Um cenário pós-desastre natural ou não natural requer o gerenciamento eficiente das equipes de resgate para o atendimento a um maior número de vítimas. Neste cenário, protocolos para descoberta de serviços são essenciais para que as vítimas localizem e aloquem os recursos necessários para o atendimento.

As Redes Móveis Ad Hoc (MANETs) são particularmente atrativas para cenários em que existe a necessidade de instalar emergencialmente uma rede, e podem possibilitar a comunicação entre as vítimas e socorristas. A complexidade para gerenciar os recursos em relação ao número de atendimentos requer o uso de algoritmos adequados. Um algoritmo de escalonamento, neste contexto, deve organizar uma ordem de atendimento, a fim de assegurar uma distribuição dos recursos disponíveis.

Em face dessas circunstâncias, nesta dissertação, apresenta-se como principal contribuição uma abordagem para escalonamento de recursos que estende o protocolo de descoberta de serviços para MANETs distribuídos em cenários pós-desastre *Location Aware Discovery Protocol* (LADP). A abordagem para o escalonamento foi modelada com base em Algoritmos Genéticos (AG) e no algoritmo de busca em grafos A-Estrela (A*).

Orientadora: Janine Kniess

Joinville, 2019

ANO 2019 MARCELO PETRI

ESCALONAMENTO DE RECURSOS PARA DESCOBERTA DE SERVIÇOS EM REDES AD HOC MÓVEIS

UDESC

UNIVERSIDADE DO ESTADO DE SANTA CATARINA – UDESC
CENTRO DE CIÊNCIAS TECNOLÓGICAS - CCT
PROGRAMA DE PÓS GRADUAÇÃO EM COMPUTAÇÃO APLICADA

DISSERTAÇÃO DE MESTRADO

ESCALONAMENTO DE RECURSOS PARA DESCOBERTA DE SERVIÇOS EM REDES AD HOC MÓVEIS

MARCELO PETRI

JOINVILLE, 2019

MARCELO PETRI

ESCALONAMENTO DE RECURSOS PARA DESCOBERTA DE SERVIÇOS EM REDES AD HOC MÓVEIS

Dissertação submetida ao Programa de Pós-Graduação em Computação Aplicada do Centro de Ciências Tecnológicas da Universidade do Estado de Santa Catarina, para a obtenção do grau de Mestre em Computação Aplicada.

Orientador: Dr. Janine Kniess (Doutora)

JOINVILLE

2019

MARCELO PETRI
ESCALONAMENTO DE RECURSOS PARA DESCOBERTA DE SERVIÇOS EM REDES AD HOC MÓVEIS/ MARCELO PETRI. – Joinville, 2019-
80 p.

Dr. Janine Kniess (Doutora)

Dissertação (mestrado) – Universidade do Estado de Santa Catarina - UDESC,2019.

1. Descoberta de Serviço. 2. Escalonamento de Recursos. 3. MANET. 4. Genético. 5. A-Estrela. I. KNIESS, JANINE. II. Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Programa de Pós-Graduação em Computação Aplicada. III. Título

Escalonamento de Recursos para Descoberta de Serviços em Redes Ad Hoc Móveis

por

Marcelo Petri

Esta dissertação foi julgada adequada para obtenção do título de

Mestre em Computação Aplicada

Área de concentração em "Ciência da Computação",
e aprovada em sua forma final pelo

CURSO DE MESTRADO ACADÊMICO EM COMPUTAÇÃO APLICADA
DO CENTRO DE CIÊNCIAS TECNOLÓGICAS DA
UNIVERSIDADE DO ESTADO DE SANTA CATARINA.

Banca Examinadora:

Profa. Dr. Rafael Stubs Parpinelli
CCT/UDESC (Presidente)

Prof. Dr. Omir Correia Alves Junior
CCT/UDESC

Prof. Dr. Célio Vinicius Neves de Albuquerque
Universidade Federal Fluminense (UFF)

Joinville,SC, 12 de fevereiro de 2019.

Dedico este trabalho aos meus familiares, amigos, colegas e professores que me acompanharam e me deram forças nessa magnífica trajetória.

AGRADECIMENTOS

Em primeiro lugar gostaria de agradecer a Deus por tudo que tem proporcionado em minha vida. Por todas as vezes que fraquejei e Ele me trouxe força e coragem para continuar. Por todos os males que Ele tirou da minha frente, me protegendo em todos os momentos que eu precisei.

Agradeço também à minha noiva Sandra, que me ajudou, dando o suporte necessário para que eu enfrentasse mais esse desafio. Além disso, soube me compreender quando tive que abdicar de muitos momentos familiares para a realização deste trabalho.

Presto meus agradecimentos especiais à minha orientadora, Prof.ª Janine Kniess, por sempre estar à disposição, ajudando e motivando nos momentos mais necessários. Também agradeço pela compreensão que ela teve diante dos meus momentos de dificuldade.

Aos membros da banca examinadora, Prof. Rafael Stubs Parpinelli, Prof. Omir Correia Alves Júnior e Prof. Célio Vinícius Neves de Albuquerque, que tão gentilmente aceitaram participar e colaborar com esta dissertação. Ainda, agradeço ao Prof. Rafael, pelas conversas breves, porém importantíssimas.

Um agradecimento muito especial aos meus pais, Paulo e Beatriz, que me deram todo o amor que podiam dar, bem como todo o suporte necessário para que eu conseguisse ter boas condições de vida.

Por fim, a todos aqueles que contribuíram, direta ou indiretamente, para a realização desta dissertação, o meu sincero agradecimento.

”Se a gente quiser modificar alguma coisa, é pelas crianças que devemos começar. Devemos respeitar e educar nossas crianças para que o futuro das nações e do planeta seja digno.”

Ayrton Senna

RESUMO

Um cenário pós-desastre natural ou não natural requer o gerenciamento eficiente das equipes de resgate para o atendimento a um maior número de vítimas. As equipes de resgate (p.ex. carros de apoio) devem realizar os atendimentos rapidamente após um desastre. Neste cenário, protocolos para descoberta de serviços são essenciais para que as vítimas localizem e aloquem os recursos necessários para o atendimento. Contudo, em situações de emergência, a infraestrutura de comunicação pode ser danificada, não sendo possível fornecer serviços adequados para a comunicação entre vítimas e socorristas.

As Redes Móveis Ad Hoc (*Mobile Ad Hoc Networks MANETs*) são particularmente atrativas para cenários em que existe a necessidade de instalar emergencialmente uma rede, e podem possibilitar a comunicação entre as vítimas e socorristas. Com as informações obtidas através da rede, os socorristas podem tomar decisões sobre os atendimentos a serem realizados.

O Corpo de bombeiros, bem como as ambulâncias, são serviços indispensáveis e nem sempre disponíveis a contento em muitas cidades. A complexidade para gerenciar os recursos em relação ao número de atendimentos requer o uso de algoritmos adequados. O número de vítimas em uma situação de emergência pode aumentar consideravelmente de acordo com a gravidade da calamidade. Além de atender ao requisito de tempo máximo para o atendimento especificado pelas vítimas, um algoritmo de escalonamento, neste contexto, deve organizar uma ordem de atendimento, a fim de assegurar uma distribuição dos recursos disponíveis.

Em face dessas circunstâncias, nesta dissertação, apresenta-se como principal contribuição uma abordagem para escalonamento de recursos que estende o protocolo de descoberta de serviços para MANETs distribuídos em cenários pós-desastre *Location Aware Discovery Protocol* (LADP). A abordagem para o escalonamento foi modelada com base em Algoritmos Genéticos (AG) e no algoritmo de busca em grafos A-Estrela (A*).

Palavras-chaves: Descoberta de Serviços, Escalonamento, Recursos, MANET, Genético, A-Estrela.

ABSTRACT

A pos-disaster scenario requires an efficient management of rescue teams to serve a larger number of victims. Rescue teams (e.g. support cars) must perform care promptly after a disaster. In this scenario, service discovery protocols are essential for victims to locate and allocate the resources. However, in emergency situations, the communication infrastructure can be damaged and services for communication between victims and rescuers cannot be available.

Mobile Ad Hoc Networks (MANETs) are particularly attractive for scenarios in which is necessary to distribute a network on demand and to enable communication between victims and rescuers. With the information obtained through the network, rescuers can make decisions about the care to be performed.

Fire brigade and ambulances are indispensable services and not always enough available. The complexity of managing scarce resources requires the use of appropriate algorithms. The number of victims in an emergency situation can increase considerably according to the severity of the disaster. Besides the maximum time to attend a request, a scheduling algorithm needs to organize a queue in order to define the care sequence.

In view of these circumstances, this work presents as a main contribution a resource-scaling approach that extends the service discovery protocol (LADP) for MANETs distributed in post-disaster scenarios. The scaling approach was modeled based on Genetic Algorithms (AG) and the A-Star (A *) graph search algorithm.

Key-words: Discovery Services. Scheduling. Resources. MANET, Genetic, A-Star.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

A* Algoritmo A-Estrela

ACO *Ant Colony Optimization*

GA *Genetic Algorithm*

BAHSD *Bee AdHoc Service Discovery*

BCO *Bee Colony Optization*

CARE *Congestion Aware Scheduling Algorithm*

CaSMA *Channel Aware Scheduling for MANETs*

DSDV *Destination-Sequenced Distance Vector*

DSR *Dynamic Source Routing*

ELQS *Energy-Efficient and Load Balanced Queue Scheduling Algorithm*

FCFS *First Come First Serve*

FIFO *First In First Out*

GN *General Node*

GUI *Graphical User Interface*

GPSR *Greedy Perimeter Stateless Routing*

IC *Inteligência Computacional*

JINI Projeto JINI da Sun

LADP *Location-Aware Discovery Protocol*

LWSD *LightWeight Service Discovery*

MANET *Mobile Ad Hoc Network*

MDS *Main-Service Directory*

NACK *Negative Acknowledgement*

NS-3 *Network Simulator 3*

OLSR *Optimized Link State Routing Protocol*

PMX *Partially Mapped Crossover*

QoS *Quality of Service*

RISED *Rescue Information System for Earthquake Disasters*

RTT *Round Trip Time*

SI *Swarm Inteligence*

SINVK *Service Invoke Packet*

SDA *Service Discovery Agent*

SOAP *Simple Object Access Protocol*

SP *Service Provider*

SRA *Swarm Routing Agent*

SRP *Service Request Packet*

SRESP *Service Response Packet*

SRP *Service Replay Packet*

SSD *Sub-Service Directory*

TTL *Time To Live*

WRR *Weighted Round Robin*

XML *Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

Desastres naturais ou não naturais ocorrem com frequência na atualidade. Na presença de um desastre, tem-se situações que põem em risco a vida das pessoas. Como consequência, equipes de resgate especializadas necessitam atender às vítimas.

Neste cenário, existe a necessidade de uma distribuição eficiente das equipes de resgate (p.ex. carros de apoio, ambulâncias) para um rápido atendimento às vítimas. A comunicação entre vítimas e socorristas é vital para agilizar os atendimentos. Contudo, em situações de emergências, a infraestrutura de comunicação pode ser danificada, não sendo possível fornecer serviços adequados para a comunicação entre vítimas e socorristas.

Neste contexto, as Redes Ad Hoc Móveis (MANETs), que são caracterizadas por uma natureza altamente dinâmica (os nós podem entrar ou sair da rede a qualquer momento) (LABIOD, 2010), são adequadas para os cenários onde exista a necessidade da distribuição de uma rede sob-demanda.

Um componente essencial para a usabilidade das MANETs é a descoberta de serviços. A descoberta de serviços permite que os dispositivos de comunicação possam localizar e selecionar os provedores de serviços na rede (p.ex. um carro do Corpo de Bombeiros ou uma ambulância), e também auxiliar os provedores de serviços a anunciarem seus próprios recursos para a rede.

A realização de um gerenciamento eficiente dos recursos disponíveis na área afetada, que normalmente não estão disponíveis a contento, contribui para o atendimento de um maior número de vítimas e dar igual probabilidade de atendimento dos requisitantes. Os protocolos para descoberta de serviços possibilitam que as vítimas localizem e aloquem os recursos necessários para o atendimento. A adição de uma abordagem para escalonamento de recursos à descoberta de serviços pode contribuir para uma melhor distribuição dos recursos entre as vítimas.

Logo, propõe-se neste trabalho uma abordagem para o escalonamento de recursos para protocolos de descoberta de serviços em Redes Ad Hoc Móveis (MANETs) com múltiplos saltos, operando em cenários pós-desastre. Nestas redes, os próprios nós são responsáveis pelo encaminhamento dos pacotes. Os nós devem, portanto, cooperar entre si, a fim de permitir a comunicação entre os nós da rede.

A Figura 1.1 representa uma MANET distribuída em uma área pós-desastre. Os provedores de recursos (equipes de resgate, são identificados como triângulos), as

vítimas (representadas pelos quadrados) e os intermediários (os círculos), transportam dispositivos móveis que possibilitam a comunicação. A transmissão ocorre salto-a-salto por meio dos dispositivos móveis transportados pelas equipes de resgate, vítimas e nós intermediários na rede. Neste cenário (denominado aqui de cenário alvo), as vítimas (pessoas necessitando de atendimento), solicitam e aguardam o atendimento de um provedor de serviço (p.ex. um carro de apoio).

