidas em todo o espaço. Nesta seção, será mostrado que, embora o conjunto $\mathcal{X}$ não satisfaça as propriedades de um espaço vetorial, este conjunto apresenta algumas propriedades locais que se assemelham aos de um espaço vetorial.

Definição 9. ***Edit move:*** *Seja* $\mathbf{x}^a$ *e* $\mathbf{x}^b \in \mathcal{X}$*. Um edit move* $V(\cdot, \cdot, \cdot)$ *é definido por:*

$$\mathbf{x}^b = V(\mathbf{x}^a, i, j) \ \Rightarrow \ \left\{ \begin{array}{l} x_i^b = x_j^a \\ x_j^b = x_i^a \\ x_k^b = x_k^a \ \forall\ k \neq i, j \end{array} \right.$$

□

Um *edit move* é uma operação elementar que é $\mathcal{X}$-invariante, o que significa que, quando esta operação é aplicada em um elemento de $\mathcal{X}$, resulta em mais um elemento de $\mathcal{X}$. O *edit move* induz uma *vizinhança*, que é o conjunto de pontos de $\mathcal{X}$ que são acessíveis a partir de um determinado ponto em $\mathcal{X}$ através de um simples *edit move*, como definido a seguir.

Definição 10. Vizinhança: *Seja* $\mathbf{x}^a \in \mathcal{X}$. *A* vizinhança *de* $\mathbf{x}^a$ *é o conjunto* $\mathcal{N}(\mathbf{x}^a)$ *definido por:*

$$\mathcal{N}(\mathbf{x}^a) = \{\mathbf{x} \in \mathcal{X} \mid \mathbf{x} = V(\mathbf{x}^a, i, j) \ \text{ para algum } i, j \in \{1, 2, \ldots, n\}\}$$

□

Qualquer ponto $\mathbf{x}^a$ pertencente a $\mathcal{X}$ pode ser transformado em qualquer outro ponto $\mathbf{x}^b$ de $\mathcal{X}$, através de uma sequência de *edit moves*. A sequência de pontos gerados por estes *edit moves*, começando em $\mathbf{x}^a$ e terminando em $\mathbf{x}^b$, é chamado de *caminho* entre $\mathbf{x}^a$ e $\mathbf{x}^b$, como consta na definição a seguir.

Definição 11. Caminho: *Seja* $\mathbf{x}^a$ *e* $\mathbf{x}^b \in \mathcal{X}$. *Qualquer sequência de pontos* $\mathbf{P} = \{\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^m\}$ *de tal forma que*

$$\begin{cases} \mathbf{p}^{i+1} \in \mathcal{N}(\mathbf{p}^i) \\ \mathbf{p}^1 = \mathbf{x}^a \\ \mathbf{p}^m = \mathbf{x}^b \end{cases}$$

é um caminho *de* $\mathbf{x}^a$ *a* $\mathbf{x}^b$. *O conjunto de todos os caminhos de* $\mathbf{x}^a$ *a* $\mathbf{x}^b$ *é denotado por* $\mathcal{P}(\mathbf{x}^a, \mathbf{x}^b)$.

□

Definição 12. Comprimento do caminho: *Seja um caminho denotado pela sequência de pontos* $\mathbf{P} = \{\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^m\}$. *O* comprimento *de* $\mathbf{P}$, *denotado por* $\ell(\mathbf{P})$, *é dado por:*

$$\ell(\mathbf{P}) = m - 1$$

□

O comprimento de um caminho corresponde ao número de *edit moves* que são executados na geração da sequência dos pontos pertencentes ao caminho. Um *caminho mínimo* entre $\mathbf{x}^a$ e $\mathbf{x}^b$ é o caminho com o comprimento mínimo no conjunto $\mathcal{P}(\mathbf{x}^a, \mathbf{x}^b)$:

Definição 13. Caminho mínimo: *Seja* $\mathbf{x}^a$ *e* $\mathbf{x}^b \in \mathcal{X}$. *Um* caminho mínimo $\mathbf{P}^*$ *de* $\mathbf{x}^a$ *a* $\mathbf{x}^b$ *é definido como*

$$\mathbf{P}^* : \begin{cases} \mathbf{P}^* \in \mathcal{P}(\mathbf{x}^a, \mathbf{x}^b) \\ \ell(\mathbf{P}^*) \leq \ell(\mathbf{P}) \ \forall \ \mathbf{P} \in \mathcal{P}(\mathbf{x}^a, \mathbf{x}^b) \end{cases}$$

O conjunto de todos os caminhos mínimos de $\mathbf{x}^a$ *a* $\mathbf{x}^b$ *é denotado por* $\mathcal{P}^*(\mathbf{x}^a, \mathbf{x}^b)$.

□

O conceito de *distância* entre $\mathbf{x}^a$ e $\mathbf{x}^b$ é enunciado como o comprimento do caminho mínimo entre $\mathbf{x}^a$ e $\mathbf{x}^b$:

Definição 14. Distância: *Seja* $\mathbf{x}^a$ *e* $\mathbf{x}^b \in \mathcal{X}$. *A* distância *entre* $\mathbf{x}^a$ *e* $\mathbf{x}^b$, *denotado por* $d(\mathbf{x}^a, \mathbf{x}^b)$, *é definido por:*

$$d(\mathbf{x}^a, \mathbf{x}^b) = \ell(\mathbf{P}^*) \quad \textit{para qualquer} \quad \mathbf{P}^* \in \mathcal{P}^*(\mathbf{x}^a, \mathbf{x}^b)$$

□

Vale a pena ressaltar que a entidade $\mathcal{X}$ não possui todas as propriedades que seriam necessárias para constituir um espaço vetorial. Em particular, esta entidade não tem nenhum ponto adequado para se tornar a origem de um espaço (outra forma de expressar a mesma ideia é dizer que não há uma forma pertinente de definir a operação de multiplicação de um vetor por um escalar). Entretanto, se um vetor fosse definido como um movimento, ao invés da definição adotada aqui, o espaço vetorial ficaria propriamente definido. Devido a isto, $\mathcal{X}$ é chamado aqui de *variedade*, indicando que as operações no espaço vetorial devem ser transladadas a fim de formar esta entidade.

Existe uma estrutura peculiar do PDCC-RSSF que o torna bem diferente do problema de permutação usual. Com base na modelagem mostrada no Capítulo 3, e no esquema de codificação discutido anteriormente neste capítulo, é possível notar que a estrutura do PDCC-RSSF compartilha algumas características com dois problemas combinatoriais clássicos:

Problema de Escalonamento: Neste problema, é necessário encontrar uma ordenação ótima para um conjunto de entidades (usualmente tarefas) com intuito de otimizar uma função objetivo (normalmente o tempo requerido para executar tais tarefas).

Problema de Partição de Conjuntos: Para resolver este problema, o otimizador deve procurar a melhor maneira de dividir um conjunto de entidades (geralmente objetos) em um número diferente de conjuntos. Frequentemente, o objetivo neste tipo de problema é criar grupos com objetos semelhantes.

Fazendo um paralelo com o Problema de Escalonamento e o Problema de Partição de Conjuntos, o PDCC-RSSF pode ser visto como um problema de criar grupos com tarefas de ativação (cada grupo representa um conjunto de nós sensores que são ativados no começo de um estágio), e encontrar o melhor *scheduling* para tais grupos (sequência de estágios), buscando sempre maximizar o tempo de vida da rede. Obviamente estes subproblemas interagem.

É importante notar que na codificação descrita anteriormente, ações de permutação dentro de um mesmo grupo podem não causar alterações no fenótipo da solução se após a alteração o grupo permanecer inalterado. Isto ocorre em virtude de todas as ações de um grupo serem executadas simultaneamente no início do estágio. Esta característica pode ser visualizada na Figura 4.7

A Figura 4.7 mostra dois diferentes indivíduos, $\mathbf{x}^a$ e $\mathbf{x}^b$. Estes indivíduos são considerados adjacentes, pois $\mathbf{x}^b = V(\mathbf{x}^a, 4, 5)$. Como $i = 4$ e $j = 5$ pertencem ao mesmo estágio

(grupo), os fenótipos de $\mathbf{x}^a$ e $\mathbf{x}^b$ são os mesmos. De fato, nestas soluções os nós presentes tanto em $\mathbf{x}^b$ como em $\mathbf{x}^a$ são ativados no mesmo instante de tempo. Essa situação de invariância do fenótipo tende a ocorrer na maioria das permutações entre coordenadas do mesmo grupo.

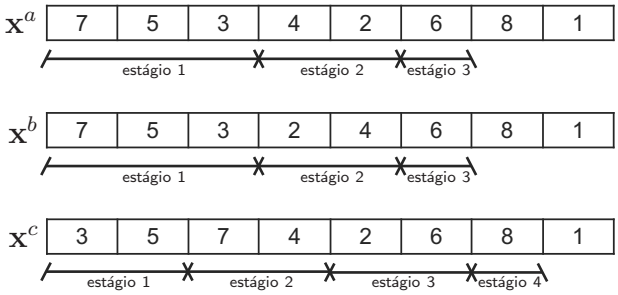


Figura 4.7 $\mathbf{x}^a$ e $\mathbf{x}^b$, apesarem de serem indivíduos diferentes, compartilham o mesmo fenótipo, pois definem a mesma sequência de *scheduling*. $\mathbf{x}^a$ e $\mathbf{x}^c$ não compartilham o mesmo fenótipo, pois definem diferentes sequências de *scheduling* dos nós sensores.

Isso significa que deve existir uma espécie de não-homogeneidade da operação *edit move* nas direções do espaço, o que irá causar impacto nas estratégias de busca. As definições que serão apresentadas a seguir têm o intuito de identificar a estrutura por trás desta não-homogeneidade.

Definição 15. Vetor de tempo de escalonamento: *Seja* $\mathbf{x}$ *pertencente à variedade da variável de decisão* $\mathcal{X}$. *O* vetor de tempo de escalonamento *de* $\mathbf{x}$, *denotado por* $\mathbf{T}(\mathbf{x})$, *é dado por:*

$$\mathbf{T}(\mathbf{x}) = \left[\begin{array}{cccccc} \tau_0 & \tau_1 & \tau_2 & \ldots & \tau_n & \tau_{n+1} \end{array} \right]$$

onde $\tau_i \leq \tau_{i+1}$, *e cada componente* τ_i *de* $\mathbf{T}(\mathbf{x})$ *corresponde ao instante de tempo em que o nó sensor* x_i *(indicado pela* i*-ésima coordenada da variável de decisão* $\mathbf{x}$*) é ativado, exceto nos casos* $\tau_0 = -\infty$ *e* $\tau_{n+1} = \infty$, *que por convenção, não têm nós correspondentes. Também por convenção, se existe um nó sensor* x_i *que não deve ser ativado, deve-se ter* $\tau_i = \infty$.

□

O vetor de tempo de escalonamento é utilizado para identificar os conjuntos de nós que são ativados ao mesmo tempo. Esses conjuntos são chamados de *grupos*:

Definição 16. Grupo: *Seja* $\mathbf{x} \in \mathcal{X}$, *com um vetor de tempo de escalonamento* $\mathbf{T}(\mathbf{x})$. *Os grupos* $\mathcal{G}_k(\mathbf{x})$ *são definidos como:*

$$\mathcal{G}_k(\mathbf{x}) = \{x_i, x_{i+1}, \ldots, x_{j-1}, x_j\}$$
$$\text{em que} \begin{cases} \tau_{i-1} < \tau_i \\ \tau_i = \tau_{i+1} = \ldots = \tau_j \\ \tau_{j+1} > \tau_j \end{cases}$$

□

Deve-se notar que $\mathbf{x} = \{\mathcal{G}_1(\mathbf{x}), \mathcal{G}_2(\mathbf{x}), \ldots, \mathcal{G}_s(\mathbf{x})\}$, para uma operação de rede com $s-1$ estágios, em que o grupo $\mathcal{G}_s$ representa o conjunto de nós que não são ativados (este conjunto pode ser vazio).

Deve-se notar, entretanto, que às vezes após uma permutação entre ações de um mesmo grupo, podem ocorrer mudanças nas fronteiras entre os grupos, como exemplificado na Figura 4.7 com a troca de $\mathbf{x}^a$ para $\mathbf{x}^c$, fazendo $\mathbf{x}^c = V(\mathbf{x}^a, 1, 3)$. Neste caso, apesar das coordenadas x_1 e x_3 pertencerem ao mesmo grupo na solução $\mathbf{x}^a$, os nós sensores 3 e 5 foram suficientes para prover a cobertura desejada. Como consequência, a ativação do nó sensor 7 pode ser prorrogada, sendo a coordenada x_3 deslocada para o próximo grupo na solução $\mathbf{x}^c$. Esse raciocínio conduz à conclusão de que a maior parte das variações de função objetivo ocorrerão em virtude de movimentos executados envolvendo trocas entre coordenadas de diferentes grupos. Apenas em algumas poucas situações os movimentos de trocas que ocorrerem entre coordenadas de um mesmo grupo causarão mudanças na função objetivo (situação ocorrida na solução $\mathbf{x}^c$ apresentada na Figura 4.7).

Somente nos casos em que o conjunto de nós sensores para serem ativados em um estágio não for um conjunto mínimo para garantir a cobertura e a conectividade, um *edit move* envolvendo permutações de elementos pertencentes a um mesmo grupo poderá levar a soluções com fenótipos diferentes. A redundância associada a tais conjuntos não mínimos pode ser removida por um procedimento simples. A fim de fazer isto, definem-se os *grupos mínimos*:

Definição 17. Grupo mínimo: *Seja* $\mathbf{x} \in \mathcal{X}$. *O grupo* $\mathcal{G}_k(\mathbf{x})$ *é um* grupo mínimo *se qualquer subconjunto próprio deste grupo não for suficiente para satisfazer as restrições de cobertura e conectividade do PDCC-RSSF.*

□

A operação $\bar{\mathcal{G}}_k(\mathbf{x}) = \text{mingr}(\mathcal{G}_k(\mathbf{x}))$ denota a extração do *grupo mínimo* de $\mathcal{G}_k(\mathbf{x})$. Deve ser notado que esta operação é de fácil implementação computacional utilizando procedimentos determinísticos[1]. O pressuposto a seguir será utilizado para simplificar a tarefa de construção do algoritmo.

Pressuposto 2. Aplicação da minimalidade: *A partir de agora, será assumido que, dada uma solução* $\mathbf{x}$*, a operação* $\bar{\mathcal{G}}_k(\mathbf{x}) = mingr(\mathcal{G}_k(\mathbf{x}))$ *será realizada em todos os grupos, sequencialmente, da esquerda para a direita. Dentro de cada grupo, os elementos serão examinados da direita para esquerda, e os elementos não pertencentes aos* grupos mínimos *(Definição 17) serão movidos para o início (posição mais à esquerda) do grupo seguinte,* $\mathcal{G}_{k+1}(\mathbf{x})$.

□

Embora este procedimento de aplicação da minimalidade possa ser realizado por diferentes sequências de operações, este formato em específico foi escolhido porque ele faz

[1]Observe que a etapa principal de tais procedimentos deva ser a verificação da viabilidade de cada subconjunto do grupo $\mathcal{G}_k(\mathbf{x})$, que é composto por todos os elementos do grupo exceto um elemento. Esta operação é executada uma vez para cada elemento do grupo. Um exemplo de tal implementação é mostrado no Algoritmo 8 (Operador de Desativação Inteligente), descrito na Seção 4.4.4

com que a mudança no genoma seja menos prejudicial do que as outras alternativas. O conceito de *fenótipo equivalente* está relacionado com o pressuposto da aplicação da minimalidade:

Definição 18. Fenótipo equivalente: *Um vetor de variáveis de decisão* $\mathbf{x} \in \mathcal{X}$ *é* fenótipo equivalente *de outro vetor* $\bar{\mathbf{x}} \in \mathcal{X}$ *se ambos os vetores, quando decodificados, levam a ativação dos mesmos nós sensores no mesmo instante de tempo.*

□

Após o pressuposto de minimalidade de grupo (Pressuposto 2), todos os *edit moves* que alteram elementos de um mesmo grupo garantidamente produzem somente soluções *fenótipo equivalente*. Este fato é enunciado formalmente na Proposição 3, mostrado mais a frente nesta mesma seção.

Algumas definições adicionais, relacionadas com as noções de *vizinhança relativa* e *conjunto span* serão úteis para abordar as características estruturais do problema.

Definição 19. Vizinhança relativa: *Considere um conjunto de* q *pares não ordenados*[2] $\mathcal{Q} = \{(i_1, j_1), (i_2, j_2), \ldots, (i_q, j_q)\}$*, e um ponto* $\mathbf{x} \in \mathcal{X}$*. Um par não ordenado* $(i_k, j_k) \in \mathcal{Q}$ *refere-se a um* edit move *que é aplicado em* $\mathbf{x}$*. Considere também a seguinte sequência de conjuntos:*

$$\begin{aligned}
\mathcal{L}^1(\mathbf{x}, \mathcal{Q}) &= \{\mathbf{x}\} \cup \{\mathbf{z} = V(\mathbf{x}, i, j) \ \forall \ (i, j) \in \mathcal{Q}\} \\
\mathcal{L}^2(\mathbf{x}, \mathcal{Q}) &= \mathcal{L}^1 \cup \{\mathbf{z} = V(\bar{\mathbf{x}}, i, j) \ \forall \ \bar{\mathbf{x}} \in \mathcal{L}^1(\mathbf{x}, \mathcal{Q}), (i, j) \in \mathcal{Q}\} \\
&\vdots \\
\mathcal{L}^\eta(\mathbf{x}, \mathcal{Q}) &= \mathcal{L}^{\eta-1} \cup \{\mathbf{z} = V(\bar{\mathbf{x}}, i, j) \ \forall \ \bar{\mathbf{x}} \in \mathcal{L}^{\eta-1}(\mathbf{x}, \mathcal{Q}), (i, j) \in \mathcal{Q}\} \\
&\vdots
\end{aligned}$$

O conjunto $\mathcal{L}^1(\mathbf{x}, \mathcal{Q})$ *é uma* vizinhança relativa *de* $\mathbf{x}$*, em relação ao conjunto* $\mathcal{Q}$*. O conjunto* $\mathcal{L}^\eta(\mathbf{x}, \mathcal{Q})$ *é uma* vizinhança relativa de grau η *de* $\mathbf{x}$*, em relação ao conjunto* $\mathcal{Q}$*.*

□

A Tabela 4.1 apresenta um exemplo da construção dos conjuntos $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^\perp)$ e $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^\oslash)$, a partir de um ponto inicial $\mathbf{x}$ utilizando diferentes conjuntos de *edit moves* denotados por $\mathcal{Q}^\perp$ e $\mathcal{Q}^\oslash$.

Observe que a sequência $[\mathcal{L}^k]$ é tal que $\mathcal{L}^i \supseteq \mathcal{L}^{i-1}$. O conjunto $\mathcal{L}^1(\mathbf{x}, \mathcal{Q})$ contém $\mathbf{x}$ e também todos os pontos em $\mathcal{X}$ que são alcançados por um *edit move* definido por um par não ordenado em $\mathcal{Q}$, a partir de qualquer ponto em $\mathcal{L}$. O conjunto $\mathcal{L}^2(\mathbf{x}, \mathcal{Q})$ contém todos os elementos do conjunto $\mathcal{L}^1(\mathbf{x}, \mathcal{Q})$ e também todos os pontos em $\mathcal{X}$ que são alcançados por um *edit move* definido por um par não ordenado em $\mathcal{Q}$, a partir de qualquer ponto em $\mathcal{L}^1$, e assim por diante. Considerando a cardinalidade finita da variedade da variável de decisão $\mathcal{X}$, a sequência $[\mathcal{L}^k]$ dos conjuntos definidos anteriormente, deve ter um conjunto maximal, que é chamado de *conjunto span*.

[2]Um par não ordenado (i, j) significa que $(i, j) = (j, i)$. Deve ser notado que $V(\mathbf{x}, i, j) = V(\mathbf{x}, j, i)$.

Tabela 4.1 Conjuntos $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\perp})$ e $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\oslash})$, obtidos a partir de um ponto inicial $\mathbf{x}$ utilizando os conjuntos de *edit moves* denotado por $\mathcal{Q}^{\perp}$ e $\mathcal{Q}^{\oslash}$. O vetor de variáveis de decisão $\mathbf{x} = \{x_1, x_2, x_3, x_4, x_5, x_6\}$ e composto por dois grupos mínimos, $\{x_1, x_2, x_3\} = \{a_1, a_2, a_3\}$ e $\{x_4, x_5, x_6\} = \{a_4, a_5, a_6\}$.

			x_1	x_2	x_3	x_4	x_5	x_6
		$\mathbf{x}$	a_1	a_2	a_3	a_4	a_5	a_6
$\mathcal{Q}^{\perp}$		$\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\perp})$	a_1	a_2	a_3	a_4	a_5	a_6
	$(1,4)$		a_4	a_2	a_3	a_1	a_5	a_6
	$(1,5)$		a_5	a_2	a_3	a_4	a_1	a_6
	$(1,6)$		a_6	a_2	a_3	a_4	a_5	a_1
	$(2,4)$		a_1	a_4	a_3	a_2	a_5	a_6
	$(2,5)$		a_1	a_5	a_3	a_4	a_2	a_6
	$(2,6)$		a_1	a_6	a_3	a_4	a_5	a_2
	$(3,4)$		a_1	a_2	a_4	a_3	a_5	a_6
	$(3,5)$		a_1	a_2	a_5	a_4	a_3	a_6
	$(3,6)$		a_1	a_2	a_6	a_4	a_5	a_3
$\mathcal{Q}^{\oslash}$		$\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\oslash})$	a_1	a_2	a_3	a_4	a_5	a_6
	$(1,2)$		a_2	a_1	a_3	a_4	a_5	a_6
	$(1,3)$		a_3	a_2	a_1	a_4	a_5	a_6
	$(2,3)$		a_1	a_3	a_2	a_4	a_5	a_6
	$(4,5)$		a_1	a_2	a_3	a_5	a_4	a_6
	$(4,6)$		a_1	a_2	a_3	a_6	a_5	a_4
	$(5,6)$		a_1	a_2	a_3	a_4	a_6	a_5

Definição 20. Conjunto *span*: *O* conjunto span $\mathcal{L}^*$ *de* $(\mathbf{x}, \mathcal{Q})$ *é o conjunto maximal da sequência* $[\mathcal{L}^k]$*:*

$$\mathcal{L}^*(\mathbf{x}, \mathcal{Q}) = \mathcal{L}^k(\mathbf{x}, \mathcal{Q}) \quad tal\ que\ \ \mathcal{L}^k(\mathbf{x}, \mathcal{Q}) = \mathcal{L}^{k-1}(\mathbf{x}, \mathcal{Q})$$

□

Agora, os pares não ordenados que definem o *edit move* são classificados de acordo com as posições relativas das coordenadas que são trocadas no movimento, seja dentro de um mesmo grupo ou envolvendo grupos diferentes:

Definição 21. Conjunto de troca: *Seja* $\mathbf{x} = \{x_1, \ldots, x_n\} \in \mathcal{X}$*, e seja* $i, j \in \{1, \ldots, n\}$*. Seja também* $\mathcal{G}_p(\mathbf{x})$ *e* $\mathcal{G}_q(\mathbf{x})$ *denotando dois grupos de* $\mathbf{x}$*. Os seguintes conjuntos de pares não ordenados são definidos:*

$$\mathcal{Q}^{\oslash} = \{(i,j) \mid x_i \in \mathcal{G}_p(\mathbf{x})\ ,\ x_j \in \mathcal{G}_q(\mathbf{x})\ ,\ p = q\ ,\ i \neq j\}$$
$$\mathcal{Q}^{\perp} = \{(i,j) \mid x_i \in \mathcal{G}_p(\mathbf{x})\ ,\ x_j \in \mathcal{G}_q(\mathbf{x})\ ,\ p \neq q\}$$
$$\mathcal{Q}^{*} = \{(i,j) \mid i,j \in \{1,2,\ldots,n\}\ ,\ i \neq j\}$$

em que o conjunto $\mathcal{Q}^{\oslash}$ *é o* conjunto de troca intragrupo*, o conjunto* $\mathcal{Q}^{\perp}$ *é o* conjunto de troca intergrupo *e o conjunto* $\mathcal{Q}^*$ *é o* conjunto de troca completo*.*

□

O conjunto $\mathcal{Q}^{\oslash}$ inclui todos os pares não ordenados que definem os *edit moves* que correspondem a permutações dos nós dentro de um mesmo grupo. O conjunto $\mathcal{Q}^{\perp}$, por outro lado, inclui todos os pares não ordenados que definem os *edit moves* que correspondem às permutações dos nós pertencentes a grupos diferentes. Finalmente, o conjunto $\mathcal{Q}^*$ inclui os pares não ordenados que definem todos os possíveis *edit moves*. O exemplo da Tabela 4.1 foi construído considerando os conjuntos $\mathcal{Q}^{\oslash}$ e $\mathcal{Q}^{\perp}$. Alguns simples dados sobre esses conjuntos são apresentados na Proposição 1

Proposição 1. *As afirmativas a seguir são verdadeiras:*

(i) $\mathcal{Q}^* = \mathcal{Q}^{\perp} \cup \mathcal{Q}^{\oslash}$*;*

(ii) $\mathcal{Q}^{\oslash} \cap \mathcal{Q}^{\perp} = \emptyset$*.*

> ***Prova:*** Essa proposição vem diretamente da complementaridade dos conjuntos $\mathcal{Q}^{\oslash}$ e $\mathcal{Q}^{\perp}$.

♢

As principais propriedades geométricas do PDCC-RSSF são demonstradas na Proposição 2, que relacionam conjuntos de troca com vizinhanças relativas.

Proposição 2. *Seja* $\mathbf{x} \in \mathcal{X}$ *qualquer instância do vetor de variáveis de decisão de um PDCC-RSSF. Então:*

(i) $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\perp}) \cap \mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\oslash}) = \{\mathbf{x}\}$

(ii) $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\perp}) \cup \mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\oslash}) = \mathcal{N}(\mathbf{x})$

(iii) $\mathcal{L}^*(\mathbf{x}, \mathcal{Q}^{\perp}) = \mathcal{L}^*(\mathbf{x}, \mathcal{Q}^*) = \mathcal{X}$

Prova:

(i) O conjunto $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\perp}) \cap \mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\oslash})$ deve incluir $\{\mathbf{x}\}$ por construção. O único elemento de $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\perp}) \cap \mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\oslash})$ deve ser $\mathbf{x}$ porque:

- $V(\mathbf{x}, i, j) \neq V(\mathbf{x}, k, w)\ \forall\ (i, j) \neq (k, w)$, para (i, j) e (k, w) dois pares não ordenados; e
- $\mathcal{Q}^{\perp}$ e $\mathcal{Q}^{\oslash}$ são conjuntos disjuntos;

o qual estabelece o ponto (i).

(ii) A fim de estabelecer este ponto, deve-se notar que $\mathcal{N}(\mathbf{x}) = \mathcal{L}^1(\mathbf{x}, \mathcal{Q}^*)$ e $\mathcal{Q}^* = \mathcal{Q}^{\perp} \cup \mathcal{Q}^{\oslash}$.

(iii) A afirmativa $\mathcal{L}^*(\mathbf{x}, \mathcal{Q}^*) = \mathcal{X}$ vem diretamente do:

- fato já conhecido de que a composição do *edit moves* do tipo $V(\mathbf{x}, i, j)$, considerando-se todos os pares (i, j), permite gerar qualquer ponto do conjunto de variáveis de decisão $\mathcal{X}$, a partir de qualquer ponto inicial deste conjunto; e
- o fato de que $\mathcal{Q}^* = \mathcal{Q}^{\perp} \cup \mathcal{Q}^{\oslash}$.

Considere $(i, j) \in \mathcal{Q}^{\oslash}$, $(i, k) \in \mathcal{Q}^{\perp}$ e $(j, k) \in \mathcal{Q}^{\perp}$. Observe que

$$V(\mathbf{x}, i, j) = V(V(V(\mathbf{x}, i, k), j, k), i, k)$$

é verdade para qualquer i, j, k. Portanto, a afirmativa $\mathcal{L}^*(\mathbf{x}, \mathcal{Q}^{\perp}) = \mathcal{L}^*(\mathbf{x}, \mathcal{Q}^*)$ é verdadeira, pois um *edit move* arbitrário que realiza a permutação de $(i, j) \in \mathcal{Q}^{\oslash}$ pode ser sintetizado por meio de três *edit moves* realizando as seguintes trocas $\{(i, k), (j, k), (i, k)\} \subset \mathcal{Q}^{\perp}$.

$\diamondsuit$

A Proposição 2, nos itens (i) e (ii), expressa uma espécie de complementaridade entre as pesquisas utilizando operadores de *edit move* $V(\mathbf{x}, i, j)$ com $(i, j) \in \mathcal{Q}^{\oslash}$ e com $(i, j) \in \mathcal{Q}^{\perp}$. Isto é análogo à soma direta de *subspaços* em espaços vetoriais usuais, na medida em que esses movimentos exploram regiões complementares da variedade das variáveis de decisão $\mathcal{X}$. Desta forma, os conjuntos $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\perp})$ e $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\oslash})$ podem ser interpretados como localmente análogos para a noção de *subespaços afins*. No entanto, a interação não linear descrita na Proposição 2(iii) faz com que esta analogia seja invalidada para casos maiores (não locais). Não obstante, a própria prova da proposição sugere que é possível definir operadores genéticos que utilizam a estrutura geométrica local de um espaço, a fim de executar pesquisas neste espaço. Isto ocorre porque a composição de alguns *edit moves* com $(i, j) \in \mathcal{Q}^{\perp}$ ainda pode ser equivalente a um *edit move* com $(i, j) \in \mathcal{Q}^{\oslash}$, embora esta situação seja bastante rara. Tal situação é rara, pois exige a ocorrência de sequências

específicas de movimentos, e que estes devam ser escolhidos a partir de conjuntos muito maiores de movimentos possíveis.

Finalmente, a Proposição 3 estabelece uma relação entre o *conjunto span*, o *conjunto de troca* $\mathcal{Q}^{\oslash}$ e o *fenótipo equivalente*.

Proposição 3. *Seja* $\mathbf{x} \in \mathcal{X}$ *qualquer instância de um vetor de variáveis de decisão de um PDCC-RSSF. Então* $\mathbf{x}$ *é fenótipo equivalente para qualquer* $\bar{\mathbf{x}} \in \mathcal{L}^*(\mathbf{x}, \mathcal{Q}^{\oslash})$.

Prova:
Essa proposição vem como consequência do pressuposto da aplicação da minimalidade (Pressuposto 2). A idea é:

- Suponha um grupo arbitrário $\mathcal{G}_k(\mathbf{x}) = \{x_j, x_{j+1}, \ldots, x_{j+v}\}$. Por definição, quando se inicia o estágio k, todos os nós armazenados nas coordenadas $\{x_j, x_{j+1}, \ldots, x_{j+v}\}$ são ativados simultaneamente, no tempo $t = t_j$, e estas ativações são capazes de restaurar a cobertura e a conectividade da rede.
- Seja um vetor de variáveis de decisão $\bar{\mathbf{x}}$ diferente de $\mathbf{x}$ devido a um *edit move* que troca dois elementos do conjunto $\{x_j, x_{j+1}, \ldots, x_{j+v}\}$.
- Em princípio, $\mathbf{x}$ e $\bar{\mathbf{x}}$ não são considerados fenótipo equivalentes:
  - **(i)** se alguns dos nós indicados em $\{x_j, x_{j+1} \ldots, x_{j+v}\}$ foram ativados antes do tempo t_j, ou
  - **(ii)** se alguns destes nós forem ativados após t_j.
- A situação (i) é impossível, já que o tempo t_j em que o estágio anterior (estágio $k-1$) terminou, não depende das variáveis $\{x_j, x_{j+1}, \ldots, x_{j+v}\}$.
- A situação (ii) também é impossível, porque o procedimento de aplicação da minimalidade garante que a ativação simultânea dos nós presentes em $(\mathcal{G}_k(\mathbf{x}) - \{x_w\})$, para qualquer $w \in \{j, j+1, \ldots, j+v\}$ não é suficiente para restaurar a cobertura e a conectividade da rede, o que significa que todos os nós indicados em $\{x_j, x_{j+1}, \ldots, x_{j+v}\}$ devem ser ativados em $t = t_j$ a fim de restaurar a cobertura e a conectividade da rede.
- Portanto, $\bar{\mathbf{x}}$ é fenótipo equivalente de $\mathbf{x}$.
- Este raciocínio se aplica também a $\bar{\mathbf{x}}$ e $\mathbf{x}$ separados por qualquer número de *edit moves*, uma vez que o caminho de um até o outro é constituído por simples *edit moves*. Para que estes vetores de variáveis de decisão tivessem fenótipos diferentes, deveria haver pelo menos um *edit move* em que o fenótipo fosse alterado.

$\diamondsuit$

A Proposição 3 apresenta formalmente a razão pela qual a maior parte do esforço de pesquisa deva ser gasto explorando a vizinhança relativa definida pelo conjunto $\mathcal{Q}^{\perp}$, que corresponde à troca de elementos pertencentes a grupos diferentes. A exploração do conjunto $\mathcal{Q}^{\oslash}$ produz informação redundante, ou seja, mais conjuntos de soluções que possuem fenótipo equivalente.

Portanto, os operadores genéticos devem ser adaptados a fim de realizar explorações utilizando *edit moves* definidos pelos pares contidos em $\mathcal{Q}^{\perp}$. Abaixo são descritos os fatos relevantes a respeito desta estratégia, são eles:

- A cardinalidade de $\mathcal{Q}^{\oslash}$ é muitas vezes comparável ao de $\mathcal{Q}^{\perp}$ (isto depende dos tamanhos dos grupos e do número de grupos);
- Neste caso, a cardinalidade do conjunto de combinações do conjunto $\mathcal{Q}^{\perp}$ torna-se muito menor que a cardinalidade do conjunto de combinações do conjunto $\mathcal{Q}^{\perp} \cup \mathcal{Q}^{\oslash}$;
- Portanto, a exploração do espaço utilizando *edit moves* $V(\mathbf{x}, i, j)$ com (i, j) feitos a partir de $\mathcal{Q}^{\perp}$, analisa, em cada movimento, um conjunto $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\perp})$ que tem uma cardinalidade muito menor que a vizinhança $\mathcal{N}(\mathbf{x})$. Por outro lado, a exploração do espaço utilizando *edit moves* $V(\mathbf{x}, i, j)$ com (i, j) feitos apartir de $\mathcal{Q}^{\perp} \cup \mathcal{Q}^{\oslash}$ examinaria, em cada movimento, um conjunto $\mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{\perp} \cup \mathcal{Q}^{\oslash}) = \mathcal{L}^1(\mathbf{x}, \mathcal{Q}^{*}) = \mathcal{N}(\mathbf{x})$.

Portanto, a busca construída a partir de $\mathcal{Q}^{\perp}$ será provavelmente mais eficiente, sem perda significativa da capacidade de busca, que a busca utilizando todo o conjunto de *edit moves* possíveis. Esta constatação será empregada na seção seguinte, na construção de um Algoritmo Genético eficiente para o PDCC-RSSF.

4.3.1 Generalidade das Entidades Geométricas

Como um dos objetivos deste trabalho é desenvolver uma abordagem baseada na geometria para problemas combinatórios que utilizam computação evolutiva, é importante comentar sobre a generalidade das entidades geométricas que foram discutidas nesta seção.

As seguintes entidades: *vetor de variáveis de decisão*; *edit move*; *vizinhança*; *caminho*; *tamanho do caminho*; *caminho mínimo*; *distância*, serão exibidas, em geral, em qualquer problema combinatório. Uma discussão sobre as possíveis diferentes formulações de um conceito de *distância*, juntamente com as consequências dessas formulações é mostrada em (Carrano et al., 2010). A *vizinhança relativa*, em que os *edit moves* são restritos a um subgrupo dos possíveis movimentos, terá formulação análoga na maioria dos problemas combinatórios. No entanto, não tendo a obrigatoriedade de estar associado a uma entidade de *grupos*.

A entidade *variedade da variável de decisão*, como apresentado aqui, aparecerá apenas em problemas combinatórios em que a codificação emprega permutações.

O conceito de *fenótipo equivalente* irá aparecer em qualquer problema combinatório em que exista uma redundância na codificação. A entidade *grupo*, nesses casos, irá se referir a genótipos que mapeiam em um único fenótipo. Em tais casos, os conjuntos de troca $\mathcal{Q}^{\perp}$, $\mathcal{Q}^{\oslash}$ e $\mathcal{Q}^{*}$ irão aparecer associados aos grupos, definindo suas respectivas *vizinhanças relativas*. A *Proposição 1*, *Proposição 2* e a *Proposição 3*, irão se manter nesses casos, embora possam depender do problema em específico, podendo estas não serem relevantes.

4.4 OPERADORES GENÉTICOS

O algoritmo genético geométrico proposto para o PDCC-RSSF utiliza a não-homogeneidade das direções do espaço, como mostrado na seção anterior, na definição dos operadores geométricos. As buscas utilizando *edit moves* definidos pelo conjunto $\mathcal{Q}^{\oslash}$ tendem a não serem muito informativas. Portanto, estas buscas são realizadas através de um mecanismo de busca local guloso que executa a operação $\bar{\mathcal{G}}_k(\mathbf{x}) = \text{mingr}(\mathcal{G}_k(\mathbf{x}))$, assim assegurando que a suposição de minimalidade de grupo seja válida (Pressuposto 2). Por outro lado, as buscas com *edit moves* de $\mathcal{Q}^{\perp}$, podem potencialmente gerar novas informações, apresentando um cenário de *fitness* com vários mínimos locais. O operador de mutação é projetado sobre este conjunto, a fim de gerar a inovação que é necessária para explorar este cenário. Estes são os princípios gerais que fundamentam o algoritmo proposto.

A escolha dos parâmetros de alguns operadores que serão apresentados a seguir foram baseados em experimentos que são descritos no Apêndice A.

4.4.1 Operador de Mutação

O operador de mutação pode ser definido com base no *edit move* que define a geometria do espaço. Em um *edit move* dois sensores são trocados de posição na codificação da solução. O operador de mutação desenvolvido aqui pode ser definido da seguinte maneira:

Definição 22. ***Operador de Mutação:*** *corresponde a executar edit moves em sensores pertencentes a diferentes grupos escolhidos de forma aleatória. Uma operação de mutação será constituída por um, dois ou três movimentos consecutivos executados independentemente. A quantidade de movimentos é escolhida de forma aleatória com igual probabilidade de ocorrência.*

Conforme a definição acima, vale a pena enfatizar que nenhum *edit move* é permitido dentro de um mesmo grupo, o que corresponde a uma projeção dos movimentos sobre o conjunto $\mathcal{Q}^{\perp}$. Isto evita o gasto desnecessário de avaliações de função após mutações que tenham conduzido a indivíduos que compartilham o mesmo fenótipo dos pais.

Um algoritmo para este operador é mostrado no Algoritmo 4. As características deste algoritmo são:

- $\mathbf{x}$ é uma solução (vetor de variáveis de decisão).
- H representa os estágios em que os sensores de $\mathbf{x}$ são ativados.
- O é a solução descendente.
- RAND($\mathcal{D}$) retorna um elemento aleatório de um conjunto $\mathcal{D}$ fornecido.

A aplicação do operador de mutação é ilustrado na Figura 4.8. Neste exemplo, $n_{op} = 3$. No primeiro momento os sensores 5 e 6 são trocados de posição, sequencialmente são trocados os sensores 3 e 4 e por fim os sensores 1 e 7 também são trocados.

Algoritmo 4 Pseudocódigo para Mutação

```
1: função MUTATE(x,H)
2:     O ← x;
3:     n_op ← rand({1,2,3})                          ▷ número de trocas
4:     para k de 1 até n_op faça
5:         r_1 ← RAND({1,...,|S|})                   ▷ tarefa 1
6:         r_2 ← RAND({1,...,|S|})                   ▷ tarefa 2
7:         enquanto H_{r_1} = H_{r_2} faça
8:             r_2 ← RAND({1,...,|S|})
9:         fim enquanto
10:        troque a tarefa r_1 e r_2 em O;
11:    fim para
12:    retorno O.
13: fim função
```

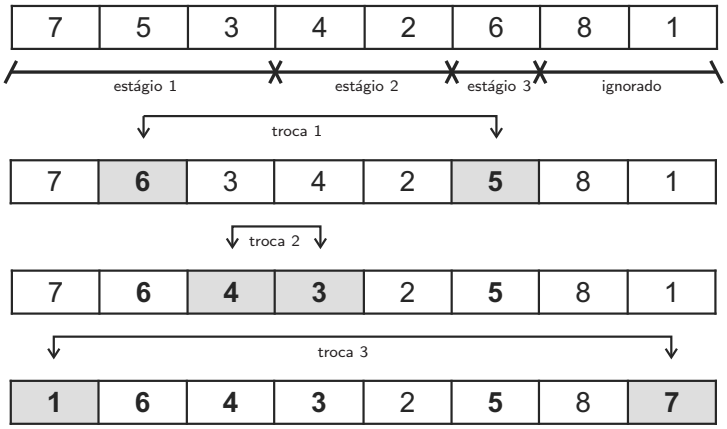


Figura 4.8 Exemplo do operador de mutação.