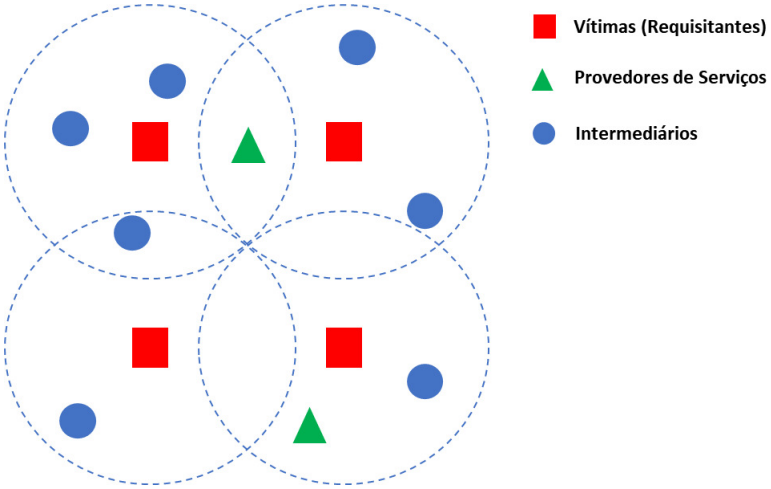


Figura 1.1 – Exemplo de cenário de aplicação. Fonte: próprio autor.

No escopo deste trabalho, os veículos utilizados pelas equipes de resgate são os provedores de recursos. Na área afetada, existem nós intermediários caracterizados como dispositivos móveis que participam da rede, porém não atuam como provedores e nem como vítimas. Tanto os provedores quanto os intermediários são móveis, e as vítimas permanecem fixas enquanto esperam pelo atendimento do provedor.

A abordagem para escalonamento de recursos proposta nesta dissertação de mestrado, estende o protocolo de descoberta de serviço, *Location Aware Discovery Protocol* (LADP) (KNIESS; LOQUES; ALBUQUERQUE, 2011), no qual questões de escalonamento dos provedores de serviços não foram consideradas. O mecanismo de escalonamento leva em consideração algumas características do cenário alvo como, a localização geográfica dos requisitantes, a velocidade de deslocamento dos nós provedores e o tempo máximo para o atendimento das vítimas para realizar o escalonamento.

Alguns protocolos de descoberta de serviços no contexto das MANETs foram desenvolvidos para ambientes pós-desastres: (JUSZCZYK; DUSTDAR, 2008), (MALLAH; QUINTERO, 2009), (ROY; KAR; MUKHERJEE, 2010), (PARK et al., 2013), (SERHANI; GADALLAH, 2010), (ARENELLA; SANTIS; MALANDRINO, 2014) e o protocolo LADP (KNIESS; LOQUES; ALBUQUERQUE, 2011). Dentre esses protocolos, destaca-se o protocolo de descoberta de serviços *Location Aware Dicovery Protocol*

(LADP) proposto por Kniess et al. (2011). A arquitetura do protocolo LADP foi especificada com a finalidade de oferecer suporte ao desenvolvimento de aplicações para descoberta, seleção e invocação de serviço em MANETs de múltiplos saltos, possibilitando aos dispositivos sem fio, utilizarem os recursos disponíveis na rede, mesmo na presença de obstáculos no caminho e na de falhas dos nós.

1.1 CONTEXTO

No cenário alvo representado por meio da Figura 1.1, assume-se o uso de um protocolo de descoberta de serviços, auxiliando as equipes de resgate na área afetada. Um exemplo de utilização do protocolo de descoberta de serviços neste cenário, ocorre quando um requisitante (vítima), provido de um dispositivo móvel (denominado aqui de nó da rede e previamente configurado com o protocolo de descoberta de serviços), solicita um ou mais provedores de serviços (p.ex. ambulância). O requisitante, a partir do seu dispositivo de comunicação, envia uma mensagem de descoberta de serviços para a rede, buscando os provedores apropriados para o atendimento.

Contudo, normalmente o número de requisitantes é bem maior do que o número de provedores disponíveis para o atendimento. A complexidade para gerenciar os provedores de serviços em relação ao número de atendimentos, e o tempo que esses provedores levam para chegar até o local do evento, requer o uso de mecanismos adequados de escalonamento dos recursos, com o objetivo de organizar os atendimentos, a fim de facilitar o deslocamento dos provedores até as vítimas.

Nos protocolos para descoberta de serviços analisados da literatura, verificou-se a ausência de um mecanismo para escalonamento de recursos em MANETs para cenário de pós-desastre. O protocolo LADP (KNIESS; LOQUES; ALBUQUERQUE, 2011) (o funcionamento deste protocolo será descrito em detalhes no Capítulo 3) atende aos requisitos do cenário alvo: os provedores são móveis; é geográfico (leva em consideração a localização geográfica dos nós); estabelece um tempo máximo para o atendimento; e considera a velocidade de deslocamento dos nós provedores. Motivos que o tornaram adequado para o desenvolvimento do mecanismo de escalonamento de recursos desenvolvido nesta dissertação.

Dentro desse contexto, o trabalho apresentado enquadra-se na área de otimização combinatória. No cenário alvo, o número de nós dentro do espaço de busca pode crescer exponencialmente e inviabiliza a utilização de métodos exatos. Em face destas características e da complexidade do problema, estudou-se duas abordagens: o algoritmo A-Estrela (A*) Seção 2.3.2 e o algoritmo Genético (AG) Seção 2.3.1.

O algoritmo de escalonamento foi modelado com base em meta-heurísticas utilizando um Algoritmo Genético, devido a sua simplificação na formulação, solução

de problemas de otimização e a possibilidade de entradas de dados do tipo inteiro para a formação do cromossomo inicial (representado neste trabalho pelos requisitantes na fila de atendimento). Para fins de comparação com o Algoritmo Genético, utilizou-se um algoritmo de escalonamento com base no Algoritmo A-Estrela, considerado na literatura um algoritmo de busca em grafos, que encontra um caminho entre um nó de origem e um nó de destino. Adicionalmente, na abordagem para escalonamento de recursos, foram desenvolvidos temporizadores para a sincronização e gerenciamento da troca de mensagens entre os nós da rede, visando assegurar que os requisitantes e provedores recebam confirmações das mensagens enviadas, e diante do não recebimento executem as recuperações necessárias.

1.2 OBJETIVO

O objetivo geral desta dissertação é especificar, desenvolver, implementar e avaliar uma solução para escalonamento de recursos para protocolos de descoberta de serviços em Redes Ad Hoc Móveis (MANETs) com múltiplos saltos, operando em cenários pós-desastre.

A abordagem para escalonamento de recursos estende o protocolo *Location Aware Discovery Protocol* (LADP) proposto por Kniess et al. (2011), e a meta é auxiliar as equipes de resgate no gerenciamento dos recursos em relação ao atendimento às vítimas.

1.3 CONTRIBUIÇÕES

Esta dissertação apresenta a especificação, a implementação e a análise de desempenho de uma abordagem para escalonamento de recursos, que estende o protocolo para descoberta de serviços *Location Aware Discovery Protocol* (LADP) (KNIESS; LOQUES; ALBUQUERQUE, 2011). Em suma, as principais contribuições deste trabalho são:

1. **Especificação de uma abordagem para escalonamento de recursos.** Concepção, implementação e avaliação de desempenho de uma solução para escalonamento de recursos, que auxilia equipes de resgate no atendimento às vítimas. A estratégia adotada para o escalonamento de recursos foi realizada considerando-se fatores como, localização geográfica dos requisitantes, tempo de deslocamento dos provedores ao local do evento e tempo máximo para o atendimento determinado pelo requisitante.
2. **Mecanismo de escalonamento com base em meta-heurísticas.** Especificação de um mecanismo para escalonamento com base em Algoritmo Genético,

para realizar o escalonamento dos provedores de serviços em relação às solicitações dos requisitantes.

3. **Temporizadores para o gerenciamento das mensagens entre os nós.** Definição de temporizadores de sincronização para gerenciar o envio e recebimento de mensagens pelo(s) provedor(es) e requisitante(s).

4. **Publicação em artigos e revistas.** Publicações em artigos e revistas (voltadas a área e afins) para demonstrar os resultados alcançados com o desenvolvimento deste trabalho de dissertação.

Por meio das contribuições apresentadas nessa subseção, pretende-se prover uma gestão eficiente dos provedores de serviços distribuídos em cenários pós-desastres, e realizar um maior número de atendimentos.

1.4 ESTRUTURA DO DOCUMENTO

Esta dissertação apresenta a seguinte estrutura organizacional: no Capítulo 2, apresenta-se a fundamentação teórica; no Capítulo 3, discutem-se os trabalhos relacionados à descoberta de serviços e escalonamento de recursos em MANETs; no Capítulo 4, demonstra-se a abordagem proposta para o escalonamento de recursos; no Capítulo 5, ilustram-se os resultados obtidos e as discussões; e, por fim, no Capítulo 6, as conclusões e os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, apresenta-se uma revisão de literatura como fundamentação teórica deste trabalho. Primeiramente, a Seção 2.1, aborda sobre o tema descoberta de serviços. Na Seção 2.2, apresentam-se os conceitos sobre escalonamento de recursos e sua relação com a MANETs. Na Seção 2.3, discutem-se as abordagens heurísticas e meta-heurísticas na otimização do escalonamento de recursos. Por fim, na Seção 2.4, são apresentadas as conclusões finais do capítulo.

2.1 DESCOBERTA DE SERVIÇOS: VISÃO GERAL

Os protocolos de descoberta de serviços são compostos de três entidades principais: o Cliente, que está interessado em encontrar e usar um serviço; o Servidor, que oferece o serviço; e o Diretório, um repositório centralizado ou distribuído na rede que hospeda totalmente ou parcialmente a informação de serviço em uma base de dados (MARIN-PERIANU; HARTEL; SCHOLTEN, 2005).

O conceito de descoberta de serviços surgiu no contexto de sistemas de informações auto-organizáveis. De uma forma geral, os protocolos de descoberta de serviços viabilizam a detecção automática de serviços oferecidos por dispositivos em uma rede de computadores (MIAN; BALDONI; BERALDI, 2009). Os elementos da descoberta de serviços em redes de computadores classificam-se em (LABIOD, 2010):

- **Descrição:** sintaxe clara, e semântica bem definida, para facilitar a busca de serviços;
- **Publicação:** consiste em anunciar descrições de serviços disponíveis a um servidor, ou diretamente a outros nós da rede;
- **Descoberta:** processo pelo qual um usuário encontra um provedor para um serviço, realiza um pedido de descoberta e determina os serviços correspondentes ao pedido. A descoberta também fornece um mecanismo de comunicação entre usuários e os prestadores de serviços;
- **Seleção:** um pedido de descoberta realizado pelo cliente pode resultar no retorno de várias respostas de provedores para o serviço solicitado. A seleção pode ser feita manualmente pelo usuário ou automaticamente, usando um algoritmo baseado em critérios, tais como distância, número de saltos, tempo de resposta ou energia disponível (MIAN; BALDONI; BERALDI, 2009);

- **Invocação:** estabelece como os serviços oferecidos pelos provedores serão acessados e usados pelos usuários.

Conforme descrito por Kniess et al. (2011), um protocolo de descoberta e de seleção de serviços é composto por três fases: descoberta de serviços, seleção de serviços e invocação de serviços. A descoberta de serviços compreende os mecanismos de busca e seleção de serviços. Após a fase de seleção, um protocolo de descoberta de serviços pode oferecer um mecanismo de invocação de serviços, que especifica como os provedores escolhidos através do mecanismo de seleção serão localizados e selecionados (MIAN; BALDONI; BERALDI, 2009).

Com base nas informações obtidas durante o processo de descoberta de serviços, o sucesso da utilização dos serviços disponíveis na rede por clientes, depende da flexibilidade que o protocolo de descoberta de serviços tem em relação à invocação de serviços. Dentre os trabalhos na área de descoberta de serviços para cenários pós-desastres, e que disponibilizam um mecanismo para a invocação de serviços, citam-se: RESCUE (JUSZCZYK; DUSTDAR, 2008), o mecanismo permite o registro de clientes em um *middleware* e que suas informações sejam enviadas para um servidor. No lado do servidor, o pedido é processado e enviado de volta ao *middleware*, que envia a resposta para o cliente solicitante da invocação; (SERHANI; GADALLAH, 2010), o nó requisitante inicia a fase de invocação de serviços com o provedor mais adequado para o atendimento. Se o provedor estiver indisponível, o solicitante enviará uma mensagem de invocação para o próximo provedor da sua lista; BAHSD (ARENELLA; SANTIS; MALANDRINO, 2014), utiliza uma abordagem entre camadas, que integra o algoritmo proposto com a camada de rede. As características da transmissão das mensagens ficam associadas ao algoritmo de roteamento. Os clientes, de posse das informações sobre os provedores aptos, enviam uma solicitação de serviço para estes provedores (invocação); e o LADP (KNIESS; LOQUES; ALBUQUERQUE, 2011), após o requisitante receber as respostas dos provedores aptos, envia uma mensagem de invocação de serviço para o(s) provedor(es) que possui(em) os serviços necessários para o atendimento, e que possam deslocar-se até o requisitante dentro do tempo definido.