4.4.1.1 Algoritmo Multiobjetivo Na modelagem multiobjetivo apresentada no Capítulo 3 a restrição de cobertura foi transformada em um novo objetivo. A representação deste objetivo foi codificada na seção 4.2.3 em um genoma individual, contendo a variável C_i. Desta forma, na versão multiobjetivo é necessário prever a possibilidade de realização de uma mutação adicional no genoma do indivíduo. A mutação desta variável é feita substituindo o valor atual por um novo valor escolhido aleatoriamente no intervalo $C_{min} \leq C_i \leq 1.00$.

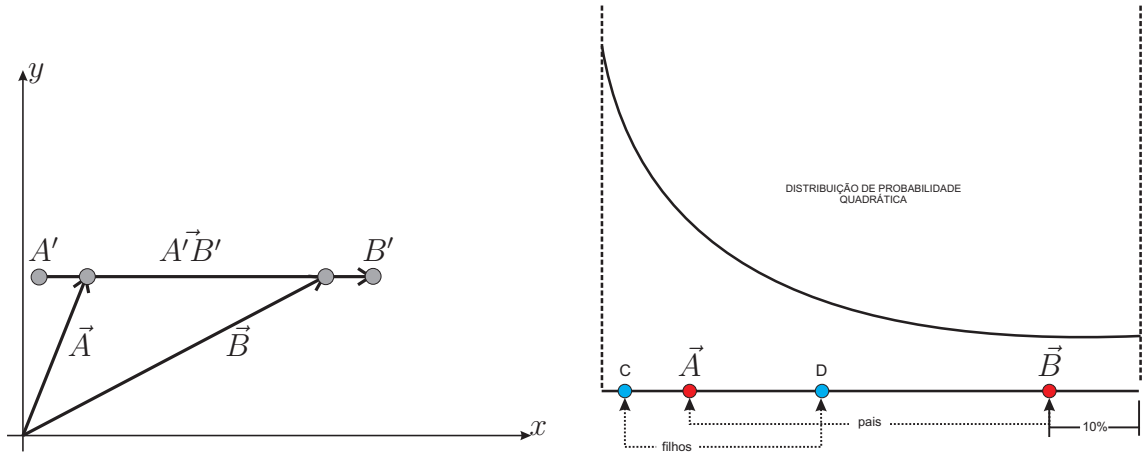
4.4.2 Operador de Cruzamento

Para construção do operador geométrico de cruzamento para o PDCC-RSSF serão utilizadas as definições de *edit move* e distância mostradas na seção 4.3. Diferente do operador de mutação, aqui serão considerados todos os pares $(i, j) \in \mathcal{Q}^*$. Tal operador é baseado no operador de Cruzamento Real-Polarizado (CRP), proposto por Takahashi et al. (2003) para otimização de funções contínuas.

Dado os vetores $\vec{A}$ e $\vec{B}$ representando dois pais, as soluções filhas são geradas sobre a linha $A'\vec{B}'$, tais que:

$$A\vec{B} = \vec{B} - \vec{A} \tag{4.1}$$
$$\vec{A}' = \vec{A} - 0.1 \cdot A\vec{B} \tag{4.2}$$
$$\vec{B}' = \vec{B} + 0.1 \cdot A\vec{B} \tag{4.3}$$
$$A'\vec{B}' = \vec{B}' - \vec{A}' \tag{4.4}$$

O vetor $A'\vec{B}'$ é ilustrado na Figura 4.9(a). O CRP gera duas soluções filhas no segmento de A' até B' que inclui os pontos que compõem a combinação convexa de A e B e também certa extrapolação fora deste segmento, na mesma linha. Um dos indivíduos descendentes é gerado segundo uma função de distribuição uniforme em $\overline{A'B'}$. O outro indivíduo filho é gerado segundo uma função de distribuição quadrática, em que a vizinhança do melhor pai tem maiores chances de receber o novo indivíduo (veja a Figura 4.9(b)). Esta última operação proporciona uma espécie de "busca descendente", uma vez que o indivíduo gerado tem maior probabilidade de estar no segmento extrapolado, fora de $\overline{AB}$, na direção em que a função é melhor.



(a) Exemplo de vetor de suporte em CRP

(b) Função de distribuição quadrática.

Figura 4.9 Operador de Cruzamento Real Polarizado

Apesar de não ser possível, literalmente, ligar duas soluções candidatas do PDCC-RSSF através de uma linha reta, um caminho mínimo[3] que liga tais soluções pode ser construído sequencialmente fazendo *edit moves*, como mostrado na Definição 14. Além disso, é também possível definir o segmento de reta extrapolado que passa sobre a solução pai do PDCC-RSSF fazendo-se alguns *edit moves* em cada solução pai que aproximem as seguintes condições:

1. $d(A, A') \approx 0.1 \cdot d(A, B)$

[3]O caminho é considerado mínimo se o número de *edit moves* necessários para transformar A em B (ou vice e versa) for mínimo, ou seja, igual à distancia ($d(A, B)$)

2. $d(B, B') \approx 0.1 \cdot d(A, B)$
3. $d(A', B') \approx 1.2 \cdot d(A, B)$

É claro que no conjunto de pontos do espaço definido pelas permutações usualmente não é possível atingir a igualdade exata nessas condições. Condições adequadas para serem aplicadas diretamente podem ser definidas da seguinte maneira:

$$\begin{aligned} &A' = \arg\max_{Z} d(B, Z) \\ &\text{sujeito a: } d(A, Z) \leq \max\{0.1 \cdot d(A, B), 1\} \end{aligned} \tag{4.5}$$

e

$$\begin{aligned} &B' = \arg\max_{Z} d(A, Z) \\ &\text{sujeito a: } d(B, Z) \leq \max\{0.1 \cdot d(A, B), 1\}. \end{aligned} \tag{4.6}$$

Um pseudocódigo pode ser visto no Algoritmo 5, sendo que:

- SHUFFLE($\mathcal{G}$) ordena os elementos de $\mathcal{G}$ aleatoriamente;
- RAND($\mathcal{D}$) retorna um elemento aleatório de um conjunto $\mathcal{D}$ fornecido.

Depois de realizar a extrapolação, a distância entre A' e B' é avaliada ($d(A', B')$), e um vetor binário W é criado. Este vetor tem o mesmo tamanho do vetor pai, e em cada uma de suas células contém 0 se A' e B' são iguais para um mesmo índice, ou 1 caso contrário. Então, um valor inteiro $0 \leq p < d(A, B)$ é gerado e, de forma aleatória, p posições não nulas de W são escolhidas para serem mudadas para 0. Por fim, alguns *edit moves* são executados de forma a garantir que A e B sejam iguais em todos os índices em que W é não nulo, ou seja, para todo $W_i = 1$. As demais posições não são alteradas. Um pseudocódigo para este operador pode ser visto no Algoritmo 6. É assumido, sem perda de generalidade, que $f_{lifetime}(A) < f_{lifetime}(B)$. No algoritmo:

- A e B são soluções pais (vetores de variáveis de decisão);
- O^A e O^B são soluções filhas;
- RANDQUAD($\mathcal{D}$) retorna um elemento aleatório de um conjunto $\mathcal{D}$ dado, conforme a função de distribuição quadrática mostrada na Figura 4.9(b). Menores valores de $\mathcal{D}$ têm maiores chances de serem escolhidos.

Este operador de cruzamento pode ser visto como uma sequência de operações de *edit moves*, que conduza das soluções pais às soluções descendentes. Entretanto, cada mutação deve ser escolhida de forma a gerar soluções que estejam necessariamente contidas no caminho mínimo que conecta A' e B'. Já o procedimento de extrapolação adotado é particularmente importante para gerar diversidade ao longo das gerações do algoritmo, principalmente quando as soluções pais se tornarem muito similares.

É importante notar que este operador de cruzamento executa pesquisas em todo conjunto de possíveis *edit moves* definido por $\mathcal{Q}^*$ e não apenas os definido por $\mathcal{Q}^{\perp}$. Isso

Algoritmo 5 Pseudocódigo para a operação de extrapolação

```
1: função EXTRAPOLATE(A, B)
2:     A' ← A
3:     B' ← B
4:     te ← 0,1                                   ▷ taxa de extrapolação
5:     dist_AB ← d(A, B)
6:     numExt ← ⌈te × dist_AB⌉                    ▷ número de movimentos para extrapolar
7:     G_dif ← {}                                 ▷ conjunto de variáveis de decisão (genes) diferentes
8:     G_eq ← {}                                  ▷ conjunto de variáveis de decisão (genes) iguais
9:     para i de 1 até |A| faça
10:        se A_i ≠ B_i então
11:            G_dif ← {G_dif  i}
12:        senão
13:            G_eq ← {G_eq  i}
14:        fim se
15:    fim para
16:    se |G_eq| = 1 então
17:        G_dif ← SHUFFLE(G_dif)
18:        j ← RAND(1, ..., |G_dif|)
19:        t_1 ← G_eq[0]                          ▷ tarefa 1
20:        t_2 ← G_dif[j]                         ▷ tarefa 2
21:        troque as tarefas t_1 e t_2 em A' e em B';
22:    fim se
23:    se |G_eq| > 1 então
24:        G_eq ← SHUFFLE(G_eq)
25:        para i e j de 1 até i < numExt e j < |G_eq| faça
26:            t_1 ← G_eq[j]                      ▷ tarefa 1
27:            t_2 ← G_eq[j + 1]                  ▷ tarefa 2
28:            troque as tarefas t_1 e t_2 em A' e em B';
29:            j ← j + 2
30:        fim para
31:    fim se
32:    retorno A', B'.
33: fim função
```

é desejável, pois ainda há algumas interações entre os conjuntos $\mathcal{Q}^{\perp}$ e $\mathcal{Q}^{\oslash}$ que devem ser investigadas. Além disso, como o cruzamento geométrico é uma pesquisa ao longo de um segmento de reta, não há explosão da cardinalidade do conjunto de busca.

Um exemplo deste operador é mostrado na Figura 4.10. Neste exemplo, três *edit moves* são executados na solução extrapolada A' de forma a aproximá-la da solução B'.

4.4.2.1 Algoritmo Multiobjetivo Na versão multiobjetivo do algoritmo o cruzamento é similar ao mostrado na versão de apenas um objetivo. Existem duas pequenas

Algoritmo 6 Pseudocódigo para o Cruzamento

```
1: função CROSSOVER(A,B)
2:     [A', B'] ← EXTRAPOLATE(A, B);
3:     avalie d(A', B')
4:     O^A ← A';
5:     O^B ← B';
6:     encontre o vetor binário W;
7:     W^A ← W;
8:     p^A ← RANDQUAD(0, d(A', B') − 1)
9:     faça p^A posições não nulas de W^A com 0.
10:    para cada posição não nula k de W^A faça
11:        t_1 ← A'_k                                   ▷ tarefa 1
12:        t_2 ← B'_k                                   ▷ tarefa 2
13:        troque as tarefas t_1 e t_2 em O^A;
14:    fim para
15:    W^B ← W;
16:    p^B ← rand({0, ..., d(A', B') − 1})
17:    faça p^B posições não nulas de W^B com 0.
18:    para cada posição não nula k em W^B faça
19:        t_1 ← A'_k                                   ▷ tarefa 1
20:        t_2 ← B'_k                                   ▷ tarefa 2
21:        troque as tarefas t_1 e t_2 em O^B;
22:    fim para
23:    retorno O^A, O^B.
24: fim função
```

diferenças, que são:

1. Ambos os indivíduos filhos são gerados segundo a distribuição uniforme no segmento já extrapolado $\overline{A'B'}$, não sendo mais utilizada a distribuição quadrática em um dos novos indivíduos. Esta mudança é necessária, pois não faz sentido seguir uma direção de busca descendente em apenas um dos objetivos, dado que o problema agora envolve dois objetivos conflitantes.

2. Como visto na seção 4.2.3, a codificação multiobjetivo possui um genoma individual contendo a variável C_i. Para fazer o cruzamento destas variáveis basta fazer uma simples interpolação entre os C_i's dos indivíduos pais (Figura 4.11).

4.4.3 Operador de Ativação Inteligente (Smart Activation Operator)

Como discutido anteriormente, o final de um estágio é caracterizado pelo surgimento de uma falha em um ou mais nós sensores. Na maioria das vezes ocorre a falha em apenas um sensor. Quando um sensor falha é necessário retirá-lo da configuração corrente da

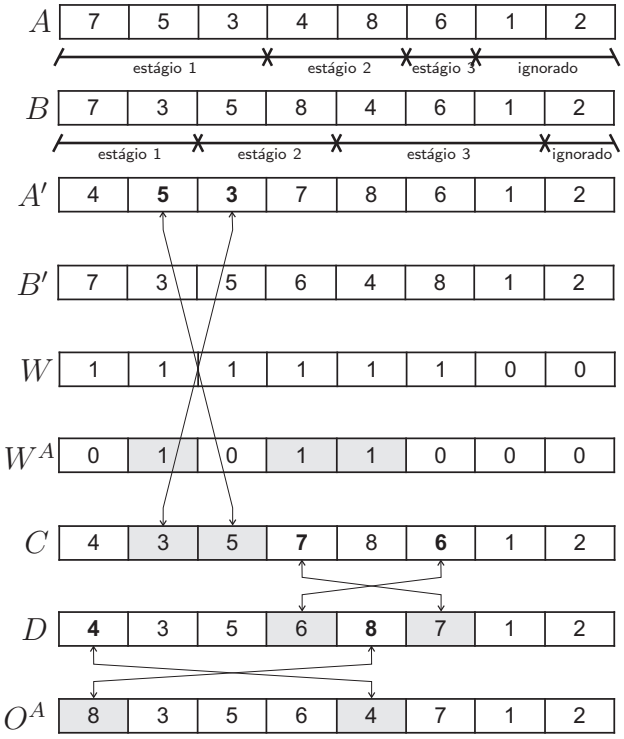


Figura 4.10 Exemplo do Operador de Cruzamento

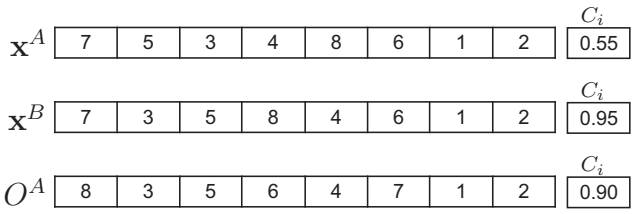


Figura 4.11 Exemplo do Operador de Cruzamento Multiobjetivo aplicado ao mesmo exemplo da Figura 4.10.

rede, pois ele não consegue mais sensoriar ou até mesmo ser usado para comunicação *multi-hop*. Consequentemente, esta falha implica uma redução na cobertura da rede, podendo levá-la à inviabilidade, devido à violação da restrição de cobertura (g_1), sendo necessário ativar nós que se encontravam adormecidos para permitir a continuidade do funcionamento da rede. Como o principal objetivo do algoritmo é estender a vida útil da rede, seria interessante ativar um pequeno grupo de nós sensores, a fim de manter um maior número de nós sensores disponíveis para os próximos estágios (Tilak et al., 2002).

O Operador de Ativação Inteligente é então proposto para tentar melhorar o tempo de vida da rede: ele corresponde a uma busca local dentro de um subconjunto de soluções adjacentes. Este operador tenta encontrar um simples nó sensor que pode restaurar a cobertura da rede após alguma falha. Se este nó sensor existe, então uma operação de troca (*edit move*) é executada, de tal forma que a ativação deste nó seja definida logo após o estágio em que a falha ocorre. Caso contrário, a operação de decodificação é executada

normalmente, como descrito na Seção 4.2.2.

Um pseudocódigo deste operador é mostrado no Algoritmo 7, sendo que:

- $\mathbf{x}$ representa a solução de entrada (vetor de variáveis de decisão);
- $\mathcal{A}_s$ o conjunto de sensores ativos no momento;
- j o índice do primeiro nó sensor de $\mathbf{x}$ que não tenha sido ainda ativado; e
- $G_{cov}[\mathcal{A}_s]$ a função que retorna o número de pontos de demanda que são cobertos pelos nós sensores que estão ativos (mostrado anteriormente na Equação 3.11).

O Operador de Ativação Inteligente deve ser executado em cada estágio, durante o processo de decodificação. Ao considerar o Algoritmo 3 mostrado na Seção 4.2.2, uma função chamada SmAct (*Smart Activation*) pode ser colocada antes da ativação do nó sensor (linha 9).

Algoritmo 7 Pseudocódigo para o Operador de Ativação Inteligente

```
1: função SmAct(x, A_s, j, C)
2:     stop ← FALSO;
3:     p ← j                                          ▷ Sensor corrente
4:     enquanto stop = FALSO e p < |S| faça
5:         ative o sensor x_p e inclua ele em A_s;
6:         se G_cov[A_s] ≥ C · |D| então
7:             troque as tarefas x_p e x_j em x;
8:             stop ← VERDADEIRO;
9:         fim se
10:        desative x_p;
11:        remova o sensor x_p de A_s;
12:        p ← p + 1;
13:    fim enquanto
14:    retorno x.
15: fim função
```

Um exemplo da aplicação deste operador é mostrado na Figura 4.12. A partir deste exemplo é possível observar que:

i) No final do Estágio 1 (após falha do nó sensor 2), o operador identifica que o próximo nó sensor na sequência (sensor 3) consegue reestabelecer a cobertura da rede.

ii) Após a falha do nó sensor 5, o operador detecta que não é possível reestabelecer a cobertura apenas com o nó sensor 4. Então, o operador tentar encontrar um nó sensor, entre os restantes na sequência, que possa restaurar o funcionamento da rede. Esta avaliação indica que o nó sensor 8 é capaz de realizar tal tarefa, resultando em uma troca entre os nós sensores 4 e 8.

iii) Quando o estágio 3 termina, o operador proposto identifica que não é possível encontrar um sensor que restaure a rede. Neste caso, o processo usual de decodificação (Algoritmo 3) é empregado. Como consequência, mais de um nó sensor é ativado (nós 1, 4 e 7).

iv) Finalmente, no final do estágio 4, após a falha do nó 1, torna-se impossível restaurar a rede com o nó sensor restante (sensor 6). Desta forma o processo de decodificação é encerrado.

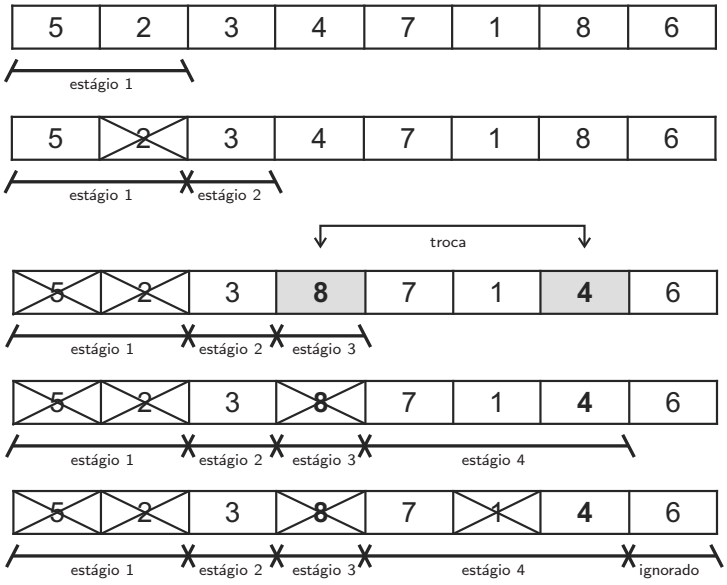


Figura 4.12 Exemplo do Operador de Ativação Inteligente.

4.4.3.1 Algoritmo Multiobjetivo Na versão multiobjetivo, pode ser utilizado o mesmo Operador de Ativação Inteligente descrito na seção anterior, sem necessidade de alterações ou adaptações.

4.4.4 Operador de Desativação Inteligente (Smart Deactivation Operator)

A cada novo estágio um novo conjunto de sensores é ativado e é possível que estes novos sensores introduzam uma redundância de cobertura na rede, em que múltiplos sensores cubram os mesmos pontos de demanda. Nestes casos, talvez seja possível desativar alguns nós sensores ativos sem que se viole a restrição de cobertura (g_1). Estas desativações podem contribuir para aumentar o tempo de vida da rede, uma vez que estes nós sensores tornam-se disponíveis para uso nas operações dos estágios seguintes.

Este operador faz com que haja uma redução do tamanho do grupo, ao remover dos grupos algumas operações de ativação de nós sensores. Na verdade, este operador corresponde a realização da operação $\bar{\mathcal{G}}_i(\mathbf{x}) = \text{mingr}(\mathcal{G}_i(\mathbf{x}))$, discutido na Seção 4.3 após a definição de **grupo mínimo** (Definição 17). Portanto, este operador desempenha um

papel fundamental, pois faz assegurar a garantia da minimalidade de grupo apresentada no Pressuposto 2 (Seção 4.3).

O funcionamento do Operador de Desativação Inteligente pode ser descrito da seguinte maneira: no início de cada estágio, depois de realizadas as ativações de todos os nós sensores, o operador tenta encontrar dentre os nós ativos quais podem ser desativados sem que a taxa de cobertura seja reduzida para patamares menores que C. Um pseudocódigo deste operador é mostrado no Algoritmo 8. Neste algoritmo:

- $\mathbf{x}$ representa a solução de entrada (vetor de variáveis de decisão);
- $\mathcal{A}_s$ o conjunto de sensores que estão ativos no estado corrente;
- j o índice do primeiro nó sensor de $\mathbf{x}$ que ainda não foi ativado; e
- $G_{cov}[\mathcal{A}_s]$ a função que retorna o número de pontos de demanda que são cobertos pelos nós sensores que estão ativos.

Assim como no Operador de Ativação Inteligente, este procedimento deve ser executado em cada etapa do processo de decodificação. No entanto, neste caso, a chamada de função deve ser colocada após a ativação dos nós sensores. Tal operador pode ser interpretado como uma sequência de buscas locais dentro dos grupos. Devido à utilização deste operador, a busca dentro dos grupos através do operador de mutação torna-se desnecessária.

Algoritmo 8 Pseudocódigo para o Operador de Desativação Inteligente

```
1: função SmDeact(x, A_s, j, C)
2:     p ← j − 1                                          ▷ sensor corrente
3:     enquanto p ≥ 0 faça
4:         desative o sensor x_p e o remova de A_s;
5:         se G_cov[A_s] ≥ C · |D| então
6:             mova x_p para o início do próximo conjunto (estágio);
7:             j ← j − 1;
8:         senão
9:             reative o sensor x_p e o recoloque em A_s;
10:        fim se
11:        p ← p − 1;
12:    fim enquanto
13:    retorno x, A_s, j.
14: fim função
```

Um exemplo de como é o funcionamento do Operador de Desativação Inteligente pode ser visto na Figura 4.13. Neste exemplo, no fim do estágio 3 é verificado que há necessidade de ativar os nós sensores 7, 1 e 4 para que seja reestabelecida a cobertura da rede. O operador de desativação proposto identifica que o sensor 1 pode ser desativado sem que a restrição de cobertura (g_1) e as restrições de conectividade sejam violadas. Desta forma, o sensor 1 é desativado e movido para o início do próximo estágio, na posição anterior ao sensor 6.

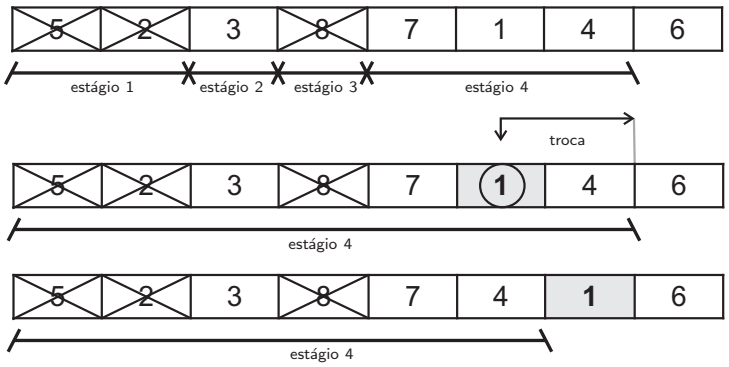


Figura 4.13 Exemplo do Operador de Desativação Inteligente.

4.4.4.1 Algoritmo Multiobjetivo Na versão multiobjetivo, pode ser utilizado o mesmo Operador de Desativação Inteligente descrito na seção anterior, sem necessidade de alterações ou adaptações.

4.4.5 Avaliação da Solução

A decodificação e os operadores de ativação inteligente e desativação inteligente descritos nas seções anteriores podem ser combinados de forma a construir um operador único de avaliação. Este operador de avaliação melhora a solução de entrada e retorna seu tempo de vida e a cobertura neste período. Um pseudocódigo pode ser visto no Algoritmo 9.

Como pode ser visto, o Algoritmo 9 retorna três variáveis, são elas:

lifetime **-** o tempo de vida da rede;

coverage **-** a média da cobertura durante o tempo de vida da rede;

H **-** o estágio em que cada nó sensor é ativado;

Deve ser notado que não há necessidade de se avaliar as restrições dentro da função, uma vez que o seu cumprimento é assegurado pelo o processo de decodificação adotado. O indivíduo X a ser avaliado deve ser uma cópia do indivíduo presente na população, pois este indivíduo pode ser alterado durante o processo de avaliação devido ao uso dos operadores de ativação e desativação inteligente. Como as possíveis alterações realizadas nos indivíduos durante o processo de avaliação são determinísticas, não há necessidade de retornar o indivíduo alterado, pois ele sempre terá o mesmo desempenho na simulação da rede.

É importante ressaltar que o mesmo operador de avaliação pode ser usado em ambas as versões do algoritmo: mono-objetivo e multiobjetvo. Na versão de apenas um objetivo o parâmetro de saída *coverage* é descartado, uma vez este não é necessário para avaliar a função *fitness* da solução.

Algoritmo 9 Pseudocódigo para Avaliação da Solução

função EVAL($\mathbf{x}$, E_{max}, C)
  $j \leftarrow 0$ ▷ sensor corrente
  $c_{stg} \leftarrow 0$; ▷ estágio corrente
  $lifetime \leftarrow 0$; ▷ tempo de vida
  $cov_{num} \leftarrow 0$;
  $E(0) = [E_{max}\ E_{max}\ \ldots\ E_{max}]'$;
  $\mathcal{A}_s \leftarrow \emptyset$; ▷ conjunto dos sensores ativos
  enquanto $j < |\mathcal{S}|$ **faça**
    $\mathbf{x} \leftarrow$ SMACT $(\mathbf{x}, \mathcal{A}_s, j, C)$
    enquanto $G_{cov}[\mathcal{A}_s] < C \cdot |\mathcal{D}|$ e $j < |\mathcal{S}|$ **faça**
      ative o sensor x_j e o **inclua** em $\mathcal{A}_s$;
      $H_{x_j} \leftarrow c_{stg} + 1$
      $j \leftarrow j + 1$;
    fim enquanto
    $[\mathbf{x}, \mathcal{A}_s, j] \leftarrow$ SMDEACT $(\mathbf{x}, \mathcal{A}_s, j, C)$
    se $G_{cov}[\mathcal{A}_s] \geq C \cdot |\mathcal{D}|$ **então**
      simule a rede até que a primeira falha ocorra entre os nós sensores de $\mathcal{A}_s$;
      $lifetime \leftarrow lifetime + F_{time}[\mathcal{A}_s, E(c_{stg})]$;
      $cov_{num} \leftarrow cov_{num} + G_{cov}[\mathcal{A}_s] \cdot F_{time}[\mathcal{A}_s, E(c_{stg})]$;
      remova os nós sensores falhos de $\mathcal{A}_s$;
      $c_{stg} \leftarrow c_{stg} + 1$;
      atualize $E(c_{stg})$;
    fim se
  fim enquanto
  $coverage \leftarrow \dfrac{cov_{num}}{lifetime}$; ▷ média ponderada da cobertura
  retorno *lifetime*, *coverage*, H.
fim função

4.5 ALGORITMO GENÉTICO PROPOSTO

Neste capítulo são propostas duas versões de um Algoritmo Genético Geométrico para Redes de Sensores Sem Fio, são elas:

Mono-Objetivo: uma versão mono-objetivo do Algoritmo Genético para Escalonamento Dinâmico de Redes de Sensores Sem Fio (**W***ireless* **S***ensor* **N***etwork* **d***ynamic* **s***cheduling* **G***enetic* **A***lgorithm* - WSNdsGA)

Multiobjetivo: uma versão multiobjetivo do WSNdsGA (**M**ulti**O**bjective **W***ireless* **S***ensor* **N***etwork* **d***ynamic* **s***cheduling* **G***enetic* **A***lgorithm* - MOWSNdsGA)

As duas versões possuem os mesmos parâmetros de entrada:

- S_{pop}: tamanho da população;

- max_{evl}: número máximo de avaliações de função;
- p_c: probabilidade de cruzamento;
- p_m: probabilidade de mutação.

Os resultados atingidos por cada abordagem e as comparações com outros trabalhos previamente introduzidos na literatura serão descritos no Capítulo 6.

4.5.1 Abordagem Mono-Objetivo - WSNdsGA

O WSNdsGA é um usual algoritmo genético com operadores de seleção, cruzamento e mutação sendo executados na sequência, e em cada geração. O torneio binário e elitismo foram adotados como método de seleção. Já os operadores de mutação, cruzamento e avaliação foram construídos tais como descritos na Seção 4.4. A população inicial é gerada por um procedimento que gera uma permutação aleatória com distribuição de probabilidade uniforme. Finalmente, o critério de parada adotado foi o número máximo de avaliações de função. Um pseudocódigo pode ser visto no Algoritmo 10, em que:

- $\mathcal{P}^t$ representa a população;
- $|\mathcal{P}^t|$ é o tamanho da população;
- $\mathcal{P}_i^t \in \mathcal{P}^t$ é a estrutura que contem informações sobre o indivíduo i:
  - $\mathcal{P}_i^t.\mathbf{x}$: armazena o vetor de variáveis de decisão;
  - $\mathcal{P}_i^t.lifetime$: armazena o valor da função objetivo que representa o tempo de vida da rede;
  - $\mathcal{P}_i^t.coverage$: armazena o valor da função objetivo que representa a média da cobertura;
  - $\mathcal{P}_i^t.H$: armazena os estágios em que os nós sensores serão ativados;
- SHUFFLE($\mathcal{P}^t$) ordena os elementos de $\mathcal{P}^t$ aleatoriamente;
- PICKBEST($\mathcal{P}^t$) retorna o melhor indivíduo da população;
- BINTOURNAMENT ($\mathcal{P}^{t-1}, s_{pop} - 1$) faz um torneio binário com reposição; e
- RANDU é um gerador de números aleatórios com distribuição uniforme em $[0, 1[$.

Algoritmo 10 Pseudocódigo para o WSNdsGA

procedimento WSNDSGA(s_{pop},max_{ev},p_c,p_m, E_{max}, C)
  $t \leftarrow 0$ e $n_{ev} \leftarrow 0$;
  gere a população inicial $\mathcal{P}^t$;
  para cada $P_i^t \in \mathcal{P}^t$ **faça**
    $\{P_i^t.lifetime, P_i^t.coverage, P_i^t.H\} \leftarrow$ EVAL($P_i^t.\mathbf{x}, E_{max}, C$);
    $n_{ev} \leftarrow n_{ev} + 1$;
  fim para
  $P_{best} \leftarrow$ PICKBEST($\mathcal{P}^t$);
  enquanto $n_{ev} < max_{ev}$ **faça**
    $t \leftarrow t + 1$;
    $\mathcal{O} \leftarrow \emptyset$;
    $\mathcal{P}^t \leftarrow P_{best} \cup$ BINTOURNAMENT $(\mathcal{P}^{t-1}, s_{pop} - 1)$; ▷ elitismo e seleção
    $\mathcal{P}^t \leftarrow$ SHUFFLE($\mathcal{P}^t$);
    para i de 1 até $\dfrac{|\mathcal{P}^t|}{2}$ **faça**
      se RANDU $\leq p_c$ **então**
        $\{I_1, I_2\} \leftarrow$ CROSSOVER$\left(P_i^t.\mathbf{x}, P_{|\mathcal{P}^t|-i+1}^t.\mathbf{x}\right)$;
        $\mathcal{O} \leftarrow \mathcal{O} \cup I_1 \cup I_2$;
      fim se
    fim para
    para cada $P_i^t \in \mathcal{P}^t$ **faça**
      se RANDU $\leq p_m$ **então**
        $I \leftarrow$ MUTATE($P_i^t.\mathbf{x}, P_i^t.H$);
        $\mathcal{O} \leftarrow \mathcal{O} \cup I$;
      fim se
    fim para
    para cada $O_i \in \mathcal{O}$ **faça**
      $\{O_i.lifetime, O_i.coverage, O_i.H\} \leftarrow$ EVAL($O_i.\mathbf{x}, E_{max}, C$);
      $n_{ev} \leftarrow n_{ev} + 1$;
    fim para
    $P_{best} \leftarrow$ PICKBEST($P_{best} \cup \mathcal{O}$);
    $\mathcal{P}^t \leftarrow \mathcal{P}^t \cup \mathcal{O}$;
  fim enquanto
fim procedimento

4.5.2 Abordagem Multiobjetivo - MOWSNdsGA

A versão MOWSNdsGA tem a mesma estrutura do algoritmo NSGA-II original proposto por Deb et al. (2002). As operações de seleção são realizadas pelos procedimentos *fast non-dominated sorting* ("rápida classificação de não dominância") e *crowding distance* ("distância de aglomeração"), mostrados respectivamente nos Algoritmos 1 e 2. Já as operações de mutação, cruzamento e avaliação de função, são realizadas exatamente como

descritas na Seção 4.4. A população inicial é também escolhida por um procedimento que gera permutações de forma aleatória com probabilidade uniforme, lembrando que na versão multiobjetivo a codificação de cada indivíduo possui um gene a mais representando a variável C. Ao gerar a população inicial, um valor aleatório entre $C_{min} \leq C_i \leq 1.00$ deve também ser escolhido para cada indivíduo i.

CAPÍTULO 5

DIFERENTES ABORDAGENS ONLINE PARA RSSF

Uma das características desejáveis de uma Rede de Sensores sem Fio (RSSF) é a tolerância a falhas. Tolerância a falhas é um dos componentes que constituem um sistema confiável (Lussier et al., 2005). Nas RSSF a ocorrência de falhas é frequente, e estas podem ser causadas por diversos fatores: fenômenos atmosféricos, interferência, desastres naturais, quebra acidental, esgotamento da bateria, etc. Normalmente uma RSSF é projetada para monitorar ambientes hostis e de difícil acesso, como florestas e áreas de risco. Como consequência, os nós sensores presentes na rede podem ser facilmente danificados ou mesmo terem seus recursos energéticos esgotados, alterando assim a topologia da rede. Devido às características das áreas monitoradas, a substituição destes sensores ou de suas baterias torna-se inviável. Sendo assim, este tipo de rede deve apresentar mecanismos para se auto-organizarem durante seu tempo de vida.

Uma das falhas recorrentes em RSSF é a falta de energia nos sensores. Tanto a carga das baterias como o consumo real dos sensores nem sempre podem ser fielmente preditos, sendo difícil sua modelagem exata. Neste capítulo será apresentada uma breve descrição do comportamento do consumo de baterias, e como este comportamento foi modelado a fim de realizar testes em uma RSSF. Além disto, será apresentado um esboço de um protocolo de tolerância a falhas, para ser possível uma auto-organização da rede na presença de falhas inesperadas.

5.1 COMPORTAMENTO DA BATERIA

Os autores de (da Silva et al., 2010) apresentam uma avaliação através de curvas temporais de consumo de energia e de capacidade remanescente de baterias alcalinas, utilizando os modelos de estimação baseados na corrente e na tensão. Para simular os experimentos os autores utilizam o nó sensor MICAz (XBOW, 2006b) e baterias alcalinas Rayovac e Duracell. Diferentemente do manual do fabricante, é observado que o consumo de energia do MICAz não é constante. Além disso, é mostrado que as baterias utilizadas apresentam variações significativas na sua capacidade efetiva de carga. Isto ocorre tanto em relação aos valores informados pelos respectivos *datasheets*, como entre baterias do mesmo fabricante. Nos testes realizados, os valores foram monitorados ao longo do tempo até que o nó sensor parasse de transmitir, ou quando verificados erros grosseiros nos valores medidos (fator que pode ocorrer quando a tensão diminui). Os autores utilizam dois modelos de estimação da capacidade remanescente de baterias, são eles:

Modelo de Estimação Baseado na Tensão: utiliza os dados de tensão coletados ao longo tempo de vida do nó sensor. A estimativa da capacidade remanescente da bateria é dada pela Equação 5.1:

$$C_{res}(t) = a \cdot V(t) - b \tag{5.1}$$

em que:

$C_{res}(t)$ é a capacidade em "mAh" remanescente no instante t;

$V(t)$ é a tensão em "volts" da bateria no instante t;

a **e** b são constantes adimensionais, valores característicos de cada bateria;

t é o tempo medido em segundos.

Apesar de ser um modelo linear, este considera efeitos eletroquímicos não lineares, como o *efeito de relaxamento*[1].

Modelo de Estimação Baseado na Corrente: este modelo utiliza os dados de corrente coletados ao longo do tempo de vida do nó sensor. A estimativa da capacidade remanescente da bateria por este modelo é dada pelas Equações 5.2 e 5.3:

$$C_{res}(t) = C_{ef} - \int_o^t I(t)dt \tag{5.2}$$

$$C_{ef} = k \cdot C_{MAX} \tag{5.3}$$

em que:

$C_{res}(t)$ é a capacidade em "mAh" remanescente no instante t;

C_{MAX} é a capacidade máxima nominal em "mAh" da bateria;

C_{ef} é capacidade efetiva em "mAh" da bateria, função da corrente nominal de descarga;

k é um fator adimensional que considera a corrente de descarga da bateria;

I é a corrente nominal de descarga em "mA";

t é o tempo medido em segundos.

Ao contrário do modelo anterior, este não considera o efeito de relaxamento.

Jongerden & Haverkort (2008) mostram outro efeito interessante do comportamento de uma bateria: a *taxa de capacidade*. Este efeito mostra que as taxas de descargas dependem da capacidade residual da bateria e da intensidade da corrente de descarga. Além disto é mostrado também que as taxas de descargas não são lineares.

Os autores de (Schneider et al., 2011) apresentam uma análise comparativa entre dois modelos analíticos de descarga de baterias em um nó sensor da família Mica Motes: o modelo Linear e o modelo de Rakhmatov-Vrudhula (Jongerden & Haverkort, 2008; Rakhmatov & Vrudhula, 2003). O modelo de Rakhmatov-Vrudhula foi escolhido devido ao mesmo conseguir capturar o efeito de taxa de capacidade, o efeito de relaxamento e ser de fácil implementação. Os autores também concluem que o modelo linear não é o mais apropriado para modelar o consumo de uma bateria.

[1]Uma recuperação de carga que surge quando a bateria está com uma corrente de descarga alta e ocorre um corte ou redução desta corrente.

5.2 AJUSTES PARA SIMULAR O COMPORTAMENTO DE UMA BATERIA

Como visto na seção anterior, a carga remanescente de uma bateria não pode ser medida precisamente. Existem modelos analíticos que ajudam a modelar o comportamento do consumo ao longo do tempo. No entanto, devido aos muitos fatores que podem interferir no consumo da bateria, não é possível prever exatamente o momento em que a bateria de um sensor não terá carga suficiente para manter seu funcionamento dentro da normalidade. Na tentativa de aproximar o efeito causado por esta imprecisão, durante as simulações as cargas iniciais das baterias de cada sensor presente na rede são alteradas. Tal alteração serve para compensar basicamente duas imprecisões:

1. Valor real da carga inicial.
2. Consumo real ao longo do tempo e consequentemente o valor real da carga remanescente.

Durante a simulação *online* da rede, a carga inicial de cada sensor é ajustada segundo uma distribuição de probabilidade normal (Figura 5.1). Como pode ser visto na Figura 5.1 a curva tem média μ e desvio padrão σ. Desta forma a energia real (EB_{real}) presente na bateria de cada sensor tem maior probabilidade de estar na faixa de $\mu - \sigma \leq EB_{real} \leq \mu + \sigma$. No modelo aqui considerado:

EB a carga inicial da bateria fornecida pelo fabricante.

S conjunto de todos os sensores presentes na rede.

$S_i.EB_{real}$ o valor estimado da carga inicial do sensor i.

randn() um gerador de números aleatórios que retorna um número aleatório segundo uma função de distribuição de probabilidade normal tendo $\mu = 0$ e $\sigma = 1$, ou seja, média zero e desvio padrão igual a um.

O Algoritmo 11 é executado antes do início da simulação *online*.

Algoritmo 11 Pseudocodigo para ajuste de carga inicial da bateria de um sensor na simulação *online*

```
1: função EnBatAjust(SensList, EB, σ)
2:     para cada sensor i presente em S faça
3:         S_i.EB_real = EB + randn() × σ
4:     fim para
5: fim função
```

Para os algoritmos que são executados de forma centralizada pelo sorvedouro, as estimativas da energia residual de cada sensor são baseadas na energia inicial fornecida pelo fabricante (EB). Já para efeito de vida útil de cada nó sensor durante a simulação *online*, o valor da carga inicial é dado por EB_{real} calculado segundo a distribuição de probabilidade gaussiana. As diferenças entre EB e EB_{real} podem gerar falhas imprevistas

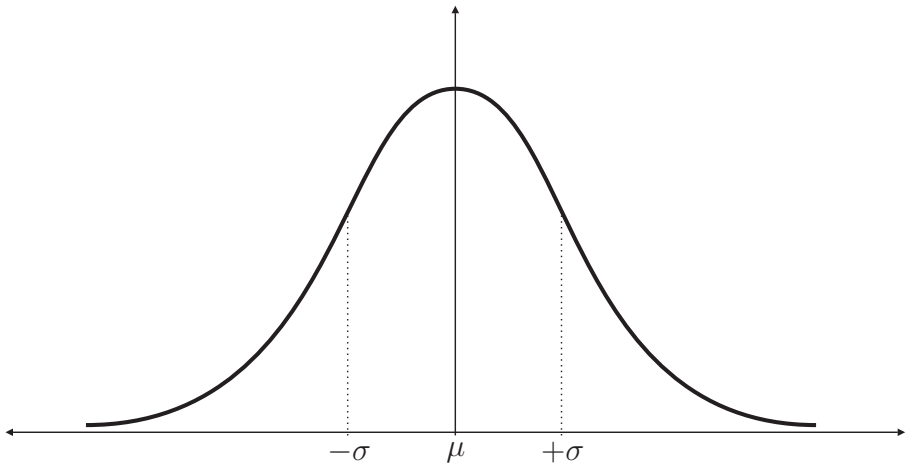


Figura 5.1 Exemplo da curva de distribuição normal utilizada para gerar EB_{real}.

em um nó sensor, pois estes podem ter um tempo maior ou menor de vida útil. Este fator pode prejudicar o *scheduling* que busca maximizar o tempo de vida da rede, realizado pelos algoritmos centralizados. A inserção deste ruído busca mostrar a robustez das soluções e também aproximar mais a simulação de um evento real.

5.3 ABORDAGEM COM TOLERÂNCIA A FALHAS INESPERADAS

Quando ocorre a falha de um sensor presente na rede, além da não cobertura da área monitorada exclusivamente por este sensor, toda informação antes passada pelo sensor falho durante o roteamento dos dados pode não mais ser entregue ao nó sorvedouro. No Capítulo 4 foi apresentado um algoritmo capaz de estender o tempo de vida de uma RSSF considerando falhas ocorridas por falta de energia nos nós sensores. O algoritmo foi modelado considerando que as falhas por falta de energia são previsíveis, no entanto como mostrado na Seção 5.1 isto nem sempre ocorre. Nesta seção será apresentado o esboço de um possível protocolo de tolerância a falhas, para agir juntamente com o algoritmo WSNdsGA apresentado no Capítulo 4, em situações em que falhas inesperadas ocorrem. Além deste protocolo, serão apresentadas três outras versões compostas por adaptações no próprio WSNdsGA de forma a torná-lo mais adequado a um funcionamento centralizado. As seguintes versões a serem apresentadas são:

- WSNdsGA-OS: Um algoritmo *Online* Sequencial (centralizado);
- WSNdsGA-OmGA: Um micro Algoritmo Genético *Online* (centralizado);
- WSNdsGA-OGA: Um Algoritmo Genético *Online* (centralizado);
- WSNdsGA-OP: Um algoritmo com restrição de Protocolo *Online* (centralizado e distribuído).

Todas estas versões utilizam o pré-processamento feito pelo algoritmo WSNdsGA para estimar qual seria o *scheduling* dos nós sensores presentes na rede caso não houvessem falhas inesperadas. De posse desta solução, é possível utilizar, sem necessidade de

adaptações, o algoritmo para avaliar uma solução (Algoritmo 9) fornecido na Seção 4.4.5, simulando assim o funcionamento de uma RSSF. Para aumentar a correspondência com uma possível situação real, ao utilizar o operador Inteligente de Desativação, a energia de ativação gasta por todos os sensores ativados no estágio corrente será computada, mesmo que este seja desativado por tal operador no mesmo estágio.

Para as ações que possuem característica centralizada, o sorvedouro é quem identifica o sensor falho e envia pela rede as ações de controle para que a rede seja reconfigurada. O sorvedouro pode identificar o sensor falho quando não mais receber os pacotes de dados que deveriam ser enviados por ele. Já as ações de controle são disseminadas pela rede através dos nós sensores ativos. Quando um sensor requisitado para ser ativado pertencer à lista de vizinhos de um nó ativo, este envia um sinal de ativação para tal sensor, conforme o Pressuposto 3.

Pressuposto 3. *A partir de agora, será assumido que, um sensor ativo é capaz de enviar uma mensagem de controle de ativação para qualquer um dos sensores presentes na sua lista de vizinhança. Além disto, também será assumido que um sensor desativado é capaz de identificar esta mensagem e alterar seu estado para ativado.* □

Na tentativa de aproximar as simulações de um caso real, em todos os ambientes de simulação *online* as cargas iniciais das baterias dos sensores são alteradas segundo o Algoritmo 11. Entretanto, para o nó sorvedouro a energia remanescente de cada nó sensor continua sendo a calculada pela Equação 3.8 mostrada no Capítulo 3, considerando o estágio atual e EB (carga inicial informada pelo fabricante).

Os resultados obtidos por cada abordagem serão mostrados no Capítulo de Resultados na Seção 6.4.

5.3.1 WSNdsGA-OS: Um algoritmo Online Sequencial centralizado

A solução do WSNdsGA fornece um escalonamento dos nós sensores, fornecendo o período de tempo em que estes devem permanecer ativos e os momentos que os mesmos devem ficar inativos. Todo controle pode ser feito de forma pré-processada ou até mesmo centralizada no nó sorvedouro. Como mostrado na Seção 5.1 uma bateria pode chegar ao fim antes ou até mesmo depois do tempo previsto de sua duração. Quando isto ocorre, a configuração de um escalonamento realizado com base em um pré-processamento pode não mais atender aos requisitos de cobertura e conectividade da rede.

Para ser possível o uso de um cálculo pré-processado em uma rede em que possa ocorrer falhas inesperadas, foi desenvolvido o WSNdsGA-OS. Este algoritmo serve para medir a robustez da solução fornecida pelo WSNdsGA em meio a falhas que podem ocorrer de forma inesperada, pois a sequência da solução não é alterada diretamente.

O WSNdsGA-OS segue as seguintes etapas:

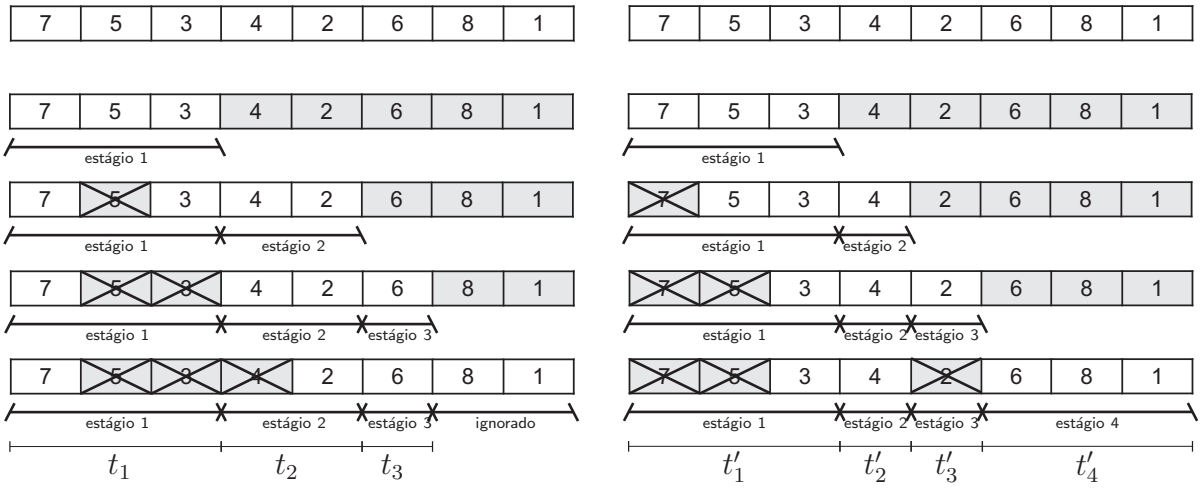
1. Inicia uma rede a partir de uma solução fornecida pelo pré-processamento através do WSNdsGA (considerando tanto a carga como o consumo das baterias presentes nos sensores totalmente previsíveis).

2. Para simular as falhas inesperadas, as cargas iniciais das baterias dos sensores são alteradas segundo o Algoritmo 11.

3. Então esta solução é simulada através do Algoritmo 9.

Como visto no Capítulo 4, o Algoritmo 9 é utilizado para avaliar uma solução do WSNdsGA, esta avaliação corresponde a uma simulação da rede projetada pelo algoritmo. Para determinar se a rede está atendendo os requisitos mínimos de cobertura e conectividade, durante a simulação mensagens são constantemente trocadas entre os nós ativos da rede e o sorvedouro. A partir destas mensagens é possível determinar o início de cada estágio. Quando o sorvedouro recebe a informação de uma falha, um novo estágio é iniciado. Ao iniciar, nós sensores são ativados para suprir a falha ocorrida. Os sensores a serem ativados são os presentes na sequência da solução. Em um ambiente em que não são consideradas as falhas inesperadas, a rede pode ser completamente previsível, ou seja, todos os estágios podem ser previamente definidos. Quando as falhas inesperadas são inseridas, tanto a duração, como quais sensores fazem parte de cada estágio, passam ser definidos a partir das mensagens trocadas entre a rede e o sorvedouro.

Para um melhor entendimento, na Figura 5.2 é mostrado um exemplo de decodificação de uma solução em dois ambientes diferentes. Na Figura 5.2(a) é mostrado uma simulação em um ambiente em que os nós sensores possuem cargas e consumo totalmente previsíveis, já na Figura 5.2(b), antes de iniciar a simulação da rede, as cargas iniciais dos sensores são alteradas através do Algoritmo 11. Com a alteração das baterias dos sensores, os estágios podem se alterar, pois podem ocorrer falhas antes ou depois dos momentos previstos. Como visto na Figura 5.2(b) o sequenciamento da solução não foi alterado, apenas os estágios em que os sensores foram ativados. No entanto, a duração de cada estágio e o tempo total de vida da rede podem se alterar. Vale a pena ressaltar que este exemplo se trata apenas de um exemplo didático, e que para facilitar o entendimento os Operadores Inteligentes de Ativação e Desativação não foram considerados.



(a) Sem falhas inesperadas (b) Com falhas inesperadas

Figura 5.2 Decodificação de uma solução pelo Algoritmo 9, sem aplicar os Operadores Inteligentes de Ativação e Desativação.

5.3.2 WSNdsGA-OmGA: Um micro Algoritmo Genético online centralizado

O micro Algoritmo Genético (mAG) sugerido inicialmente por Goldberg (1989b) e implementado pela primeira vez por Krishnakumar (1990), faz uso de um algoritmo genético com seus parâmetros de população e taxas de cruzamento e mutação reduzidos. Kazarlis et al. (2001) sugerem a generalização do mAG como um operador de busca local para o AG. Já em Takahashi & Martins (2004) é mostrado o potencial do mAG para buscas de tendência global.

Dado a eficiência do mAG tanto como operador de busca de tendência global como operador de busca local, uma versão deste algoritmo foi implementada para ser executada no sorvedouro quando este recebe um sinal de falha inesperada, ou seja, a falha de um sensor que não estava na sequência de falhas esperada pelo sorvedouro. A esta versão deu-se o nome de WSNdsGA-OmGA. Este algoritmo recebe como entrada a rede atual e retorna como resultado um novo *scheduling* com base na maximização do tempo de vida da rede. A solução corrente fornecida pelo WSNdsGA ou já atualizada pelo WSNdsGA-OmGA, é inserida na população inicial do mAG para o cálculo do novo *scheduling*, retirando os sensores que estão falhos no estado corrente.

Como os parâmetros utilizados no mAG são reduzidos, consequentemente o número de avaliações de função objetivo é bem menor, possibilitando fornecer uma resposta de maneira rápida. Para execução do WSNdsGA-OmGA é necessário estimar a carga da bateria de cada sensor no momento atual. Para isto, é utilizado como referência o valor inicial da bateria de acordo com o fabricante, sem alterações realizadas pelo Algoritmo 11. Considerando este o valor inicial, o valor da energia residual presente em cada nó sensor é estimada com base na Equação 3.8 mostrada no Capítulo 3 e o estágio atual. Para efeito de simulação, após executado o WSNdsGA-OmGA, o valor real da carga presente na bateria volta a ser o calculado pela Equação 3.8 tendo como referência a carga inicial calculada pela Equação 11 e o estágio corrente. Isto é necessário, pois como mostrado na Seção 5.1, na realidade o sorvedouro não tem como saber ao certo o valor da energia residual presente em cada sensor.

As etapas presentes no WSNdsGA-OmGA são:

- Inicia-se uma rede a partir de uma solução fornecida pelo pré-processamento através do WSNdsGA (considerando tanto a carga como o consumo das baterias presentes nos sensores totalmente previsíveis).
- Para simular as falhas inesperadas, as cargas iniciais das baterias dos sensores são alteradas segundo o Algoritmo 11.
- Esta solução é então simulada através do Algoritmo 12, onde:
  - $falhaEsperada$ é uma variável *booleana* que assume $true$ se a falha ocorrida na rede estava prevista no *scheduling*, e $false$ caso contrário.
  - WSNdsGA-OmGA é a chamada do mAG com parâmetros reduzidos a $\vartheta\%$ do tamanho da população e do número de avaliações de função objetivo do algoritmo genético original.

Algoritmo 12 Pseudocódigo para Simulação de uma Solução no WSNdsGA-OmGA

1: **função** EVAL-MGA($\mathbf{x}$, E_{max}, E_{real}, C)
2: $j \leftarrow 0$ ▷ sensor corrente
3: $c_{stg} \leftarrow 0$; ▷ estágio corrente
4: $lifetime \leftarrow 0$; ▷ tempo de vida
5: $cov_{num} \leftarrow 0$;
6: $E(0) = [E_{max}\ E_{max}\ \ldots\ E_{max}]'$;
7: $E_r(0) = [EB_{real}\ EB_{real}\ \ldots\ EB_{real}]'$; ▷ Algoritmo 11
8: $\mathcal{A}_s \leftarrow \emptyset$; ▷ conjunto dos sensores ativos
9: **enquanto** $j < |\mathcal{S}|$ **faça**
10: $\mathbf{x} \leftarrow \text{SMACT}\,(\mathbf{x}, \mathcal{A}_s, j, C)$
11: **enquanto** $G_{cov}[\mathcal{A}_s] < C \cdot |\mathcal{D}|$ e $j < |\mathcal{S}|$ **faça**
12: **ative** o sensor x_j e o **inclua** em $\mathcal{A}_s$;
13: $H_{x_j} \leftarrow c_{stg} + 1$
14: $j \leftarrow j + 1$;
15: **fim enquanto**
16: $[\mathbf{x}, \mathcal{A}_s, j] \leftarrow \text{SMDEACT}\,(\mathbf{x}, \mathcal{A}_s, j, C)$
17: **se** $G_{cov}[\mathcal{A}_s] \geq C \cdot |\mathcal{D}|$ **então**
18: **simule** a rede até que a primeira falha ocorra entre os nós sensores de $\mathcal{A}_s$;
19: $lifetime \leftarrow lifetime + F_{time}[\mathcal{A}_s, E(c_{stg})]$;
20: $cov_{num} \leftarrow cov_{num} + G_{cov}[\mathcal{A}_s] \cdot F_{time}[\mathcal{A}_s, E(c_{stg})]$;
21: **remova** os nós sensores falhos de $\mathcal{A}_s$;
22: **se** $falhaEsperada = false$ **então**
23: **atualize** $E(c_{stg})$;
24: $\mathbf{x}$ = WSNdsGA-OmGA($\mathcal{A}_s, E(c_{stg})$) ▷ parâmetros reduzidos
25: $j = 0$
26: **fim se**
27: $c_{stg} \leftarrow c_{stg} + 1$;
28: **atualize** $E(c_{stg})$;
29: **fim se**
30: **fim enquanto**
31: $coverage \leftarrow \dfrac{cov_{num}}{lifetime}$; ▷ média ponderada da cobertura
32: **retorno** *lifetime*, *coverage*, H.
33: **fim função**

5.3.3 WSNdsGA-OGA: Um Algoritmo Genético online centralizado

A implementação do WSNdsGA-OGA é semelhante à do WSNdsGA-mGA. A única diferença é a substituição do mAG (linha 25 do Algoritmo 12) pelo mesmo AG utilizado no WSNdsGA, inclusive com os mesmos parâmetros. Como se trata de um algoritmo muito mais custoso, pode não ser viável em algumas aplicações. No entanto, sua implementação neste caso foi realizada para que seus resultados sirvam como referência de comparação em relação aos resultados obtidos pelas demais abordagens *online*.

5.3.4 WSNdsGA-OP: Um algoritmo com restrição de protocolo online

Aqui será apresentado um esboço de um possível protocolo para tratar falhas em uma RSSF de forma distribuída. Este protocolo é implementado localmente, e um algoritmo centralizado é utilizado para reorganizar a rede de forma a buscar um maior tempo de vida da mesma.

O WSNdsGA-OP busca identificar a falha de um nó sensor através de trocas de mensagens de confirmação de recepção, através da camada de controle de acesso ao meio (MAC), semelhante ao protocolo PROC apresentado em (Macedo et al., 2005). No PROC um contador registra quantos ACKs (*acknowledgement frames*) (quadros de confirmação de recebimento do protocolo MAC) consecutivos foram perdidos. Quando o contador ultrapassa certo limiar, o nó sensor assume que seu pai falhou. No mesmo sentido, o WSNdsGA-OP faz uso dos seguintes pacotes de controle ACK-RTS-CTS (Figura 5.3).

ACK - (*Acknowledgement frame*): Confirmação de recebimento de pacotes de dados.

RTS - (*Request To Send*): Requisição para envio de pacotes de dados.

CTS - (*Clear To Send*): Resposta da requisição para envio de pacotes de dados.

Um contador registra quantos ACKs consecutivos não foram recebidos do sensor pai, para identificar se este falhou. Já o pai do possível sensor falho consegue identificar uma possível falha após uma quantidade de não envios de mensagens RTS deste sensor dentro do mesmo estágio da rede.

(a) O nó sensor A envia pacote RTS ao nó B que responde com pacote CTS informando que os dados podem ser enviados.

(b) O nó sensor A envia pacote RTS ao nó B, no entanto o sensor B não responde com pacote CTS. O sensor B falhou ou está impossibilitado para receber dados de A.

(c) Após o sensor A receber a confirmação CTS do sensor B, ele inicia a transmissão dos dados. Logo após a transmissão do pacote o sensor B envia um pacote ACK de confirmação de recebimento.

(d) Possivelmente devido à alguma falha do sensor B, este não envia o pacote ACK de confirmação de recebimento.

Figura 5.3 Exemplo dos pacotes de controle ACK-RTS-CTS. As cores dos sinais de cada sensor representam seu estado: azul: desativado, verde: ativado e vermelho: falho. A linha laranjada tracejada representa um "link" de comunicação estabelecido entre os sensores. (Legenda Figura 4.5)

Como mencionado existem duas formas de identificar um sensor falho, este pode ser identificado tanto por seus filhos como pelo seu pai. Os limiares são ajustados de forma que a identificação seja feita primeiramente pelo sensor pai, para que este possa enviar uma mensagem de *controle de ativação* (Pressuposto 3) para um de seus vizinhos,

tentando assim restaurar a cobertura e a conectividade da rede. Caso o sensor ativado não seja suficiente para que a cobertura e a conectividade sejam reestabelecidas, cada um dos sensores filhos do sensor falho também pode enviar mensagens de ativação para um de seus vizinhos.

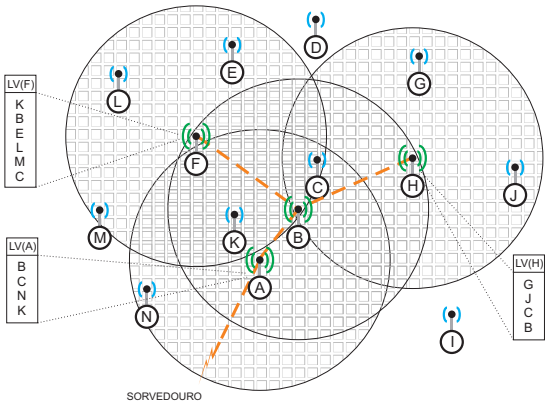
Nas Figuras 5.4 e 5.5 é possível visualizar dois exemplos do funcionamento do WSNdsGA-OP. Na Figura 5.4, quando o sensor A deixa de receber γ mensagens RTS do sensor B (Figura 5.4(b)), ele retira o sensor B de sua lista de vizinhos[2] e procura dentro desta mesma lista o sensor mais próximo do sensor B (sensor falho). Então é escolhido o sensor C, e este mantém-se ativo e passa a ser filho do sensor A (Figura 5.4(c)). Neste mesmo período de tempo, os sensores filhos do sensor B (F e H), deixam de receber ρ mensagens ACK do sensor B. Após $\rho + 1$ mensagens sem confirmação, os sensores F e H buscam dentre seus vizinhos se existe algum sensor ativo que eles possam utilizar para rotear os dados. No exemplo da Figura 5.4 o sensor ativado C estava presente na lista de vizinhos tanto de F como de H, sendo necessário apenas mudar a rota de comunicação (Figura 5.4(d)). No entanto, podem ocorrer situações como mostrado na Figura 5.5, em que nenhum sensor ativo pertence à lista de algum ou de todos os filhos do sensor falho. Neste caso, eles também enviam um sinal de ativação para o seu vizinho que é mais próximo do sensor falho B (Figura 5.5(d)). Este protocolo visa recuperar a conectividade e a cobertura perdida por um sensor que venha a falhar em uma RSSF.

Após a reestruturação local realizada pelo WSNdsGA-OP, são necessários ajustes na solução fornecida pelo WSNdsGA para que o sorvedouro possa prosseguir a decodificação da solução no processo de gerência da rede. O WSNdsGA-OP altera a topologia da rede ativando um ou mais nós sensores. O sorvedouro identifica os novos sensores presentes na rede e atualiza a sequência do *scheduling*. Um exemplo pode ser visto na Figura 5.6. Neste exemplo o sensor 5 falhou no estágio corrente e o sensor 6 foi escolhido pelo protocolo *online* para substituí-lo. A atualização da sequência é feita movendo o sensor 6 para o final do estágio corrente. Após a ação do protocolo *online* as novas informações de cobertura e conectividade podem ser calculadas pelo nó sorvedouro, e este pode tomar as seguintes decisões:

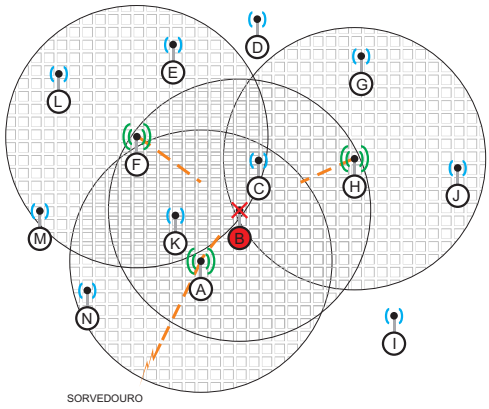
- Caso continuem sendo atendidos os requisitos de cobertura e conectividade, o estágio corrente permanece o mesmo e no instante de tempo seguinte é aplicado o operador de Desativação Inteligente (Algoritmo 8, Seção 4.4.4), na tentativa de eliminar redundâncias na rede.

- Se mesmo com a ação do WSNdsGA-OP os requisitos de cobertura e conectividade não forem reestabelecidos, a rede passa para o próximo estágio, continuando a sequência do *scheduling*.

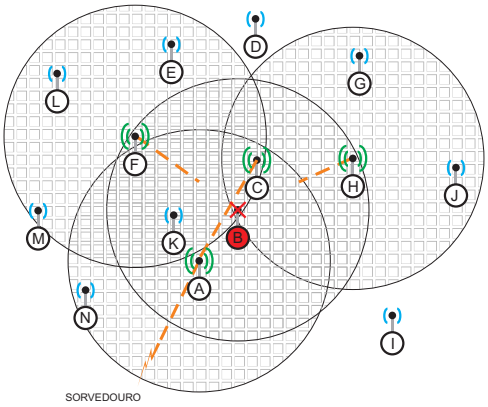
[2]Um sensor B é vizinho de um sensor A se é possível estabelecer comunicação entre eles.



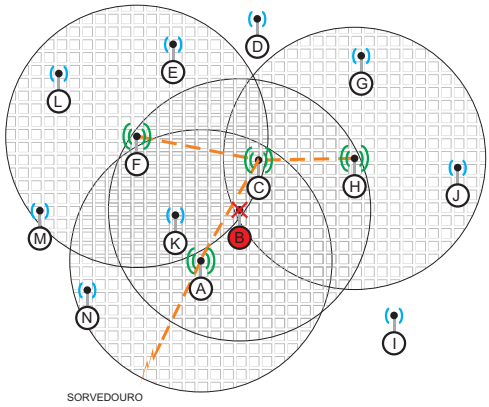
(a) Situação da rede antes de uma falha e lista dos nós sensores vizinhos. LV(X) representa a lista de vizinhos do nó X.



(b) Falha do nó sensor B e tentativa de comunicação dos nós sensores A, F e H.

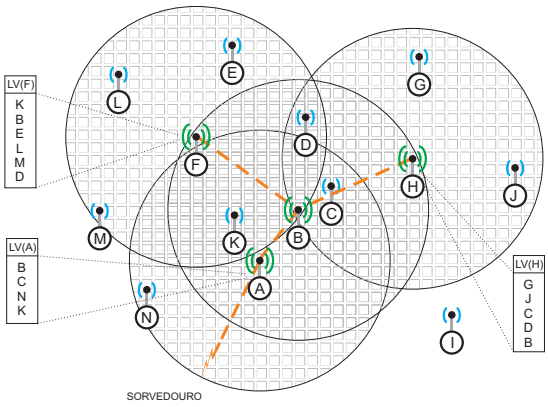


(c) Sinal de ativação enviado para o sensor C, após γ tentativas de comunicação com o sensor B.

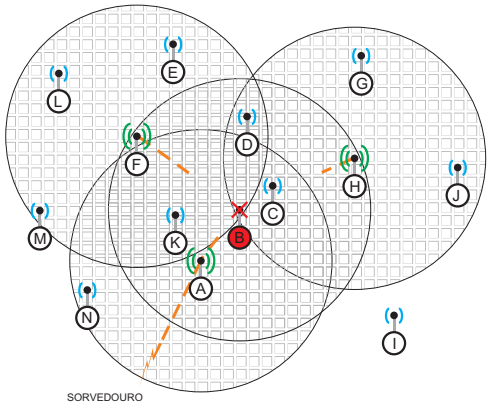


(d) Após $\rho + 1$ tentativas de comunicação com o sensor B, os sensores F e H encontram outras rotas para enviar suas mensagens.

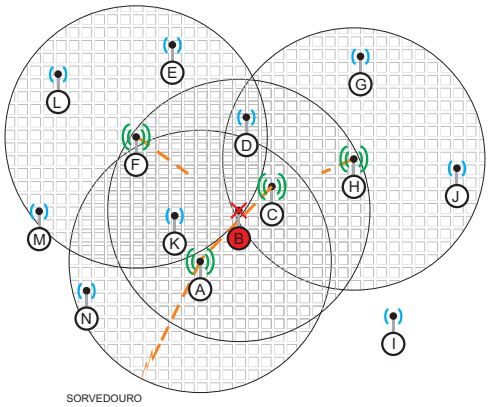
Figura 5.4 Exemplo A: funcionamento do protocolo WSNdsGA-OP. A área hachurada representa o raio de comunicação do sensor. As cores dos sinais de cada sensor representam seu estado: azul: desativado, verde: ativado e vermelho: falho. A linha laranjada tracejada representa um "link" de comunicação estabelecido entre os sensores. (Legenda Figura 4.5)



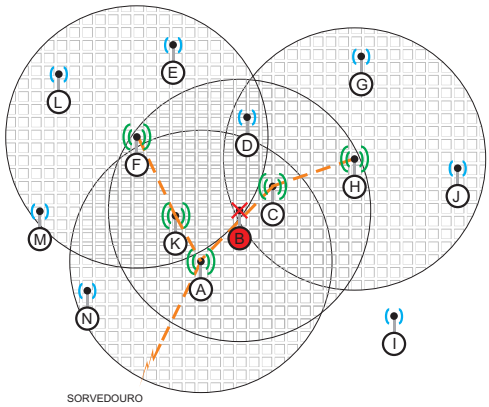
(a) Situação da rede antes de uma falha e lista dos nós sensores vizinhos. LV(X) representa a lista de vizinhos do nó X.



(b) Falha do nó sensor B e tentativa de comunicação dos nós sensores A, F e H.



(c) Sinal de ativação enviado para o sensor C, após γ tentativas de comunicação com o sensor B.



(d) Após $\rho + 1$ tentativas de comunicação com o sensor B, o sensor H encontra outra rota para enviar suas mensagens. Já o sensor F precisou enviar um sinal de ativação para o sensor K para reconstruir sua rota.

Figura 5.5 Exemplo B: funcionamento do protocolo WSNdsGA-OP. A área hachurada representa o raio de comunicação do sensor. As cores dos sinais de cada sensor representam seu estado: azul: desativado, verde: ativado e vermelho: falho. A linha laranjada tracejada representa um "link" de comunicação estabelecido entre os sensores. (Legenda Figura 4.5)

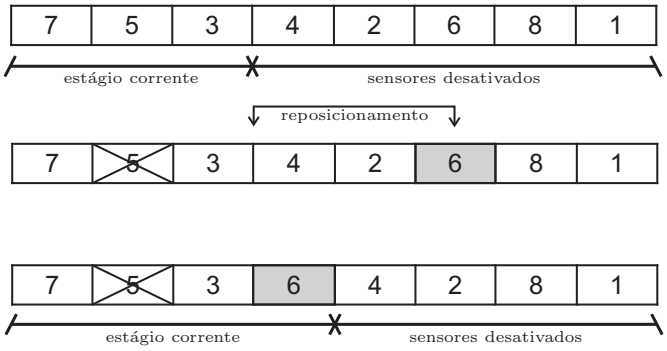


Figura 5.6 Exemplo da ação do protocolo *online*.

CAPÍTULO 6

RESULTADOS

Os testes apresentados se dividem basicamente em dois diferentes tipos de ambiente. Um com falhas previsíveis e outro com falhas prováveis.

Ambiente com Falhas Previsíveis: trata de um ambiente onde cada sensor da rede está sujeito apenas a falhas por falta de energia. Neste primeiro momento as falhas são fielmente previsíveis, ou seja, dadas as características de consumo de cada nó sensor, seu tempo em operação e sua carga inicial, é possível prever com exatidão o momento em que ele não terá mais energia disponível em sua bateria. Os resultados para este ambiente são apresentados nas Seções 6.2 e 6.3.

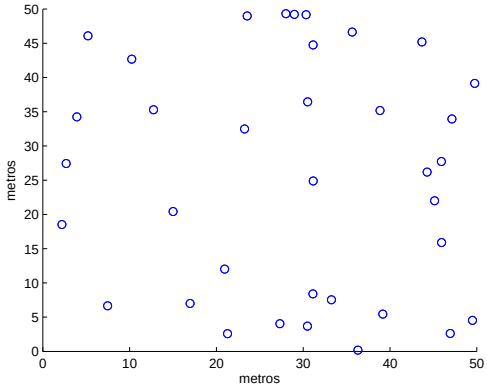
Ambiente com Falhas Prováveis: simula um ambiente em que o consumo da bateria não tem um comportamento regular e preciso. Dado as características de consumo de cada sensor, seu tempo em operação e sua carga inicial, uma previsão da vida útil do sensor é estimada. A partir desta previsão é construída a solução, a qual pode ser alterada caso algum evento não previsto ocorra. Resultados da simulação das RSSF neste ambiente são apresentados na Seção 6.4.

Dispondo da área a ser monitorada, primeiramente devem-se inserir os nós sensores nesta região. Uma das formas usuais é lançar estes sensores através de um avião, pois muitas vezes, estas áreas são de difícil acesso (Ruiz, 2003). Para simular a posição que cada sensor obteve nesta região, pontos nesta área foram gerados de forma aleatória. A distribuição usada foi a uniforme, ou seja, para cada ponto da área, a probabilidade de ter um sensor é a mesma. Na Figura 6.1 pode-se ver a distribuição aleatória, com distribuição uniforme, de 36, 49, 64, 81 e de 100 sensores em uma área de $2500m^2$. Observa-se que a instância de 49 sensores foi gerada a partir da instância de 36 sensores, ou seja, adicionando mais 13 sensores de forma aleatória e uniforme. As demais instâncias foram geradas da mesma forma, sempre adicionando os sensores na rede já existente. Decidiu-se fazer desta maneira, para que o testes sejam comparados de forma mais precisa, sendo possível então, uma identificação de forma mais correta, da influência da inserção de novos nós sensores em uma RSSF.

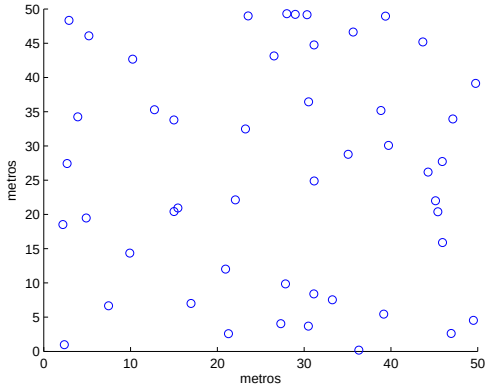
Todos os testes realizados a seguir correspondem a monitoração nesta área de $2500m^2$. Cada tipo de sensor possui características e parâmetros individuais, e todos os valores aqui apresentados, são baseados no nó sensor comercial Mica2dot (XBOW, 2006a). Tem-se os seguintes parâmetros para cada sensor da RSSF[1]:

$RS \leftarrow 15m$ (raio de sensoriamento).

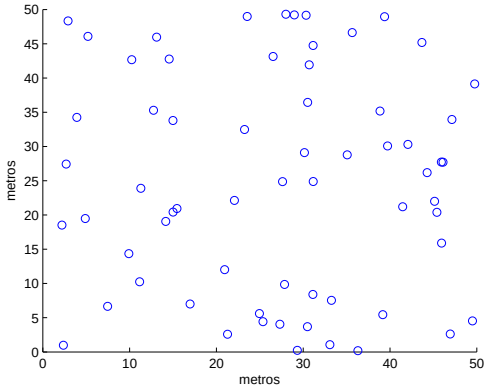
[1]Supondo fixada uma tensão na bateria, a energia armazenada ou retirada da bateria é proporcional à *corrente* x *tempo*, podendo ser medida na unidade A.h. A potência, por sua vez, é proporcional à corrente, podendo então ser medida em A.



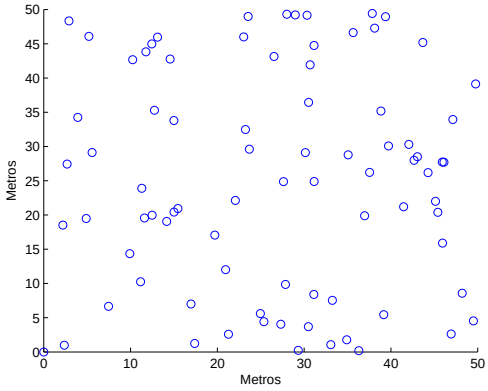
(a) Distribuição uniforme de 36 sensores



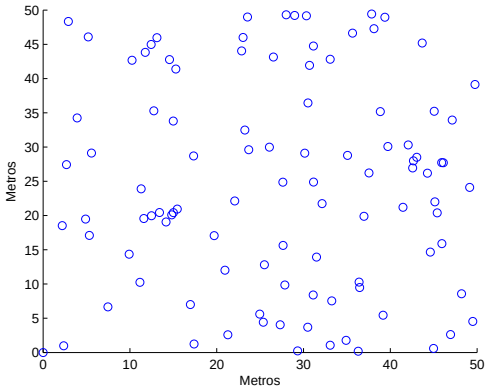
(b) Distribuição uniforme de 49 sensores



(c) Distribuição uniforme de 64 sensores



(d) Distribuição uniforme de 81 sensores



(e) Distribuição uniforme de 100 sensores

Figura 6.1 Ilustração da distribuição usada para 36, 49, 64, 81 e 100 sensores em uma área de $2500m^2$. As coordenadas dos sensores são mostradas no Apêndice C

Tabela 6.1 Consumo de corrente do nó sensor com a transmissão. (Nakamura, 2010)

Alcance (m)	**Corrente (mA)**
$[00,000; 05,142]$	$8,6$
$]05,142; 05,769]$	$8,8$
$]05,769; 07,263]$	$9,0$
$]07,263; 08,150]$	$9,1$
$]08,150; 10,260]$	$9,3$
$]10,260; 11,512]$	$9,5$
$]11,512; 12,916]$	$9,7$
$]12,916; 14,492]$	$9,9$
$]14,492; 16,261]$	$10,1$
$]14,492; 18,245]$	$10,4$
$]18,245; 20,471]$	$10,6$
$]20,471; 22,969]$	$10,8$
$]22,969; 25,000]$	$11,1$

$RC \leftarrow 25m$ (raio de comunicação).

$AE \leftarrow 5mAh$ (energia de ativação).

$MP \leftarrow 13mAh$ (potência de manutenção).

$RP \leftarrow 2mAh$ (potência de recepção).

$EB \leftarrow 1000mAh$ (energia da bateria).

$Tempo\ de\ transmissão \leftarrow 0,25u.t.$

A potência de transmissão (TP) é calculada com base na distância em que o nó sensor transmite os dados. Os valores respectivos da energia consumida de acordo com a distância podem ser vistos na Tabela 6.1.

A AE é retirada de cada sensor quando ele passa do estado de não ativo para o estado de ativo. A MP é a estimativa da soma das energias consumidas com processamento e manutenção do sensor; essa energia é gasta a cada unidade de tempo. Já a RP e TP são proporcionais à quantidade de dados que são recebidos e transmitidos respectivamente. Os valores para RP e TP descritos acima, correspondem respectivamente ao gasto para receber e transmitir um pacote de dados em uma unidade de tempo. Cada sensor coleta um pacote de dados a cada unidade de tempo e transmite o seu pacote, mais os pacotes que foram recebidos de seus filhos (sensores conectados a ele). Deve-se lembrar que cada sensor gasta apenas $\frac{1}{4}$ da unidade de tempo do seu ciclo de funcionamento transmitindo seus pacotes. Para todos os testes e simulações realizadas, um sensor foi considerado falho por falta de energia quando sua bateria atingiu um valor de carga igual ou inferior a 10% de sua carga inicial.

6.1 MODELO DE COMUNICAÇÃO

Para todos os testes realizados, não foram utilizados efetivamente um protocolo de roteamento para RSSF. O modelo de comunicação implementado foi simplificado, pois não era o foco do trabalho. Para estabelecer uma árvore de comunicação para que os dados pudessem ser disseminados pela rede, a rota de cada sensor foi calculada e estabelecida pelo caminho mínimo entre o nó sensor ativo e o nó sorvedouro, de forma que apenas sensores ativos pudessem pertencer à rota de comunicação.