Neste contexto, durante a invocação de serviços, um algoritmo de escalonamento pode ser utilizado para auxiliar a definir uma ordem de atendimento, a fim de assegurar uma distribuição dos recursos disponíveis e atender um maior número de invocações de serviços.

2.2 ESCALONAMENTO: ASPECTOS FUNDAMENTAIS

Algoritmos de escalonamento são fundamentais para garantir o provisionamento de Qualidade de Serviço (QoS) e compartilhar recursos em redes de comunicação. Em redes que suportam integração de serviços, torna-se necessário providenciar uma partilha de recursos. Por meio da garantia de QoS, um aplicativo de rede pode obter serviços que atendam a seus requisitos específicos, como por exemplo: menor atraso, maior taxa de transferência, otimização dos serviços e disponibilidade dos recursos (FATTAH; LEUNG, 2002).

Devido a natureza dinâmica das MANETs, que resultam em alterações frequentes de topologia e quebras de enlaces, o desenvolvimento de algoritmos de escalonamento torna-se mais difícil (ENZAI; RAIS; DARUS, 2010). Os algoritmos de escalonamento realizam duas funções (RUELA, 2006): i) decidir a ordem de serviço de fluxos em competição (pedidos de serviços); ii) gerar as filas de pedidos de serviços.

Os algoritmos de escalonamento são usados em qualquer sistema ou camada de protocolos em que ocorra controle de recursos, por exemplo, em elementos de rede (roteadores e comutadores), processos em sistemas operacionais ou recursos disponíveis na rede. Algoritmos de escalonamento desempenham um papel importante na provisão de diferentes níveis de QoS e para diferentes aplicações, permitindo controle diferenciado de atraso, largura de banda ou taxa de perdas (JAMES; KEITH, 2013).

Os requisitos essenciais para o desenvolvimento de um mecanismo de escalonamento depende das características do ambiente no qual ele será aplicado. No contexto deste trabalho, os requisitos listados a seguir foram identificados como essenciais para o escalonamento de recursos no cenário alvo (descrito no Capítulo 1):

- **Provedores móveis.** os provedores de recurso são móveis e se deslocam fisicamente até o local do atendimento;
- **Localização geográfica.** as informações geográficas dos nós possibilitam o conhecimento da localização dos provedores e requisitantes;
- **Tempo máximo para o atendimento.** o tempo máximo especificado pelo requisitante permite determinar através do escalonamento, quais requisitantes podem ser atendidos pelos provedores;
- **Velocidade de deslocamento dos nós provedores.** informações sobre a velocidade de deslocamento dos provedores é fundamental para determinar o tempo de chegada do provedor aos requisitantes.

Na literatura, encontram-se estratégias para o escalonamento de recursos em MANETs. Os trabalhos propostos por (CHUN; BAKER, 2002), (CHEN; GE; ZHAO, 2006), (ENZAI; RAIS; DARUS, 2010) e (YIN; YANG, 2009), apresentam soluções para escalonamento em redes ad hoc móveis.

Chun e Baker (2002), analisaram a dinâmica de filas em redes ad hoc móveis. Avaliou-se o efeito de diferentes algoritmos de escalonamento no desempenho da rede com os protocolos de roteamento *Dynamic Source Routing* (DSR) e *Greedy Perimeter Stateless Routing* (GPSR). As dinâmicas de enfileiramento com diferentes graus de mobilidade e protocolos de roteamento, mostram que a composição de pacotes na fila determina o efeito em dar prioridade aos pacotes de controle, ou estabelecer prioridades entre os pacotes de dados, especialmente para o atraso médio. A utilização da métrica, menor distância, junto ao algoritmo de escalonamento, possibilita que os pacotes de dados possam ser entregues rapidamente em uma rede congestionada, sem troca adicional de pacotes de controle pelos algoritmos. Os algoritmos de escalonamento analisados pelos autores foram: i) prioridade para controle de tráfego (*No-priority Scheduling* e *Priority Scheduling*); ii) prioridade de configuração no tráfego de dados (*Weighted-hop Scheduling*, *Weighted-distance Scheduling*, *Round Robin Scheduling* e *Greedy Scheduling*).

Chen, Ge e Zhao (2006), analisam o esquema de prioridade simples (caminho de atraso mais curto, dando prioridade ao roteamento de pacotes sobre pacotes de dados). Neste trabalho, os autores afirmam que o esquema de prioridade simples não representa a rede real, especialmente durante o congestionamento, e um nó deve considerar o nó vizinho, que compartilha o mesmo canal sem fio no escalonamento. Neste sentido, o algoritmo *Congestion Aware Scheduling Algorithm* (CARE) foi proposto. Nas MANETs, por causa da mudança dinâmica de topologia e escassez de largura de banda, um algoritmo de escalonamento deve responder rapidamente às mudanças. Assim, a informação de carga é selecionada como indicador de congestionamento. Como os pacotes de roteamento são importantes para a conectividade da rede e contêm as informações de carga, é dado a eles alta prioridade. Se dois fluxos diferentes tiverem a mesma prioridade, o escalonamento do tipo FIFO é aplicado. Quando o *buffer* está cheio, os pacotes pertencentes ao fluxo de prioridade mais baixa são descartados primeiro, para liberar espaço para os pacotes de fluxos de alta prioridade.

No trabalho de Enzai, Rais e Darus (2010), propõe-se um mecanismo de escalonamento chamado *Channel Aware Scheduling for MANETs* (CaSMA). O CaSMA, concentra-se no reconhecimento do canal e representa a qualidade do canal em termos da vida útil do caminho. O mecanismo considera o tamanho da fila para tornar o escalonamento ciente de congestionamento. No entanto, uma limitação do CaSMA é

que o algoritmo assume uma estimativa de tempo de vida do caminho. O desempenho do CaSMA varia com a precisão da estimativa dos enlaces.

O algoritmo *Energy-Efficient and Load Balanced Queue Scheduling Algorithm* (ELQS) proposto por Yin e Yang (2009), aborda o escalonamento no contexto do consumo de energia. Nesta abordagem, a prioridade de pacotes é designada dinamicamente de acordo com a capacidade atual do nó quanto a energia residual. O ELQS propõe reduzir o atraso de transmissão e prolongar o tempo de vida da rede.

2.3 ABORDAGEM HEURÍSTICA E META-HEURÍSTICA NA OTIMIZAÇÃO PARA ESCALONAMENTO DE RECURSOS

Os algoritmos heurísticos e os meta-heurísticos são alternativas em relação aos métodos exatos devido ao alto custo computacional. Os métodos heurísticos procuram alcançar os resultados para os problemas de forma rápida, mas não garantem que a solução obtida seja a solução ótima. Contudo, apresentam resultados satisfatórios em um tempo computacional aceitável. Os métodos e algoritmos meta-heurísticos são de uso, aplicação geral e buscam o ótimo global (TÁVORA et al., 2011). Assim como os métodos heurísticos, os meta-heurísticos também procuram alcançar os resultados para os problemas rapidamente, sem garantir que a solução obtida seja a solução ótima. Diferentemente dos métodos heurísticos, os meta-heurísticos contem vários parâmetros que devem ser ajustados, de acordo com o problema a ser resolvido.

A otimização, ou programação matemática, refere-se em escolher o melhor elemento em um conjunto de alternativas disponíveis, por meio de uma função objetivo específica que apresenta ou um valor próximo desejado (BECCENERI, 2008). Por exemplo, o objetivo desta dissertação é otimizar o gerenciamento dos provedores de serviços em relação ao número de atendimentos, que significa, maximizar o número de requisitantes atendidos. Para este trabalho utilizou-se programação linear, por estar sujeita a um conjunto de restrições, e inteira porque as variáveis se restringem a valores inteiros.

O problema abordado nesse trabalho de dissertação referente ao escalonamento de recursos para MANETs enquadra-se na área de otimização combinatória. No cenário alvo, o número de nós dentro do espaço de busca pode crescer exponencialmente, para determinar o custo da função de avaliação, tem-se a complexidade do algoritmo, representado pela função *f(n) = (n-1)!*. Por exemplo, 7 (sete) requisitantes em fila, equivale a 720 (setecentos e vinte) combinações possíveis (f(n) = *(7-1)!*). Em face destas características, estudou-se duas abordagens: o Algoritmo A-Estrela (A*) na Seção 2.3.2 e o Algoritmo Genético (AG) na Seção 2.3.1.

2.3.1 Fundamentação do A-Estrela (A*)

O algoritmo A-Estrela (A*) é reconhecido na literatura por sua precisão ao encontrar o melhor caminho para problemas de otimização. O algoritmo A* proposto por Hart, Nilsson e Raphael (1968), avalia os nós, combinando *g(n)*, que corresponde ao custo para alcançar cada nó, e *h(n)*, heurística que define o custo estimado do melhor caminho de *n* para ir do nó até o seu objetivo. Uma boa função heurística deve ser admissível, nunca superestimar o custo real da solução (p.ex. em uma distância direta, é admissível porque o caminho mais curto entre dois pontos é sempre uma linha reta).

Segundo Pearl (1984), nas propriedades do A-Estrela, se *h* é admissível, *f(n)* nunca irá superestimar o custo real da melhor solução através de *n*, isso porque a função *f(n)* sempre cresce analisando os caminhos de cima para baixo. Para provar a otimalidade do algoritmo é necessário que a monotonicidade seja garantida, ou seja, nunca deve ocorrer que *f(n')* < *f(n)*, onde *n* é o pai de *n'*, uma vez que todo o caminho que passa por *n'* passa também pro *n* (seu pai) (PEARL, 1984). A Equação 2.1 representa o cálculo realizado pelo algoritmo A-Estrela para avaliar os nós (HART; NILSSON; RAPHAEL, 1968).

$$f(n) = g(n) + h(n) \tag{2.1}$$

Dessa forma, para encontrar a solução de custo mais baixo, é experimentar primeiro o nó com o menor valor de *g(n) + h(n)*. Portanto, a Equação 2.1 é dito o custo estimado da melhor solução que passa por *n*.

O cálculo da função *h(n)* utilizado neste trabalho, é realizado com base no algoritmo mínimo *spanning tree* (PRIM, 1957), para encontrar o caminho com menor custo. O mínimo *spanning tree* é um algoritmo que têm um caráter guloso: em cada iteração, pega a aresta que parece mais promissora localmente sem preocupar-se com o efeito global dessa escolha. Eles são os protótipos da estratégia gulosa, que se aplica a vários outros problemas computacionais. A formulação do mínimo *spanning tree* é dado pela Equação 2.2.

$$min \sum_{i,j \in E}^{n} d_{ij} x_{ij} \tag{2.2}$$

Sujeito às restrições:

$$\sum_{i,j \in E}^{n} x_{ij} = n - 1 \qquad \forall i \in N \tag{2.3}$$

$$\sum_{i,j \in E : i \in S, j \in S} x_{ij} \qquad \leq |S| - 1 \forall S \subseteq V \tag{2.4}$$

Na Equação 2.2, o custo associado ao deslocamento entre os nós $i, j \subset N$ é dado por d_{ij}, e x_{ij} representa a ligação entre o nó *i* e o nó *j* (designadas como variáveis de decisão). A Equação 2.3 estabelece as restrições para que o provedor passe uma única vez em cada nó requisitante, devendo partir para o próximo ponto até que todo o percurso seja realizado. A Equação 2.4 representa confirmação de deslocamento x_{ij}.

Diferentemente do algoritmo de *Dijkstra* (CORMEN et al., 2002), o algoritmo A-Estrela é considerado como o melhor algoritmo de *pathfinding*, devido a utilização de uma heurística aplicada em *h(n)*. No *Dijkstra*, a procura é feita em todas as direções, e normalmente explora uma área muito maior do mapa. Em decorrência dessa limitação, o algoritmo de *Dijkstra* é mais lento que o A-Estrela. O fato do A-Estrela ser um algoritmo que implementa uma heurística de plano de corte, torna-o mais eficiente entre os algoritmos de busca, característica interessante para a resolução do problema do escalonamento no cenário alvo.

2.3.2 Fundamentação do Algoritmo Genético (AG)

Os Algoritmos Genéticos (AGs) foram citados pela primeira vez em 1975, por Holland et al. (1992). Posteriormente, passaram a ser relacionados com os termos de computação evolucionária ou algoritmos evolucionários. Os AGs, em geral, consistem em uma técnica de Inteligência Computacional (IC), fundamentada em teorias, conceitos da genética e evolução das espécies (HOLLAND et al., 1992). Os Algoritmos Genéticos se propõem a resolver problemas complexos do mundo real, no qual métodos tradicionais de otimização possuem dificuldades em encontrar soluções viáveis (SHUKLA; PANDEY; MEHROTRA, 2015).

A motivação original para a meta-heurística foi uma analogia biológica (GLOVER; KOCHENBERGER, 2006). Na criação de plantas e animais, existe a seleção com base em características desejáveis, as quais são determinadas geneticamente, de acordo com a forma como se combinam os cromossomos dos pais. No caso dos AGs, a população utilizada é composta por sequências, relacionadas como cromossomos, e a recombinação das sequências comumente se dá pelo uso de simples analogias com mutações (operador realiza a escolha aleatória de um subconjunto de genes e seus valores são trocados pelos de seus genes alelos, que se refere às diferenças) e o *crossover* (operador que realiza uma combinação entre os genes dos

pais). Entende-se por gene, a representação da solução, onde cada parâmetro é de acordo com o alfabeto utilizado (binário, inteiro ou real) (LINDEN, 2008).