A cada novo estágio da rede, uma nova rota de comunicação é calculada. Um grafo contendo os sensores ativos é gerado, sendo que existem arestas apenas entre os sensores ativos e para os quais a comunicação possa ser direta (sensores dentro do raio de comunicação). Para fins de cálculo do caminho mínimo (Dijkstra, 1959), o peso considerado nas arestas foi o gasto com transmissão de um pacote de dados entre os sensores.

6.2 ABORDAGEM MONO-OBJETIVO - WSNdsGA

Duas diferentes formas de analisar o desempenho do algoritmo mono-objetivo foram utilizadas:

- **Parte A:** Os resultados obtidos aplicando o algoritmo nas instâncias mostradas anteriormente são comparados com os resultados obtidos utilizando o Ilog CPLEX (CPLEX, 2006) para as mesmas instâncias. Para cada uma das instâncias o nível de cobertura adotado foi de 100% ($C = 1,00$).

- **Parte B:** O algoritmo mono-objetivo proposto é comparado com dois algoritmos já publicados e que fundamentaram esta tese; o MultiOnHA (Martins et al., 2011) e o PAWSN (Martins et al., 2010). Neste caso, os níveis de cobertura adotados foram de 70% e 95%, uma vez que estes foram os requisitos de cobertura adotados nas referências originais.

Tanto nos testes **Parte A**, como **Parte B**, o WSNdsGA foi executado 33 vezes para fins de caracterização estatística, pois se trata de um algoritmo não determinístico. Como mostrado na Seção 4.5, existem parâmetros que devem ser inicializados; os valores utilizados foram:

- $S_{pop} = 200$ (tamanho da população);
- $n_{evl} = 28.000$ (número de avaliações de função objetivo);
- $p_c = 0,90$ (taxa de cruzamento);
- $p_m = 0,10$ (taxa de mutação).

Os valores adotados foram obtidos após vários testes com diferentes combinações de parâmetros. Os testes de ajustes de parâmetros podem ser vistos no Apêndice B.

Em cada um dos testes os resultados obtidos foram avaliados utilizando o teste de hipótese conforme as Equações 6.1 e 6.2.

$$\begin{cases} H0 : \mu_a - T_{PLI} = 0 \\ H1 : \mu_a - T_{PLI} \neq 0 \end{cases} \tag{6.1}$$

$$\begin{cases} H0 : \mu_a - \mu_b = 0 \\ H1 : \mu_a - \mu_b \neq 0 \end{cases} \tag{6.2}$$

onde:

$H0$: representa a hipótese nula, ou seja, a hipótese que traduz a ausência do efeito que se quer verificar.

$H1$: representa a hipótese alternativa, ou seja, a hipótese que o investigador quer verificar.

μ_x : representa a média das amostras do algoritmo x.

T_{PLI} representa o resultado do modelo PLI utilizando o CPLEX.

Estes testes foram realizados utilizando o Teste do Sinal (Conover, 1980) e a significância foi corrigida utilizando a Correção de Bonferroni (Dunn, 1961). A significância global utilizada foi de $\alpha = 0,01$. O Teste do Sinal e a Correção de Bonferroni foram escolhidos pois são baseados em poucas suposições:

Teste do Sinal: O Teste do Sinal é um procedimento não paramétrico que pode ser utilizado para comparar a mediana de uma ou duas amostras. Neste teste, é construído um teste para a distribuição do sinal da diferença entre as amostras. O número observado da diferença de negativos (ou positivos) é utilizado para estimar o p-valor sob uma distribuição binomial. Tal teste baseia-se no pressuposto de que as amostras são i.i.d. (independentes e identicamente distribuídos).

Correção de Bonferroni: A Correção de Bonferroni é utilizada para ajustar, sob múltiplas comparações, a significância de cada teste estatístico, a fim de assegurar que a significância global não seja maior do que a estabelecida previamente. No teste de Bonferroni, a significância (α ou probabilidade de erro tipo I) utilizada em cada teste passa a ser α/n, considerando n testes realizados. Este método não requer nenhuma suposição adicional.

Esta construção faz com que o procedimento de comparação seja bem geral, isto é, ele pode ser aplicado a praticamente qualquer tipo de algoritmo de comparação estocástico sem violar as premissas do teste. Como desvantagem, estes métodos são consideravelmente menos poderosos do que os procedimentos paramétricos. Isso significa que a probabilidade de se rejeitar uma hipótese nula quando ela é falsa é menor quando este tipo de teste é empregado.

Os resultados obtidos pelo algoritmo WSNdsGA e pelos outros métodos a serem comparados, são mostrados a seguir.

6.2.1 Parte A - Comparação com o CPLEX ($C = 1,00$)

Em Martins et al. (2007); Park et al. (2001); Rajagopalan et al. (2005), o PDCC-RSSF é formulado como um modelo de Programação Linear Inteira (PLI). A modelagem adotada pode ser vista nas Equações (6.3) e (6.4).

$$\mathcal{U}^* = \arg\min_{\mathcal{U}} f_{cons}[\mathcal{U}] \tag{6.3}$$

$$\text{sujeito a: } \begin{cases} g_1''' \\ g_2 \\ g_3 \\ g_4 \\ g_5 \\ g_6 \\ g_7 \end{cases} \tag{6.4}$$

em que:

$$f_{cons}[\mathcal{U}] = \sum_{k=1}^{n_{stg}} \sum_{i \in \mathcal{A}_s} e_i^{cons}(k) \tag{6.5}$$

$$g_1''' \ : \ h_j(k) > 0 \qquad \forall j \in \mathcal{D} \ , \ k \leq n_{stg} \tag{6.6}$$

$$\mathcal{A}_s = \{X(k) + U(k) + Z(k)\} \tag{6.7}$$

e $h_j(k)$ é uma variável binária que recebe valor 1 quando o ponto de demanda j é coberto no estágio k e 0 caso contrário.

Nas declarações acima, a função objetivo tem como objetivo minimizar a soma da energia total consumida nos nós sensores em cada estágio. Além disto, o modelo requer que em cada estágio, todos os pontos de demanda sejam atendidos por pelo menos um nó sensor ativo (Restrição g_1'''). Esta formulação é usualmente adotada pois o problema resultante é linear inteira, e portanto pode ser resolvido utilizando ferramentas para PLI, tais como as usadas pelo CPLEX (CPLEX, 2006)[2].

Sob o ponto de vista do problema prático, as Equações (6.3) e (6.4) podem ser interpretadas como uma aproximação da formulação proposta no Capítulo 3. O objetivo real do problema, que é a maximização do tempo de vida da rede, é substituído pelo objetivo de minimizar a energia consumida, que é fácil de expressar como uma função linear, adequada para ser resolvida com um pacote de programação linear inteira. Além disto, outra diferença pode ocorrer no processo de solução do PLI. O acoplamento entre os estágios pode ou não totalmente considerado. Para lidar com o sistema dinâmico, pode ser resolvido um único problema de PLI multiestágio, com um número de estágios pré-definidos e todos com a mesma duração. Outra alternativa, aqui adotada, é resolver um PLI para cada estágio, com duração fixa ou não, correspondente a um problema estático pra cada estágio. Neste caso, os estágios são otimizados do primeiro para o último em uma abordagem gulosa, não considerando a interação entre os estágios.

[2]Tais ferramentas asseguram que o ótimo global é alcançado, no entanto o custo computacional é, em geral, muito elevado.

Dado que a ferramenta de otimização adotada neste trabalho é um algoritmo evolutivo, a função objetivo e as restrições não necessitam obrigatoriamente ser lineares. Portanto, as Equações (3.18) e (3.19) apresentadas no Capítulo 3, que têm exatamente o tempo de vida da rede como função objetivo, podem ser adotadas no algoritmo proposto.

Outra característica interessante do WSNdsGA é a sua capacidade de lidar diretamente com qualquer nível de cobertura no intervalo $C = [0.00, 1.00]$. O relaxamento da restrição de cobertura total pode ser desejável em algumas aplicações. Em muitos casos, um agente gerenciador da rede provavelmente irá preferir aumentar o tempo de vida da rede em detrimento de certo grau de cobertura. Os resultados aqui apresentados mostram que pequenas reduções na cobertura podem levar a um grande ganho na vida útil da rede. Por fim, não deve ser ignorado que os resolvedores de PLI demandam um tempo computacional elevado, o que pode tornar inviável o seu uso para problemas com grande quantidade de nós sensores.

Nesta seção o WSNdsGA é comparado com a abordagem PLI resolvida pelo pacote CPLEX, estágio a estágio. Para todas as instâncias é adotada como limite a cobertura mínima de 100%. O algoritmo proposto é desenvolvido para resolver as Equações (3.18) e (3.19), com $C = 1.00$. Já o CPLEX busca resolver o modelo linear mostrado nas Equações (6.3) e (6.4). Embora os métodos utilizados para resolver o problema possuam formulações diferentes, a comparação entre eles pode ser considerada significativa, desde que seja utilizado um objetivo adequado para comparação: Tanto a formulação apresentada nas Equações (3.18) e (3.19), quanto a formulação por PLI apresentada nas Equações (6.3) e (6.4), têm como objetivo principal maximizar o tempo de vida da rede.

Os resultados obtidos pelo algoritmo proposto e pela abordagem PLI são mostrados nas Figuras 6.2, 6.3, 6.4, 6.5 e 6.6. Três soluções alcançadas pelo WSNdsGA são mostradas nas figuras: o pior caso (linha azul pontilhada), o melhor caso (linha tracejada preta) e o caso médio (círculos verdes), tais soluções foram obtidas após 33 execuções do algoritmo para cada instância. Como o PLI é resolvido deterministicamente, basta mostrar uma solução (linha rosa). A unidade de tempo final em que cada solução conseguiu permanecer com o nível de cobertura em 100% pode ser visto na Tabela 6.2.

Com base na Tabela 6.2 e nas Figuras 6.2, 6.3, 6.4, 6.5 e 6.6, é possível afirmar que o WSNdsGA superou o PLI (resolvido pelo CPLEX) em todas as instâncias quando comparado o tempo de vida da rede. Considerando que a hipótese nula (Equação 6.1) pode ser rejeitada se o respectivo p-valor for menor que 0,002 ($0,01/5$ testes), então, a rejeição das cinco hipóteses nulas apoiam a hipótese de que as diferenças observadas são estatisticamente significativas. Uma análise ingênua de tais resultados poderia sugerir que tais resultados são inconsistentes, uma vez que se tem um algoritmo heurístico com desempenho melhor do que um método exato. No entanto, deve-se observar que os algoritmos foram desenvolvidos para resolver duas formulações diferentes do PDCC-RSSF. Uma vez que não era necessário considerar apenas funções lineares ao resolver o problema por meio de um algoritmo genético, o modelo apresentado no Capítulo 3 e resolvido pelo WSNdsGA se mostrou mais aderente ao objetivo final de estender o tempo de vida da rede, atingindo consequentemente melhores resultados. E quanto maior o número de sensores maior o tempo de vida.

Apesar de atingir a otimalidade da função objetivo de gasto de energia, estágio a

Tabela 6.2 Algoritmo mono-objetivo – Resultados A: tempo de vida de cada solução (em *u.t.*)

	WSNdsGA								
nSens	$q_{0.00}$	$q_{0.25}$	$q_{0.50}$	$q_{0.75}$	$q_{1.00}$	PLI	p_{val}	T_{WSN}	T_{PLI}
36	56	56	57	57	57	45	$< \mathbf{10^{-8}}$	66	38
49	110	116	118	120	121	85	$< \mathbf{10^{-8}}$	155	134
64	132	139	145	145	150	100	$< \mathbf{10^{-8}}$	229	677
81	135	153	164	168	180	110	$< \mathbf{10^{-8}}$	279	2194
100	162	173	179	180	180	114	$< \mathbf{10^{-8}}$	374	9972

onde:

- nSens: representa a instância e a quantidade de sensores;
- $q_{0.00}$ é o pior resultado atingido pelo WSNdsGA (em *u.t.*);
- $q_{0.25}$ é o primeiro quartil dos resultados atingidos pelo WSNdsGA (em *u.t.*);
- $q_{0.50}$ é a mediana dos resultados atingidos pelo WSNdsGA (em *u.t.*);
- $q_{0.75}$ é o terceiro quartil dos resultados atingidos pelo WSNdsGA (em *u.t.*);
- $q_{1.00}$ é o melhor resultado atingido pelo WSNdsGA (em *u.t.*);
- PLI: indica resultados atingidos por PLI (em *u.t.*);
- p_{val} é o p-valor observado no teste estatístico;
- T_{WSN} é o tempo médio, em segundos, necessário para realizar uma execução do algoritmo WSNdsGA;
- T_{PLI} é o tempo de execução do PLI[a].

[a] T_{WSN} e T_{PLI} correspondem ao tempo de processador, utilizando apenas um núcleo, em simulações realizadas em um computador com processador Intel(R) Xeon(R) 2.00GHz 64bits e 8GB RAM DDR3 1067MHZ. Todas as Implementações foram em JAVA.

estágio (não considera a otimalidade considerando todos os estágios simultaneamente), a solução obtida pelo PLI apresenta um desempenho inferior quando comparado o tempo de vida da rede. A menor diferença é atingida na instância de 36 sensores, nela o WSNdsGA permanece por pelo menos 21% mais unidades de tempo do que a solução por PLI. Já quando comparado o pior caso da instância de 100 sensores, o resultado atingido pelo algoritmo proposto chega ser mais de 40% superior. Esta diferença está relacionada com a formulação apresentada nas Equações (6.3) e (6.4), pois ela se mostra ineficiente para expressar o objetivo de estender o tempo de vida da rede, e também por ser uma formulação não dinâmica para otimizar a sequência dos estágios.

Alguns aspectos que justificam tais diferenças significativas podem ser evidenciados nas Figuras 6.7 e 6.8. Estas figuras mostram a energia consumida e a energia residual

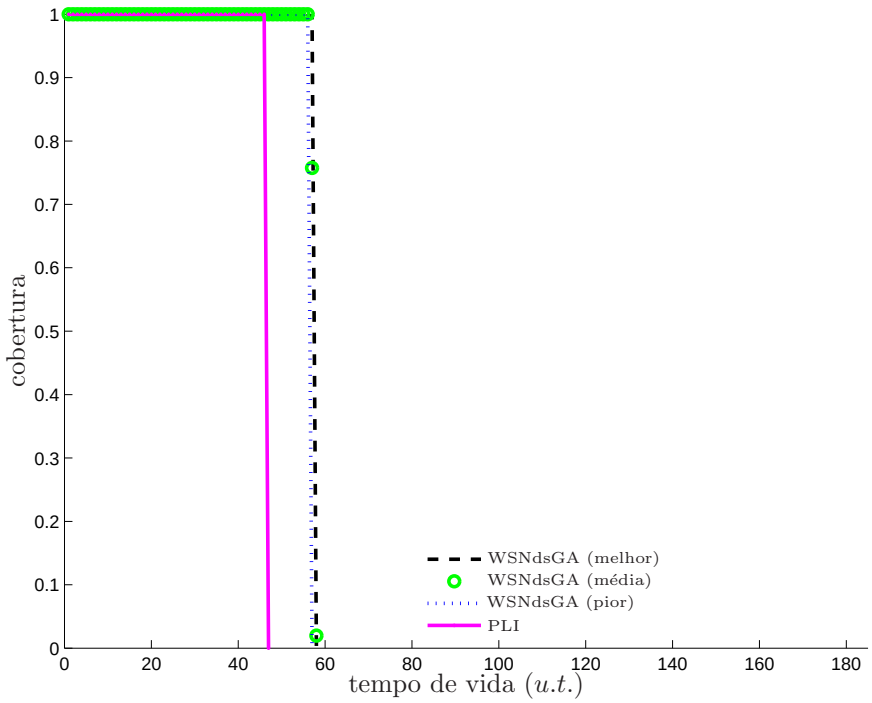


Figura 6.2 Algoritmo mono-objetivo – Resultados A: 36 sensores: cobertura × tempo de vida

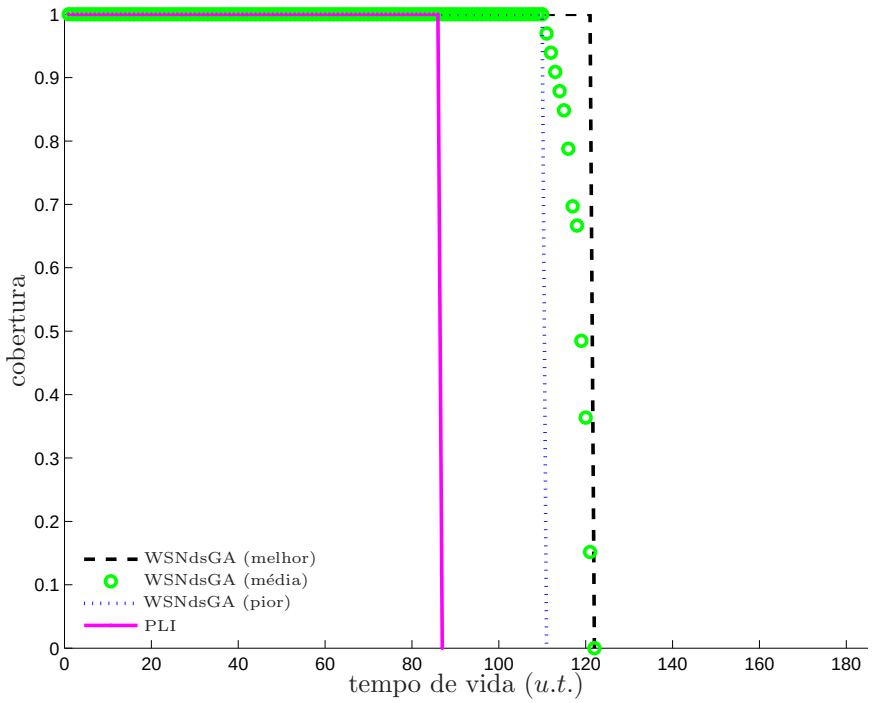


Figura 6.3 Algoritmo mono-objetivo – Resultados A: 49 sensores: cobertura × tempo de vida

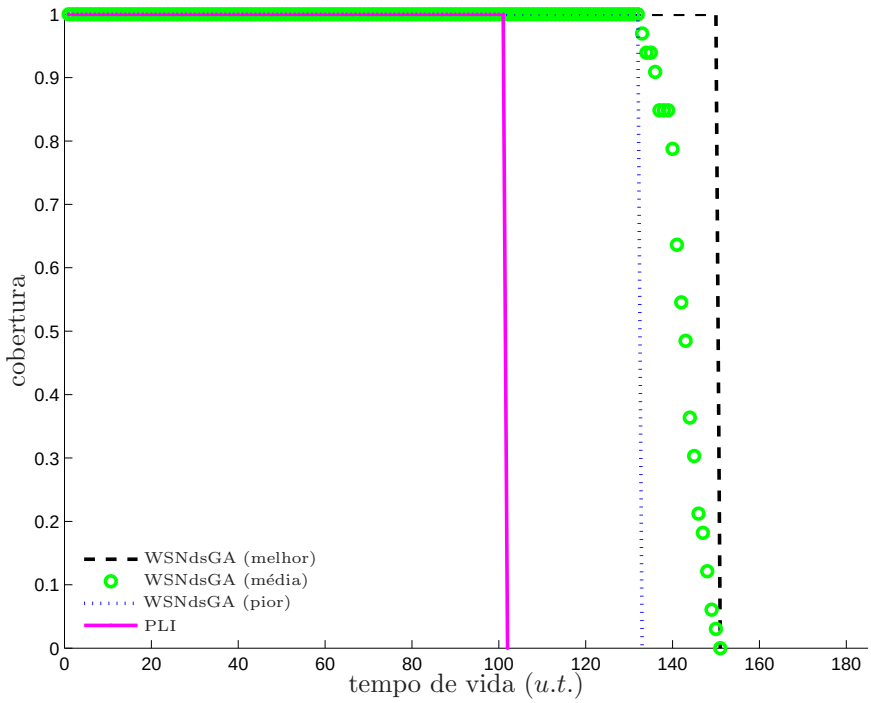


Figura 6.4 Algoritmo mono-objetivo – Resultados A: 64 sensores: cobertura × tempo de vida

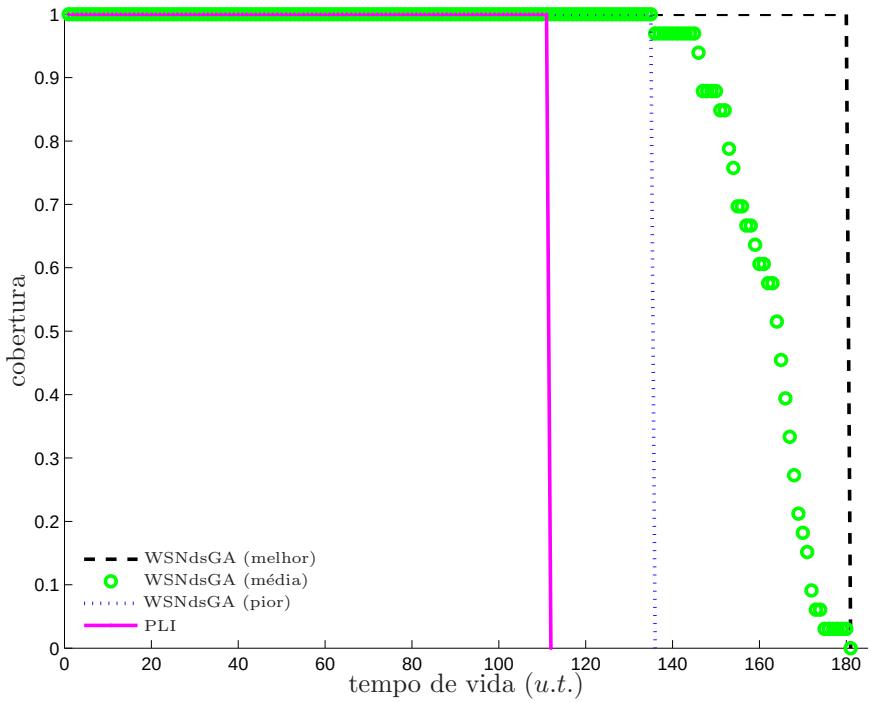


Figura 6.5 Algoritmo mono-objetivo – Resultados A: 81 sensores: cobertura × tempo de vida

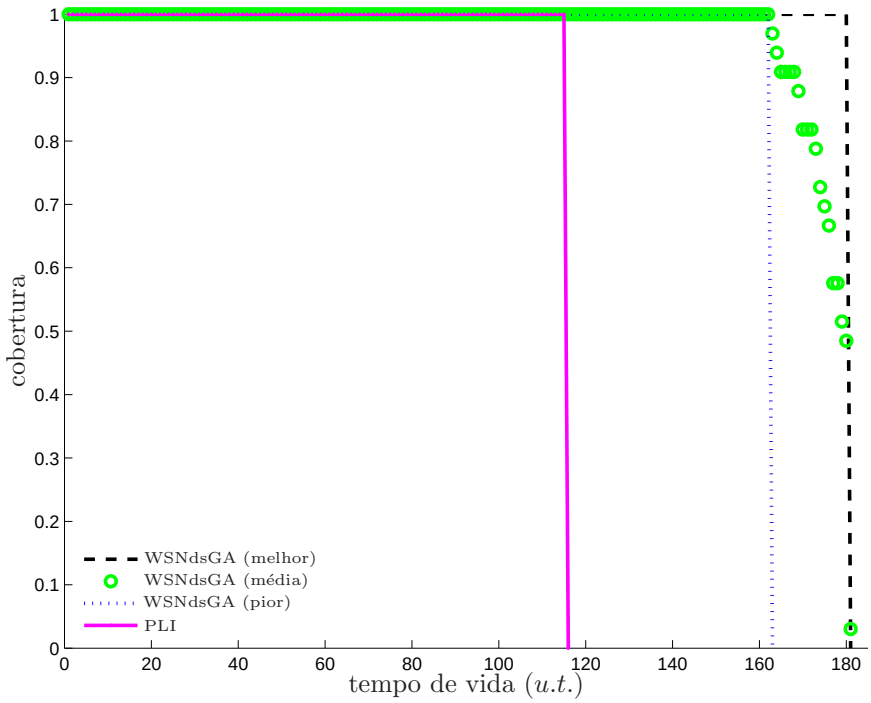


Figura 6.6 Algoritmo mono-objetivo – Resultados A: 100 sensores: cobertura × tempo de vida

ao longo da vida útil da rede para a instância de 100 nós sensores. A solução do PLI consome menos energia até 115 *u.t.*, e a rede torna-se inativa a partir deste momento devido à sua incapacidade de cumprir os requisitos de cobertura e conectividade. Isto está de acordo com o que seria esperado, pois o PLI busca atingir menores valores de energia consumida, uma vez que esta é a função objetivo otimizada pela sua formulação. A Figura 6.8 mostra que há uma grande quantidade de energia residual presente na rede após 115 *u.t.*, o que indica que alguns nós sensores ainda têm carga, mas estes nós não podem ser utilizados para construir uma rede viável, ou seja, atendendo os requisitos de cobertura e conectividade.

Diferente da solução apresentada pelo PLI, o WSNdsGA gasta mais energia ao longo da vida útil da rede, mas mantém o funcionamento da rede por mais tempo. A energia residual da rede em tais soluções é muito menor quando estas se tornam indisponíveis, o que indica que o algoritmo proposto fornece uma melhor gestão da capacidade de energia disponível na rede.

A partir dos tempos computacionais, T_{WSN} e T_{PLI}, mostrados na Tabela 6.2, é traçado um gráfico (Figura 6.9) no qual mostra o crescimento destes valores. Dado a escala logarítmica do gráfico, fica evidente que o PLI apresenta um crescimento computacional extremamente mais elevado. Conforme visto na Tabela 6.2, para a instância com 100 nós sensores, o WSNdsGA chega ser mais de 26 vezes mais rápido. Considerando que mais de 90% do tempo de processamento do WSNdsGA é gasto com as 28.000 chamadas de avaliação de função objetivo, o tempo requerido pelo algoritmo pode ser drasticamente

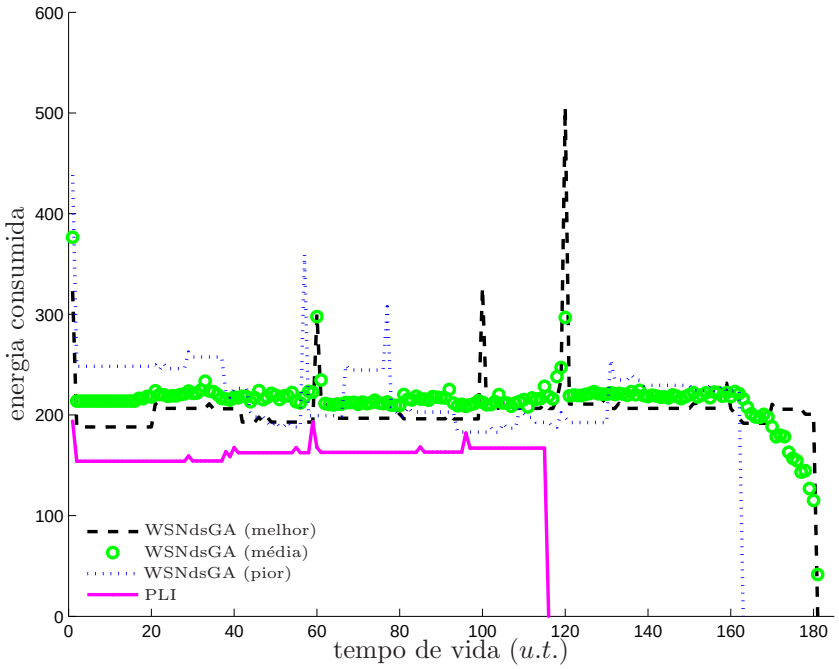


Figura 6.7 Algoritmo mono-objetivo – Resultados A: 100 sensores: energia consumida × tempo de vida

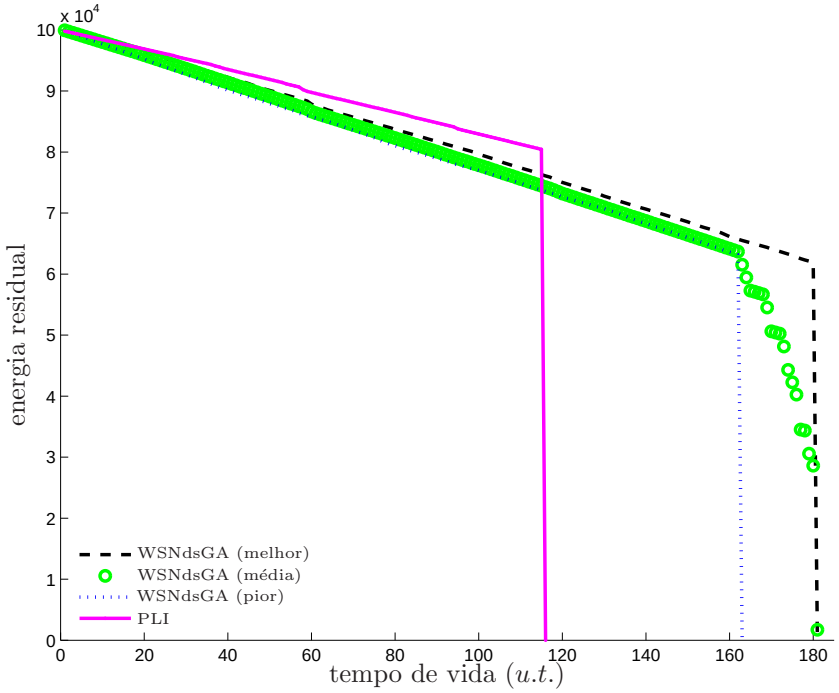


Figura 6.8 Algoritmo mono-objetivo – Resultados A: 100 sensores: energia residual × tempo de vida

reduzido, para isto, basta efetuar estas avaliações em paralelo, em um modelo síncrono.

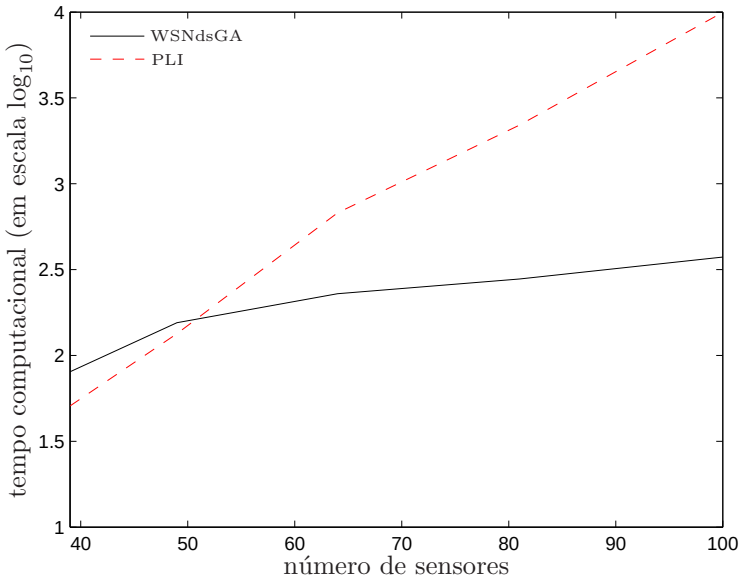


Figura 6.9 Algoritmo mono-objetivo – Resultados A: Média do tempo computacional: WSNdsGA × PLI

6.2.2 Parte B - Comparação com a Literatura ($C = 0,70$ e $C = 0,95$)

Como discutido anteriormente neste capítulo, o WSNdsGA pode ser utilizado com diferentes níveis de cobertura C, sendo $C \leq 1$. Esta característica torna o algoritmo adequado para situações práticas em que não é necessário assegurar a cobertura total da área de interesse.

O algoritmo WSNdsGA é comparado com outros dois algoritmos evolutivos publicados anteriormente e que fundamentaram esta tese, destinados a resolver o PDCC-RSSF. Abaixo é feita uma pequena descrição de cada abordagem:

- Em Martins et al. (2011), os autores propuseram um Algoritmo *Online* Híbrido Multiobjetivo (*Multiobjective Online Hybrid Algorithm* - MultiOnHA), em que cada estágio é resolvido de forma individual, ora utilizando um algoritmo genético (estratégia global), ora através de um algoritmo de busca local (estratégia local). A escolha da estratégia a ser adotada em cada estágio dependia do número de nós sensores que tinham seus estados alterados em relação ao estágio anterior. A modelagem utilizada pelos autores foi a mesma apresentada nas Equações (6.3) e (6.4), tendo a restrição g_1''' (cobertura) relaxada como um objetivo a ser maximizado. O algoritmo apresenta como solução final um conjunto de soluções eficientes, que têm desempenhos diferentes no que diz respeito aos requisitos de cobertura e energia consumida.

- Em Martins et al. (2010), os autores propuseram um Algoritmo Pré-processado para Redes de Sensores sem Fio (*Preprocessed Algorithm for Wireless Sensor Networks* - PAWSN), cujo o objetivo é maximizar o tempo de vida da rede. Tal algoritmo busca resolver a mesma declaração do problema apresentado na modelagem proposta no Capítulo 3. As principais diferenças implementadas no WSNdsGA em relação ao PAWSN são:

  – O operador de mutação baseado em subespaços;

  – O operador de Desativação Inteligente (*Smart Deactivation Operator*) (implementado para a operação mingr$(\cdot)$);

  – O operador de Cruzamento Real-Polarizado (CRP).

  Em termos das entidades geométricas, o PAWSN não contava com um operador de mutação em $\mathcal{L}^{\eta}(\mathbf{x}, \mathcal{Q}^{\perp})$, nem com um operador de busca local comprometido com a exploração do subespaço complementar $\mathcal{L}^{*}(\mathbf{x}, \mathcal{Q}^{\oslash})$. Além disso, o PAWSN também não foi desenvolvido utilizando o operador de pesquisa de descida CRP.

Até onde é de conhecimento do autor, as referências (Martins et al., 2010, 2011) constituem o estado-da-arte sobre o tema, considerando as abordagens heurísticas. Eles superaram outros algoritmos em problemas de um único objetivo e são as únicas referências disponíveis na abordagem multiobjetivo.

O WSNdsGA foi executado para dois diferentes níveis de cobertura: 70% e 95%. Estes níveis foram escolhidos para estabelecer uma comparabilidade entre os métodos, dado que Martins et al. (2011) apresenta soluções com níveis de cobertura 70% e 95% e o PAWSN apresentado em Martins et al. (2010) mostra resultados com níveis de cobertura em 95%. A média dos resultados atingidos por estes três métodos na instância com 100 nós sensores é mostrado na Figura 6.10. O tempo de vida de tais soluções é mostrado na Tabela 6.3.

Estes resultados mostram que o WSNdsGA superou os outros métodos para níveis de cobertura equivalentes. Além disso, a solução obtida pelo método proposto para $C = 0,95$ dura mais tempo do que a solução obtida por MultiOnHA para $C = 0,70$. Tais conclusões podem ser confirmadas pelos testes estatísticos mostrados na Tabela 6.4. Quando realizada uma comparação entre a Tabela 6.3 e a Tabela 6.2, é possível ver que a redução da cobertura pode aumentar o tempo de vida da rede consideravelmente. Para soluções relaxando 5% ($C = 0,95$) e 30% ($C = 0,70$) de cobertura é possível atingir, em média, resultados superiores em 213,1% e 306,3% unidades de tempo respectivamente.

Baseado na Tabela 6.3, é possível notar que o MultiOnHA possui um tempo computacional inferior aos apresentados pelos algoritmos WSNdsGA e PAWSN. Entretanto, o MultiOnHA necessita ser executado durante a operação da rede. Os demais algoritmos apresentados são executados de forma pré-processada, antes do início efetivo da rede, sendo este tempo não muito relevante. Já a diferença entre os tempos gastos pelo PAWSN e o WSNdsGA é justificado pelos novos operadores introduzidos nesta nova versão. Ao analisar os tempos computacionais do algoritmo WSNdsGA, nota-se que o esforço para resolver um problema com a exigência de cobertura de pelo menos 70% foi maior.Tal

Tabela 6.3 Algoritmo mono-objetivo – Resultados B: 100 sensores: tempo de vida de cada solução (em $u.t.$)

Algoritmo	$q_{0.00}$	$q_{0.25}$	$q_{0.50}$	$q_{0.75}$	$q_{1.00}$	T
WSNdsGA-95	354	369	376	380	387	611
WSNdsGA-70	525	535	539	542	552	737
PAWSN-95	269	279	284	286	296	380
MultiOnHA-95	239	266	280	286	299	15
MultiOnHA-70	262	327	348	361	397	13

onde:

- $q_{0.00}$ é o pior resultado atingido por cada algoritmo (em $u.t.$);
- $q_{0.25}$ é o primeiro quartil dos resultados atingidos por cada algoritmo (em $u.t.$);
- $q_{0.50}$ é a mediana dos resultados atingidos por cada algoritmo (em $u.t.$);
- $q_{0.75}$ é o terceiro quartil dos resultados atingidos por cada algoritmo (em $u.t.$);
- $q_{1.00}$ é o melhor resultado atingido por cada algoritmo (em $u.t.$);
- T é o tempo médio, em segundos, necessário para realizar uma execução de cada algoritmo[a].

[a] T corresponde ao tempo de processador, utilizando apenas um núcleo, em simulações realizadas em um computador com processador Intel(R) Xeon(R) 2.00GHz 64bits e 8GB RAM DDR3 1067MHZ. Todas as Implementações foram em JAVA.

Tabela 6.4 Algoritmo mono-objetivo – Resultados B: p-valores[b]

		p-valor
WSNdsGA 70% *vs.* WSNdsGA 95%	:	$< \mathbf{10^{-9}}$
WSNdsGA 70% *vs.* MultiOnHA 70%	:	$< \mathbf{10^{-9}}$
WSNdsGA 70% *vs.* MultiOnHA 95%	:	$< \mathbf{10^{-9}}$
WSNdsGA 70% *vs.* PAWSN 70%	:	$< \mathbf{10^{-9}}$
WSNdsGA 95% *vs.* MultiOnHA 70%	:	$< \mathbf{10^{-4}}$
WSNdsGA 95% *vs.* MultiOnHA 95%	:	$< \mathbf{10^{-9}}$
WSNdsGA 95% *vs.* PAWSN 70%	:	$< \mathbf{10^{-9}}$
MultiOnHA 70% *vs.* MultiOnHA 95%	:	$< \mathbf{10^{-8}}$
MultiOnHA 70% *vs.* PAWSN 70%	:	$< \mathbf{10^{-8}}$
MultiOnHA 95% *vs.* PAWSN 70%	:	0.3616

[b] A significância de cada teste foi ajustada para 0,001 (0,01 / 10 testes).

fenômeno ocorre pois a expectativa do tempo de vida de uma rede passa a ser maior quando a exigência de cobertura é reduzida. Sabe-se que grande parte do esforço com-

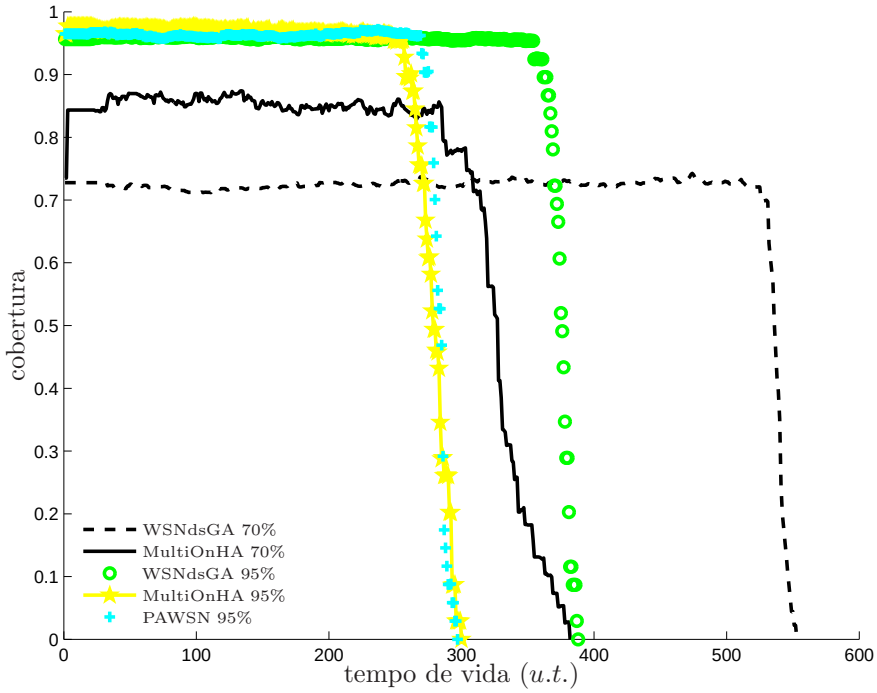


Figura 6.10 Algoritmo mono-objetivo – Resultados B: 100 sensores: cobertura × tempo de vida

putacional do algoritmo está concentrado na avaliação da função objetivo do problema. Como a função objetivo está diretamente ligada ao tempo de vida da rede, é de se esperar que o tempo computacional seja superior ao reduzir a exigência da cobertura.