De forma simplificada, os AGs funcionam da seguinte maneira: é gerada uma população inicial de *n* indivíduos aleatoriamente (solução candidata). Cada um dos indivíduos da população (conjunto de indivíduos) representa uma possível solução para o problema, ou seja, um ponto no espaço de soluções. Em seguida, busca-se um processo iterativo para gerar uma boa solução por meio da evolução das melhores soluções da população atual, de acordo com sua função de aptidão (*fitness*), avaliando-se a qualidade do indivíduo. Três operadores genéticos geralmente são aplicados: elitismo (cópia simples das melhores soluções), cruzamento (combinação das partes de pares de soluções) e mutação (altera a composição do indivíduo para causar uma ruptura no processo de convergência) (CARVALHO et al., 2011).

Na implementação do algoritmo, por meio de observações, optou-se pela escolha das seguintes técnicas: método de torneio para a seleção dos indivíduos; permutação de 2 (dois) pontos para o cruzamento entre os indivíduos, para que não ocorra repetição dos segmentos, *Partially Mapped Crossover (PMX)*; o elitismo que faz cópia simples das melhores soluções e; para a mutação nos indivíduos, utilizou-se a técnica de mutação aleatória, responsável por mudar a ordem dos genes no indivíduo (LINDEN, 2008).

Uma das vantagens dos AGs consiste na implementação simples que oferece uma fonte de obtenção de soluções de busca global, tornando-o atrativo para a resolução do problema do escalonamento do cenário alvo.

2.4 CONCLUSÕES DO CAPÍTULO

A descoberta de serviços possibilita a detecção de serviços oferecidos por dispositivos em uma rede, sendo composta pelas fases de descoberta, seleção e invocação de serviços. Na fase de invocação, os requisitantes podem alocar e usar os serviços oferecidos na rede.

No cenário de motivação, apresentado no Capítulo 1, a descoberta de serviços contribui para a descoberta, seleção e invocação dos provedores de recursos. Contudo, o principal desafio é gerenciar os recursos em relação ao número de atendimentos, porque o número de vítimas em uma situação de emergência pode aumentar de acordo com a gravidade da tragédia. Neste sentido, soluções para o escalonamento dos provedores são essenciais para uma distribuição justa dos provedores entre as vítimas.

As abordagens apresentadas para escalonamento de recursos em MANETs (CHUN; BAKER, 2002), (CHEN; GE; ZHAO, 2006), (ENZAI; RAIS; DARUS, 2010)

e (YIN; YANG, 2009), focam no escalonamento de recursos da rede como, largura de banda, memória e processadores, não sendo adequadas para o escalonamento no cenário alvo deste trabalho. Soluções para MANETs que consideram o escalonamento de provedores de recursos móveis com velocidade de deslocamento variadas não foram encontradas na literatura. Dado a complexidade do problema apresentado, utilizou-se os métodos discutidos nesse capítulo para possibilitar encontrar uma solução fidedigna para o escalonamento de recursos no cenário alvo.

No próximo capítulo, apresentam-se os trabalhos relacionados à descoberta de serviços para MANETs, operando em cenários pós-desastres. Discute-se as características das soluções investigadas em relação à descoberta e seleção de serviços, a invocação de serviços e escalonamento dos recursos.

3 TRABALHOS RELACIONADOS

Neste capítulo, alguns trabalhos que abordam o tema da descoberta de serviços em MANETs são apresentados. Trabalhos envolvendo a descoberta de serviços em cenários de emergência estão na Seção 3.1, enquanto trabalhos relacionados à descoberta de serviços baseados em meta-heurísticas estão na Seção 3.2. Ao final do capítulo, apresenta-se uma comparação entre os trabalhos relacionados e as conclusões finais sobre os trabalhos investigados.

3.1 DESCOBERTA DE SERVIÇOS EM CENÁRIOS DE EMERGÊNCIA

No trabalho proposto por Juszczyk e Dustdar (2008), os autores desenvolveram uma abordagem denominada RESCUE. A abordagem consiste em um *middleware* para MANETs em ambientes pós-desastres, e que oferece mecanismos para a descoberta e invocação de serviços. No RESCUE os nós localizados na rede anunciam seus serviços e escutam os anúncios de outros nós que se encontram na mesma rede. Informações sobre os serviços descobertos, sendo locais ou remotos, são mantidas em um banco de dados que contém metadados publicados (descrições) sobre todos os serviços. Os metadados consistem em registros obrigatórios que são necessários para identificar e invocar os serviços. O RESCUE, oferece três serviços: (a) um serviço de implantação que está disponível apenas localmente, no próprio nó; (b) um serviço de informações que fornece metadados sobre os serviços implantados e; (c) um serviço para receber respostas para invocações dos requisitantes.

Os aplicativos do cliente procuram os serviços consultando o banco de dados, ou colocando inscrições para serem notificados quando os serviços estiverem disponíveis. Devido ao protocolo de publicação ser reativo, é possível que os clientes sejam notificados quase que imediatamente após a aparição de um serviço na rede. Na fase de invocação, os clientes são registrados em um *middleware* e o pedido é anotado com cabeçalhos SOAP (*Web Service* baseado em XML), e posteriormente enviados a um servidor. No lado do servidor, o pedido é processado e enviado de volta ao *middleware*. O *middleware*, por sua vez, consulta o seu banco de dados local para verificar se o cliente ainda está disponível e, em seguida, envia a resposta para o cliente solicitante. Em MANETs, como no cenário alvo, o uso de mecanismos de anúncios periódicos de serviços pode gerar um número excessivo de mensagens na rede. Neste trabalho, questões de escalonamento de recursos não foram consideradas pelos autores.

No trabalho de Roy et al. (2010), foi proposta uma solução para situações de emergência baseada no *framework* JINI (WALDO, 1999). Os provedores de serviços

e clientes trabalham de forma colaborativa na MANET. O sistema é composto de nós móveis, que são classificados como: provedores de serviços, clientes e hospedeiros de serviços (líderes). As unidades da equipe de resgate de posse de dispositivos habilitados em modo *ad hoc*, organizam-se em grupos atuando como nós líderes. O aplicativo de um cliente pode descobrir e se registrar nos líderes mais próximos, fornecendo informações como, por exemplo, localização e outros dados vitais, por meio de uma interface gráfica do utilizador (GUI). As equipes enviam uma resposta aos sobreviventes, indicando o tempo para chegar ao local de atendimento. A seleção e a invocação de serviços são centralizadas no líder. A escolha de centralizar as duas funções no líder, pode levar a uma queda rápida da bateria do dispositivo, bem como tornar o líder um potencial ponto de falha do sistema.

Jang et al. (2009) propuseram o *Rescue Information System for Earthquake Disasters* (RISED), um sistema de coleta de informações sobre terremotos para operações de atendimento e resgate. O objetivo do RISED é fornecer atualizações relacionadas às informações utilizadas em resgate, tais como, localização, possíveis danos à vida e construções, salvamento, socorro e gerenciamento de recursos disponíveis. O sistema é implantado em uma arquitetura de duas camadas: o RISED Central e o RISED Local. O RISED Central é implantado em uma central de comando de operação de resgate, responsável pela avaliação de desastres e gerenciamento de recursos. O RISED Local opera em um centro de comando local, usado para recolher dados sísmicos, e atualizar a RISED Central periodicamente. O RISED Local pode operar de forma independente, caso não tenha acesso ao RISED Central. O RISED atua em dispositivos já configurados em uma rede ad hoc. A invocação de recursos é realizado pelo RISED Central. A política de escalonamento de recursos adotada pelo RISED Central vem por meio das estatísticas geradas pela coleta das informações do RISED, o comando de resgate central pode alocar e despachar recursos apropriados para as áreas atingidas com base em suas reais necessidades.

Serhani e Gadallah (2010), apresentam um protocolo de descoberta de serviços para MANETs em cenários de desastre. Os provedores de serviços enviam anúncios para uma central que realiza a seleção dos provedores para os clientes. O protocolo permite localizar os serviços com base nos critérios especificados pelo cliente: proximidade do prestador de serviços (localização) e gravidade da emergência. O protocolo faz a divisão da rede em diferentes zonas geográficas. Cada zona contém os provedores móveis de diferentes categorias. Os provedores podem se mover dentro de suas zonas e/ou de uma zona para outra e os requisitantes são estacionários. Ao receber as respostas com as informações sobre um ou mais provedores para atender a uma solicitação, o nó requisitante inicia a fase de invocação de serviço com o mais apto para o atendimento. Se o provedor estiver disponível, ele realizará o atendimento

para este solicitante. Por outro lado, se o provedor já estiver reservado para outro atendimento, o solicitante enviará uma mensagem de invocação para o próximo provedor da sua lista.

No trabalho de Park et al. (2013), apresenta-se uma arquitetura distribuída de descoberta e compartilhamento de serviços para MANETs. A proposta consiste em integrar o protocolo de roteamento pró-ativo *Destination-Sequenced Distance Vector (DSDV)* com o protocolo de descoberta de serviços. O protocolo DSDV mantém as informações utilizadas pelo usuário na tabela de roteamento. Essas informações servem para descobrir e localizar o serviço de acordo com o tipo de recurso desejado. São utilizadas de cada nó informações como, posição geográfica, topologia da rede e o tipo de serviço. A topologia da MANET e a preferência do usuário são utilizadas na distribuição dos serviços. A arquitetura do protocolo compreende: a descoberta do serviços, o registro do serviços e a invocação do serviços.

A estrutura da rede do mecanismo proposto é composto por *Service Provider* (SP), *General Node* (GN), *Main-Service Directory* (MSD) e *Sub-Service Directory* (SSD). GN e MSD mantêm as informações de serviço gerenciadas em sua região. SSDs contêm as informações de cada prestador de serviços de acordo com o tipo de serviço registrado no MSD. O GN faz a solicitação de serviços, transmitindo um *Service Request Packet* (SRP) para o SSD mais próximo, contendo o tipo de serviço necessário. O SSD, por sua vez, procura pelo serviço solicitado e gera uma descrição do serviço solicitado pelo GN. Uma descrição composta pela, identificação do tipo de serviço, código, parâmetro de invocação, TTL e serviço expirados são acessados com o endereço de rede do SP. Depois disso, o SSD envia a descrição para o solicitante GN. O GN recebe um *Service Replay Packet* (SREP) e armazena a descrição do serviço, escolhe um SP e envia o *Service Invoke Packet* (SINVK) para invocar o serviço do SP. O SP recebendo o SINVK envia um *Service Response Packet* (SRESP) para o GN realizar sua reserva. A integração do protocolo de descoberta de serviço com o protocolo de roteamento, limita o funcionamento do mecanismo de descoberta ao protocolo de roteamento usado.

O protocolo de descoberta *Location Aware Dicovery Protocol* (LADP) proposto por (KNIESS; LOQUES; ALBUQUERQUE, 2011) foi desenvolvido para atender aos requisitos de sistemas para ambientes de desastres naturais e não-naturais. A finalidade da arquitetura do protocolo LADP foi especificada para oferecer suporte no desenvolvimento de aplicações para descoberta, seleção e invocação de serviços em MANETs de múltiplos saltos.

No protocolo LADP, as vítimas iniciam o processo de descoberta de serviços através de um dispositivo móvel interconectado à MANET. Os nós são classificados em requisitantes (vítimas), provedores de serviços e intermediários, que atuam como

relay, repassando as informações. O protocolo de descoberta de serviços deve estar previamente configurado nos dispositivos das vítimas, provedores e intermediários.

A mensagem irá trafegar dentro de um raio R_i. O raio é criado em relação ao requisitante, com base no tempo máximo para atendimento da requisição, definido pelo requisitante na mensagem de descoberta de serviços, e na velocidade máxima de deslocamento dos nós na rede. Qualquer provedor de recursos fora do raio, que receba a mensagem de descoberta, irá descartá-la. O raio R_i é dado pela Equação 3.1, o valor de v_{max} especifica a velocidade máxima do nó e Δ_{tmax} representa o tempo máximo de atendimento para uma requisição (KNIESS; LOQUES; ALBUQUERQUE, 2011):

$$R_i \leftarrow v_{max} \times \Delta_{tmax} \tag{3.1}$$

No protocolo LADP, um mecanismo de seleção de serviços, seleciona automaticamente as respostas dos melhores provedores, durante o encaminhamento da mensagem na rede, e as direciona para os requisitantes. Após receber as respostas dos provedores aptos, o requisitante envia uma mensagem de invocação de serviço para o(s) provedor(es) selecionado(s), ou seja, o(s) provedor(es) que possui(em) os recursos necessários para o atendimento, e que possa(m) deslocar-se até o requisitante dentro do tempo definido. A arquitetura do protocolo de descoberta de serviço proposta por Kniess et al. (2011) é mostrada na Figura 3.1, incluindo os componentes para descoberta, seleção e invocação.