6.3 ABORDAGEM MULTIOBJETIVO - MOWSNdsGA

O MOWSNdsGA é utilizado para resolver a formulação multiobjetivo mostrada ao longo do Capítulo 4. Devido ao comportamento ser bem semelhante em todas as instâncias, serão apresentados os resultados apenas para a instância de 100 nós sensores. A variável de cobertura C_i foi definida de modo a aceitar os valores no intervalo $[0,50,\ldots,1,00]$, com intervalos de $0,05$. Redes com cobertura mínima abaixo de 50% não foram consideradas. O tamanho escolhido para o intervalo não foi tão pequeno para evitar a ocorrência de soluções excessivamente semelhantes no conjunto Pareto-ótimo.

Novamente o MultiOnHA e o PAWSN foram utilizados como base de comparação para avaliar o algoritmo proposto. Os resultados aqui apresentados são organizados da seguinte maneira:

- **Parte A:** Em primeiro lugar, as soluções fornecidas por cada um dos três algoritmos para os níveis de cobertura 70% e 95% são comparados.

- **Parte B:** Em seguida, a qualidade das fronteiras não dominadas obtidas com o método proposto são comparadas. Nessa comparação, um limite superior da

fronteira não dominada é estimado utilizando várias execuções do algoritmo mono-objetivo WSNdsGA.

6.3.1 Parte A - Comparação dos Algoritmos

Nesta comparação, o MOWSNdsGA foi executado até se atingir a aproximação do conjunto Pareto-ótimo. Então, duas soluções foram escolhidas: as soluções com maior tempo de vida que garantam o nível de cobertura mínimo acima de 70% e 95%. Este critério é o mesmo considerado por Martins et al. (2011) e foi adotado aqui para permitir uma comparabilidade. Estas soluções são então comparadas com as soluções de 70% e 95% de cobertura fornecidas pelo MultiOnHA e a solução de 95% de cobertura fornecida pelo PAWSN. Estas soluções podem ser encontradas nas referências (Martins et al., 2011) e (Martins et al., 2010), respectivamente.

O desempenho do tempo de vida da rede versus cobertura das soluções obtida pelo MOWSNdsGA e pelos outros dois métodos, são mostrados na Figura 6.11. A quantidade de unidades de tempo em que estas soluções são capazes de manter a cobertura superior ao nível mínimo requerido, podem ser vistos na Tabela 6.5. Os resultados apresentados são baseados em 33 execuções (por método).

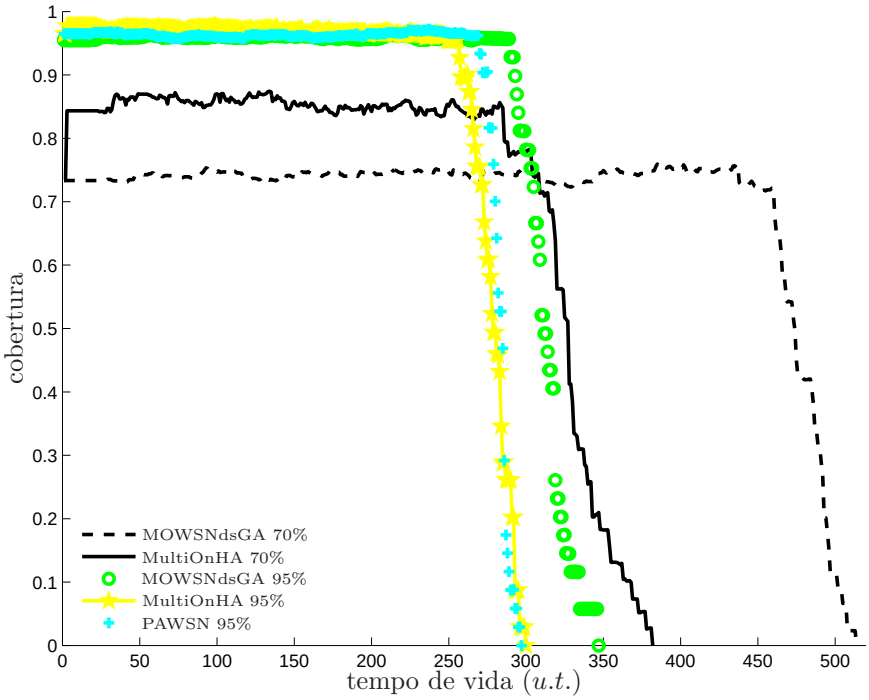


Figura 6.11 Algoritmo multiobjetivo – Resultados A: cobertura × tempo de vida

A partir da Figura 6.11 e da Tabela 6.5, é possível notar que o MOWSNdsGA é capaz de aumentar o tempo de vida da rede consideravelmente, para um certo nível de cobertura dado:

- para o nível de cobertura mínimo em 70%, o melhor tempo de vida da rede passou

Tabela 6.5 Algoritmo multiobjetivo – Resultados A: tempo de vida da rede

Algoritmo	$q_{0.00}$	$q_{0.25}$	$q_{0.50}$	$q_{0.75}$	$q_{1.00}$	T
MOWSNdsGA-95	289	304	313	319	346	501
MOWSNdsGA-70	437	467	486	494	513	555
PAWSN-95	269	279	284	286	296	380
MultiOnHA-95	239	266	280	286	299	15
MultiOnHA-70	262	327	348	361	397	13

onde:

- $q_{0.00}$ é o pior resultado atingido por cada algoritmo (em $u.t.$);
- $q_{0.25}$ é o primeiro quartil dos resultados atingidos por cada algoritmo (em $u.t.$);
- $q_{0.50}$ é a mediana dos resultados atingidos por cada algoritmo (em $u.t.$);
- $q_{0.75}$ é o terceiro quartil dos resultados atingidos por cada algoritmo (em $u.t.$);
- $q_{1.00}$ é o melhor resultado atingido por cada algoritmo (em $u.t.$);
- T é o tempo médio, em segundos, necessário para realizar uma execução de cada algoritmo[a].

[a] T corresponde ao tempo de processador, utilizando apenas um núcleo, em simulações realizadas em um computador com processador Intel(R) Xeon(R) 2.00GHz 64bits e 8GB RAM DDR3 1067MHZ. Todas as Implementações foram em JAVA.

das 397 u.t. (MultiOnHA-0.70) para 513 u.t. (MOWSNdsGA-0.70), o que significa um ganho de $29,2\%$ em relação à solução do MultiOnHA;

- para o nível de cobertura de 95%, o uso da nova abordagem traz um aumento no tempo de vida útil de 299 u.t. (MultiOnHA-95) ou 296 u.t. (PAWSN-0.95) para 346 u.t. (MOWSNdsGA-0.95). Esta solução proporciona um ganho de $15,7\%$ quando comparado ao MultiOnHA e de $16,9\%$ quando comparado ao PAWSN.

Portanto, a abordagem aqui proposta parece representar uma ferramenta de projeto melhor para RSSF's, alcançando soluções com um maior tempo de vida associado. Além disto, o MOWSNdsGA também apresenta outra vantagem quando comparado com as abordagens de um único objetivo: ele pode estimar soluções para vários níveis de cobertura ao mesmo tempo, em apenas uma execução. Os algoritmos de um único objetivo (WSNdsGA e PAWSN) requerem N execuções para encontrar soluções para N níveis de cobertura diferentes.

A energia consumida e a energia residual das soluções comparadas são mostradas nas Figuras 6.12 e 6.13 respectivamente.

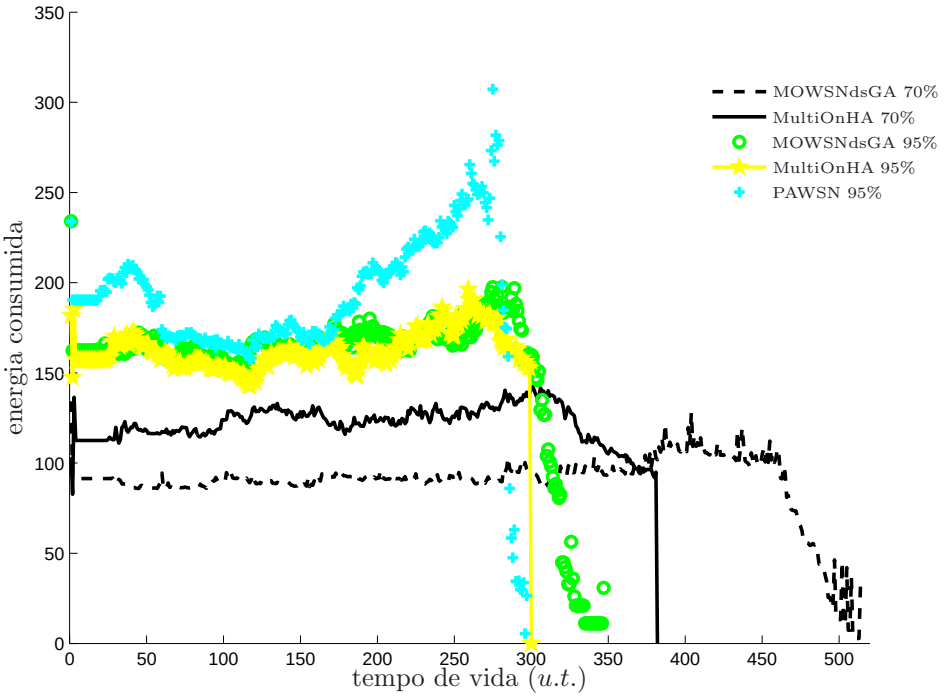


Figura 6.12 Algoritmo multiobjetivo – Resultados A: Energia Consumida × tempo de vida

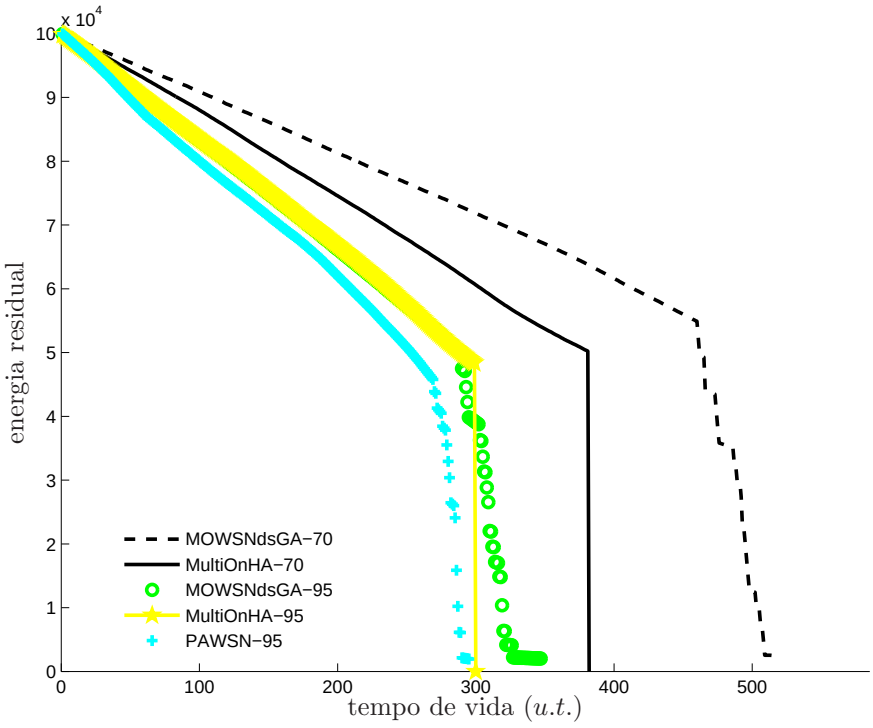


Figura 6.13 Algoritmo multiobjetivo – Resultados A: energia residual × tempo de vida

6.3.2 Parte B - Validação do Conjunto Pareto-Ótimo

No intuito de avaliar o desempenho do MOWSNdsGA, duas diferentes comparações são realizadas:

i) Foi realizada uma série de 33 execuções do algoritmo a fim de gerar uma figura mostrando a variabilidade dos resultados.

ii) A versão mono-objetivo do algoritmo, WSNdsGA, foi executada de modo a permitir uma avaliação das melhorias que podem ser obtidas em uma abordagem escalarizada, na qual cada ponto do conjunto Pareto-ótimo é gerado separadamente. Sabe-se que esta versão faz uma busca mais intensa das soluções, à custa de um maior esforço computacional.

O procedimento a seguir foi utilizado para análise dos resultados das 33 execuções do algoritmo:

- Agregar os 33 conjuntos eficiente alcançados em um único conjunto $\mathcal{P}$.
- Dividir o nível de cobertura em 11 faixas, como se segue: $[50\%, 55\%[$, $[55\%, 60\%[$, $[60\%, 65\%[$, $[65\%, 70\%[$, $[70\%, 75\%[$, $[75\%, 80\%[$, $[80\%, 85\%[$, $[85\%, 90\%[$, $[90\%, 95\%[$, $[95\%, 100\%[$ e $[100\%, 100\%]$.
- Para cada faixa i:
  - formar um conjunto $\mathcal{P}_i$ com todas as soluções de $\mathcal{P}$ que se encontram na faixa i.
  - tomar três soluções de $\mathcal{P}_i$: a de maior tempo de vida ("melhor"), a média do tempo de vida ("média") e a de menor tempo de vida ("pior").

A execução destes passos possibilita gerar três diferentes tipos de aproximações da fronteira de Pareto: "melhor", "média" e "pior".

A versão mono-objetivo do algoritmo, WSNdsGA, foi utilizada para realizar uma busca ϵ-restrita para compor um conjunto de soluções multiobjetivo. Neste caso, para cada execução do algoritmo, uma solução multiobjetivo é encontrada, para um determinado nível de cobertura. Um total de 33 execuções foram realizadas para cada um dos 11 níveis de cobertura indicados na Equação 6.8.

$$\mathcal{S}_{cov} = \{0.50, 0.55, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, 0.90, 0.95, 1.00\} \tag{6.8}$$

Este procedimento proporciona um conjunto de 11 soluções aproximadamente equidistantes. Ele força uma busca mais intensa do que no caso da versão multiobjetivo do algoritmo, o que tende a levar a melhores soluções. No entanto, o custo computacional total desta abordagem é aproximadamente 11 vezes maior do que na abordagem multiobjetivo.

Os resultados das 33 execuções do MOWSNdsGA e também os resultados da versão ϵ-restrita do WSNdsGA são apresentados na Figura 6.14. Pode-se notar que as soluções

"melhor" e "média" da aproximação da fronteira de Pareto são próximas das soluções ϵ-restrita. Este fato sugere que o MOWSNdsGA é eficiente para realizar este tipo de pesquisa e que os parâmetros do algoritmo encontrados para a versão mono-objetivo são também adequados para a versão multiobjetivo.

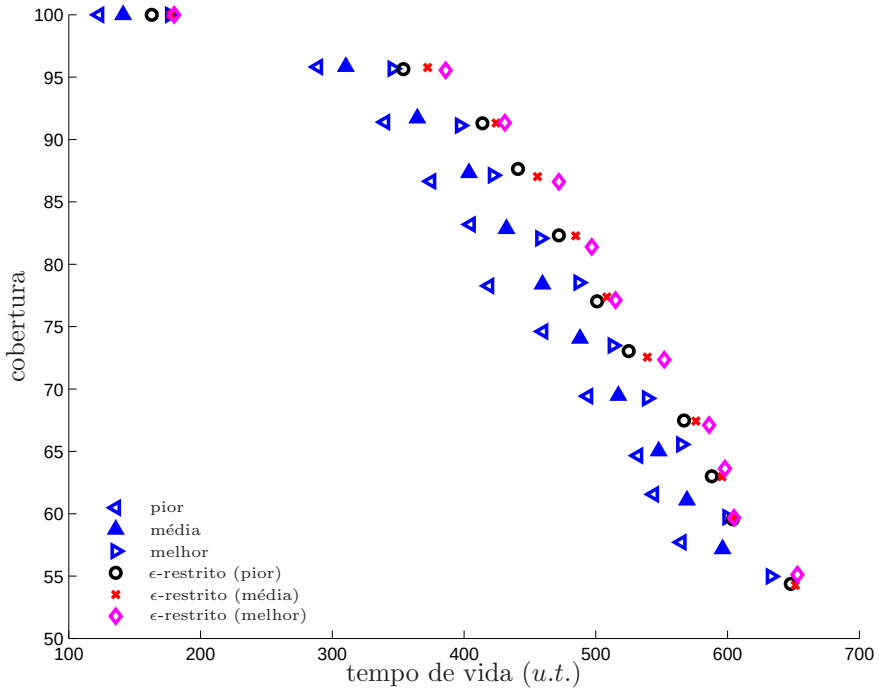


Figura 6.14 Algoritmo multiobjetivo – Resultados B: Avaliação da qualidade da fronteira de Pareto.

6.4 ABORDAGEM ONLINE COM TOLERÂNCIA A FALHAS INESPERADAS

Na tentativa de diminuir a diferença entre o caso imune a falhas imprevistas e o comportamento real das RSSF, nesta seção serão apresentados os resultados obtidos por quatro abordagens *online* sugeridas no Capítulo 5, são elas:

- WSNdsGA-OS: Um algoritmo *Online* Sequencial (centralizado);
- WSNdsGA-OmGA: Um micro Algoritmo Genético *Online* (centralizado);
- WSNdsGA-OGA: Um Algoritmo Genético *Online* (centralizado);
- WSNdsGA-OP: Um algoritmo com restrição de Protocolo *Online* (centralizado e distribuído).

Todas as abordagens utilizam como solução inicial o *scheduling* calculado pelo WSNdsGA. Esta solução é simulada em um **Ambiente com Falhas Prováveis** como definido anteriormente. Este ambiente foi criado alterando as cargas iniciais dos sensores presentes na rede pelo Algoritmo 11. A alteração das cargas iniciais dos sensores pode provocar

falhas imprevistas, alterando a topologia da rede. Isto ocorre, pois o sensor pode parar de funcionar antes do tempo previsto, podendo a rede ficar sem cobertura e conectividade. Apesar de ter sido modelado apenas com falhas de energia, a forma pela qual o Ambiente com Falhas Prováveis foi concebido, possibilita interpretar algumas falhas como sendo de outros tipos. A título de exemplificação, pode-se ter um sensor cuja carga inicial seja bastante inferior à média, em alguns casos este sensor pode nunca ser ativado, pois sua capacidade energética pode não ser suficiente para ele entrar em operação. Este tipo de falha pode ser interpretado de várias formas, como: falha mecânica, defeito causado no lançamento, ou até mesmo defeito de fábrica.

Os parâmetros utilizados no Algoritmo 11 para gerar o Ambiente com Falhas Prováveis foram:

desvio padrão: $\sigma \leftarrow 100mAh$

média: $EB \leftarrow 1000mAh$.

Com a aplicação destes parâmetros a curva de probabilidade para o valor inicial da carga de um sensor da rede é apresentada na Figura 6.15.

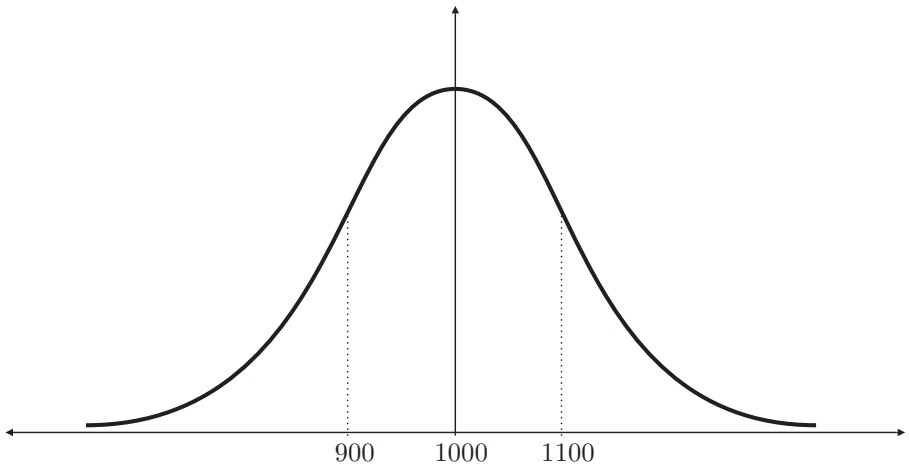


Figura 6.15 Exemplo da aplicação dos parâmetros na curva de distribuição normal utilizada para gerar EB_{real}.

Para realizar os testes com as quatro abordagens *online*, foram escolhidas aleatoriamente 21 das 33 soluções obtidas nas simulações do WSNdsGA para a instância de 100 nós sensores em que foram exigidas 100% de cobertura a cada unidade de tempo. Os resultados aqui apresentados foram divididos em três etapas:

- **Parte A:** Avaliação do comportamento de cada uma das 21 soluções aplicadas nas quatro diferentes abordagens *online*. Cada uma das soluções foram submetidas a 10 diferentes tipos de instâncias[3] geradas através do Algoritmo 11, o qual altera a carga inicial das baterias.

[3]Estas instâncias podem ser consultadas no Apêndice C.

- **Parte B:** Comparação dos resultados obtidos na Parte A, descrita acima, com os resultados da otimização do WSNdsGA apresentados na Seção 6.2.1.
- **Parte C:** Uma simulação de Monte Carlo (Metropolis & Ulam, 1949) é realizada com intuito de mostrar a robustez das soluções fornecidas pelo WSNdsGA, e do protocolo WSNdsGA-OP.

Os testes aqui realizados buscam se aproximar de um caso real. Contudo, a aplicação de operadores como o operador Inteligente de Desativação é realizado de forma centralizada, no nó sorvedouro. Sendo assim, a energia de ativação gasta por todos os sensores ativados no estágio corrente é computada, mesmo que este seja desativado por tal operador no mesmo estágio.

6.4.1 Parte A - Resultados das Abordagens Online

Nesta parte dos testes, 210 diferentes instâncias foram avaliadas nas quatro diferentes abordagens *online*. Como informado no Capítulo 5 uma das abordagens *online*, a WSNdsGA-OGA, foi construída com intuito de servir como referência para as demais soluções.

Como o WSNdsGA-OmGA faz uso de um micro algoritmo genético (mAG) seus parâmetros precisam ser definidos. Como mostrado na Seção 5.3.2, uma característica do mAG é possuir valores reduzidos de população e número de avaliações de função objetivo. Portanto, a seguir são mostrados os parâmetros adotados pelo WSNdsGA-OmGA:

- $S_{pop} = 20$;
- $n_{evl} = 2.800$;
- $p_c = 0,90$;
- $p_m = 0,10$.

O desempenho médio do tempo de vida da rede versus a cobertura obtida por cada método, pode ser visto na Figura 6.16. A quantidade de unidades de tempo em que estas soluções são capazes de manter a cobertura exigida, é mostrada na Tabela 6.6.

Os resultados apresentados na Figura 6.16 mostram que realmente o WSNdsGA-OGA realiza seu papel de limite de referência. Isto já era de se esperar, pois a cada falha inesperada a rede é reconfigurada pelo sorvedouro através do algoritmo WSNdsGA, que apresentou bons resultados ao resolver o novo modelo proposto para o PDCC-RSSF. No entanto, executar o WSNdsGA a cada falha inesperada, pode não ser muito eficiente devido ao alto custo computacional dessa operação. Na Tabela 6.6 também é possível ver o tempo médio de CPU de simulação gasto por cada abordagem. Os valores mostrados representam apenas o tempo gasto para simular a rede, não foi considerado o tempo para projetar a solução inicial. Uma alternativa para reduzir o custo computacional, tentando manter a eficiência do algoritmo foi a implementação de um mAG. Foi então sugerido o WSNdsGA-OmGA, cujo os resultados também são mostrados. O WSNdsGA-OmGA conseguiu atingir resultados bem próximos ao limite de referência estabelecido

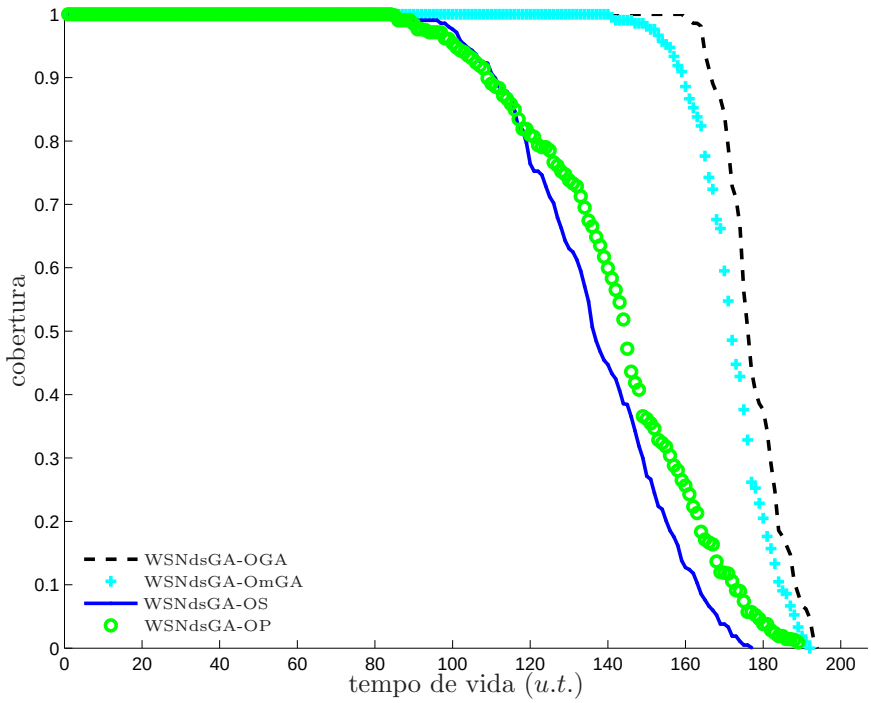


Figura 6.16 Abordagem Online – Resultados A - 100 sensores: cobertura x tempo de vida.

em um tempo computacional aproximadamente 10 vezes menor. A redução do esforço computacional, com a aplicação do mAG, pode ampliar a viabilidade da aplicação desta abordagem.

Para aplicações que exigem um tempo mais rápido de resposta, foram sugeridos dois outros algoritmos cujos resultados também são apresentados.

WSNdsGA-OS: O desempenho deste algoritmo em relação ao tempo de vida da rede foi em média 21% inferior ao WSNdsGA-OmGA, no entanto foi em média 11.757 vezes mais rápido. Isto ocorre, pois este algoritmo é apenas uma decodificação da solução pré-processada pelo WSNdsGA.

WSNdsGA-OP: Este é um algoritmo que trabalha junto com um protocolo para reconfiguração local da rede. Seu desempenho em relação ao tempo de vida da rede foi em média 17% inferior ao WSNdsGA-OmGA, no entanto foi em média 15.132 vezes mais rápido. Ao comparar com a versão sequencial, a inserção do protocolo online trouxe benefício em relação ao tempo de vida da rede, mantendo o tempo computacional na mesma ordem de grandeza.

Como visto na Tabela 6.6, a variação dos resultados apresentados por estas duas abordagens foram bem superiores. Isto ocorre pois não há uma ação efetiva de reconstrução do *scheduling* da rede ao surgir uma falha inesperada. A reconstrução é feita de forma local, não tratando a natureza dinâmica do PDCC-RSSF.

Tabela 6.6 Abordagem Online – Resultados A - 100 sensores: tempo de vida da rede

Algoritmo	$q_{0.00}$	$q_{0.25}$	$q_{0.50}$	$q_{0.75}$	$q_{1.00}$	T
WSNdsGA-OGA	158	171	176	182	193	3481,8424
WSNdsGA-OmGA	140	165	171	178	191	326,8436
WSNdsGA-OS	83	120	136	151	176	0,0278
WSNdsGA-OP	84	127	143	159	188	0,0216

onde:

- $q_{0.00}$ é o pior resultado atingido por cada algoritmo (em $u.t.$);
- $q_{0.25}$ é o primeiro quartil dos resultados atingidos por cada algoritmo (em $u.t.$);
- $q_{0.50}$ é a mediana dos resultados atingidos por cada algoritmo (em $u.t.$);
- $q_{0.75}$ é o terceiro quartil dos resultados atingidos por cada algoritmo (em $u.t.$);
- $q_{1.00}$ é o melhor resultado atingido por cada algoritmo (em $u.t.$);
- T é o tempo médio, em segundos, necessário para realizar uma execução de cada algoritmo[a].

[a] T corresponde ao tempo de processador, utilizando apenas um núcleo, em simulações realizadas em um computador com processador Intel(R) Xeon(R) 2.00GHz 64bits e 8GB RAM DDR3 1067MHZ. Todas as Implementações foram em JAVA.

Como já informado, o tempo computacional (T) mostrado na Tabela 6.6 representa a execução total do algoritmo na simulação de uma solução. Sendo assim, é importante ressaltar que nas abordagens WSNdsGA-OGA e WSNdsGA-OmGA os tempos apresentados correspondem à soma de todas as chamadas dos algoritmos para recálculo do escalonamento. Feita estas considerações, fica claro que o algoritmo a ser escolhido depende muito da aplicação. Se o tempo para reestabelecer a rede é algo crítico, os algoritmos mais adequados seriam os WSNdsGA-OS e WSNdsGA-OP. No entanto, se a aplicação tolerar tempos de respostas da ordem de 5 minutos, uma boa solução é aplicar o WSNdsGA-OmGA. Por fim, se a aplicação suportar uma reestruturação que possa demorar da ordem de 30 minutos pode-se aplicar o WSNdsGA-OGA.

6.4.2 Parte B - Comparações com a Otimização pelo WSNdsGA e com PLI

Nesta seção serão comparados os resultados das simulações pelas abordagens *online* e os resultados obtidos pelo algoritmo de otimização WSNdsGA. As soluções obtidas pelas abordagens são as mesmas apresentadas na seção anterior, resultados das 210 diferentes instâncias. Como já mencionado, as instâncias são fruto de uma combinação das 21 soluções otimizadas pelo WSNdsGA e 10 diferentes instâncias para carga inicial das baterias de cada sensor. Já os resultados referente ao WSNdsGA são os mesmos apresentados na Seção 6.2.1, ou seja, as mesmas 21 soluções citadas acima. Deve-se lembrar

que os resultados do WSNdsGA e do PLI foram obtidos em um ambiente com falhas previsíveis (considerado o comportamento analítico das baterias e cargas iniciais conforme informado pelo fabricante). Devido a esse motivo os resultados são os 21 melhores resultados encontrados em 21 execuções do algoritmo na instância de 100 nós sensores. Uma comparação do tempo de vida médio da rede obtido por cada uma das abordagens e também a atingida pelo PLI pode ser visto na Figura 6.17. Um quadro comparativo entre as soluções é mostrado na Tabela 6.7.

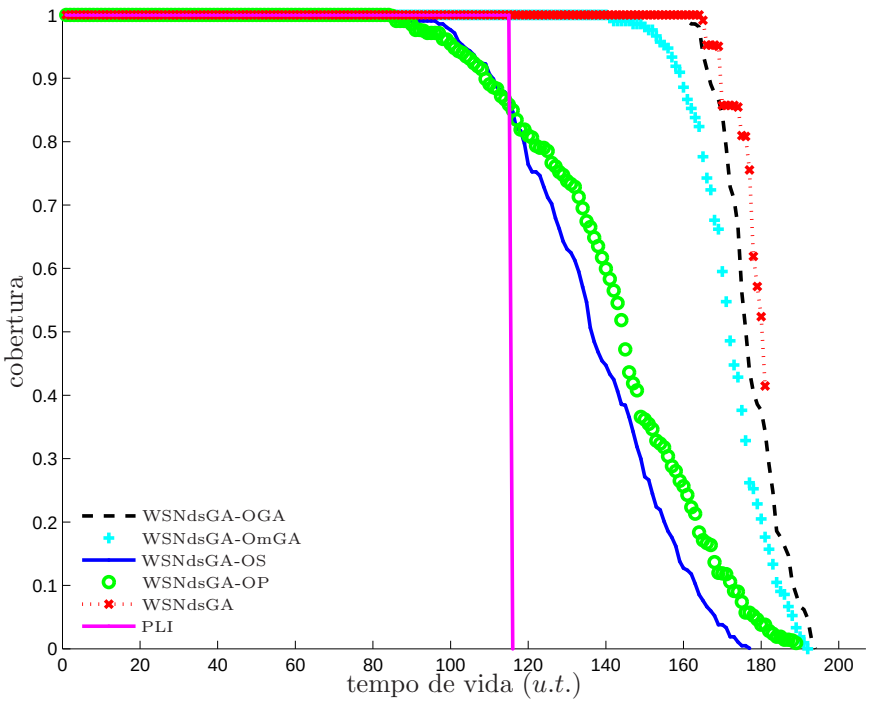


Figura 6.17 Abordagem Online - Resultados B - 100 sensores: cobertura × tempo de vida. As curvas apresentadas correspondem ao caso médio apresentado na Tabela 6.7. A curva WSNdsGA é uma média em 21 soluções. Já as curvas dos algoritmos *online* são uma média da combinação destas 21 soluções coma 10 diferentes instâncias para carga inicial das baterias.

Tabela 6.7 Abordagem Online – Resultados B - 100 sensores: tempo de vida da rede

Algoritmo	$q_{0.00}$	$q_{0.25}$	$q_{0.50}$	$q_{0.75}$	$q_{1.00}$
WSNdsGA-OGA	158	171	176	183	193
WSNdsGA-OmGA	140	165	171	178	191
WSNdsGA-OS	83	120	136	151	176
WSNdsGA-OP	84	127	143	159	188
WSNdsGA	164	175	180	180	180
PLI	115				

Como observado na Figura 6.17, os resultados obtidos pelo WSNdsGA-OGA, abordagem que serviu como referência de comparação na Parte A, ficaram bem próximos da

versão pré-processada considerando um ambiente totalmente controlável. Ao analisar os resultados é possível perceber que, mesmo se os sensores possuírem uma bateria cuja carga inicial varie de acordo com uma função densidade de probabilidade normal, com o desvio padrão sendo 10% da média, a rede consegue se reorganizar ao ponto de conseguir resultados similares. Comparando pela Tabela 6.7 é possível perceber que apenas nos extremos ($q_{0.00}$ e $q_{1.00}$) os resultados não são muito próximos. Isto ocorre devido à própria característica da distribuição normal: ora podem surgir baterias com baixíssimo valor de carga, ora com valor alto. Como consequência pode-se ter uma rede com baixo ou alto tempo de vida.

Quando os resultados são comparados com a solução da outra modelagem por PLI, os resultados continuam surpreendentes. Pela Figura 6.17 é possível ver que os resultados de WSNdsGA-OGA e WSNdsGA-OmGA superam em muito os atingidos por PLI. Pela Tabela 6.7 é possível ver que, no pior caso o WSNdsGA-OmGA chega a ser mais de 21,73% melhor. A mesma tabela mostra que a partir do primeiro quartil ($q_{0.25}$) as outras abordagens passam também a superar o PLI. Os tempos computacionais não foram abordados nesta tabela por se tratarem de duas abordagens distintas, sendo uma *online* e outra pré-processada, não fazendo sentido tal comparação.

6.4.3 Parte C - Simulação de Monte Carlo

Aqui foi empregado o Método de Monte Carlo para avaliar o comportamento de duas diferentes abordagens: WSNdsGA-OS e WSNdsGA-OP. Para cada uma das 21 soluções iniciais geradas pelo WSNdsGA, foram utilizadas 1.000 diferentes instâncias geradas através do Algoritmo 11, o qual altera a carga inicial das baterias. Sendo assim, foram utilizadas 21.000 instâncias diferentes no Método de Monte Carlo.

A Figura 6.18 e a Tabela 6.8 mostram que os resultados tiveram comportamento similar ao mostrado na Parte A.

Tabela 6.8 Abordagem Online – Resultados C - 100 sensores: tempo de vida da rede

Algoritmos	$q_{0.00}$	$q_{0.25}$	$q_{0.50}$	$q_{0.75}$	$q_{1.00}$	T
WSNdsGA-OS	75	120	134	147	200	0,000249
WSNdsGA-OP	63	126	142	156	205	0,046208

Ao observar os histogramas apresentados nas Figuras 6.19 e 6.20, é possível notar que grande parte das soluções se encontram na faixa de tempo de vida entre 120 a 160 $u.t.$. Estes valores estão aproximadamente entre o primeiro e quarto quartil e são superiores ao valor de 114 $u.t.$ alcançados pelo PLI. Estes resultados mostram novamente a robustez dos métodos apresentados. Mesmo em uma gama muito grande de instâncias os resultados se mantiveram similares. Ao comparar a Tabela 6.8 com a Tabela 6.7 é possível notar que apenas nos extremos ($q_{0.00}$ e $q_{1.00}$) os resultados não foram muito próximos. Isto novamente se justifica devido à característica da distribuição normal, função de densidade de probabilidade utilizada na geração das cargas iniciais das baterias, ao gerar as instâncias utilizadas nas simulações.

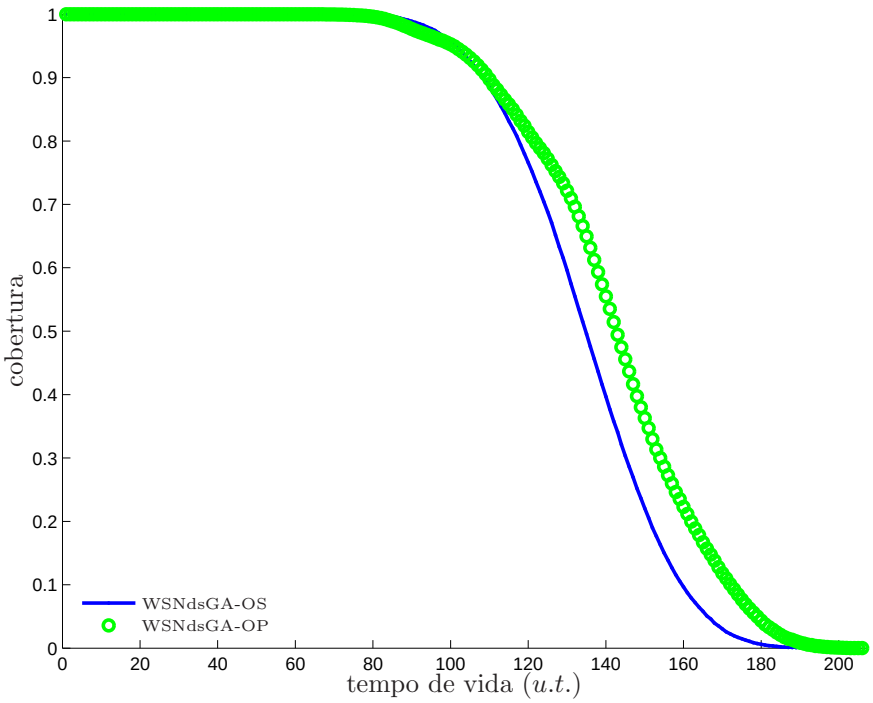


Figura 6.18 Abordagem Online - Resultados C - 100 sensores: cobertura × tempo de vida.

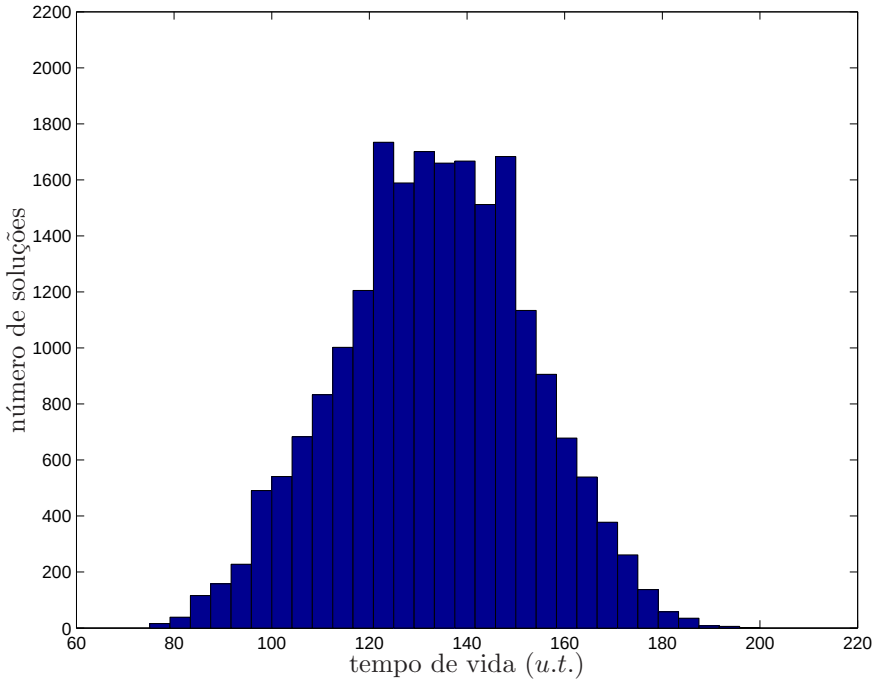


Figura 6.19 Abordagem Online - Resultados C - 100 sensores: histograma da simulação de Monte Carlo do WSNdsGA-OS.