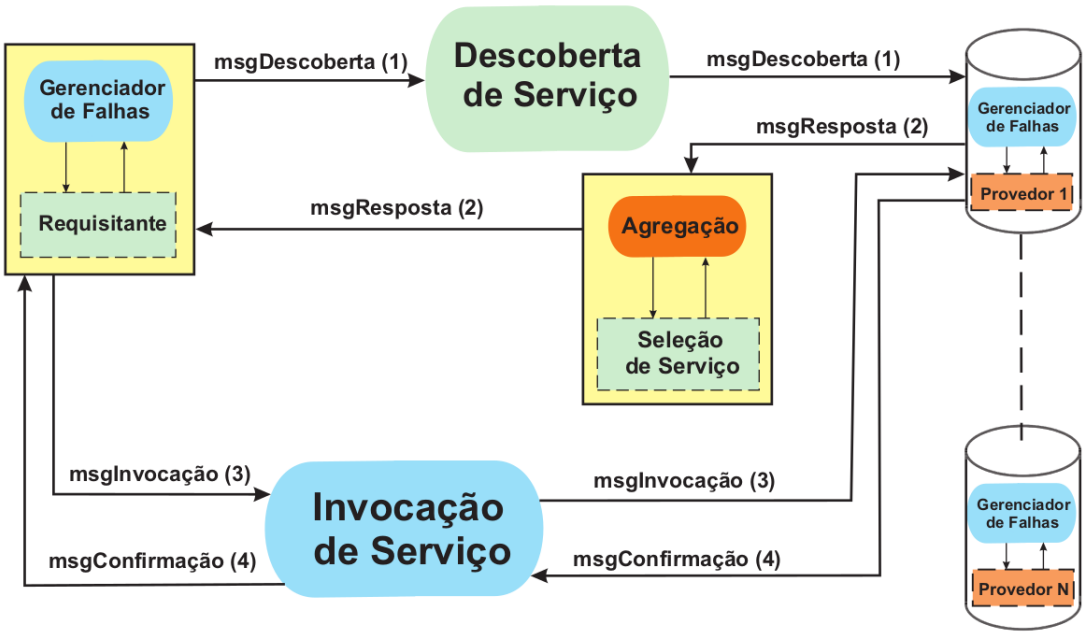


Figura 3.1 – Arquitetura do protocolo LADP. Fonte: Kniess et al. (2011)

Segundo Kniess et al. (2011), os requisitantes iniciam o processo de descoberta de serviço por meio de um dispositivo móvel interconectado à MANET. A mensagem de descoberta (*msgDescoberta*) será transmitida para a rede, salto a salto, por meio de um mecanismo de difusão do tipo *flooding*. Na mensagem de descoberta, o nó requisitante envia as seguintes informações: uma identificação do nó, uma identificação da requisição, o número de sequência da requisição, suas coordenadas geográficas, o tempo máximo para atendimento da requisição (Δ_{tmax}), o serviço procurado e o número de provedores desejado. Um mecanismo de seleção de serviços executa um filtro seletivo nos nós intermediários durante o encaminhamento da mensagem de resposta (*msgResposta*) na rede, e envia as respostas dos provedores mais aptos na direção do nó requisitante. Após selecionar os provedores, o requisitante envia uma mensagem de invocação de serviço (*msgInvocação*) para o(s) provedor(es) selecionado(s). Este, por sua vez, deve enviar uma mensagem de confirmação de recebimento da invocação (*msgConfirmação*) para o requisitante. Por fim, o requisitante envia uma mensagem de reconhecimento da confirmação (*msgRecConfirmação*) ao provedor. Cada mensagem de reconhecimento da confirmação recebida tem por função notificar ao requisitante que o provedor irá se deslocar fisicamente até o local onde o recurso é necessário.

Na Figura 3.2 (KNIESS; LOQUES; ALBUQUERQUE, 2011), descreve-se as etapas a partir do envio do pedido pelo nó requisitante, até o recebimento pelo provedor da mensagem de reconhecimento da confirmação.

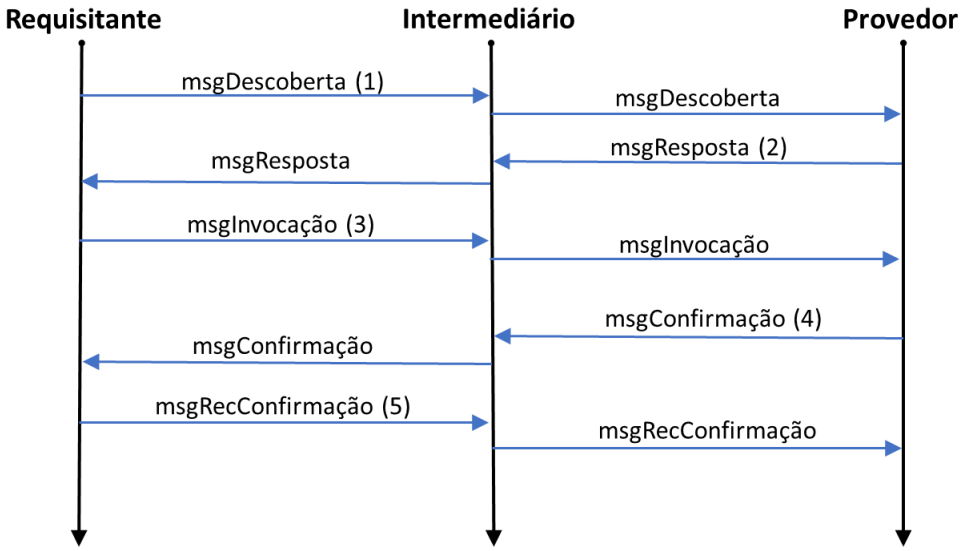


Figura 3.2 – Diagrama de transição: descoberta de serviços. Fonte: Kniess et al. (2011)

Observa-se na Figura 3.2, que o evento (1) corresponde ao envio da mensagem de descoberta na rede pelo nó requisitante. O provedor apto para o atendimento recebe a mensagem de descoberta de serviço e envia uma mensagem de resposta

para o requisitante (2). Após a fase de seleção de serviços, o requisitante envia uma mensagem de invocação de serviço (3) para o provedor selecionado. Em seguida, o provedor envia uma mensagem de confirmação (4) para o nó requisitante. O requisitante, por fim, envia uma mensagem de reconhecimento da confirmação (5) para o provedor, que, ao receber a mensagem, inicia seu deslocamento na direção do primeiro requisitante, cuja a mensagem de reconhecimento da confirmação (5) tenha chegado no provedor. Neste protocolo, não existe uma política de escalonamento de recursos adequada, por exemplo, o requisitante atendido pelo provedor pode estar geograficamente mais distante que outros, cuja as mensagens de reconhecimento da confirmação chegaram depois no provedor, em consequência de terem seguido caminhos mais congestionados.

Entretanto, o protocolo LADP atende a todos os requisitos listados na Seção 2.2 do Capítulo 2, cruciais para o desenvolvimento de soluções para o cenário alvo, provedores móveis, protocolo geográfico e tempo máximo para o atendimento dos pedidos. Neste sentido, considerou-se o LADP o protocolo de descoberta de serviço mais adequado para o desenvolvimento da abordagem de escalonamento de recursos (descrita no Capítulo 4) proposta neste trabalho de dissertação. Definiu-se que o mecanismo de escalonamento de recursos atuará na fase de invocação de serviço do protocolo LADP, visando o compartilhamento de recursos para o máximo de atendimentos possível.

3.2 DESCOBERTA DE SERVIÇOS E META-HEURÍSTICAS

A crescente complexidade dos problemas do mundo real tem motivado os cientistas da computação a buscarem métodos eficientes na resolução de problemas de permutação (PARPINELLI; LOPES, 2011). O escalonamento de recursos no cenário alvo foi identificado como sendo um problema de permutação da área de otimização combinatória. Destaca-se na área de otimização as meta-heurísticas: Colônia de Formigas - *Ant Colony Optimization* (ACO) (SINGH; KUMAR; VERMA, 2012), Colônia de Abelhas - *Bee Colony Optimization* (BCO) (KARABOGA; AKAY, 2009) e os Algoritmos Genéticos - *Genetic Algorithms* (GAs) (apresentado na Seção 2.3.2).

O ACO é um algoritmo do tipo meta-heurística para problemas de otimização combinatória. Os algoritmos de otimização de colônias de formigas focam na combinação de informações. A priori sobre a estrutura de um resultado com informações; e, a posteriori sobre a estrutura dos resultados obtidos anteriormente. Colônias de formigas são capazes de encontrar o caminho mais curto entre seu ninho e uma fonte de alimento, depositando e reagindo à trilha de feromônio, que fornecem ajuda às futuras formigas, na intenção de otimizar caminhos para a alimentação (SINGH; KUMAR; VERMA, 2012).

O BCO é uma meta-heurística que baseia-se nos padrões de busca das abelhas por alimento. O objetivo das abelhas, quando estão procurando comida, é maximizar a quantidade de alimento obtido dentro de um período de tempo. Para isso, dois fatores são levados em consideração: a distância da fonte de comida até a colmeia e a facilidade de aquisição desta comida. A busca das abelhas pelo alimento é feita em duas partes. Na primeira parte, abelhas exploradoras são enviadas para encontrar os pontos de coleta de alimentos, que inicialmente ocorre de forma aleatória. Elas retornam, depositam o pólen recolhido, e fazem uma dança conhecida por *waggle dance*. Esta dança é utilizada para passar as informações sobre o local em que cada abelha exploradora fez sua coleta de pólen. Com a dança, elas indicam a distância da fonte de comida, a direção na qual ela se encontra e a quantidade de comida existente. No segundo passo, são recrutadas as abelhas seguidoras, que buscarão a comida nos locais indicados pelas abelhas exploradoras. Os pontos mais promissores recebem um número maior de abelhas seguidoras, ou seja, mais abelhas são enviadas para onde existe maior possibilidade de arrecadar mais comida. Cada vez que as abelhas retornam à colmeia, ocorre novamente a dança, e outras abelhas seguidoras são enviadas para os locais promissores (KARABOGA; AKAY, 2009).

Alguns trabalhos foram propostos levando em conta o uso de meta-heurísticas na descoberta de serviços. Dentre estes trabalhos citam-se: (BITAM; BATOUCHE, 2008), (SIDDARTH; SEETHARAMAN, 2012) e (ARENELLA; SANTIS; MALANDRINO, 2014).

No estudo realizado por Bitam e Batouche (2008), utiliza-se um sistema de colônia de abelhas para descoberta da topologia de rede e serviços disponíveis. A arquitetura foi inspirada na comunicação entre as abelhas. Em outras palavras, no BCO, cada nó da rede é capaz de coletar informações de topologia da rede, e conhecer situações dos outros nós (p.ex. distâncias em relação aos outros nós). Considerado um sistema adaptativo, é caracterizado pela atualização em tempo real. O sistema considera que cada nó possui uma zona de cobertura, sendo dividido em dois conjuntos. O primeiro conjunto de nós contém os nós próximos, e o segundo conjunto contém os nós mais distantes. Cada nó calcula as distâncias e as direções, ou seja, os nós transferem as informações da topologia descoberta para cada nó. A grande importância do sistema nesta área é a posse dos dados globais (toda a topologia da rede) a partir de dados parciais (no nível do nó), sem qualquer administração central. A arquitetura do BCO contém: descrição do sistema, *round dance* que descreve a localização do alimento (nós ou serviços da rede) e *waggle dance* onde é feito o cálculo da distância que separa a fonte de alimento da colméia. Uma característica observada pelos autores é que o sistema evita inundação de mensagens decorrentes dos nós mais distantes, diminuindo, dessa forma, o consumo de energia dos nós mais próximos.

No trabalho de Siddarth e Seetharaman (2012), propõe-se uma arquitetura de descoberta de serviço com QoS baseada em um cluster que, utiliza inteligência de enxame *Ant Colony Optimization* (ACO). Sua arquitetura é composta por dois agentes: 1) *Service Discovery Agent* (SDA), que é independente do protocolo de roteamento e; 2) *Swarm Routing Agent* (SRA), responsável pelo roteamento de atividades para propagar mensagens de descoberta de serviço e rastrear alterações da topologia. O funcionamento ocorre da seguinte forma: uma solicitação de serviço é encaminhada pelo cliente. A mensagem de solicitação é transmitida juntamente com os parâmetros de QoS necessários, como potência, distância e velocidade da CPU para o líder do *cluster* de origem. A inteligência de enxame (*SWARM Inteligence*) é usada para estabelecer o roteamento de caminho mais curto intra e inter *cluster*. Um servidor ciente de QoS é selecionado para processar a solicitação de serviço e enviar a resposta de volta ao cliente. Algumas limitações podem ser encontradas nesse tipo de arquitetura, por exemplo, a não-convergência para a solução ótima em problemas maiores e diversidade de soluções boas (não-ótima).

O algoritmo de descoberta e seleção de serviço *Bee Ad Hoc Service Discovery* (BAHSD) proposto por Arenella et al. (2014) é baseado no conceito de *(Bee Colony Optization - BCO)* (TEODOROVIC et al., 2006). BCO é uma meta-heurística derivada do comportamento de uma colônia de abelhas para resolver problemas combinatórios. O algoritmo BAHSD utiliza uma abordagem *cross-layer* (entre camadas), que integra o algoritmo proposto com a camada de rede. Especificamente, o algoritmo de descoberta de serviço opera juntamente com o algoritmo de roteamento *Bee Ad Hoc* proposto por Teodorovic et al. (2006). O *Bee Ad Hoc* coleta informações de roteamento, tais como quebras de rota e atualizações, a fim de minimizar o número de mensagens de controle e o consumo de energia. Devido ao fato da função de descoberta e seleção de serviço estar integrada ao protocolo de roteamento *Bee Ad Hoc*, as características da transmissão das mensagens do protocolo BAHSD ficam associadas ao protocolo de roteamento, diminuindo a interoperabilidade do protocolo. Os clientes, ao receberem as informações sobre os provedores aptos, enviam uma solicitação de serviço para estes provedores (invocação). Questões relacionadas ao escalonamento de recursos não são abordadas pelos autores.