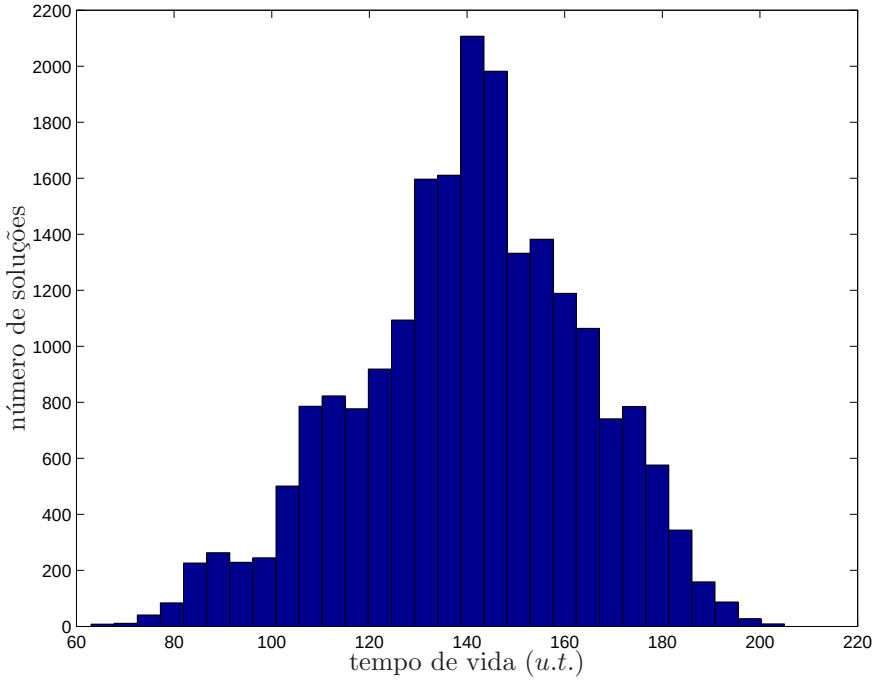


Figura 6.20 Abordagem Online - Resultados C - 100 sensores: histograma da simulação de Monte Carlo do WSNdsGA-PO.

CAPÍTULO 7

CONSIDERAÇÕES FINAIS

7.1 CONCLUSÃO

Inspirado nos métodos de espaço vetorial, o estudo apresentado aqui é parte de um esforço teórico para o desenvolvimento de uma metodologia abrangente, para análise e síntese de algoritmos combinatórios evolucionários. Os resultados aqui obtidos partem do ponto de vista de que a estrutura formal de um espaço vetorial pode constituir um guia poderoso para capturar algumas características essenciais de problemas combinatórios em uma estrutura abstrata, permitindo o desenvolvimento de operadores que são análogos aos conhecidos procedimentos de busca em um espaço vetorial genérico. Em particular, neste trabalho foram exploradas as entidades geométricas de *direções de descida* e *subespaço*. Estas entidades foram introduzidas para fins de descrição da estrutura do problema e para a síntese de um algoritmo genético de alto desempenho para um problema prático e específico: O Problema Dinâmico de Cobertura e Conectividade em Redes de Sensores Sem Fio - PDCC-RSSF.

De posse de um breve fundamento teórico sobre Redes de Sensores Sem Fio (RSSF) mostrado no Capítulo 2, o PDCC-RSSF foi definido e uma nova modelagem para sua solução foi proposta no Capítulo 3. Este novo modelo foi fundamentado em um espaço vetorial, e a partir das entidades geométricas de *direção de descida* e *subespaço* foi construído um Algoritmo Genético de escalonamento dinâmico para Redes de Sensores sem Fio (WSNdsGA - *Wireless Sensor Network dynamic scheduling Genetic Algorithm*). O objetivo deste algoritmo é encontrar uma sequência (*scheduling*) de ativação e controles de desativação de sensores, a fim de maximizar o tempo de vida da rede, garantindo os requisitos de cobertura e conectividade. A RSSF foi modelada como um sistema dinâmico, capturando toda a dinamicidade do PDCC-RSSF. A cada falha de um nó sensor, foi caracterizado um novo estágio do sistema. O modelo procura maximizar o tempo de vida da RSSF considerando a interação entre todos os estágios do sistema simultaneamente. O problema foi tratado como um problema de controle ótimo, em que as ações de controle eram escolhidas de forma a seguir o sinal de referência (limite de cobertura desejável), assegurando a conectividade e buscando maximizar o tempo de vida da rede.

Usualmente os algoritmos encontrados na literatura, buscam estender o tempo de vida de uma RSSF a partir da minimização do consumo de energia da rede, a cada estágio, sem considerar a dinamicidade real do problema. Esta modelagem geralmente é escolhida por ser de fácil representação e linearização, possibilitando utilizar técnicas de Programação Linear Inteira (PLI) para resolver o problema. O modelo aqui apresentado caminhou em outra ótica, em que o próprio tempo de vida da rede foi o objetivo a ser maximizado. Como a ferramenta de otimização adotada neste trabalho foi um algoritmo evolutivo, a função objetivo e as restrições não necessitam obrigatoriamente ser lineares.

A construção do WSNdsGA foi mostrada com detalhes no Capítulo 4. A estrutura do problema levou à construção de um operador de mutação projetado em um *subespaço* definido. Utilizando esta definição, foi também projetado um operador de busca local que efetua uma busca no *subespaço* complementar. Já o conceito de *direção de descida* foi firmado utilizando o operadore de Cruzamento Real Polarizado. Este operador gera descendentes em um segmento extrapolado, e em uma região com tendência de melhoria de função objetivo. Em paralelo, ao longo deste capítulo, uma extensão multiobjetivo do algoritmo foi proposta (MOWSNdsGA - *MultiObjective Wireless Sensor Network dynamic scheduling Genetic Algorithm*), em que a restrição de cobertura foi relaxada, transformando-se em um objetivo a ser maximizado.

O WSNdsGA foi desenvolvido com intuito de resolver o novo modelo sugerido para o PDCC-RSSF (Capítulo 3). Os resultados obtidos foram contrastados com a solução exata de outra modelagem, na qual minimiza a energia consumida da rede a cada período de tempo na tentativa de estender seu tempo de vida (Nakamura et al., 2005; Nakamura, 2010). Os resultados obtidos foram mostrados na Seção 6.2. Ao resolver uma instância de 100 nós sensores, o novo modelo proposto chegou a superar em mais de 40% no tempo de vida da rede, em seu pior caso, e em mais 56% em seu melhor caso. Como o WSNdsGA foi desenvolvido de forma a suportar diferentes restrições de cobertura, foi possível compará-lo com dois outros trabalhos do autor desta tese, o MultiOnHA (Martins et al., 2011) e o PAWSN (Martins et al., 2010). Em todos os casos os resultados foram de 31% a 82% superiores.

Na Seção 6.3 foram apresentados os resultados obtidos pela abordagem multiobjetivo MOWSNdsGA. A grande vantagem mostrada por este algoritmo foi tornar possível, em apenas uma execução, escolher diferentes soluções eficientes para diferentes restrições de cobertura. Os resultados foram também comparados com o PAWSN e com o algoritmo multiobjetivo MultiOnHA, sendo superior em todos os casos. Além desta comparação, o desempenho do MOWSNdsGA foi analisado, realizando uma comparação de seu Pareto, com um conjunto de soluções multiobjetivo formada por uma busca ϵ-restrita pelo algoritmo WSNdsGA.

As soluções apresentadas pelos algoritmos WSNdsGA e MOWSNdsGA podem ser utilizadas como referência para o projeto de uma RSSF. Todas as soluções foram geradas em um ambiente com falhas totalmente previsíveis, considerando que as cargas iniciais de cada bateria presente nos sensores eram exatamente iguais às informadas pelos fabricantes, e que seu consumo era totalmente estável e linear. Entretanto, foi mostrado no Capítulo 5, que a carga inicial e o comportamento real de uma bateria na maioria das vezes não pode ser fielmente previsível. Para adaptar a solução fornecida pelo WSNdsGA, em um ambiente que não pode ser totalmente previsível, neste mesmo capítulo foram propostos quatro diferentes abordagens, são elas:

- WSNdsGA-OS: Um algoritmo *Online* Sequencial (centralizado);
- WSNdsGA-OmGA: Um micro Algoritmo Genético *Online* (centralizado);
- WSNdsGA-OGA: Um Algoritmo Genético *Online* (centralizado);

- WSNdsGA-OP: Um algoritmo com restrição de Protocolo *Online* (centralizado e distribuído).

Todas as abordagens foram construídas com capacidade de reestruturar a rede quando falhas imprevisíveis ocorrerem. Um método para simular falhas imprevisíveis na rede foi sugerido e os resultados foram mostrados na Seção 6.4. Em média, as soluções apresentadas pelas abordagens WSNdsGA-OGA e WSNdsGA-OmGA, conseguiram atingir resultados muito próximos à referência projetada pelo WSNdsGA. Conforme mostrado, apesar de fornecer soluções pouco inferiores, o WSNdsGA-OmGA pode ser mais adequado à aplicação em uma RSSF, devido ao seu menor custo computacional. Os resultados obtidos pelos algoritmos WSNdsGA-OS e WSNdsGA-OP foram inferiores, no entanto seus tempos computacionais foram da ordem de 11.000 vezes menores.

A nova modelagem dinâmica proposta para resolver o PCC-RSSF tem se mostrado bastante promissora, atingindo resultados bem superiores aos já encontrados na literatura. Estes resultados são justificados pelos algoritmos que buscam moldar a natureza intrinsicamente dinâmica do problema. Além de superar os resultados de outra modelagem da literatura, os tempos computacionais para obter tais soluções foram bem inferiores, possibilitando o uso destes algoritmos em aplicações mais reais, com grande quantidade de sensores.

7.2 ARTIGOS RELACIONADOS AO PROBLEMA PUBLICADOS DURANTE O DOUTORADO

[Martins et al., 2009] Martins, F. V. C., Carrano, E. G., Wanner, E. F., Takahashi, R. H. C., Mateus, G. R. (2009). Hybrid multiobjective approach for designing wireless sensor networks. Proceedings of the 12th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems.

[Martins et al., 2010] Martins, F. V. C., Silva, C., Nakamura, F. G., Mateus, G. R., Takahashi, R. H. C. (2010). Controle Multiobjetivo para o Problema de Cobertura em uma Rede de Sensores Sem Fio Plana com Conectividade Garantida. Anais do XVIII Congresso Brasileiro de Automática.

[Martins et al., 2010] Martins, F. V. C., Carrano, E. G., Wanner, E. F., Takahashi, R. H. C., Mateus, G. R. (2010). An evolutionary dynamic approach for designing wireless sensor networks for real time monitoring. In Proc. IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications.

[Martins et al., 2011] Martins, F. V. C., Carrano, E. G., Wanner, E. F., Takahashi, R. H. C., Mateus, G. R. (2011). A hybrid multiobjective evolutionary approach for improving the performance of wireless sensor networks. IEEE Sensors Journal, 11(3), 545-554.

7.3 ARTIGO RELACIONADO AO PROBLEMA SUBMETIDO DURANTE O DOUTORADO E AINDA SEM RESPOSTA

[Martins et al., 2012] Martins, F. V. C., Carrano, E. G., Wanner, E. F., Nakamura, F. G., Takahashi, R. H. C., Mateus, G. R. (2012). On a Vector Space Representation in Genetic Algorithm for Sensor Scheduling in Wireless Sensor Networks. Evolutionary Computation.

7.4 TRABALHOS FUTUROS

O próximo passo consiste em desenvolver novos operadores para o WSNdsGA apoiados no conceito de entidades geométricas. A partir destes novos operadores espera-se construir um algoritmo ainda mais estável e robusto para o projeto de RSSFs. Além de buscar melhorias no desempenho do algoritmo proposto, serão desenvolvidas novas abordagens *online* e distribuídas, a partir da rede projetada pelo WSNdsGA. O intuito destas propostas são de aproximar o modelo dinâmico introduzido neste trabalho, com as situações reais presentes nas RSSFs.

Outro passo será expandir o modelo proposto em um modelo que contemple uma RSSF hierárquica. A ideia principal consiste em dividir a área a ser monitorada em blocos menores, de forma que em cada bloco haja uma rede independente com seus *cluster heads*. Utilizando pelo menos dois *sinks* móveis, todos os dados enviados para os *cluster heads* poderão ser coletados pelos *sinks*. A cada estágio da rede pelo menos um *sink* deverá permanecer fixo coletando as informações enquanto os outros se locomovem para um ponto em que seja possível a coleta de informações dos outros blocos da área. A modelagem dinâmica já apresentada será estendida de forma a conseguir uma solução que busque otimizar o tempo de vida de todos os blocos da rede ao mesmo tempo, ou seja, não os tratando de forma independente. Com esta nova abordagem espera-se conseguir formar redes com grande quantidade de nós sensores, pois o problema seria dividido em problemas menores com possibilidade de boas soluções.

Até onde é de conhecimento deste autor, os sensores usualmente utilizados em uma RSSF não são capazes, enquanto desativados, de receber mensagens de *controle de ativação*, como mencionado no Pressuposto 3 (Capítulo 5). Na tentativa de aproximar a solução proposta nesta tese de um modelo real, este pressuposto será desconsiderado e três formas de tratar as falhas serão comparadas, são elas:

1. Além dos estados “ativado”, “desativado” e “falho”, cada sensor também poderá entrar no estado “dormindo”. Neste estado, o sensor irá ligar seu rádio em intervalos regulares, apenas para receber mensagens. Consequentemente, o sensor que estiver “dormindo”, não estará apto a participar do roteamento e nem do monitoramento da rede, e seu gasto de energia será mínimo. Ao estabelecer a rede, todos os nós sensores estarão inicialmente “ativados”. Após o sorvedouro executar um algoritmo de controle de densidade, uma mensagem será disseminada pela rede informando quais sensores deverão alterar seu estado para “dormindo”. Os quatro algoritmos para tratamento de falhas inesperadas propostos no Capítulo 5 (WSNdsGA-OS, WSNdsGA-OmGA, WSNdsGA-OGA e WSNdsGA-OP), serão novamente testados

considerando este novo estado para o sensor. Neste sentido, as mensagens de *controle de desativação* serão alteradas para mensagens de *controle de dormindo*, e a mensagem de *controle de ativação* pode ser recebida pelo sensor que estiver "dormindo", podendo este alterar seu estado para "ativo" quando requisitado.

2. Ao estabelecer a rede, os nós sensores permanecerão todos ativados até que o sorvedouro calcule um *scheduling* para a rede (poderá ser utilizando um algoritmo baseado no WSNdsGA proposto nesta tese). Cada nó sensor será programado para que ele tenha conhecimento de quais serão os intervalos que ele deverá permanecer "ativado", "desativado" ou "dormindo". Desta forma, será possível criar um "ponto de verificação", ou seja, programar todos os sensores para permanecerem "ativados" no mesmo intervalo de tempo, e por troca de mensagens, o sorvedouro terá o conhecimento da situação atual da rede. A partir desta informação, um novo *scheduling* será calculado e disseminado pela rede para que os nós sensores se reprogramem. É de se esperar que nesta abordagem ocorra perda de cobertura entre os "pontos de verificação", pois podem ocorrer, neste intervalo, falhas inesperadas em alguns nós sensores presentes na rede.

3. Cada sensor será programado conforme seu *scheduling* inicial, e manterá na memória todas as possíveis alterações deste *schedulings* caso um de seus vizinhos mude de estado de forma diferente do previsto. Ao identificar a alteração inesperada do estado de um vizinho, o sensor é reprogramado segundo o *scheduling* correspondente. A viabilidade desta abordagem dependerá da capacidade de memória interna do nó sensor.

REFERÊNCIAS BIBLIOGRÁFICAS

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: A survey. *Computer Networks*, 38, 393–422.

Anastasi, G., Conti, M., Di Francesco, M., & Passarella, A. (2009). Energy conservation in wireless sensor networks: A survey. *Ad Hoc Networks*, 7, 537–568.

Barbancho, J., Leon, C., Molina, J., & Barbancho, A. (2006). Giving neurons to sensors. QoS management in wireless sensors networks. In *Emerging Technologies and Factory Automation, 2006. ETFA '06. IEEE Conference on* (pp. 594 –597).

Belmont-Moreno., E. (2001). The role of mutation and population size in genetic algorithms applied to physics problems. *International Journal of Modern Physics C*, 9(12), 1345–1355.

Bertsekas, D. P. (1995). *Dynamic Programming and Optimal Control*. Athena Scientific.

Blahut, R. (1983). *Theory and Practice of Error Control Coding,*. Addison-Wesley.

Boukerche, A., Oliveira, H., Nakamura, E., & Loureiro, A. (2007). Localization systems for wireless sensor networks. *Wireless Communications, IEEE*, 14(6), 6 –12.

Carballido, J. A., Ponzoni, I., & Brignole, N. B. (2007). CGD-GA: A graph-based genetic algorithm for sensor network design. *Information Sciences*, 177(22), 5091 – 5102.

Cardei, M. & Wu, J. (2006). Energy-efficient coverage problems in wireless ad-hoc sensor networks. *Computer Communications*, 29, 413–420.

Carrano, E. G., Takahashi, R. H. C., Fonseca, C. M., & Neto, O. M. (2010). Nonlinear network optimization – an embedding vector space approach. *IEEE Transactions on Evolutionary Computation*, 14(2), 206–226.

Cerpa, A. & Estrin, D. (2002). ASCENT: Adaptive Self-Configuring Sensor Networks Topologies. In *Proc. International Joint Conference IEEE Computer and Communications Society (INFOCOM'02)*, volume 3 (pp. 1278–1287).

Chang, C. Y. & Chang, H. R. (2008). Energy-aware node placement, topology control and MAC scheduling for wireless sensor networks. *Computer Networks*, 52, 2189–2204.

Chen, J., Díaz, M., Llopis, L., Rubio, B., & Troya, J. M. (2011). A survey on quality of service support in wireless sensor and actor networks: Requirements and challenges in the context of critical infrastructure protection. *Journal of Network and Computer Applications*, 34(4), 1225 – 1239.

Choi, D. H. & Oh, S. Y. (2000). A new mutation rule for evolutionary programming motivated from backpropagation learning. *IEEE Transactions on Evolutionary Computation*, 2(4), 188–190.

Conover, W. (1980). *Practical Nonparametric Statistics*. New York, NY, USA: Wiley, 2nd edition.

CPLEX, I. (2006). *Ilog CPLEX source*. available in *http://www.ilog.com/products/cplex/*.

da Silva, C. V., Frery, A. C., & Viana, P. (2010). Avaliação do consumo de energia e tempo de vida de redes de sensores sem fio comerciais. *Asociación Argentina de Mecánica Computacional*, 29, 2663–2672.

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley-Interscience Series in Systems and Optimization. John Wiley & Sons.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6, 182–197.

Demirkol, I., Ersoy, C., & Alagoz, F. (2006). MAC protocols for wireless sensor networks: A survey. *Communications Magazine, IEEE*, 44(4), 115 – 121.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1, 269–271.

Du, X., Xiao, Y., Chen, H.-H., & Wu, Q. (2006). Secure cell relay routing protocol for sensor networks: Research articles. *Wireless Communications and Mobile Computing*, 6(3), 375–391.

Dubois-ferrière, H., Estrin, D., & Vetterli, M. (2005). Packet combining in sensor networks. In *In Proc. III International Conference on Embedded Networked Sensor Systems*.

Dunn, O. J. (1961). Multiple comparisons among means. *Journal of American Statistical Association*, 56, 52–64.

Ergen, S. & Varaiya, P. (2005). *TDMA scheduling algorithms for sensor networks*. Technical report, Department of Electrical Engineering and Computer Sciences University of California, Berkeley.

Fan, H. Y., Lu, J. W. Z., & Xu., Z. B. (2000). An empirical comparison of three novel genetic algorithms. *Engineering Computations*, 8(17), 981–1001.

Firoze, A., Ju, L., & Kwong, L. (2007). PR-MAC A priority reservation MAC protocol for wireless sensor networks. In *Proc.: International Conference on Electrical Engineering*.

Ganti, R. K., Jayachandran, P., Luo, H., & Abdelzaher, T. F. (2006). Datalink streaming in wireless sensor networks. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, SenSys '06 (pp. 209–222). New York, NY, USA: ACM.

Goldberg, D. (1989a). *Genetic Algorithms in Search, Optimization and Learning*. Addison-Wesley.

Goldberg, D. E. (1989b). Sizing populations for serial and parallel genetic algorithms. In *Proceedings of the Third International Conference on Genetic Algorithms* (pp. 70–79). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Guimarães, I. L. & Martins, F. V. C. (2011). Jsensors: Uma ferramenta para simulação de algoritmos em redes de sensores sem fio. Monografia apresentada ao curso de Sistemas de Informação do Instituto de Ciências Exatas e Aplicadas da Universidade Federal de Ouro Preto.

Gungor, V. C. & Özgür B. Akan (2006). DST: delay sensitive transport in wireless sensor networks. In *ISCN'06* (pp. 116–122).

Hasancebi, O. & Erbatur, F. (2000). Evaluation of crossover techniques in genetic algorithm based optimum structural design. *Computers e Structures*, 1-3(78), 435–448.

Hu, X. M., Zhang, J., Yu, Y., Chung, H. S. H., Li, Y. L., Shi, Y. H., & Luo, X. N. (2010). Hybrid genetic algorithm using a forward encoding scheme for lifetime maximization of wireless sensor networks. *IEEE Transactions on Evolutionary Computation*, 14, 766–781.

Iyer, Y. G. (2005). Stcp: A generic transport layer protocol for wireless sensor networks. In *Proc. of IEEE International Conference on Computer Communications and Networks (ICCCN)* (pp. 449–454).

Jiang, H., Chen, L., Wu, J., Chen, S., & Leung, H. (2009). A reliable and high-bandwidth multihop wireless sensor network for mine tunnel monitoring. *IEEE Sensors Journal*, 9, 1511–1517.

Jongerden, M. R. & Haverkort, B. R. H. M. (2008). *Battery Modeling*. Technical Report TR-CTIT-08-01, Centre for Telematics and Information Technology University of Twente, Enschede.

Kazarlis, S., Papadakis, S., Theocharis, J., & Petridis, V. (2001). Microgenetic algorithms as generalized hill-climbing operators for GA optimization. *IEEE Transactions on Evolutionary Computation*, 5(3), 204 –217.

Khoukhi, L. & Cherkaoui, S. (2008). Experimenting with fuzzy logic for QoS management in mobile ad hoc networks. *International Journal of Computer Science and Network Security*, 8(8), 372–386.

Krishnakumar, K. (1990). Microgenetic algorithms for stationary and nonstationary function optimization. In G. Rodriguez (Ed.), *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 1196 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* (pp. 289–296).

Kulkarni, R., Fö andrster, A., & Venayagamoorthy, G. (2011). Computational intelligence in wireless sensor networks: A survey. *Communications Surveys Tutorials, IEEE*, 13(1), 68 –96.

Lazzerini, B., Marcelloni, F., Vecchio, M., Croce, S., & Monaldi, E. (2006). A fuzzy approach to data aggregation to reduce power consumption in wireless sensor networks. In *Fuzzy Information Processing Society, 2006. NAFIPS 2006. Annual meeting of the North American* (pp. 436 –441).

Leung, H., Chandana, S., & Wei, S. (2008). Distributed sensing based on intelligent sensor networks. *IEEE Circuits and Systems Magazine*, 8, 38–52.

Liu, H., Ma, H., El Zarki, M., & Gupta, S. (1997). Error control schemes for networks: an overview. *Mobile Networks and Applications*, 2(2), 167–182.

Loureiro, A. A. F., Nogueira, J. M. S., Ruiz, L. B., de Freitas Mini, R. A., Nakamura, E. F., & Figueiredo, C. M. S. (2003). Redes de sensores sem fio. In *Simpósio Brasileiro de Redes de Computadores*.

Lussier, B., Chatila, R., Guiochet, J., Ingrand, F., Lampe, A., olivier Killijian, M., & Powell, D. (2005). Fault tolerance in autonomous systems: How and how much? In *In Proceedings of The IV IARP - IEEE / RAS - EURON Joint Workshop On Technical Challenge for Dependable Robots In Human Environments* (pp. 16–18).

Macedo, D. F., Correia, L. H. A., dos Santos, A. L., Loureiro, A. A. F., & Nogueira, J. M. S. (2005). A pro-active routing protocol for continuous data dissemination in wireless sensor networks. In *Proceedings of the 10th IEEE Symposium on Computers and Communications*, ISCC '05 (pp. 361–366). Washington, DC, USA: IEEE Computer Society.

Mainwaring, A., Culler, D., Polastre, J., Szewczyk, R., & Anderson, J. (2002). Wireless sensor networks for habitat monitoring. In *Proc. ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'02)* (pp. 88–97). New York, USA.

Mao, J., Wu, Z., & Wu, X. (2007). A TDMA scheduling scheme for many-to-one communications in wireless sensor networks. *Computer Communications*, 30(4), 863 – 872.

Marks, M. & Niewiadomska Szynkiewicz, E. (2007). Two-phase stochastic optimization to sensor network localization. In *International Conference on Sensor Technologies and Applications, 2007. SensorComm 2007.* (pp. 134 –139).

Martínez, J.-F., Garcí, A.-B., Corredor, I., López, L., Hernández, V., & Dasilva, A. (2007). QoS in wireless sensor networks: survey and approach. In *Proceedings of the 2007 Euro American conference on Telematics and information systems*, EATIS '07 (pp. 20:1–20:8). New York, NY, USA: ACM.

Martins, F. V. C., Carrano, E. G., Wanner, E. F., Takahashi, R. H. C., & Mateus, G. R. (2010). An evolutionary dynamic approach for designing wireless sensor networks for real time monitoring. In *Proceedings of the 2010 IEEE/ACM 14th International Symposium on Distributed Simulation and Real Time Applications*, DS-RT '10 (pp. 161–168). Washington, DC, USA: IEEE Computer Society.

Martins, F. V. C., Carrano, E. G., Wanner, E. F., Takahashi, R. H. C., & Mateus, G. R. (2011). A hybrid multiobjective evolutionary approach for improving the performance of wireless sensor networks. *IEEE Sensors Journal*, 11(3), 545–554.

Martins, F. V. C., Nakamura, F. G., Quintao, F. P., & Mateus, G. R. (2007). Model and algorithms for the density, coverage and connectivity control problem in flat WSNs. In *Proc. International Network Optimization Conference (INOC'07)*.

Martins, F. V. C., Nakamura, F. G., & Takahashi, R. H. C. (2008). Uma análise multiobjetivo para o problema de cobertura e conectividade em uma rede de sensores sem fio plana. In *In Proc.: XL Simpósio Brasileiro de Pesquisa Operacional*.

Mascarenas, D., Flynn, E., Farrar, C., Park, G., & Todd, M. (2009). Powering and interrogation of structural health monitoring sensor networks. *IEEE Sensors Journal*, 9, 1719–1726.

Metropolis, N. & Ulam, S. (1949). The monte carlo method. *Journal of the American Statistical Association*, 44, 335–341.

Miu, A., Balakrishnan, H., & Koksal, C. E. (2005). Improving loss resilience with multi-radio diversity in wireless networks. In *Proceedings of the 11th annual international conference on Mobile computing and networking*, MobiCom '05 (pp. 16–30). New York, NY, USA: ACM.

Moraglio, A. (2007). *Towards a geometric unification of evolutionary algorithms*. PhD thesis, University of Essex.

Moraglio, A., Kim, Y., Yoon, Y., & Moon, B. (2007). Geometric crossovers for multiway graph partitioning. *Evolutionary Computation*, 14, 445–474.

Moraglio, A. & Poli, R. (2004). Topological implementation of crossover. In *Proc. Genetic and Evolutionary Computation Conference*.

Munir, S., Bin, Y. W., Biao, R., & Man, M. (2007). Fuzzy logic based congestion estimation for QoS in wireless sensor network. In *Wireless Communications and Networking Conference, 2007.WCNC 2007. IEEE* (pp. 4336 –4341).

Nakamura, E. F. (2007). *Fusão de dados em redes de sensores sem fio*. PhD thesis, Universidade Federal de Minas Gerais.

Nakamura, F., Martins, F. V. C., & Quintão, F. P. (2007). Controle de densidade em redes de sensores: Modelos e algoritmos. In *Anais do XXXIX SBPO - Simpósio Brasileiro de Pesquisa Operacional*.

Nakamura, F. G. (2010). *Algoritmos para controle de densidade em redes de sensores sem fio*. PhD thesis, Universidade Federal de Minas Gerais.

Nakamura, F. G., Quintao, F. P., Menezes, G. C., & Mateus, G. R. (2005). An optimal node scheduling for flat wireless sensor networks. In *Proc. IEEE International Conference Networking (ICN'05)*, volume 3420 (pp. 475–483).

Nan, G.-F., Li, M.-Q., & Li, J. (2007). Estimation of node localization with a real-coded genetic algorithm in WSNs. In *Machine Learning and Cybernetics, 2007 International Conference on*, volume 2 (pp. 873 –878).

Park, S., Savvides, A., & Srivastava, M. B. (2001). Simulating networks of wireless sensors. In *Proc. Conference on Winter Simulation (WSC'01)* (pp. 1330–1338). Washington, USA.

Park, S.-J., Vedantham, R., Sivakumar, R., & Akyildiz, I. F. (2004). A scalable approach for reliable downstream data delivery in wireless sensor networks. In *Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '04 (pp. 78–89). New York, NY, USA: ACM.

Podpora, J., Reznik, L., & Von Pless, G. (2008). Intelligent real-time adaptation for power efficiency in sensor networks. *IEEE Sensors Journal*, 8, 2066–2073.

Potts, J. C., Giddens, T. D., & Yadav, S. B. (1994). The development and evaluation of an improved genetic algorithm based on migration and artificial selection. *IEEE Transactions on Systems, Man and Cybernetics*, 1(24), 73–86.

Quintao, F., Nakamura, F., & Mateus, G. R. (2004). A Hybrid Approach to solve the Coverage and Connectivity Problem in Wireless Sensor Networks. In *Proceedings of the IV European Workshop on Meta-heuristics (EU/ME04)*, volume 1.

Quintao, F., Nakamura, F., & Mateus, G. R. (2005). Evolutionary Algorithm for the Dynamic Coverage Problem Applied to Wireless Sensor Networks Design. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC05)*, volume 2 (pp. 1589–1596).

Rajagopalan, R., Mohan, C., Varshney, P., & Mehrotra, K. (2005). Multi-objective mobile agent routing in wireless sensor networks. In *Proc. IEEE Congress on Evolutionary Computation*.

Rajagopalan, R. & Varshney, P. (2006). Data-aggregation techniques in sensor networks: A survey. *Communications Surveys Tutorials, IEEE*, 8(4), 48 –63.

Rajendran, V., Obraczka, K., & Garcia-Luna-Aceves, J. J. (2003). Energy-efficient collision-free medium access control for wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03 (pp. 181–192). New York, NY, USA: ACM.

Rakhmatov, D. & Vrudhula, S. (2003). Energy management for battery-powered embedded systems. *ACM Transactions in Embedded Computing Systems*, 2(3), 277–324.

Ruiz, L. B. (2003). *MANÁ: Uma Arquitetura para Gerenciamento de Redes de Sensores Sem Fio*. PhD thesis, Universidade Federal de Minas Gerais.

Ruiz, L. B., Correia, L. H. A., Vieira, L. F. M., Macedo, D. F., Nakamura, E. F., Figueiredo, C. M. S., Vieira, M. A. M., Bechelane, E. H., Camara, D., Loureiro, A. A., Nogueira, J. M. S., da Silva Jr., D. C., & Fernandes, A. O. (2004a). *Arquiteturas para Redes de Sensores Sem Fio*, chapter IV. SBRC Tutorial.

Ruiz, L. B., Nogueira, J. M., & Loureiro, A. A. F. (2004b). *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*, chapter Sensor Networks Management. CRC Press.

R.Williams (1979). *The Geometrical Foundation of Natural Structure: A Source Book of Design*. New York: Dover.

Sareni, B. & Krahenbuhl, L. (1998). Fitness sharing and niching methods revisited. *IEEE Transactions on Evolutionary Computation*, 3(2), 97–106.

Saxena, N., Roy, A., & Shin, J. (2008). Dynamic duty cycle and adaptive contention window based QoS-MAC protocol for wireless multimedia sensor networks. *Computer Networks*, 52(13), 2532–2542.

Schneider, K. K., Sausen, A. T. Z. R., & Sausen, P. S. (2011). Análise comparativa do tempo de vida de baterias em dispositivos móveis a partir da utilização de modelos analíticos. *Tendência em Matemática Aplicada e Computacional (TEMA)*, 12, 43–54.

Takahashi, R. H. C. (2004). Otimização escalar e vetorial. Notas de Aula.

Takahashi, R. H. C. & Martins, F. V. C. (2004). Avaliação e interpretação do operador mag com algoritmo genético real-polarizado. In *Inproc: XXXVI Simpósio Brasileiro de Pesquisa Operacional*.

Takahashi, R. H. C., Vasconcelos, J. A., Ramirez, J. A., & Krahenbuhl, L. (2003). A multiobjective methodology for evaluating genetic operators. *IEEE Transactions on Magnetics*, 3(39), 1321–1324.

Tilak, S., Abu-Ghazaleh, N. B., & Heinzelman, W. (2002). Infrastructure tradeoffs for sensor networks. In *Proc. ACM International Workshop on Wireless Sensor Networks and Application (WSNA'02)* (pp. 49–58).

Tsow, F., Forzani, E., Rai, A., Wang, R., Tsui, R., Mastroianni, S., Knobbe, C., Gandolfi, A. J., & Tao, N. J. (2009). A wearable and wireless sensor system for real-time monitoring of toxic environmental volatile organic compounds. *IEEE Sensors Journal*, 9, 1734–1740.

Vasconcelos, J. A., Ramirez, J. A., Takahashi, R. H. C., & Saldanha, R. R. (2001). Improvements in genetic algorithms. *IEEE Transactions on Magnetics*, 5(37), 3414–3417.

Vieira, M., Coelho, C.N., J., da Silva, D.C., J., & da Mata, J. (2003). Survey on wireless sensor network devices. *Emerging Technologies and Factory Automation, 2003. Proceedings. ETFA '03. IEEE Conference*, 1, 537–544 vol.1.

Wazed, S., Bari, A., Jaekel, A., & Bandyopadhyay, S. (2007). Genetic algorithm based approach for extending the lifetime of two-tiered sensor networks. In *Wireless Pervasive Computing, 2007. ISWPC '07. 2nd International Symposium on*.

Wu, Q., Rao, N. S. V., Barhen, J., Iyengar, S. S., Vaishnavi, V. K., Qi, H., & Chakrabarty, K. (2004). On computing mobile agent routes for data fusion in distributed sensor networks. *IEEE Transactions on Knowledge and Data Engineering*, 16(6), 740–753.

XBOW (2006a). *Mica2dot - Wireless Measurement System*. XBOW, available in *http://www.xbow.com/*.

XBOW (2006b). *MICAz - Wireless Measurement System*. XBOW, available in *http://www.xbow.com/*.

Ye, F., Zhong, G., Cheng, J., Lu, S., & Zhang, L. (2003). PEAS: A robust energy conserving protocol for long-lived sensor networks. In *Proc. International Conference on Distributed Computing Systems (ICDCS'03)* (pp. 28–37).

Yick, J., Mukherjee, B., & Ghosal, D. (2008). Wireless sensor network survey. *Computer Networks*, 52(12), 2292–2330.

Yigitel, M. A., Durmaz Incel, O., & Ersoy, C. (2010). Diff-MAC: A QoS-aware MAC protocol with differentiated services and hybrid prioritization for wireless multimedia sensor networks. In *Proceedings of the 6th ACM workshop on QoS and security for wireless and mobile networks*, Q2SWinet '10 (pp. 62–69). New York, NY, USA: ACM.

Yin, J. & Madria, S. (2006). SecRout: A secure routing protocol for sensor networks. In *Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 01*, AINA '06 (pp. 393–398). Washington, DC, USA: IEEE Computer Society.

Yu, Y., Prasanna, V. K., & Krishnamachari, B. (2006). Energy minimization for real-time data gathering in wireless sensor networks. *IEEE Transactions on Wireless Communication*, 5, 3087–3096.

Zhang, R., Zhao, H., & Labrador, M. A. (2006). The Anchor Location Service (ALS) protocol for large-scale wireless sensor networks. In *Proceedings of the first international conference on Integrated internet ad hoc and sensor networks*, InterSense '06 New York, NY, USA: ACM.

Zhao, L. & Liang, Q. (2005). Fuzzy deployment for wireless sensor networks. In *Computational Intelligence for Homeland Security and Personal Safety, 2005. CIHSPS 2005. Proceedings of the 2005 IEEE International Conference on* (pp. 79 –83).

Zhou, Y., Lyu, M. R., & Liu, J. (2005). Port: A price-oriented reliable transport protocol for wireless sensor networks. In *In International Symposium on Software Reliability Engineering (ISSRE)* (pp. 117–126).

APÊNDICE A

EXPERIMENTOS COM DIFERENTES OPERADORES

Para chegar à conclusão de que o algoritmo a ser utilizado seria o Algoritmo Genético Real Polarizado, alguns testes com variações de algoritmos genéticos (AG) foram realizados. Todas as versões aqui apresentadas seguiram o mesmo processo de codificação, decodificação e avaliação de função apresentados no Capítulo 4. Suas diferenças giram em torno dos operadores de cruzamento e mutação. Cinco diferentes tipos de AG foram testados, são eles:

- **NM -** uma versão simples de um AG, sem a extrapolação ou polarização pelo operador de cruzamento apresentados na Seção 4.4.2.
- **RP -** versão com operador de cruzamento real polarizado (Seção 4.4.2).
- **PL -** uma variação de **RP**, utilizando apenas polarização, ou seja, sem possibilidade de extrapolação durante o cruzamento.
- **EX -** uma variação de **RP**, utilizando apenas extrapolação, sem uso de polarização pelos objetivos.
- **MV -** a mesma versão **NM** com um operador de mutação variável. Diferente do operador apresentado na Seção 4.4.1, a escolha da quantidade de movimentos não têm igual probabilidade. No **MV** a probabilidade para escolha da quantidade de movimentos é a seguinte:

  1 movimento: 50%;

  2 movimentos: 30%;

  3 movimentos: 20%.

Utilizando como referência o algoritmo genético para redes de sensores sem fio adotado em Martins et al. (2010), foi utilizado a instância de 100 sensores, o tamanho da população foi fixado em 300 indivíduos e o número de gerações do AG foi fixado em 50. Como as versões alteram os operadores de cruzamento e mutação, o desempenho de cada algoritmo pode ser sensível aos parâmetros adotados no AG. Então foram utilizadas duas probabilidades de cruzamento ($p_c = 0,6$ e $p_c = 0,9$) e três probabilidades de mutação ($p_m = 0,10$, $p_m = 0,25$ e $p_m = 0,40$) testadas em conjunto para identificar a melhor configuração destes parâmetros para cada algoritmo. Os resultados para 21 execuções de cada algoritmo podem ser vistos na Tabela A.1, onde:

- $\overline{X}$: a média das soluções;
- s: o desvio padrão;

- M: o melhor resultado atingido por cada algoritmo;
- P: o pior resultado atingido por cada algoritmo.

Tabela A.1 Resultado para as probabilidades de mutação e cruzamento em diferentes operadores

	pm	0,10				0,25				0,40			
	pc	$\overline{X}$	s	M	P	$\overline{X}$	s	M	P	$\overline{X}$	s	M	P
NM	0,6	164,5	7,7	179	151	162,1	9,5	180	145	161,6	7,6	179	147
	0,9	166,2	8,8	180	145	165,6	7,6	177	153	173,8	7,2	180	157
RP	0,6	**167,7**	**9,9**	**180**	**152**	**174,7**	**7,6**	**180**	**152**	**174,4**	**6,4**	**180**	**158**
	0,9	**176,4**	**5,6**	**180**	**163**	**174,4**	**5,5**	**180**	**165**	**175,7**	**6,4**	**180**	**161**
PL	0,6	163,9	7,8	179	154	162,4	7,6	175	152	166,1	8,5	179	148
	0,9	172,2	7,5	180	155	169,0	7,7	180	155	171,9	6,2	180	159
EX	0,6	167,3	5,5	180	156	173,8	4,1	180	167	173,1	5,6	180	159
	0,9	168,1	5,4	178	159	169,9	4,9	178	161	170,5	5,4	180	157
MV	0,6	164,3	10,0	180	148	165,2	4,4	174	158	162,6	7,7	180	152
	0,9	169,2	8,8	180	154	169,8	7,1	180	160	167,9	7,7	180	154

A partir da tabela apresentada fica claro que o algoritmo **RP** foi a que apresentou os melhores resultados. A média ($\overline{X}$) em todas as combinações de pm e pc superaram os demais algoritmos. Os demais índices de comparação, desvio padrão (s), melhor (M) e pior (P) soluções, foram superiores na maioria dos casos, justificando a escolha do Algoritmo Real Polarizado.

APÊNDICE B

AJUSTE DE PARÂMETROS

Os algoritmos propostos possuem quatro parâmetros a serem estabelecidos: n_{evl}, S_{pop}, p_m e p_c. Para ajustar os parâmetros: S_{pop}, p_m e p_c, foi utilizado a instância de 100 sensores, a mesma apresentada em Martins et al. (2011, 2010). Já o parâmetro n_{evl} foi ajustado em 28.000 para todos os experimentos, dado que este é o valor estimado do número de avaliações de função nas referências mencionada.

A avaliação dos parâmetros foi dividida em três experimentos:

- Para a primeira avaliação de parâmetros, o tamanho da população foi fixada em 300 indivíduos, como adotado em Martins et al. (2010). Então, duas probabilidades de cruzamento ($p_c = 0,6$ e $p_c = 0,9$) e três probabilidades de mutação ($p_m = 0,10$, $p_m = 0,25$ e $p_m = 0,40$) foram testadas em conjunto para identificar a melhor configuração destes parâmetros para o algoritmo.

- No segundo experimento, o conjunto de probabilidades escolhidos no primeiro teste é utilizado para avaliar cinco tamanhos de população: 100, 200, 300, 400 e 500 indivíduos.

- Por fim, uma análise é realizada para verificar se o valor escolhido para n_{evl} é satisfatório.

Para cada conjunto de parâmetro adotado, o algoritmo foi executado 21 vezes, e ao final de cada execução foi registrado o valor da função objetivo que mede o tempo de vida da rede. Foram considerados quatro indicadores para comparação: média das amostras ($\overline{X}$), desvio padrão das amostras (s), o melhor (melhor) e o pior valor presente (pior).

Sabe-se que as probabilidades dos operadores e o tamanho da população podem ter alguma dependência, no entanto, realizar um teste combinatório entre todas estas variáveis conduziria a um número muito elevado de execuções do algoritmo, o que seria necessário um tempo computacional considerável. Portanto, nenhuma dependência entre as probabilidades dos operadores e o tamanho da população foi considerada nos ajustes de parâmetros.

Os resultados obtidos no primeiro experimento são apresentados na Tabela B.1. Com base nesta tabela, é possível notar que o conjunto ($p_m = 0,10, p_c = 0,90$) foi o que atingiu melhores resultados, pois foi o que alcançou melhores valores nos indicadores: média, melhor e pior, e o desvio padrão foi ligeiramente maior do que o observado para o conjunto ($p_m = 0,25, p_c = 0,90$).

Dado o resultado do primeiro teste, no segundo experimento as probabilidades de mutação e cruzamento foram definidas com os seguintes valores: $p_m = 0,10$ e $p_c = 0,90$. A partir da fixação destes parâmetros, o tamanho da população foi avaliado. Tais resultados

Tabela B.1 Resultado para as probabilidades de mutação e cruzamento

p_m	***0,10***				0,25				0,40			
p_c	$\overline{X}$	s	melhor	pior	$\overline{X}$	s	melhor	pior	$\overline{X}$	s	melhor	pior
0,60	167,7	9,9	180	152	174,7	7,6	180	152	174,4	6,4	180	158
0,90	**176,4**	**5,6**	**180**	**163**	174,4	5,5	180	165	175,7	6,4	180	161

são mostrados na Tabela B.2. É possível notar que o tamanho da população de 200 indivíduos foi o que atingiu os melhores resultados, uma vez que atingiu os melhores valores para os quatro indicadores escolhidos para comparação.

Tabela B.2 Resultados para o tamanho da população

S_{pop}	$\overline{X}$	s	melhor	pior
100	174,1	11,4	180	142
200	**176,86**	**4,7**	**180**	**164**
300	176,4	5,6	180	163
400	174,6	5,9	180	163
500	174,5	7,0	180	160

Com base nos experimentos de ajuste de parâmetros, para os demais experimentos deste trabalho foram adotados os seguintes valores para cada parâmetro:

- $S_{pop} = 200$,
- $p_m = 0,10$,
- $p_c = 0,90$.

Por fim, é feito uma análise para saber se número de avaliações de função objetivo fixada em 28.000 é suficiente para convergência do algoritmo. A partir da Figura B.1 é fácil notar que em média a função objetivo da população começa a estabilizar por volta de 15.000 avaliações. Quatro curvas são mostradas, seguindo as mesmas características.

Com intuito de medir a convergência da população até a região próxima ao ótimo, foi escolhido no espaço de objetivos o valor de 180 *u.t.* como um ponto de referência ótimo (melhor valor encontrado em todos os testes apresentados pelo WSNdsGA). Na Figura B.2 é mostrado um gráfico contendo: a distância média da população em relação ao valor de função objetivo da referência ótima estabelecida versus o número de avaliações de função. É possível notar que a população do algoritmo genético presente no WSNdsGA começa a concentrar próximo a região da referência ótima por volta de 15.000 avaliações, e a partir deste ponto começa a convergir mais lentamente. Já a Figura B.3 mostra a velocidade de convergência da melhor e da pior solução encontrada dentre as 21 execuções. A partir deste gráfico é fácil perceber que a melhor solução encontra o ótimo por volta

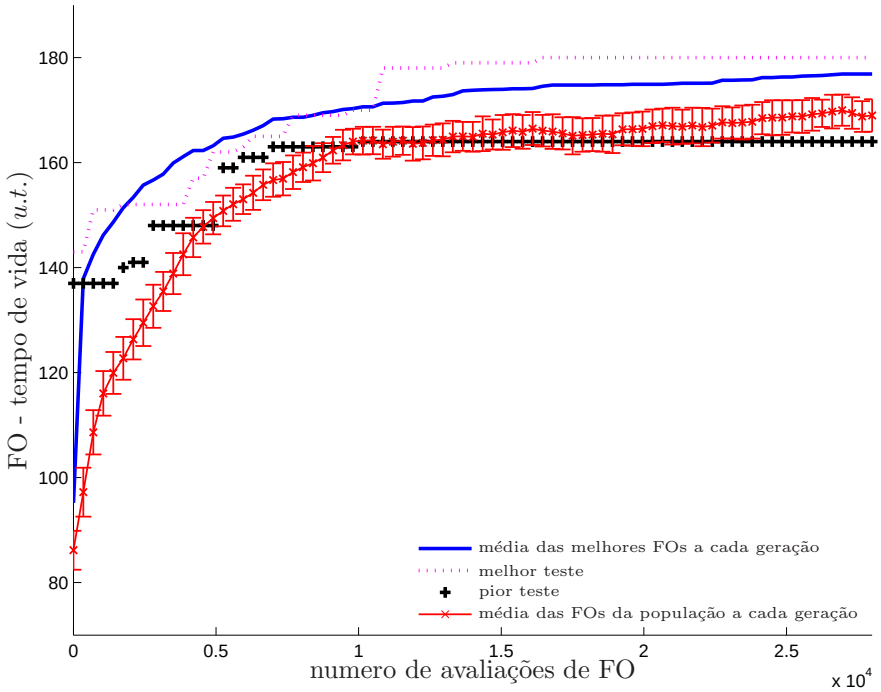


Figura B.1 Valor da função objetivo x número de avaliações. A barra de erro mostrada na média das FOs de toda a população foi baseada no desvio padrão.

de 16.000 avaliações, já a pior solução obtém uma convergência prematura por volta de 10.000 avaliações de função. Com base nestes dados apresentados é possível concluir que o valor de 28.000 estipulado para o número máximo de avaliações de função objetivo (n_{evl}) é um valor satisfatório, pois ao atingir este número de avaliações o algoritmo se apresentou estável de acordo com os parâmetros estabelecidos.

Embora o ajuste de parâmetros tenha sido realizado apenas para o algoritmo mono-objetivo, os mesmos parâmetros também foram adotados na versão multiobjetivo.

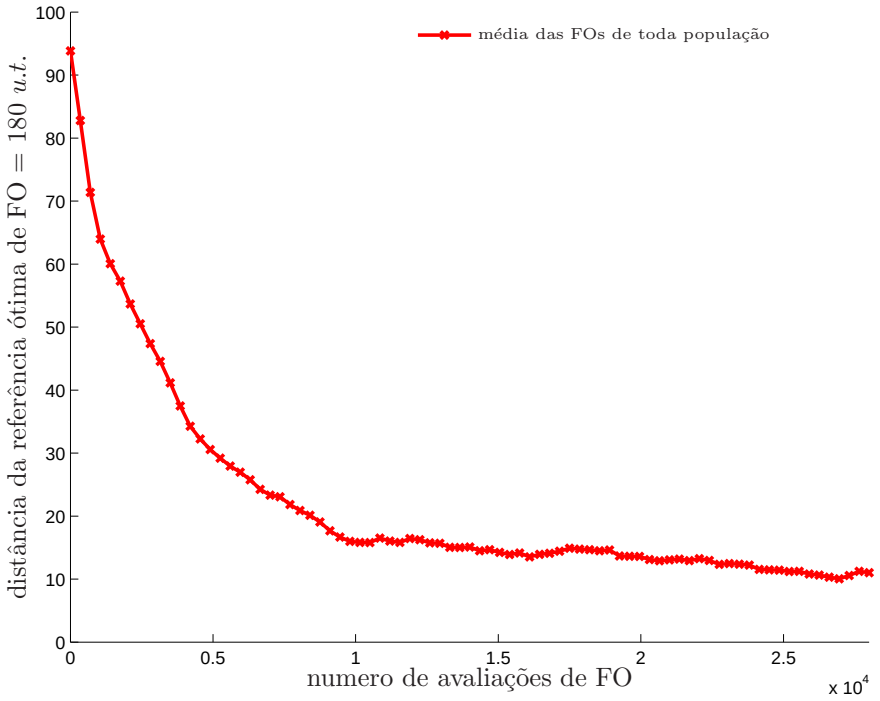


Figura B.2 Distância média da melhor FO x número de avaliações.

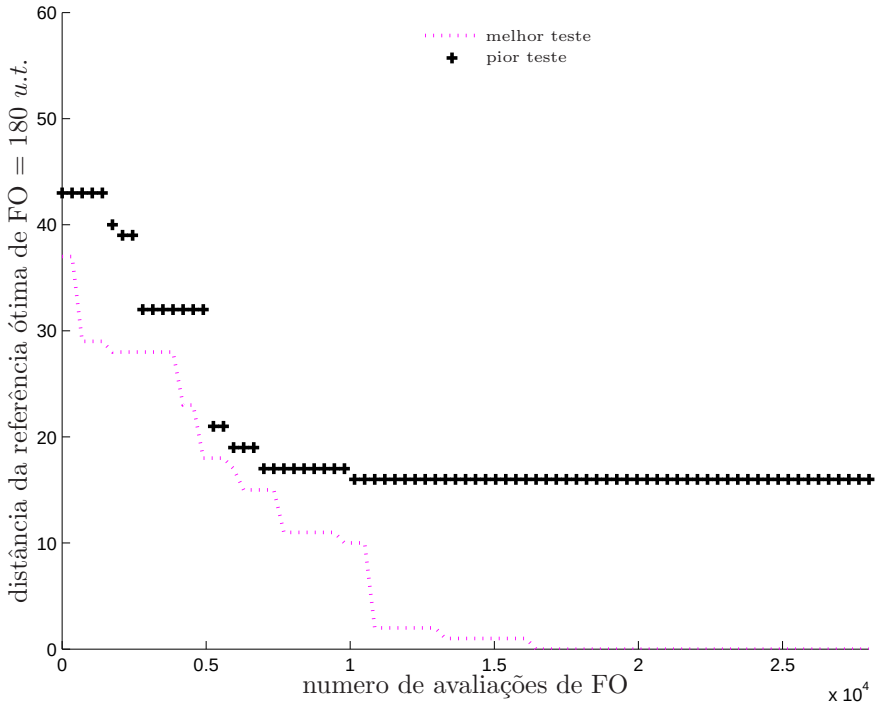


Figura B.3 Valor da função objetivo x número de avaliações.

APÊNDICE C

INSTÂNCIAS UTILIZADAS

Neste apêndice serão descritas mais detalhadamente as instâncias utilizadas em cada um dos testes apresentados nesta tese, são elas:

- Coordenadas dos sensores;
- 21 melhores soluções do WSNdsGA para a instância de 100 sensores;
- 10 instâncias de cargas iniciais dos sensores.

C.1 COORDENADAS DOS SENSORES

Como já informado a posição de cada sensor na região foi gerada de forma aleatória, simulando um lançamento por uma aeronave. A distribuição usada foi a uniforme, ou seja, para cada ponto da área, a probabilidade de surgir um sensor é igual. Foi gerado então, de forma aleatória instâncias de 36, 49, 64, 81 e de 100 sensores em uma área de $2500m^2$. É importante observar que a instância de 49 sensores foi gerada a partir da instância de 36 sensores, ou seja, adicionando mais 13 sensores de forma aleatória e uniforme. As demais instâncias foram geradas da mesma forma, sempre adicionando os sensores na rede já existente. Abaixo são mostradas as coordenadas dos 100 sensores utilizados. Caso o leitor queira saber a instância de 36 sensores, basta informar as 36 primeiras coordenadas listadas. Para as demais instâncias basta utilizar o mesmo processo, pois as coordenadas estão listadas de forma sequencial.

id	x_1	x_2
0	15.006539	20.417550
1	45.943058	15.892827
2	2.700347	27.428959
3	10.252378	42.678584
4	5.207658	46.074841
5	38.852614	35.174274
6	28.007046	49.311303
7	33.255557	7.530841
8	43.678227	45.186682
9	30.491431	3.689700
10	46.941936	2.626156
11	44.278614	26.181380
12	23.244782	32.477366
13	7.466317	6.669233
14	16.968735	7.006792
15	31.160181	24.879301

id	x_1	x_2
16	39.169601	5.442743
17	31.118732	8.397891
18	2.208403	18.525066
19	3.928493	34.247601
20	23.557755	48.979934
21	21.290877	2.595716
22	49.511591	4.535577
23	27.322586	4.052772
24	47.133226	33.936341
25	36.305246	0.194194
26	35.649047	46.633143
27	20.958661	12.021002
28	28.979833	49.207167
29	12.758464	35.287904
30	30.522097	36.440446
31	31.152092	44.744417

id	x_1	x_2
32	45.136691	21.999393
33	30.333862	49.162189
34	49.765017	39.139981
35	45.925896	27.729502
36	9.920917	14.352499
37	26.539219	43.142407
38	4.893781	19.485067
39	22.075030	22.132684
40	15.460105	20.935468
41	39.382078	48.945534
42	39.724311	30.083717
43	27.857609	9.862230
44	35.075057	28.783420
45	2.918275	48.342984
46	15.004159	33.792357
47	2.379834	0.977780
48	45.415975	20.396204
49	30.700448	41.927758
50	13.117078	45.972703
51	42.069353	30.301051
52	14.570887	42.777151
53	11.158535	10.242968
54	11.296715	23.901715
55	14.161682	19.060050
56	41.460292	21.207534
57	27.625718	24.872471
58	25.368018	4.421279
59	30.151622	29.104264
60	33.077355	1.055168
61	29.330868	0.236150
62	46.119512	27.704451
63	24.959614	5.600514
64	34.910338	1.768979
65	37.553466	26.211055
66	12.492128	19.961725
67	12.457367	44.971536
68	48.210367	8.571748
69	17.388831	1.235923
70	23.697485	29.609128
71	43.048912	28.510964
72	5.580006	29.123091
73	19.714216	17.066903
74	42.663657	27.977985
75	36.967120	19.883684
76	38.124119	47.284475
77	11.773028	43.825201
78	37.851403	49.422472
79	11.607776	19.568494
80	23.042955	46.005762
81	22.875353	44.043346
82	33.082143	42.812992
83	26.072492	29.977132
84	49.145061	24.112329
85	44.983459	0.600445
86	31.510220	13.939457
87	17.341297	28.706474
88	15.291968	41.396001
89	45.061950	35.228722
90	5.329803	17.092921
91	14.815935	20.101318
92	42.577868	26.945195
93	36.470679	9.487227
94	32.139082	21.734568
95	13.409387	20.434167
96	36.406493	10.281825
97	44.610249	14.669699
98	25.496457	12.806153
99	27.654486	15.630539

A projeção destes sensores em uma área de $2500m^2$ pode ser vista na Figura C.1. Esta figura foi retirada da ferramenta JSensors que será introduzida no Apêndice D.

C.2 SOLUÇÕES DO WSNDSGA PARA 100 SENSORES

As 21 soluções mostrada a seguir são fruto das 21 execuções do algoritmo WSNdsGA apresentados na Seção 6.2. Estes são os mesmos sequenciamentos utilizados nas abordagens *online* cujos resultados são apresentados na Seção 6.4. As 21 sequências são:

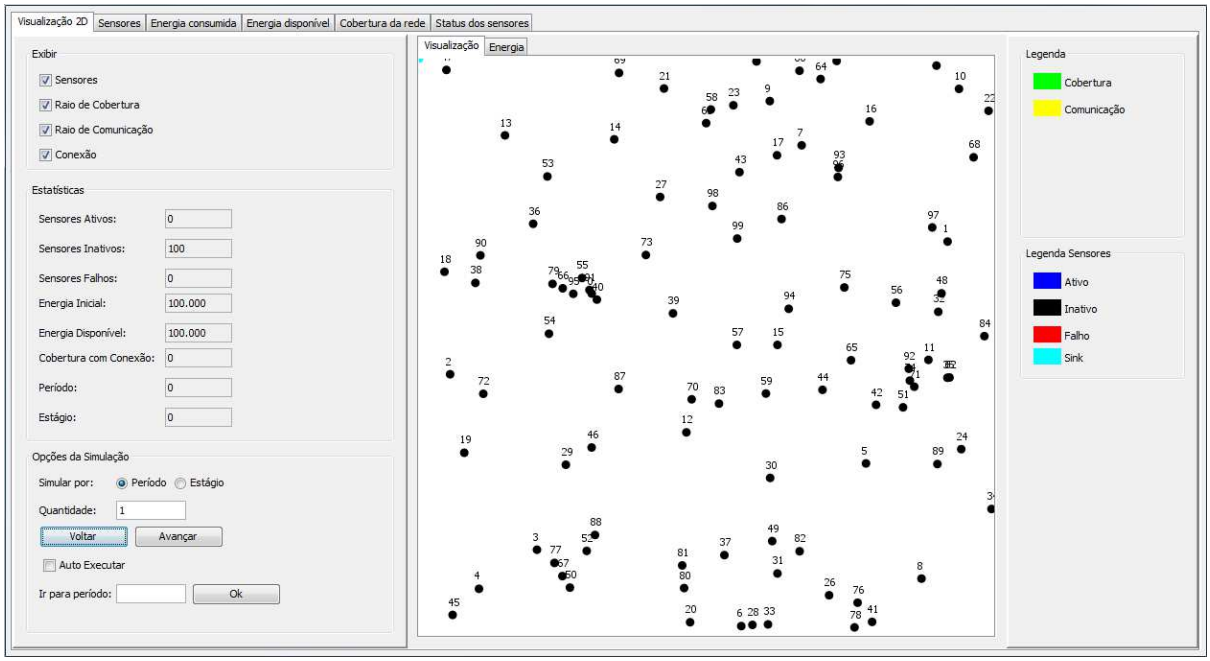


Figura C.1 Projeção dos 100 sensores em uma área de $2500m^2$.

Sequenciamento 1																	
14	96	42	31	83	8	25	79	57	84	92	35	17	22	47	11	81	···
10	93	77	90	51	94	49	3	37	55	13	34	60	6	50	80	30	···
69	98	86	20	24	75	97	27	15	64	68	21	16	1	78	19	63	···
95	46	71	66	44	87	36	23	48	43	76	7	45	82	91	29	38	···
85	18	89	32	61	41	72	26	39	40	70	65	67	58	62	12	5	···
28	53	9	88	74	33	52	2	4	73	0	54	99	56	59			

Sequenciamento 2																	
26	52	4	71	82	69	94	73	8	95	21	42	0	5	49	66	93	···
41	90	13	96	85	92	88	22	30	33	46	51	63	37	58	62	15	···
59	99	7	16	56	68	48	9	35	80	53	38	86	50	91	55	11	···
36	87	14	10	45	18	12	32	29	57	25	24	34	77	78	43	31	···
40	98	1	28	2	83	74	72	17	67	81	23	27	20	44	39	61	···
60	79	97	65	75	64	19	6	70	89	54	47	84	3	76			

Sequenciamento 3																	
74	25	59	60	44	13	64	29	46	34	3	36	66	32	42	80	91	···
97	31	45	92	55	53	71	73	76	12	33	96	52	38	17	79	62	···
21	19	14	23	89	86	24	61	8	72	39	82	30	41	1	9	50	···
84	85	26	95	49	18	22	7	68	65	83	51	87	10	94	48	56	···
69	20	47	70	98	27	75	77	99	35	88	16	11	40	90	2	6	···
93	81	4	0	67	63	15	43	28	37	57	58	54	78	5			

Sequenciamento 4																	
50	44	75	34	96	31	53	92	24	64	15	14	52	13	1	17	12	···
7	45	20	33	36	90	81	46	25	32	6	37	79	23	62	55	39	···
16	94	80	48	68	91	93	82	51	86	76	27	47	95	56	71	66	···
9	77	49	59	99	19	70	26	60	29	83	43	0	67	84	3	78	···
2	4	61	87	98	72	85	42	22	11	8	21	73	30	58	74	5	···
89	41	57	63	69	97	54	35	10	65	18	28	40	38	88			

Sequenciamento 5																	
42	4	30	77	11	91	14	6	98	79	23	26	28	86	37	93	71	···
87	39	94	49	34	90	27	84	65	78	50	38	67	12	66	88	53	···
31	21	64	29	69	55	16	44	68	41	32	13	25	3	61	80	47	···
51	81	83	56	33	18	73	52	74	40	75	35	5	20	45	62	85	···
24	82	9	89	46	60	97	10	70	17	99	43	57	0	96	22	8	···
1	48	63	92	72	95	7	76	58	59	2	54	36	19	15			

Sequenciamento 6																	
93	57	55	98	87	21	28	46	51	69	82	71	70	76	56	9	37	···
74	3	66	96	91	58	63	83	44	20	89	13	17	7	23	59	25	···
16	94	92	60	36	10	78	4	19	90	34	95	72	75	43	81	97	···
77	64	86	8	73	6	39	0	18	29	30	79	84	67	85	68	80	···
65	27	40	41	35	22	52	53	38	88	2	11	62	47	24	48	33	···
42	32	15	54	1	14	49	12	5	26	50	99	61	45	31			

Sequenciamento 7																	
52	69	74	11	64	17	93	89	38	61	94	85	95	90	75	30	4	···
62	13	66	71	21	80	46	8	79	53	3	77	14	44	72	16	65	···
41	29	1	81	67	57	6	24	91	88	5	50	45	34	87	28	18	···
59	58	26	27	2	63	25	39	73	15	7	55	32	48	10	92	33	···
23	76	19	60	47	82	31	43	97	70	49	51	12	54	36	0	56	···
22	98	9	68	40	96	86	20	78	42	84	35	83	37	99			

Sequenciamento 8																
75	71	49	53	69	25	43	37	9	27	92	78	34	3	1	95	44	···
84	38	90	26	12	60	30	45	77	10	73	89	98	59	40	32	93	···
4	29	65	88	68	52	17	66	22	5	61	8	50	18	67	55	14	···
58	41	96	51	74	87	7	23	35	28	16	20	0	70	99	39	48	···
82	64	31	46	86	21	94	11	79	91	62	13	80	83	36	2	19	···
76	97	85	24	33	15	6	56	47	57	72	54	81	42	63			

Sequenciamento 9																
29	19	60	55	24	77	46	66	35	8	53	56	71	91	67	95	49	···
99	13	33	25	47	78	28	34	96	5	26	70	90	57	83	27	85	···
87	38	30	79	88	50	82	65	72	41	64	84	52	7	14	1	76	···
23	39	21	73	17	98	10	20	37	97	93	51	92	48	16	32	59	···
61	69	58	18	40	11	44	54	4	45	3	2	22	94	81	74	0	···
36	86	31	15	63	75	12	80	68	43	89	42	6	62	9			

Sequenciamento 10																
38	14	55	79	32	11	49	69	31	80	60	8	96	33	16	53	42	···
78	15	90	34	27	93	59	40	67	91	89	68	46	86	56	50	84	···
48	95	45	18	22	35	30	25	13	36	44	82	41	62	6	97	10	···
9	19	87	65	98	3	88	20	37	2	66	24	99	12	74	7	72	···
5	57	61	43	77	28	71	83	63	23	26	81	64	47	0	51	21	···
17	92	54	4	58	85	94	70	75	76	1	39	73	52	29			

Sequenciamento 11																
95	30	13	38	60	56	68	96	7	51	8	67	59	15	22	69	53	···
55	66	34	80	50	58	86	90	73	65	20	27	10	32	14	1	35	···
97	33	63	29	28	70	61	72	62	89	24	75	17	0	12	74	93	···
41	44	25	94	6	21	16	76	82	79	31	57	52	84	18	81	26	···
78	46	88	71	99	85	40	49	48	4	23	45	3	43	5	19	39	···
54	47	36	11	98	92	37	87	2	77	83	64	42	9	91			

Sequenciamento 12																
53	90	26	88	78	65	46	29	67	14	69	96	21	81	55	44	13	···
19	16	91	95	50	3	66	92	10	5	24	52	83	97	51	68	70	···
15	1	38	6	32	7	93	85	47	8	4	18	30	39	41	9	36	···
71	45	56	27	73	2	60	49	74	82	86	23	35	76	64	77	63	···
43	62	57	33	89	31	28	59	94	22	48	37	12	0	72	61	99	···
58	98	80	17	20	87	11	34	25	42	84	54	79	40	75			

Sequenciamento 13																	
28	66	6	53	24	75	89	68	30	11	8	21	94	31	92	87	51	···
69	27	63	17	77	12	23	97	50	22	64	65	78	93	52	79	80	···
14	26	85	55	95	42	19	74	47	20	44	57	46	35	99	60	67	···
38	29	33	16	45	37	90	62	2	41	48	40	83	7	59	25	56	···
39	32	86	5	88	58	34	49	76	0	4	81	91	70	71	73	82	···
43	98	18	9	72	84	36	1	96	15	10	13	54	61	3			

Sequenciamento 14																	
52	46	53	14	25	11	35	24	27	65	5	73	64	57	74	17	21	···
29	3	82	51	66	22	62	76	38	97	36	94	69	72	49	7	79	···
75	67	18	50	96	70	90	33	98	47	9	48	59	45	71	84	12	···
19	95	15	85	41	28	56	87	92	91	99	1	4	44	26	54	34	···
60	32	6	23	68	80	43	2	83	89	78	86	0	39	63	77	16	···
42	58	40	55	13	93	20	31	88	8	37	81	61	10	30			

Sequenciamento 15																	
51	75	91	5	8	96	90	53	64	20	59	88	12	78	6	60	67	···
79	29	7	69	48	50	99	73	82	71	30	18	11	46	22	55	2	···
39	17	93	89	36	66	35	9	83	54	87	47	84	10	4	86	28	···
95	16	92	0	42	27	34	57	41	72	37	19	44	97	70	14	38	···
26	45	3	77	58	76	61	68	25	81	40	15	94	32	31	24	52	···
1	80	56	85	98	21	74	33	65	13	23	43	49	63	62			

Sequenciamento 16																	
26	99	27	18	97	92	34	7	40	16	70	30	51	22	95	90	33	···
61	29	96	48	84	91	93	71	11	56	64	3	98	79	13	68	76	···
59	41	8	65	87	75	25	55	53	23	17	94	15	37	78	5	19	···
1	77	89	88	46	69	62	73	57	52	24	39	44	74	38	6	80	···
54	32	45	72	63	60	0	9	36	49	10	66	35	31	50	82	42	···
12	47	43	67	21	20	86	83	28	58	81	4	2	14	85			

Sequenciamento 17																	
27	84	94	64	36	48	53	1	30	67	28	12	68	73	77	4	78	···
26	63	47	56	51	45	34	37	21	57	3	42	80	91	69	95	55	···
89	62	98	15	35	59	14	23	83	38	44	88	9	86	16	13	33	···
41	82	6	24	74	97	20	60	2	75	81	50	29	32	25	43	10	···
0	39	17	90	18	7	79	49	85	22	66	8	46	54	52	31	76	···
96	65	61	87	92	5	40	70	72	19	11	99	71	58	93			

Sequenciamento 18																	
11	47	53	29	91	21	43	39	50	85	95	16	76	92	3	44	65	···
37	25	4	72	36	82	55	78	33	31	88	38	14	19	80	20	90	···
58	79	28	12	34	93	94	41	87	70	75	45	6	46	63	23	66	···
27	56	49	26	24	98	13	51	18	83	62	81	71	48	64	84	99	···
2	5	15	30	1	74	97	61	69	54	89	35	73	10	8	17	7	···
22	40	52	96	59	57	60	9	68	67	77	42	86	32	0			

Sequenciamento 19																	
93	79	6	83	55	34	46	13	64	21	4	62	14	49	7	19	82	···
8	11	60	65	26	20	77	71	24	27	75	69	89	38	25	41	48	···
53	84	70	17	99	50	95	43	92	72	63	33	94	59	73	2	12	···
58	98	88	23	9	87	29	37	81	45	76	3	78	90	5	91	57	···
52	16	80	36	44	47	30	18	61	39	35	56	86	66	42	97	31	···
1	67	68	85	10	0	51	22	32	15	96	54	74	40	28			

Sequenciamento 20																	
86	18	78	88	75	93	16	20	76	25	4	84	85	62	13	60	90	···
55	8	36	69	66	15	27	46	91	24	53	38	70	28	3	81	80	···
77	68	64	95	11	6	23	22	43	67	34	10	44	61	29	65	14	···
0	99	37	98	71	30	63	79	49	42	48	9	87	89	57	92	51	···
47	52	17	59	72	19	82	96	97	41	56	83	21	7	32	50	1	···
94	39	5	74	54	58	40	12	33	2	26	35	45	31	73			

Sequenciamento 21																	
71	1	50	45	28	93	88	60	36	61	4	2	78	7	27	96	63	···
64	90	68	69	67	85	21	56	48	81	99	26	24	18	17	47	51	···
38	80	20	55	97	3	53	77	62	66	65	72	12	25	59	89	22	···
49	82	58	32	83	34	29	19	13	8	5	57	30	98	86	42	6	···
52	87	10	44	33	54	37	46	76	11	94	73	35	0	16	39	84	···
31	9	91	70	95	41	15	79	74	75	23	43	40	92	14			

C.3 INSTÂNCIAS PARA CARGAS INICIAIS DE 100 SENSORES

Aqui são apresentadas as 10 instâncias geradas a partir do Algoritmo 11, no qual altera a carga inicial das baterias. Estas instâncias foram combinadas com as 21 soluções apresentadas na seção anterior, gerando assim as 210 instâncias utilizadas nos testes da Parte A, apresentados na Seção 6.4.1. O intuito destas instâncias foi forçar um ambiente com falhas prováveis (imprevisíveis).

Instância 1							
id	EB_{real}	id	EB_{real}	id	EB_{real}	id	EB_{real}
0	1054	25	1103	50	914	75	860
1	1183	26	1073	51	1008	76	858
2	774	27	970	52	879	77	1049
3	1086	28	1029	53	889	78	982
4	1032	29	921	54	999	79	980
5	869	30	1089	55	1153	80	1142
6	957	31	885	56	923	81	1029
7	1034	32	893	57	1037	82	1020
8	1358	33	919	58	977	83	1159
9	1277	34	706	59	1112	84	920
10	865	35	1144	60	891	85	1070
11	1303	36	1033	61	1003	86	1084
12	1073	37	925	62	1055	87	976
13	994	38	1137	63	1110	88	1022
14	1071	39	829	64	1154	89	883
15	980	40	990	65	1009	90	885
16	988	41	976	66	851	91	1010
17	1149	42	1032	67	926	92	1072
18	1141	43	1031	68	894	93	1259
19	1142	44	914	69	1235	94	933
20	1067	45	997	70	938	95	1019
21	879	46	984	71	1075	96	992
22	1072	47	1063	72	981	97	807
23	1163	48	1109	73	1089	98	956
24	1049	49	1111	74	924	99	821

Instância 2							
id	EB_{real}	id	EB_{real}	id	EB_{real}	id	EB_{real}
0	1084	25	973	50	1052	75	1045
1	911	26	1110	51	998	76	987
2	1010	27	972	52	997	77	1018
3	946	28	1070	53	920	78	952
4	1030	29	795	54	1102	79	1086
5	940	30	965	55	987	80	864
6	1049	31	918	56	929	81	1046
7	1074	32	842	57	1135	82	915
8	1171	33	1051	58	978	83	967
9	981	34	1028	59	941	84	1055
10	786	35	1003	60	971	85	1104
11	916	36	867	61	915	86	888
12	1135	37	1113	62	888	87	1126
13	893	38	1035	63	1253	88	1066
14	1096	39	970	64	1166	89	993
15	1012	40	1002	65	1031	90	980
16	1144	41	974	66	874	91	978
17	804	42	825	67	913	92	970
18	980	43	971	68	982	93	1002
19	879	44	917	69	1079	94	1005
20	1291	45	902	70	867	95	1083
21	1083	46	884	71	767	96	1153
22	1138	47	947	72	855	97	1047
23	894	48	800	73	1033	98	979
24	953	49	1096	74	1039	99	1063

Instância 3							
id	EB_{real}	id	EB_{real}	id	EB_{real}	id	EB_{real}
0	1018	25	956	50	927	75	906
1	897	26	984	51	997	76	868
2	1095	27	1028	52	1023	77	1092
3	1031	28	974	53	1043	78	1000
4	1014	29	1044	54	963	79	995
5	1052	30	1039	55	976	80	1091
6	1026	31	875	56	1202	81	1059
7	906	32	905	57	774	82	1035
8	984	33	926	58	1223	83	1125
9	985	34	949	59	1034	84	1093
10	947	35	968	60	1100	85	1024
11	1168	36	1001	61	834	86	931
12	912	37	697	62	941	87	935
13	952	38	954	63	972	88	1119
14	929	39	1124	64	1042	89	839
15	883	40	893	65	833	90	998
16	981	41	1093	66	1047	91	805
17	973	42	1035	67	879	92	1102
18	1153	43	997	68	1007	93	1086
19	975	44	1018	69	1065	94	1000
20	894	45	843	70	1033	95	993
21	1160	46	992	71	1108	96	751
22	1123	47	1160	72	1101	97	1058
23	977	48	1010	73	935	98	781
24	849	49	1004	74	1026	99	768

Instância 4							
id	EB_{real}	id	EB_{real}	id	EB_{real}	id	EB_{real}
0	1008	25	1040	50	1088	75	910
1	905	26	907	51	1032	76	971
2	1041	27	982	52	922	77	1035
3	1068	28	787	53	819	78	816
4	1086	29	1115	54	1186	79	1104
5	931	30	937	55	940	80	1242
6	1045	31	880	56	1010	81	1096
7	1010	32	975	57	1056	82	968
8	1083	33	857	58	1011	83	1043
9	1054	34	998	59	910	84	896
10	1090	35	944	60	953	85	1188
11	987	36	1218	61	988	86	1094
12	985	37	1114	62	1148	87	1079
13	1101	38	750	63	914	88	912
14	788	39	1044	64	1078	89	1032
15	950	40	860	65	1031	90	944
16	873	41	974	66	977	91	969
17	962	42	1016	67	894	92	943
18	1065	43	1075	68	972	93	897
19	1083	44	973	69	991	94	909
20	899	45	1158	70	853	95	979
21	953	46	952	71	1019	96	830
22	1014	47	1033	72	918	97	1061
23	971	48	1066	73	991	98	988
24	1030	49	1009	74	1034	99	1070

Instância 5							
id	EB_{real}	id	EB_{real}	id	EB_{real}	id	EB_{real}
0	1027	25	1011	50	994	75	1041
1	1049	26	1181	51	798	76	901
2	852	27	1031	52	902	77	1076
3	898	28	1180	53	1061	78	934
4	955	29	928	54	995	79	940
5	1011	30	1053	55	888	80	1018
6	1113	31	974	56	937	81	969
7	971	32	1060	57	1025	82	987
8	1126	33	1059	58	901	83	1060
9	1048	34	781	59	1097	84	1105
10	1117	35	867	60	936	85	980
11	1013	36	856	61	1181	86	1033
12	934	37	1040	62	892	87	976
13	852	38	1147	63	1020	88	1023
14	1016	39	967	64	848	89	1044
15	1082	40	1081	65	928	90	938
16	971	41	1055	66	941	91	1027
17	946	42	895	67	1040	92	1060
18	969	43	1040	68	1094	93	1009
19	890	44	925	69	1030	94	1173
20	951	45	1152	70	963	95	939
21	982	46	997	71	1082	96	926
22	1005	47	1164	72	1080	97	825
23	994	48	957	73	1012	98	1091
24	1061	49	1059	74	1057	99	1087

Instância 6							
id	EB_{real}	id	EB_{real}	id	EB_{real}	id	EB_{real}
0	992	25	1004	50	956	75	876
1	1090	26	1223	51	959	76	781
2	1018	27	993	52	1098	77	967
3	1029	28	949	53	970	78	1071
4	1011	29	1024	54	1114	79	1032
5	1044	30	1025	55	947	80	1041
6	1010	31	1007	56	1097	81	942
7	1279	32	939	57	948	82	1014
8	883	33	878	58	1018	83	836
9	815	34	1032	59	1097	84	924
10	886	35	866	60	959	85	918
11	891	36	897	61	956	86	1052
12	957	37	1133	62	1200	87	999
13	983	38	958	63	1095	88	884
14	978	39	986	64	957	89	999
15	1054	40	1090	65	1065	90	931
16	1039	41	970	66	964	91	933
17	1075	42	1103	67	1071	92	1086
18	1178	43	965	68	1142	93	1011
19	1122	44	1101	69	840	94	1040
20	872	45	1063	70	1103	95	1088
21	767	46	979	71	1146	96	1018
22	1090	47	913	72	1005	97	1055
23	816	48	896	73	1175	98	1068
24	1007	49	973	74	1016	99	1117

Instância 7							
id	EB_{real}	id	EB_{real}	id	EB_{real}	id	EB_{real}
0	1048	25	958	50	852	75	1106
1	1141	26	1122	51	996	76	1116
2	1002	27	996	52	1096	77	1005
3	995	28	1058	53	1174	78	871
4	1170	29	899	54	957	79	963
5	949	30	1006	55	837	80	924
6	1000	31	1060	56	1017	81	944
7	1092	32	864	57	1038	82	1056
8	1015	33	1035	58	977	83	944
9	1140	34	982	59	885	84	910
10	1103	35	906	60	1202	85	959
11	1029	36	996	61	764	86	984
12	922	37	810	62	949	87	1041
13	1057	38	787	63	868	88	905
14	862	39	882	64	936	89	1032
15	1024	40	901	65	1032	90	1008
16	1081	41	883	66	1014	91	1132
17	1021	42	827	67	929	92	979
18	1088	43	1029	68	1078	93	987
19	1204	44	841	69	1062	94	883
20	1092	45	1011	70	1065	95	861
21	1027	46	1079	71	957	96	1031
22	1064	47	1000	72	1105	97	975
23	1043	48	1009	73	1066	98	1050
24	869	49	962	74	1251	99	911

Instância 8							
id	EB_{real}	id	EB_{real}	id	EB_{real}	id	EB_{real}
0	1191	25	949	50	1146	75	845
1	1012	26	910	51	1205	76	1017
2	1105	27	880	52	1012	77	994
3	977	28	1104	53	901	78	1120
4	984	29	915	54	1120	79	1080
5	1069	30	983	55	941	80	1105
6	1056	31	879	56	953	81	925
7	888	32	970	57	1089	82	906
8	847	33	677	58	861	83	873
9	890	34	891	59	804	84	1050
10	858	35	857	60	1042	85	1279
11	1006	36	899	61	1040	86	1073
12	959	37	979	62	1010	87	923
13	963	38	967	63	1050	88	1084
14	864	39	1194	64	1108	89	887
15	1078	40	943	65	1097	90	858
16	1044	41	975	66	943	91	1072
17	991	42	843	67	1081	92	922
18	1102	43	952	68	1017	93	1032
19	913	44	866	69	949	94	1141
20	1041	45	1003	70	881	95	1040
21	1035	46	1085	71	1065	96	1093
22	1035	47	1040	72	965	97	839
23	927	48	930	73	1005	98	1066
24	1033	49	837	74	921	99	1214

Instância 9							
id	EB_{real}	id	EB_{real}	id	EB_{real}	id	EB_{real}
0	1054	25	946	50	858	75	853
1	846	26	909	51	927	76	837
2	980	27	1065	52	1115	77	804
3	950	28	927	53	1060	78	1261
4	1038	29	1054	54	872	79	1097
5	1041	30	1098	55	780	80	1026
6	1041	31	984	56	943	81	903
7	964	32	1028	57	1021	82	885
8	940	33	1064	58	1094	83	1055
9	941	34	992	59	1009	84	1157
10	1085	35	1054	60	888	85	831
11	815	36	874	61	1031	86	955
12	979	37	1111	62	883	87	992
13	1027	38	901	63	904	88	801
14	935	39	817	64	935	89	1084
15	1048	40	1138	65	877	90	959
16	993	41	994	66	973	91	1191
17	906	42	1045	67	910	92	961
18	1016	43	964	68	971	93	1041
19	973	44	898	69	954	94	886
20	959	45	693	70	959	95	938
21	929	46	1063	71	950	96	883
22	1006	47	971	72	1123	97	1039
23	815	48	980	73	1061	98	1130
24	960	49	1041	74	1006	99	941

Instância 10							
id	EB_{real}	id	EB_{real}	id	EB_{real}	id	EB_{real}
0	1044	25	1077	50	975	75	1029
1	950	26	1078	51	981	76	1000
2	1010	27	852	52	897	77	1037
3	1120	28	1054	53	968	78	1353
4	1012	29	991	54	1077	79	989
5	896	30	924	55	1174	80	844
6	914	31	931	56	884	81	1192
7	983	32	1128	57	1238	82	1061
8	981	33	919	58	1153	83	935
9	913	34	876	59	1017	84	1262
10	1018	35	1021	60	970	85	1055
11	1127	36	1201	61	930	86	1029
12	975	37	1003	62	1083	87	922
13	980	38	1031	63	931	88	894
14	780	39	906	64	954	89	823
15	923	40	1167	65	1088	90	958
16	861	41	1012	66	1044	91	895
17	961	42	1053	67	1090	92	1065
18	1053	43	905	68	1050	93	968
19	1152	44	1085	69	960	94	1177
20	1180	45	1039	70	949	95	1151
21	988	46	884	71	1080	96	1016
22	968	47	1004	72	933	97	972
23	1082	48	955	73	1119	98	1115
24	1049	49	1011	74	1079	99	885

APÊNDICE D

FERRAMENTA PARA VISUALIZAÇÃO DAS SIMULAÇÕES DE RSSF

Neste apêndice será introduzida a ferramenta **JSensors** (Guimarães & Martins, 2011), desenvolvida juntamente com um aluno da Universidade Federal de Ouro Preto, tendo o presente autor o orientado em seu trabalho de monografia. O JSensors é uma ferramenta desenvolvida para visualizar o comportamento de uma rede de sensor sem fio, ela pode ser utilizada de duas maneiras:

Simulação *Online* - neste modo a ferramenta funciona como uma API executada em paralelo acoplado com o código desenvolvido pelo utilizador. Neste modo, o usuário pode utilizar a ferramenta para acompanhar a execução em tempo real do seu próprio algoritmo.