Para o desenvolvimento do algoritmo de escalonamento de recursos apresentado no Capítulo 4, foi escolhido o algoritmo genético, porque sua proposta permite a resolução de problemas de permutação, considerando que já existe um ponto de partida, o cromossomo inicial. Outra vantagem dos algoritmos genéticos é o fato de ser um métodos de busca global (MICHALEWICZ, 1996). Além disso, os AGs são capazes de realizar buscas em espaços de soluções (hipóteses) com regiões complexas, em que o impacto de cada parte da solução do problema pode ser de difícil modelagem,

quando feita por meio de técnicas convencionais (MITCHELL et al., 1997).

Na Tabela 3.1, mostra-se uma comparação dos protocolos de descoberta de serviços apresentados neste capítulo. Para fins de comparação, utilizou-se os critérios apresentados no Capítulo 2 na Seção 2.2: se o provedor é móvel e desloca-se até o requisitante (Provedor Móvel); se utiliza a localização geográfica (Geográfico); se estabelece um limite máximo para o atendimento (Tempo de Atendimento); e se possui mecanismo de escalonamento de recursos (Escalonamento de Recursos).

Tabela 3.1 – Comparação: protocolos de descoberta de serviços (MANETs).

Autor/Ano	Provedor Móvel	Geográfico	Tempo de Atendimento	Escalonamento de Recursos
(BITAM; BATOUCHE, 2008)	-	SIM	-	-
(JUSZCZYK; DUSTDAR, 2008)	-	SIM	-	-
(JANG; LIEN; TSAI, 2009)	SIM	SIM	SIM	-
(ROY; KAR; MUKHERJEE, 2010)	SIM	SIM	-	-
(SERHANI; GADALLAH, 2010)	SIM	SIM	-	-
(KNIESS; LOQUES; ALBUQUERQUE, 2011)	SIM	SIM	SIM	-
(PARK et al., 2013)	SIM	-	-	-
(SIDDARTH; SEETHARAMAN, 2012)	-	SIM	-	-
(ARENELLA; SANTIS; MALANDRINO, 2014)	SIM	-	-	-
ABORDAGEM PROPOSTA (MARCELO, 2019)	SIM	SIM	SIM	SIM

A discussão apresentada sobre os protocolos de descoberta de serviços, reforça a necessidade de um mecanismo de escalonamento de recursos que contemple as peculiaridades do cenário alvo deste trabalho, no qual os provedores devem se deslocar para o local do atendimento dentro de um tempo limite, e atender ao maior número de vítimas.

Em face das limitações dos protocolos de descoberta de serviços, este trabalho oferece uma solução para o escalonamento de recursos, considerando a localização geográfica dos requisitantes, o tempo de deslocamento dos provedores ao local do evento e o tempo máximo para atendimento determinado pelo requisitantes, a fim de aumentar o número de atendimentos dentro de um tempo hábil. A solução descrita no Capítulo 4 estende o protocolo LADP (KNIESS; LOQUES; ALBUQUERQUE, 2011), com o objetivo de prover um melhor gerenciamento dos provedores de serviço em relação às vítimas.

3.3 CONCLUSÕES DO CAPÍTULO

Neste capítulo, foram apresentados os trabalhos relacionados à descoberta de serviços em MANETs e, uma comparação destes trabalhos em relação aos requisitos considerados essenciais para o desenvolvimento de uma estratégia de escalonamento de recursos para cenários pós-desastres assistidos por MANETs.

O problema de escalonamento deste trabalho enquadra-se na área de otimização combinatória, por necessitar de múltiplas variáveis para o escalonamento de

recursos, e apresentar um número expressivo de possibilidades de combinação entre os requisitantes. No cenário alvo, o número de nós dentro do espaço de busca pode crescer exponencialmente. Em face destas características, escolheu-se o algoritmo genético como base para a especificação da abordagem para escalonamento de recursos (ver Capítulo 4).

O protocolo LADP (KNIESS; LOQUES; ALBUQUERQUE, 2011) atende a todos os requisitos especificados para o cenário alvo. Entretanto, na versão original do LADP, questões de escalonamento de provedores não foram consideradas. No Capítulo 4, discute-se a abordagem proposta para solucionar as limitações encontrados na versão original do LADP em relação ao escalonamento dos provedores de serviços.

4 ABORDAGEM PROPOSTA

Este capítulo descreve o funcionamento da abordagem proposta para escalonamento de recursos, que estende o protocolo de descoberta de serviços LADP de Kniess et al. (2011) para cenários pós-desastres. O objetivo principal é prover o gerenciamento dos provedores de recursos em relação aos atendimentos que precisam ser realizados. Os detalhes técnicos do protocolo LADP, necessários para o entendimento da proposta são apresentados na Seção 3.1. Os mecanismos para escalonamento de recursos com base nos algoritmos Genético e A-Estrela são ilustrados nas Seções 4.1.3 e 4.1.2, respectivamente.

4.1 ABORDAGEM PARA ESCALONAMENTO DE RECURSOS

Com base no diagrama de transição de mensagens proposto por Kniess et al. (2011), do processo de descoberta de serviços apresentado na Figura 3.2, intrínseca na Seção 3.1, observa-se que o protocolo LADP não trata do escalonamento dos recursos entre os requisitantes e provedores. No LADP, reserva-se o recurso (provedor) para o requisitante, cuja a mensagem de reconhecimento da confirmação (*msgRecConfirmação*) for a primeira a chegar no provedor.

No diagrama de transição representado pela Figura 4.1, ilustram-se as etapas adicionadas ao protocolo LADP em relação ao escalonamento de recursos. No protocolo LADP, os requisitantes enviam uma mensagem de invocação de serviço *msgInvocação* (3), após receber as mensagens de respostas dos provedores aptos para o atendimento. A partir deste ponto, foram desenvolvidos no escopo deste trabalho, mecanismos para o escalonamento dos recursos (provedores) entre os requisitantes.

O nó provedor envia ao requisitante uma mensagem de confirmação da invocação *msgConfirmação* (4) e aguarda a mensagem de reconhecimento da confirmação do requisitante *msgRecConfirmação* (5). Em seguida, o nó provedor envia uma mensagem ao requisitante, informando que a solicitação deste requisitante foi incluída na fila *msgFila* (6). Por fim, após o processamento da fila, o provedor envia uma mensagem de atendimento *msgAtendimento* (7) ao(s) requisitante(s) selecionado(s) para o atendimento pelo algoritmo de escalonamento de recursos.

A troca de mensagens ilustrada na Figura 4.1 demanda um controle temporal entre elas, visando assegurar o correto funcionamento do protocolo LADP e do escalonamento de recursos. No contexto do escalonamento de recursos, foram desenvolvidos temporizadores para gerenciar o envio e recebimento de mensagens pelo(s) provedor(es) e requisitante(s). Por exemplo, o provedor, após receber a primeira mensa-

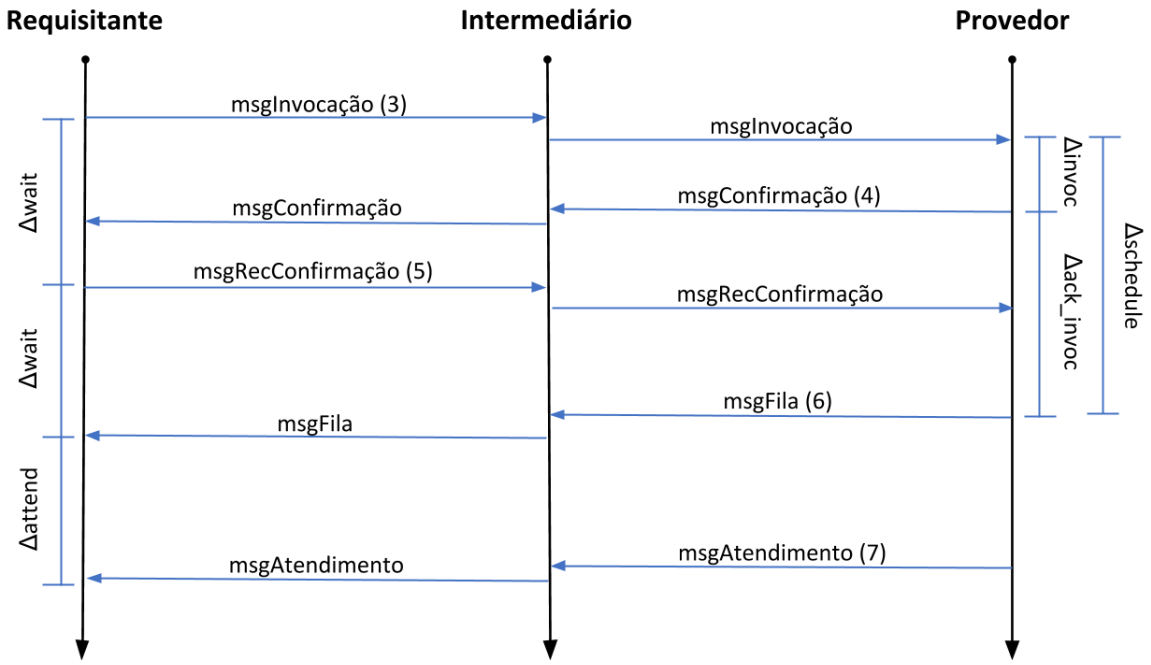


Figura 4.1 – Diagrama de transição: escalonamento de recursos, adaptado de Kniess et al. (2011).

gem *msgInvocação* (3) de um requisitante, aciona o temporizador, denominado Δ_{invoc}. O temporizador Δ_{invoc} e o temporizador Δ_{ack_invoc} foram desenvolvidos para definir o tempo total em que o provedor espera receber mensagens dos requisitantes, para os quais o provedor enviou mensagem de resposta (*msgResposta*) Figura 3.2.

O temporizador Δ_{invoc} é calculado conforme a Equação 4.1. Definiu-se que o temporizador Δ_{invoc} é baseado nas mensagens de resposta enviadas pelo provedor para os requisitantes. Esta escolha, deve-se ao fato do provedor conhecer quantas mensagens de resposta enviou, e qual a distância geográfica dele para cada requisitante. As coordenadas geográficas do requisitante são enviadas ao provedor na mensagem de descoberta. O tempo de Δ_{invoc} será ajustado com as coordenadas geográficas do requisitante mais distante em relação ao provedor. Os requisitantes permanecem fixos enquanto esperam pelo atendimento.

$$\Delta_{invoc} = K \times (d_{ij}/R_t) \tag{4.1}$$

Na Equação 4.1, o valor de *K* é calculado como, α + β. Conforme definido no protocolo LADP (KNIESS; LOQUES; ALBUQUERQUE, 2011), foi atribuído o valor de 100ms para o fator α, que especifica o atraso de ida e volta para 1 (um) salto na rede, e o valor de 0,02ms ao fator β para determinar o tempo máximo para o armazenamento das respostas recebidas pelo nó intermediário mais próximo ao requisitante. O valor de d_{ij} representa a distância entre o nó requisitante *i* e o nó provedor *j*. Considera-se o requisitante mais distante do provedor no cálculo do temporizador para que todos os

requisitantes tenham a chance de receber a mensagem de confirmação da invocação (*msgConfirmação*) do provedor. O alcance da antena é dado por R_t. Um provedor não envia mais mensagens de resposta após receber a primeira mensagem de invocação de um requisitante (*msgInvocação*).

O provedor, após enviar a primeira mensagem de confirmação (*msgConfirmação*), inicia o temporizador Δ_{ack_invoc}, que estabelece o tempo limite de espera pela mensagem de reconhecimento da confirmação da invocação (*msgRecConfirmação*) de um requisitante *i*. O temporizador Δ_{ack_invoc} é calculado conforme a Equação 4.2.

$$\Delta_{ack\_invoc} = \eta + (\alpha \times (d_{ij}/R_t)) \tag{4.2}$$

Na Equação 4.2, Δ_{ack_invoc} representa o tempo de espera no nó provedor por mensagem(s) de reconhecimento da confirmação da invocação (*msgRecConfirmação*). Valor do fator η corresponde ao tempo de ida e volta (RTT) do envio da mensagem de confirmação (*msgConfirmação*) e o recebimento da mensagem de reconhecimento da confirmação da invocação (*msgRecConfirmação*) pelo provedor. O valor de α (100ms) especifica o atraso de ida e volta para 1 (um) salto na rede. Na equação, d_{ij} representa a distância do nó requisitante *i* mais distante do provedor *j*. O valor de d_{ij} é sempre ajustado pelo provedor, quando o mesmo receber uma mensagem de reconhecimento da confirmação *msgRecConfirmação* de um requisitante que esteja mais distante que o anteriormente usado para o cálculo. A distância entre o requisitante e provedor é calculada com base nas coordenadas geográficas enviadas pelo requisitante na mensagem de descoberta de serviços recebida pelo provedor. O valor de R_t define o alcance da antena.

Caso o provedor receba o reconhecimento da confirmação (*msgRecConfirmação*) de um requisitante após Δ_{ack_invoc} expirar, o provedor irá retirar este pedido do cálculo da sequência de atendimento antes de iniciar o processamento da fila. Dessa maneira, o requisitante terá que reiniciar o processo de descoberta de serviço. Após o recebimento de uma mensagem de reconhecimento da confirmação da invocação (*msgRecConfirmação*), o provedor envia uma mensagem de aviso informando ao(s) requisitante(s) sua inclusão no processamento da fila (*msgFila*). Na mensagem de processamento da fila (*msgFila*), o provedor envia ao requisitante uma estimativa do tempo de processamento da fila de atendimento. Esta estimativa (Δ_{proc_queue}) é baseada no número de mensagens de resposta (*msgResposta*) enviada pelo provedor aos requisitantes. A estimativa é utilizada pelo requisitante para definir o tempo de espera pelo envio da mensagem de atendimento (*msgAtendimento*) do provedor.

Logo, o tempo máximo pelo qual um provedor espera pelas mensagens de confirmação da invocação dos requisitantes antes de organizar a fila de atendimento

é dado pelo temporizador $\Delta_{schedule}$, definido na Equação 4.3.

$$\Delta_{schedule} = \Delta_{invoc} + \Delta_{ack\_invoc} \tag{4.3}$$

Quando o temporizador $\Delta_{schedule}$ expira, o provedor executa o processamento da fila de atendimento com base no algoritmo de escalonamento escolhido, $\Delta_{AGschedule}$ (Seção 4.1.3) ou $\Delta_{ASschedule}$ (Seção 4.1.2).

O mecanismo de escalonamento de recursos organiza a sequência de atendimento, visando maximizar o número de requisitantes dentro da janela de tempo definida em Δ_{tmax} de cada requisitante. Ao final do processamento da fila, o provedor envia uma mensagem de aviso de atendimento aos requisitantes, confirmando o atendimento (*msgAtendimento*), e inicia o deslocamento na direção do primeiro requisitante da fila. Ressalta-se que o processo é contínuo, e a estimativa do tempo de processamento da fila é com base na quantidade de mensagens de resposta (*msgResposta*) enviadas pelo provedor aos requisitantes. Dessa forma, a estimativa do tempo de processamento pode ser maior que o tempo real de processamento da fila, uma vez que o número de requisitantes na fila pode ser menor que o número de mensagens de respostas (*msgResposta*) enviadas pelo provedor.

O temporizador Δ_{wait} define o tempo que o requisitante aguardará a mensagem de confirmação da invocação (*msgConfirmação*). Caso o período definido em Δ_{wait} expirar, e a mensagem de confirmação da invocação (*msgConfirmação*) não tiver sido recebida, o requisitante assume que a mensagem não alcançou o provedor, e faz mais uma tentativa de envio da mensagem de invocação (*msgInvocação*) (ver Figura 4.2).

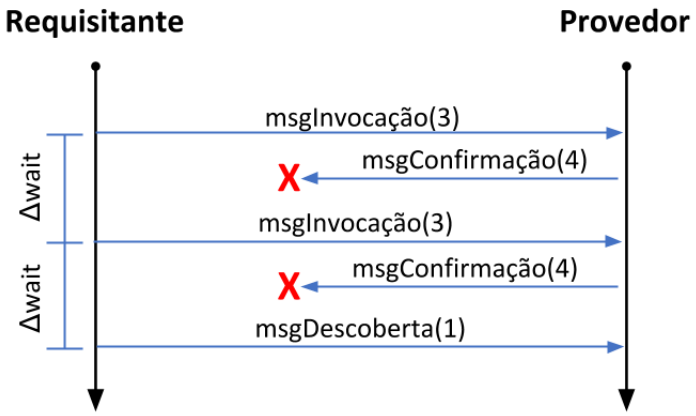


Figura 4.2 – Nova tentativa de envio da mensagem de invocação (*msgInvocação*). Fonte: próprio autor.

Se o requisitante não receber novamente a mensagem de confirmação da invocação (*msgConfirmação*), será reiniciado o processo de descoberta de serviço.

Δ_{wait} é definido na Equação 4.4.

$$\Delta_{wait_i} = \alpha \times (d_{ij}/R_t) \tag{4.4}$$

Na Equação 4.4, o valor de α (100ms) especifica o atraso de ida e volta para 1 (um) salto na rede. O valor de d_{ij} representa a distância entre o nó requisitante *i* e o provedor *j*. O alcance da antena é dado por R_t. O temporizador Δ_{wait} (Equação 4.4) também é utilizado para definir o tempo que o requisitante aguardará a mensagem de inclusão de processamento de fila (*msgFila*) após o envio da mensagem de reconhecimento da confirmação (*msgRecConfirmação*). Nesta etapa, se o temporizador Δ_{wait} terminar, e a mensagem de inclusão no processamento da fila (*msgFila*) não tiver sido recebida, o requisitante assume que a mensagem não alcançou o provedor, e faz mais uma tentativa de envio da mensagem de reconhecimento da confirmação (*msgRecConfirmação*). Se o requisitante não receber novamente a mensagem de fila (*msgFila*), será reiniciado o processo de descoberta de serviço (Figura 4.3).

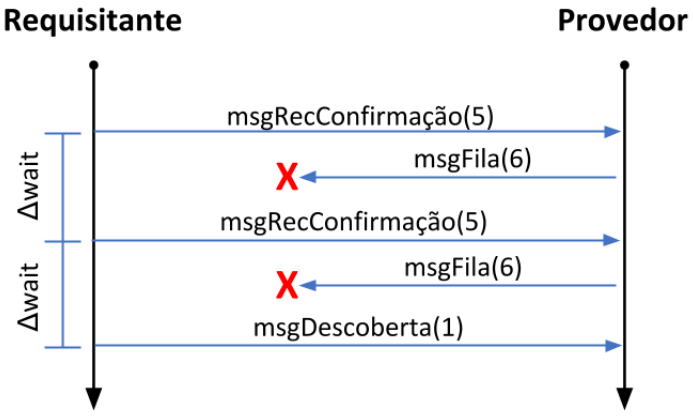


Figura 4.3 – Nova tentativa de envio da mensagem de reconhecimento da confirmação (*msgRecConfirmação*). Fonte: próprio autor.

Após o requisitante tomar ciência da inclusão do seu pedido na fila de atendimento, aguardará pelo recebimento da mensagem de atendimento (*msgAtendimento*) que será enviado pelo provedor. Esse tempo de espera é estabelecido pelo temporizador Δ_{attend}. Caso o temporizador Δ_{attend} expire, e a mensagem de atendimento (*msgAtendimento*) não tiver sido recebida, o requisitante assume que não será atendido pelo provedor e reinicia o processo de descoberta (Figura 4.4).

Caso o requisitante não receba a mensagem de atendimento (*msgAtendimento*) dentro do tempo definido por Δ_{proc_queue}, ainda durante Δ_{attend}, o requisitante fará mais uma tentativa de envio de mensagem de reconhecimento da confirmação

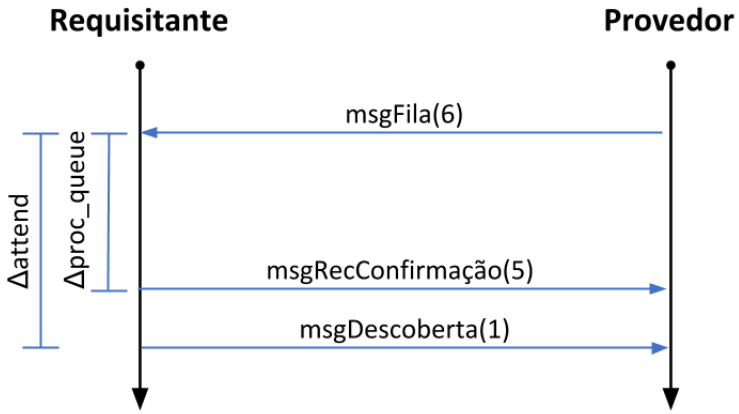


Figura 4.4 – Nova tentativa de envio da mensagem de reconhecimento da confirmação (*msgRecConfirmação*). Fonte: próprio autor.

(*msgRecConfirmação*). O temporizador Δ_{attend} é definido na Equação 4.5.

$$\Delta_{attend_i} = \Delta_{proc\_queue} + \Delta_{wait_i} \tag{4.5}$$

Na Equação 4.5, considera-se que o valor de Δ_{proc_queue} é uma estimativa do tempo de processamento da fila enviada previamente pelo provedor para o requisitante na mensagem de fila (*msgFila*). O requisitante fará mais uma tentativa após Δ_{proc_queue} expirar. Por exemplo, suponha que a mensagem de atendimento tenha sido enviada pelo provedor, porém foi perdida na rede. Neste caso, se o requisitante esperar pela mensagem de atendimento até que o temporizador Δ_{attend} termine sem ter realizado a segunda tentativa em Δ_{proc_queue}, poderá reiniciar o processo de descoberta desnecessariamente.

4.1.1 Mecanismo de Escalonamento, Definição da Fila de Atendimento

Na abordagem para o escalonamento de recursos proposta neste trabalho, a entidade responsável pelo agendamento é o nó provedor. Após o provedor ser selecionado, conforme estabelece o mecanismo de seleção de serviço do protocolo LADP (KNIESS; LOQUES; ALBUQUERQUE, 2011), o requisitante envia uma mensagem de invocação *(msgInvocação)* ao provedor. O temporizador $\Delta_{schedule}$ (Equação 4.3) define o tempo de espera pelo provedor para o processamento da fila de atendimento.

Quando o tempo definido em $\Delta_{schedule}$ expirar, um algoritmo irá gerar uma matriz de adjacência (um exemplo pode ser visualizado na Tabela 4.1) contendo informações relacionadas ao tempo de deslocamento do provedor para cada um dos requisitantes e entre seus pares adjacentes. Outra informação gerada consiste em uma variável do tipo vetor contendo o tempo estipulado pelo requisitante (valor de Δ_{tmax} enviado na mensagem de descoberta) para o atendimento, denominado janelamento.

Tabela 4.1 – Matriz de adjacência (tempo em segundos). Fonte: próprio autor.

		Para							
		P	R7	R2	R3	R1	R6	R4	R5
De	P	0	62	257	268	197	154	279	307
	R7		0	63	39	201	230	114	248
	R2			0	41	138	167	139	196
	R3				0	176	199	100	210
	R1					0	47	254	142
	R6						0	264	105
	R4							0	233
	R5								0

Os valores na matriz de tempo de deslocamento (Tabela 4.1) correspondem à distância euclidiana entre dois pontos dividido pela velocidade do provedor, sendo cada ponto uma representação dos nós da rede (provedores e requisitantes) e seus nós adjacentes. Utilizou-se um sistema de coordenadas cartesianas, considerando que a posição de um nó no plano é especificada por meio do conjunto de coordenadas X e Y. O cálculo da distância entre dois pontos é apresentado pela Equação 4.6 (DANIELSSON, 1980).

$$d = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}, \tag{4.6}$$

Analisando a Figura 4.5, suponha que os requisitantes *R1*, *R2*, *R3*, *R4*, *R5*, *R6* e *R7* enviem uma mensagem de invocação de serviços (*msgInvocação*) para um provedor *P* solicitando um atendimento, e o tempo máximo para o atendimento (Δ_{tmax}) for de 6min. Quando o provedor *P* receber a primeira mensagem de invocação (p.ex. requisitante *R7*), o provedor inicia o temporizador $\Delta_{schedule}$. Como resposta, o nó provedor enviará aos requisitantes uma mensagem de confirmação da invocação (*msgConfirmação*) em *unicast* e aguardará a mensagem de reconhecimento da confirmação (*msgRecConfirmação*). Após receber a mensagem de reconhecimento da confirmação de um requisitante, o provedor envia a mensagem de aviso de inclusão na fila (*msgFila*). As mensagens de invocação recebidas durante o tempo definido pelo temporizador $\Delta_{schedule}$ são enfileirados no provedor *P*.

Na Figura 4.5, o tempo de deslocamento do provedor *P* para cada requisitante está representado em segundos. Por exemplo, o tempo que o provedor *P* levará para se deslocar da sua posição até o requisitante *R7* é de 62 segundos. Já para o requisitante *R5*, levará o maior tempo de deslocamento do provedor, que equivale a 307 segundos.

Cada nó é considerado um possível caminho, porém, foram consideradas al-

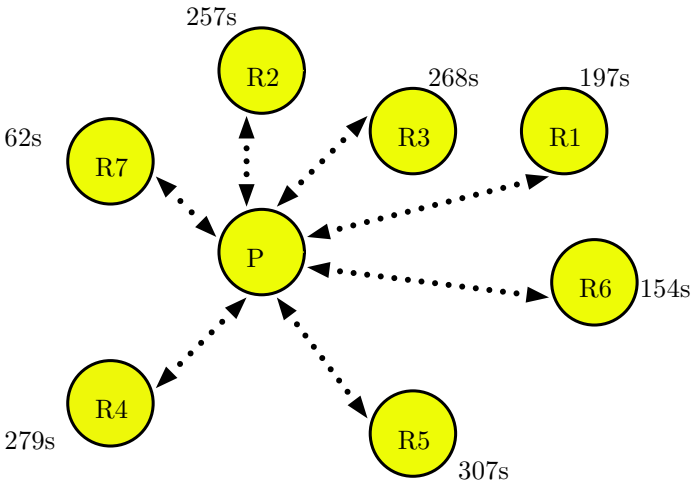


Figura 4.5 – Geração da fila no provedor. Fonte: próprio autor.

gumas restrições como, por exemplo, a viabilidade do provedor atender a um requisitante dentro do tempo limite Δ_{tmax}, ou se o provedor continua para o próximo atendimento. Considera-se uma solução válida, o resultado que atenda a todos os requisitantes em Δ_{tmax}. Para os casos que não resulte em uma solução que atenda a todos os requisitantes, escolhe-se aquele que oferecer o melhor resultado, ou seja, a maior quantidade de requisitantes para atendimento.