Simulação *Offline* - este é um modo mais simples, em que o usuário pode passar um arquivo texto com todas as informações da simulação da rede. A partir deste arquivo é possível simular todos os períodos da rede. Apesar de mais simples, todas as funcionalidades da ferramenta permanecem as mesmas.

A partir da Figura D.1 é possível ver as funcionalidades do JSensors. A ferramenta possui 6 principais "abas": Visualização 2D, Sensores, Energia consumida, Energia disponível, Cobertura da rede e Status dos sensores.

D.1 VISUALIZAÇÃO 2D

Nesta parte é possível visualizar a rede a cada período de tempo. Dentro do quadro "Exibir" o usuário pode escolher em visualizar de forma simultânea ou individual os sensores, raio de cobertura, raio de comunicação e a conectividade entre os sensores. Conforme a legenda mostrada a direita, o estado dos sensores é diferenciado conforme a cor. Informações da rede a cada período é mostrada no campo "Estatísticas". É possível acompanhar o número de sensores ativos, inativos e falhos; a energia disponível e a cobertura a cada período de tempo.

A partir do quadro "Opções da Simulação" o usuário possui total controle da simulação, podendo acompanhá-la por períodos ou estágios[1]. A rede pode avançar ou voltar de acordo com a quantidade de períodos ou estágios inseridos no campo "Quantidade". Caso não queira acompanhar a cada período ou estágio é possível marcar "Auto Executar" para que a rede continue funcionando até seu fim. Caso queira ir direto para um determinado período basta informar no campo "Ir para período".

[1] a definição do que é um estágio na rede é definida pelo usuário.

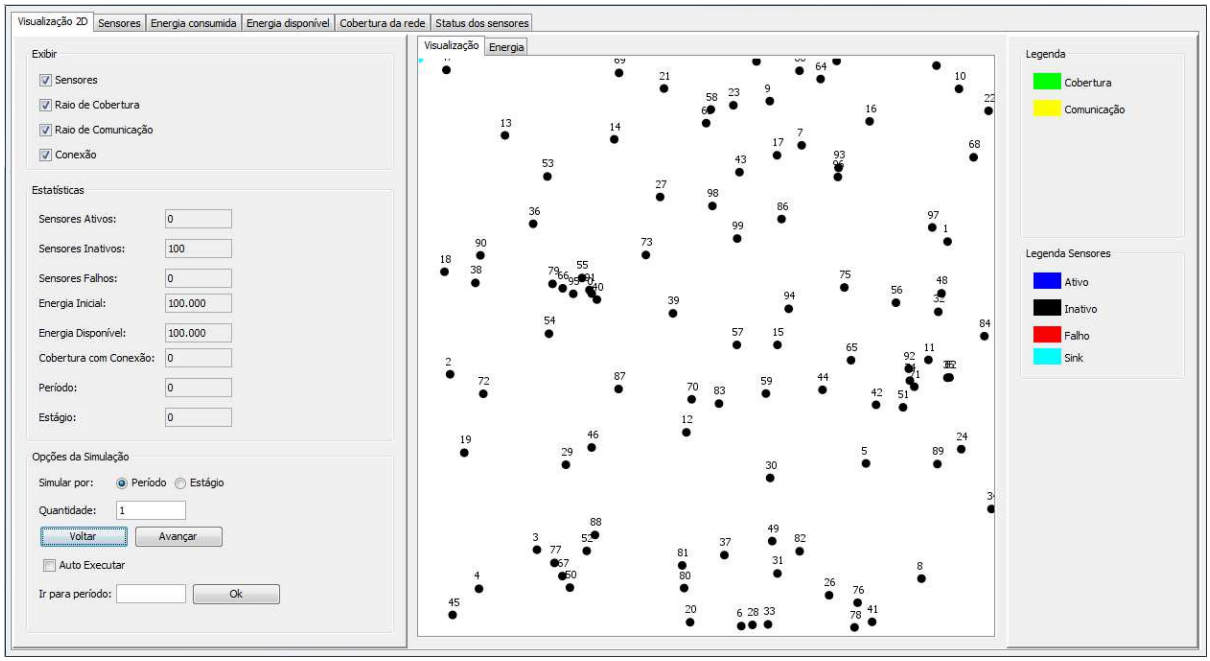


Figura D.1 Exemplo do JSensors.

Nesta aba de Visualização 2D existem duas "sub-abas" "Visualização" e "Energia". Na aba energia é mostrado um mapa de energia da rede a cada período de tempo. Nas Figuras D.2 a D.7 são mostrados exemplos de diferentes períodos de tempo da rede e também o mapa de energia da rede nestes períodos. Nas Figuras D.6 e D.7 é possível visualizar que a rede não mais conseguiu manter a cobertura em 100%. Pelo mapa de energia é fácil notar que um dos motivos é a falta de energia próxima ao sorvedouro.

D.2 SENSORES, ENERGIA CONSUMIDA, ENERGIA DISPONÍVEL, COBERTURA DA REDE E STATUS DOS SENSORES

As outras cinco principais abas também forcem informações do comportamento da rede a cada período de tempo. Na aba "Sensores" (Figura D.8) é mostrada a informação individual de cada nó sensor. Através do seu "id" é possível visualizar se este sensor está ativo, falho e qual sua carga atual de bateria.

Nas demais abas as informações são mostradas através de gráficos, cuja atualização é realizada em tempo de simulação. São mostrados gráficos de energia consumida, energia disponível, cobertura da rede e "Status" dos sensores, exemplos destes gráficos podem ser vistos respectivamente nas Figuras D.9, D.10, D.11 e D.12.

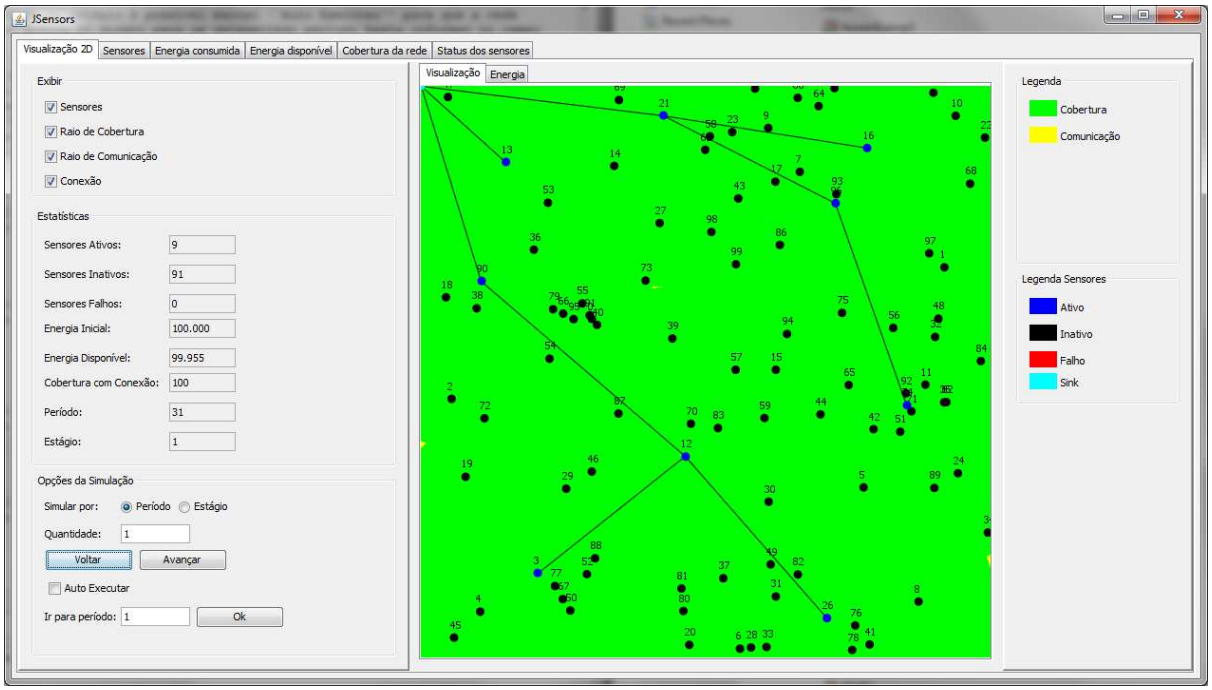


Figura D.2 JSensors - Visualização do estado de 100 sensores no período de tempo 31.

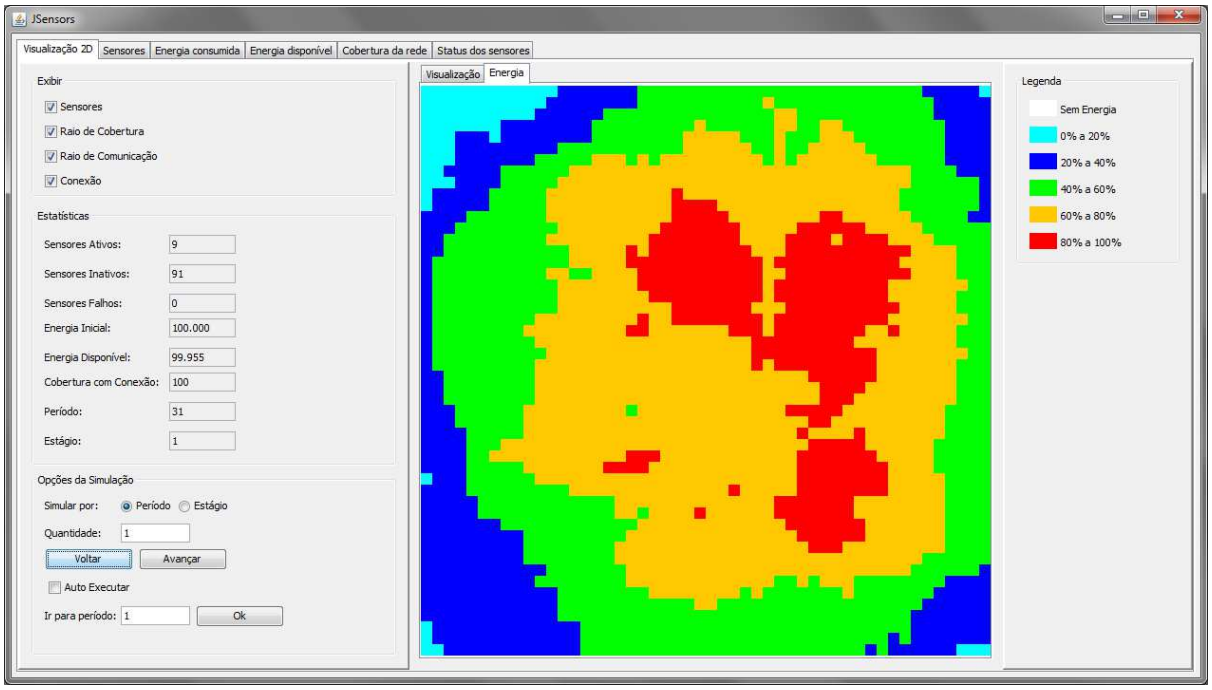


Figura D.3 JSensors - Mapa de energia da rede de 100 sensores no período de tempo 31.

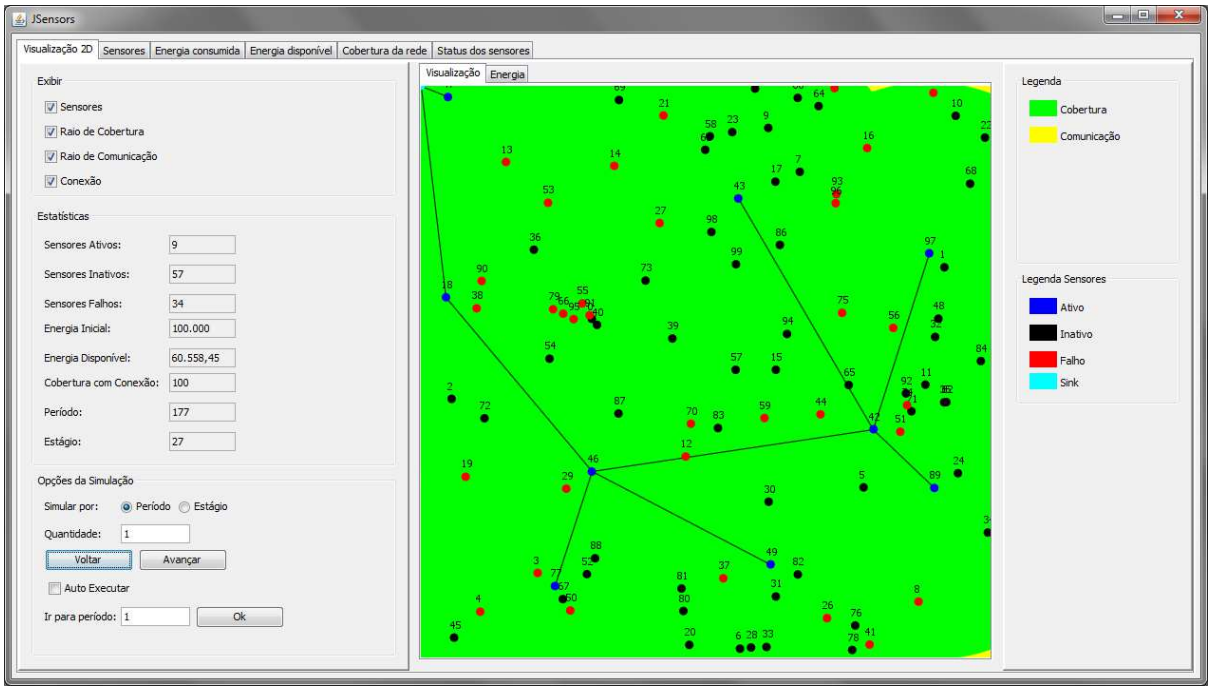


Figura D.4 JSensors - Visualização do estado de 100 sensores no período de tempo 177.

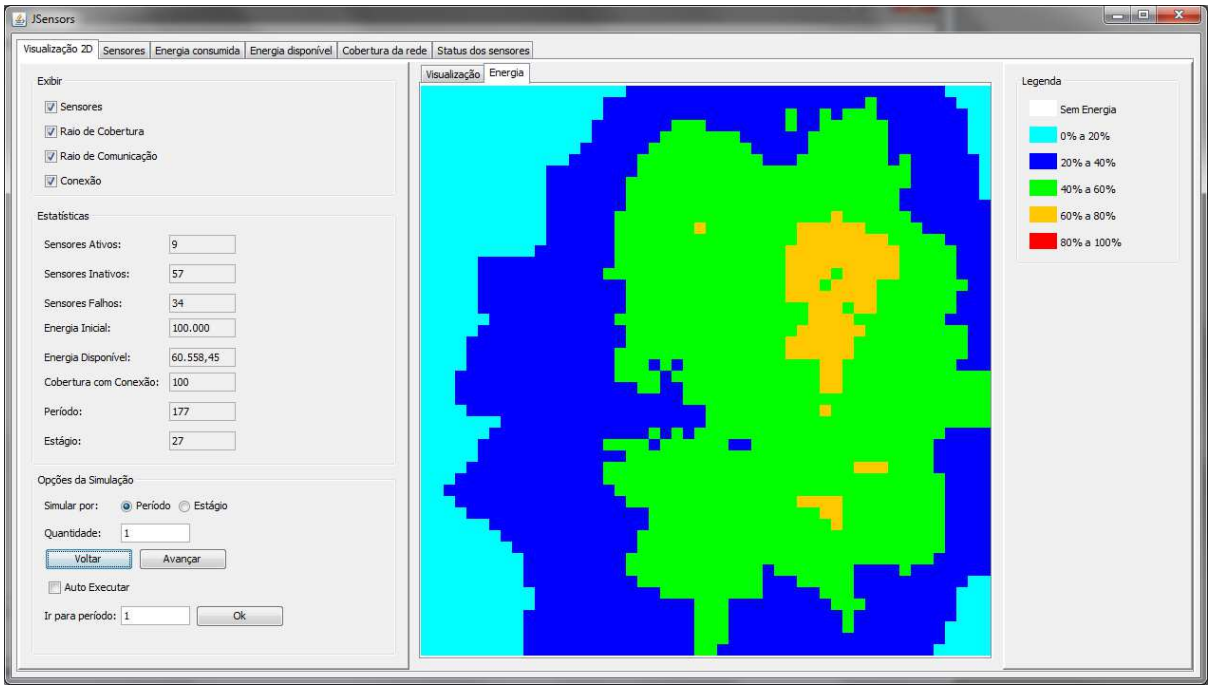


Figura D.5 JSensors - Mapa de energia da rede de 100 sensores no período de tempo 177.

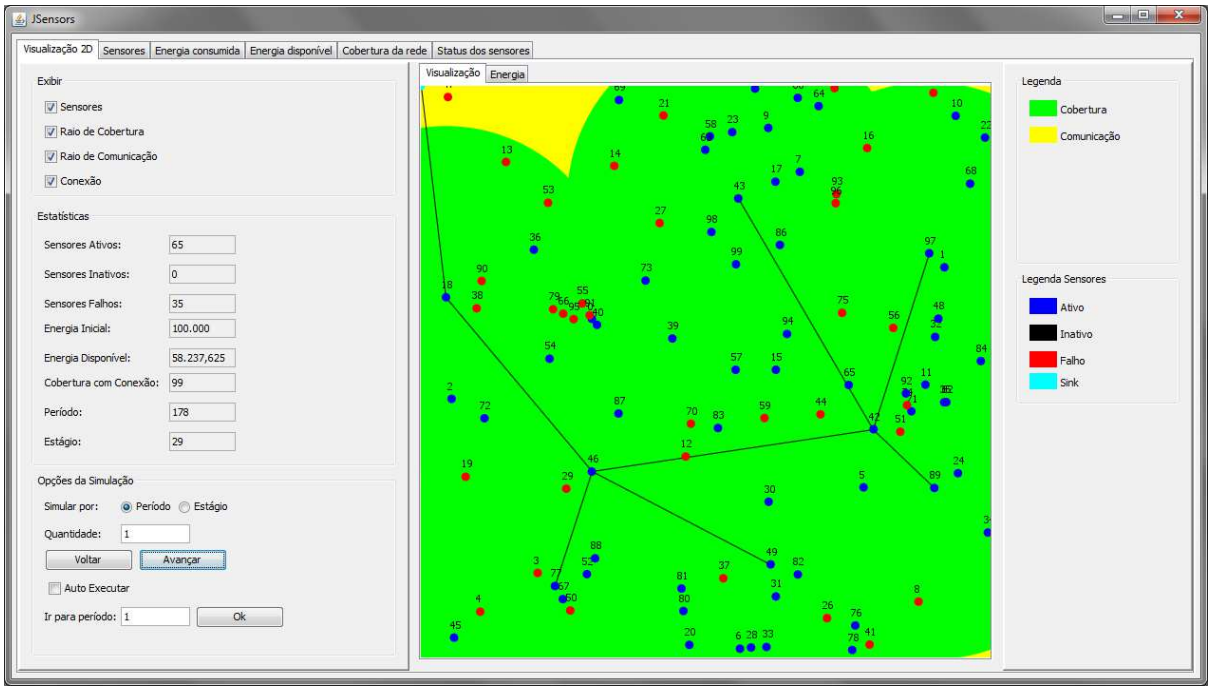


Figura D.6 JSensors - Visualização do estado de 100 sensores no período de tempo 178.

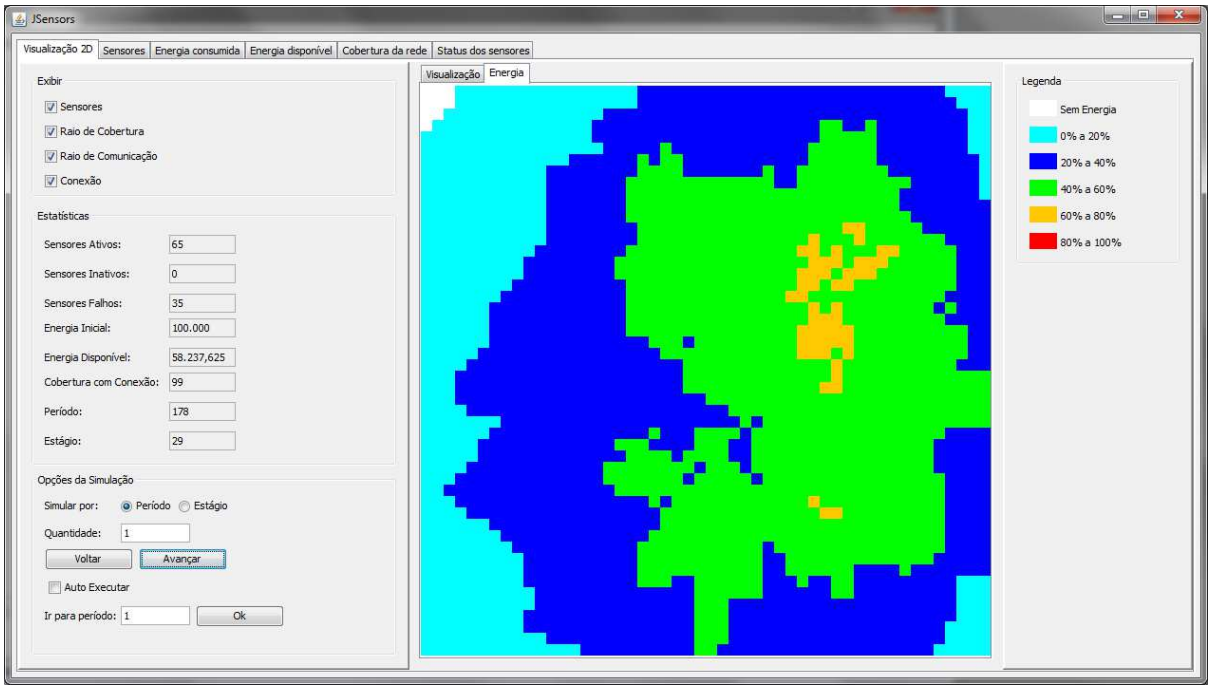
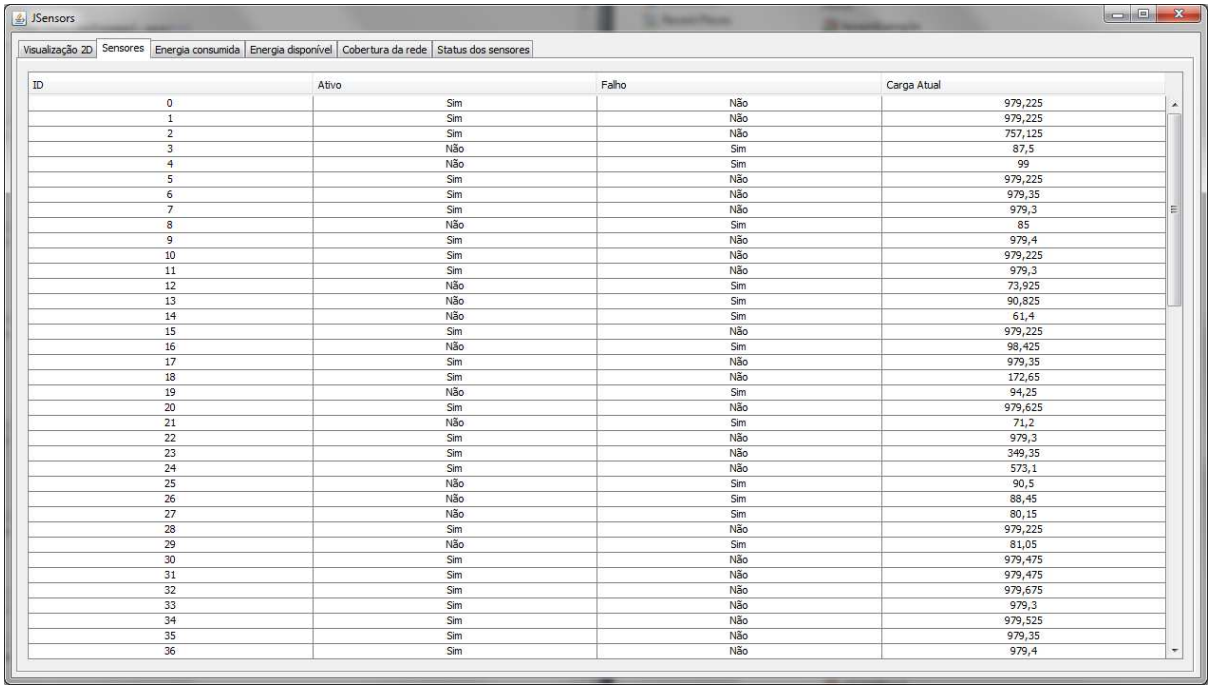


Figura D.7 JSensors - Mapa de energia da rede de 100 sensores no período de tempo 178.



ID	Ativo	Falho	Carga Atual
0	Sim	Não	979,225
1	Sim	Não	979,225
2	Sim	Não	757,125
3	Não	Sim	87,5
4	Não	Sim	99
5	Sim	Não	979,225
6	Sim	Não	979,35
7	Sim	Não	979,3
8	Não	Sim	85
9	Sim	Não	979,4
10	Sim	Não	979,225
11	Sim	Não	979,3
12	Não	Sim	73,925
13	Não	Sim	90,825
14	Não	Sim	61,4
15	Sim	Não	979,225
16	Não	Sim	98,425
17	Sim	Não	979,35
18	Sim	Não	172,65
19	Não	Sim	94,25
20	Sim	Não	979,625
21	Não	Sim	71,2
22	Sim	Não	979,3
23	Sim	Não	349,35
24	Sim	Não	573,1
25	Não	Sim	90,5
26	Não	Sim	88,45
27	Não	Sim	80,15
28	Sim	Não	979,225
29	Não	Sim	81,05
30	Sim	Não	979,475
31	Sim	Não	979,475
32	Sim	Não	979,675
33	Sim	Não	979,3
34	Sim	Não	979,525
35	Sim	Não	979,35
36	Sim	Não	979,4

Figura D.8 JSensors - Informações individuais de cada sensor presente na rede.

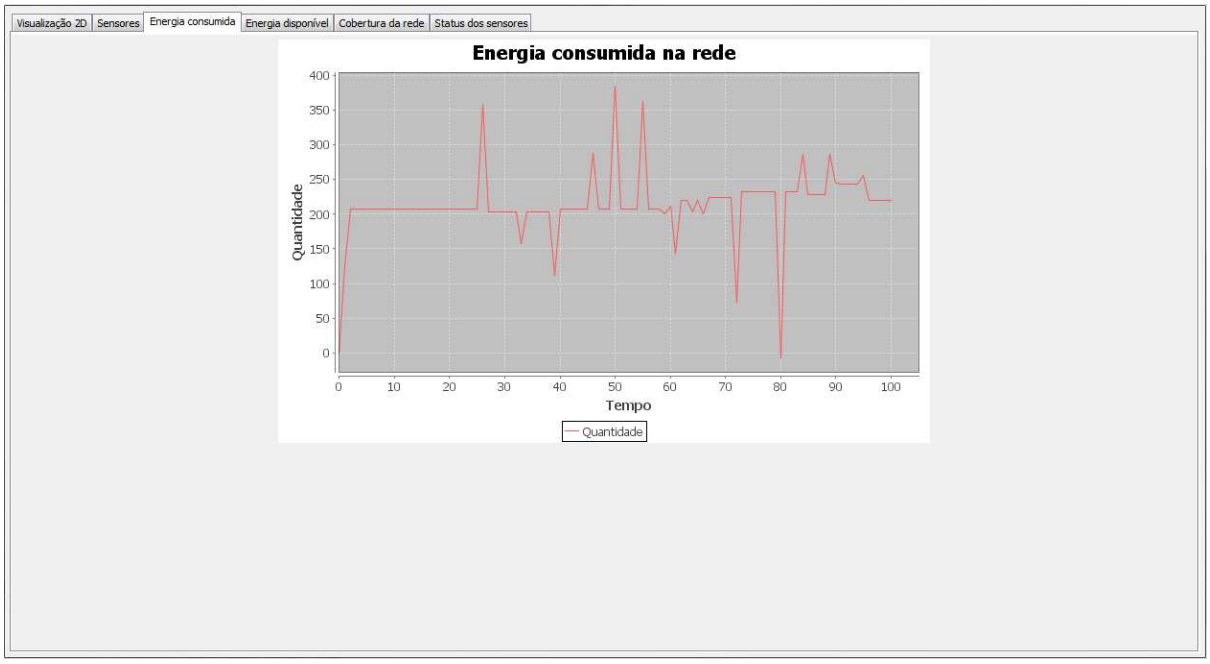


Figura D.9 JSensors - gráfico de energia consumida x tempo.

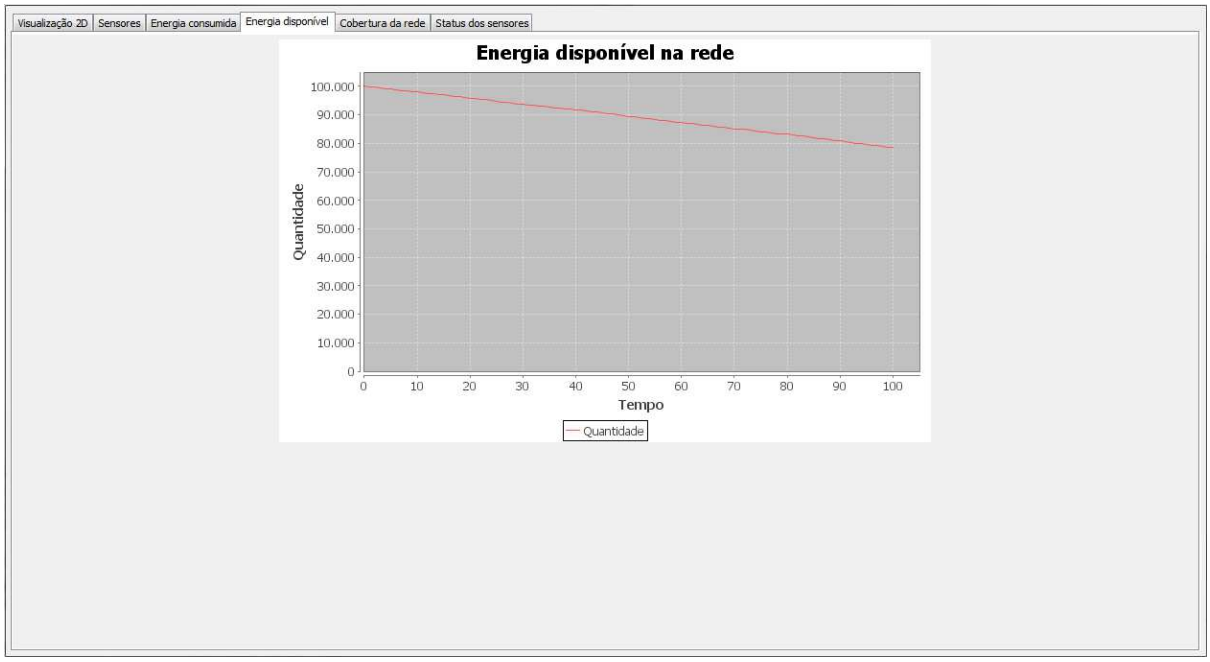


Figura D.10 JSensors - gráfico de energia disponível x tempo.

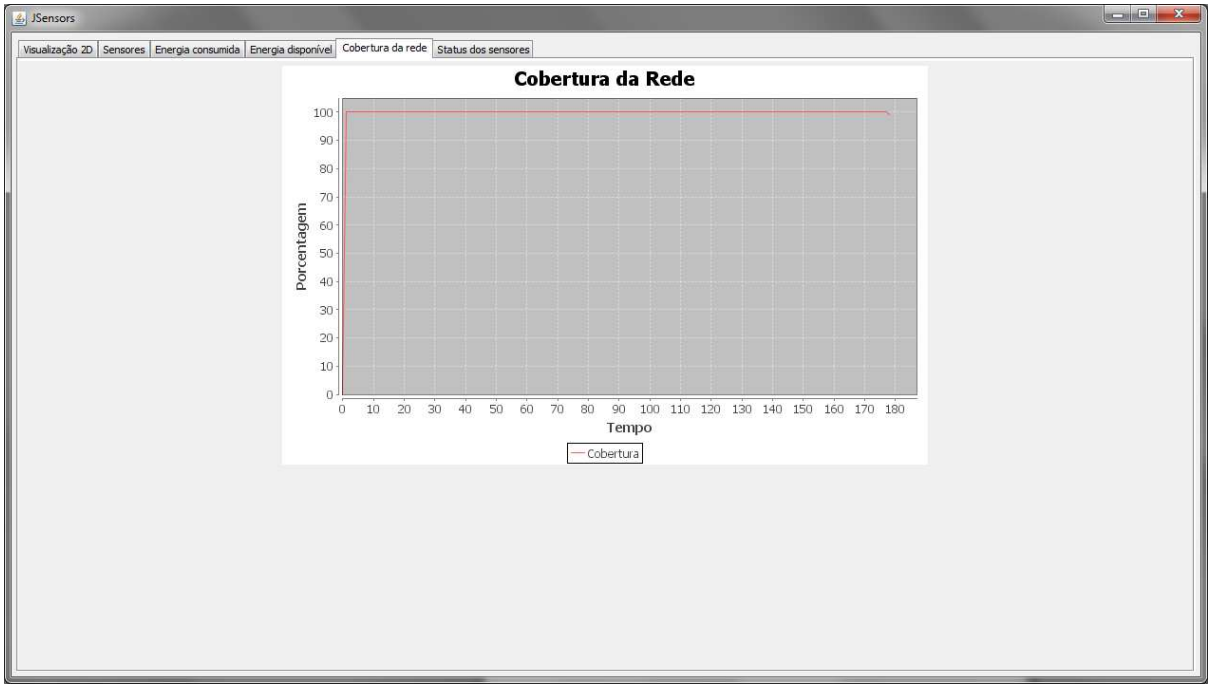


Figura D.11 JSensors - gráfico de cobertura x tempo.

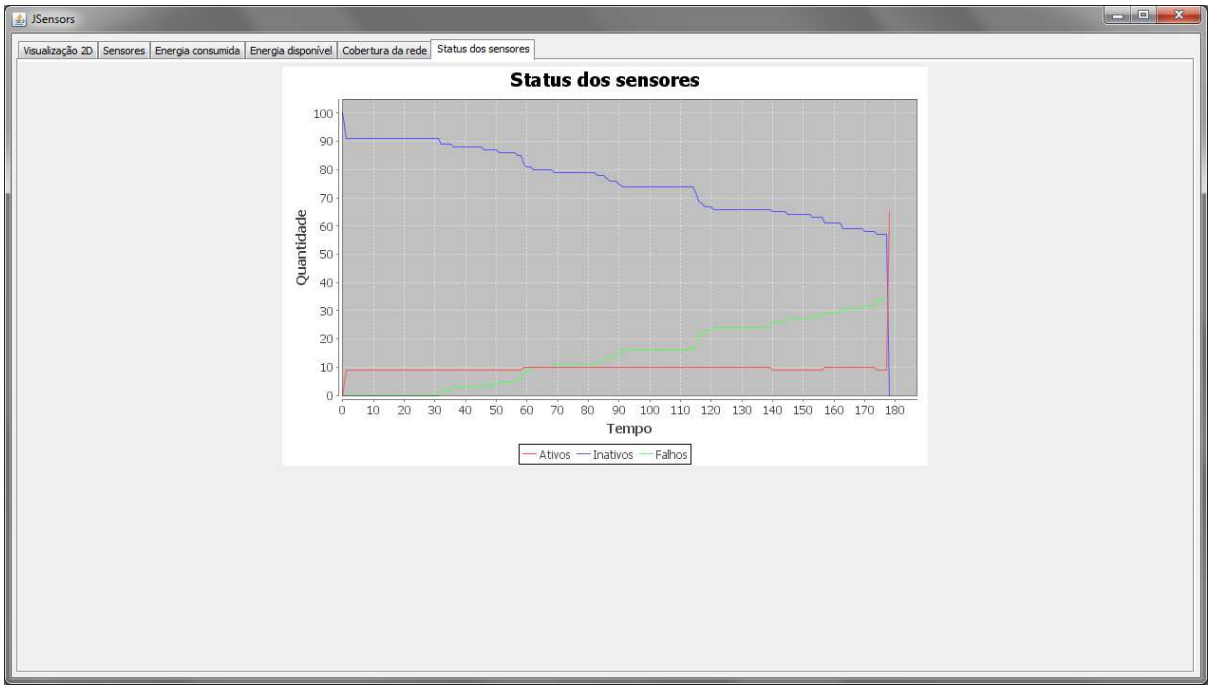


Figura D.12 JSensors - gráfico do estado dos sensores x tempo.