A Figura 4.6 representa uma possível solução de acordo com a matriz de tempo de deslocamento (Tabela 4.1). Na Figura 4.6, demonstra-se 7 (sete) requisitantes, todos com Δ_{tmax} igual 360s e um provedor *P*. Levando em consideração o tempo de deslocamento do provedor até os requisitantes e o tempo de atendimento informado em Δ_{tmax}, o algoritmo de escalonamento encontrou uma solução, definindo um trajeto que atenda a 4 (quatro) requisitantes. O provedor *P* levou 338s para o deslocamento entre os 4 (quatro) requisitantes definidos pela solução.

Em outras palavras, foi possível atender quatro requisitantes dentro do tempo solicitado sem infringir o limite Δ_{tmax} dos quatro requisitantes. Quando a ordem de atendimento (*R7*, *R2*, *R3*, *R1*) for definida, o provedor envia uma mensagem de aviso de atendimento (*msgAtendimento*) aos requisitantes (*R7*, *R2*, *R3*, *R1*) e inicia o deslocamento na direção do primeiro requisitante da fila (*R7*). Os requisitantes que não puderam ser atendidos reiniciarão o processo de descoberta, a fim de localizar um novo provedor apto para o atendimento.

No protocolo LADP (KNIESS; LOQUES; ALBUQUERQUE, 2011) possíveis casos de falhas durante a execução do protocolo em uma MANET são tratados. Com a adição da abordagem para o escalonamento de recursos e por ser um processo

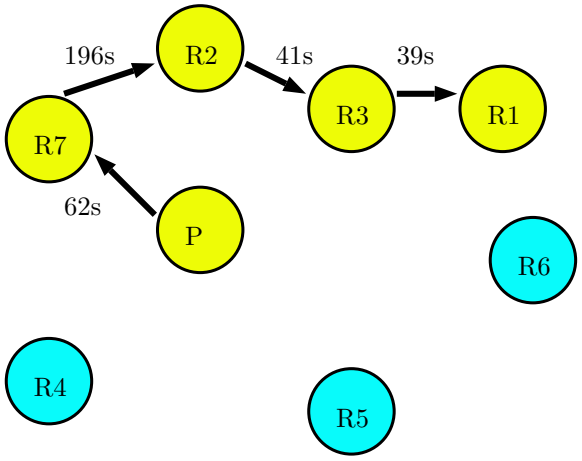


Figura 4.6 – Exemplo de uma solução. Fonte: próprio autor.

contínuo, falhas não previstas na versão original do protocolo podem ocorrer.

Para exemplificar, suponha-se que um provedor receba uma mensagem de invocação (*msgInvocação*) de um novo requisitante quando já estiver em movimento e na direção do primeiro requisitante a ser atendido. No decorrer do primeiro atendimento, o dispositivo móvel no provedor analisará as novas mensagens de invocação (*msgInvocação*) recebidas. Caso o nó provedor verifique a viabilidade de incluir algum novo atendimento no percurso, sem prejuízo no janelamento previamente definido, o provedor enviará uma mensagem de alerta ao(s) novo(s) requisitante(s) e aos já selecionado(s) previamente, anunciando o novo tempo para o atendimento.

A Tabela 4.2 sintetiza a notação apresentada neste trabalho referente ao protocolo LADP (KNIESS; LOQUES; ALBUQUERQUE, 2011). A Tabela 4.3 ilustra a notação utilizada na abordagem para escalonamento de recursos e que estende o protocolo LADP.

No Algoritmo 1, descreve-se o papel do requisitante no escalonamento de recursos. No Algoritmo 2, as estratégias para o escalonamento de recursos realizado pelo provedor selecionado pelo(s) requisitante(s) são apresentadas.

Tabela 4.2 – Notação do LADP (KNIESS; LOQUES; ALBUQUERQUE, 2011).

Símbolo	Definição
Δ_{tmax}	tempo máximo de atendimento para uma requisição
$coord_x$, $coord_y$	coordenadas geográficas dos nós
d_{ij}	distância de um nó inicial *i* a um nó final *j*
$id_{Request}$	identificador da requisição
$id_{Requisitante}$	identificador do requisitante
$id_{Provider}$	identificador do provedor
id_{ACK}	identificador do recebimento das mensagens
$seq_{Request}$	identificador do número de sequência da requisição
$max_{provider}$	número máximo de provedores esperados pelo nó requisitante
s	tipo de recurso
R_i	raio de alcance da busca de um nó *i*
R_t	raio de alcance da antena
K	α + β
α	atraso de ida e volta para 1 (um) salto na rede
β	representa o tempo máximo de armazenamento das respostas recebidas no nó intermediário mais próximo ao requisitante

Tabela 4.3 – Notação do algoritmo de escalonamento de recursos.

Símbolo	Definição
Δ_{ack_invoc}	temporizador para o recebimento da mensagem de reconhecimento de confirmação da invocação
Δ_{attend}	temporizador para o recebimento da mensagem de atendimento
Δ_{invoc}	temporizador para o recebimento da mensagem de invocação do requisitante
Δ_{wait}	temporizador para o recebimento da mensagem de confirmação e mensagem de fila do provedor
Δ_{proc_queue}	estimativa de tempo para processamento da fila de atendimento
$\Delta_{schedule}$	tempo total para tratamento dos pedidos dos requisitantes (Δ_{invoc} + Δ_{ack_invoc})
Q	fila de atendimento gerada pelo algoritmo de escalonamento
Q'	fila de agendamento gerada pelo algoritmo de escalonamento enquanto o provedor se movimenta
$AG_{schedule}$	Algoritmo Genético para escalonamento de recursos
fitness	valor aptidão de cada indivíduo
pn	número de penalidades
m	valor da penalidade
d_{ij}	custo associado ao tempo de deslocamento entre *i* e *j*
i	nó de origem
j	nó de destino
$AS_{schedule}$	Algoritmo A*(A-Estrela) para escalonamento de recursos
f(n)	função objetivo do $AS_{schedule}$
g(n)	custo atual entre um nó *i* e um nó *j*
h(n)	estimativa do custo restante para os nós adjacentes ao nó *j*
t_i	tempo máximo para deslocamento do provedor ao nó *i*
P	nó provedor
R	nó requisitante
η	RTT: envio da mensagem de confirmação e recebimento da mensagem de reconhecimento da confirmação da invocação

Algoritmo 1: Algoritmo para Escalonamento de Recursos-Requisitante(*i*)

```
1  início
2      função ProcEnvMsgInvocação(id_Request, id_Requisitante, id_Provider, coord_x, coord_y, Δ_tmax, Seq_Request,
         max_provider, s)
3      nó i envia msgInvocação para o provedor j;
4      nó i aciona Δ_wait;
5      nr_tentativas ← 0 ;
6      se temporizador Δ_invoc_i terminar então
7          se nó i não recebeu msgConfirmação do provedor j então
8              se nr_tentativas = 0 então
9                  nó i reenvia msgInvocação ;
10                 nr_tentativas ← nr_tentativas + 1;
11                 nó i inicia um novo temporizador Δ_wait;
12     função ProcRecvMsgConfirmação
13     nó i recebe msgConfirmação do provedor j;
14     nó i envia msgRecConfirmação para o provedor j;
15     nó i aciona Δ_wait;
16     nr_tentativas ← 0 ;
17     se temporizador Δ_wait terminar então
18         se nó i não recebeu msgFila do provedor j então
19             se nr_tentativas = 0 então
20                 nó i reenvia msgRecConfirmação ;
21                 nr_tentativas ← nr_tentativas + 1;
22                 nó i inicializa um novo temporizador Δ_wait;
23     função ProcRecvMsgFila(Δ_proc_queue)
24     nó i recebe msgFila do provedor j;
25     nó i aciona Δ_attend;
26     nr_tentativas ← 0 ;
27     se temporizador Δ_proc_queue terminar então
28         se nó i não recebeu msgAtendimento do provedor j então
29             se nr_tentativas = 0 então
30                 nó i reenvia msgRecConfirmação ;
31                 nr_tentativas ← nr_tentativas + 1;
32                 nó i inicializa um novo temporizador Δ_attend;
33     se temporizador Δ_attend terminar então
34         se nó i não recebeu msgAtendimento do provedor j então
35             nó i reinicia o processo de descoberta ;
```

Algoritmo 2: Algoritmo para Escalonamento de Recursos-Provedor(*j*)

```
1  início
2      função ProcRecvMsgInvocacao
3      nó j recebe msgInvocação do nó i;
4      se nó i não constar em Q então
5          se primeira msgInvocação então
6              nó j aciona Δ_invoc;
7          nó j adiciona nó i na fila Q;
8          nó j envia msgConfirmação para nó i;
9      função ProcRecvMsgRecConfirmação
10     nó j recebe msgRecConfirmação do nó i;
11     nó j envia msgFila para nó i;
12     Δ_ack_invoc ← η × ⟨ α × ⟨ d_ij / R_t ⟩ ⟩ ;
13     se primeira msgRecConfirmação então
14         nó j aciona Δ_ack_invoc;
15         Δ_schedule ← Δ_invoc + Δ_ack_invoc;
16     se Δ_schedule terminar então
17         se Q > 0 então
18             nó j processa algoritmo de escalonamento ;
19             nó j envia msgAtendimento para nó i;
20             nó j inicia deslocamento para atendimento ao primeiro nó i;
```

4.1.2 Escalonamento com Algoritmo A-Estrela (A*)

Para fins de comparação com o algoritmo genético que faz parte dos sistemas computacionais inspirados em modelos de processos naturais (mais especificamente da computação evolutiva), modelou-se o problema do escalonamento de recursos com base no algoritmo A-Estrela (A*) (HART; NILSSON; RAPHAEL, 1968). O fluxograma do mecanismo de escalonamento desenvolvido para encontrar uma solução por meio do algoritmo A-Estrela é ilustrado na Figura 4.7.

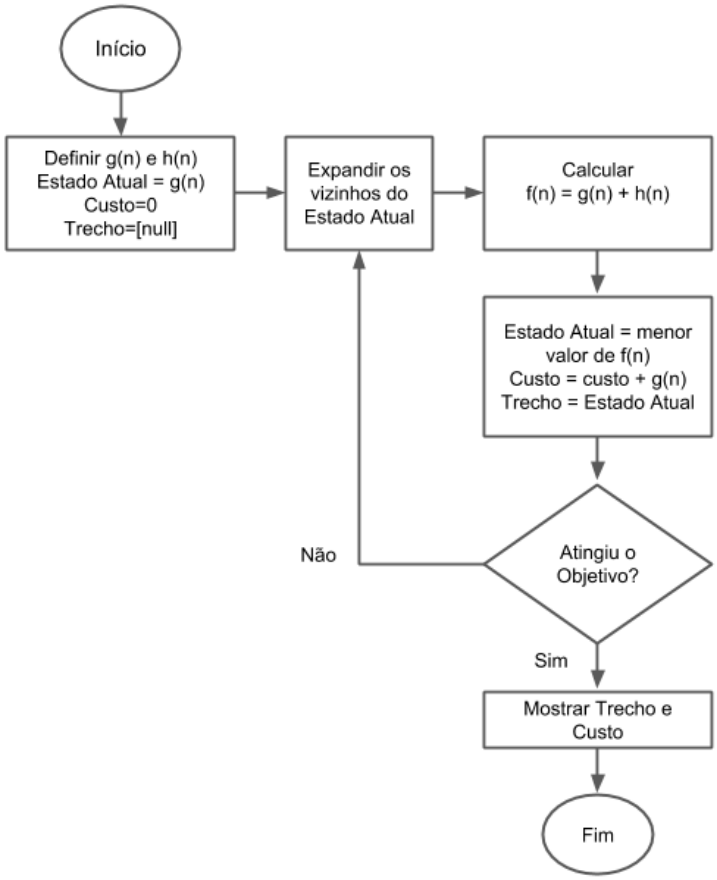


Figura 4.7 – Fluxograma de execução do algoritmo A-Estrela. Fonte: próprio autor.

Conforme apresentado no Capítulo 2 Seção 2.3.1 a Equação 2.1 representa o cálculo realizado pelo algoritmo A-Estrela para avaliar os nós. Para o desenvolvimento do algoritmo de escalonamento de recursos com base no A-Estrela, adicionou-se a restrição definida na Equação 4.7.

$$\sum_{i,j \in N}^{n} d_{ij} \leq \Delta_{tmax_j} \qquad \forall i, j \in N \tag{4.7}$$

O algoritmo A-Estrela (denominado aqui, $AS_{schedule}$) encontra o caminho com base no mínimo *spanning tree*. Somente o nó mais promissor, aquele com o menor *f(n)*

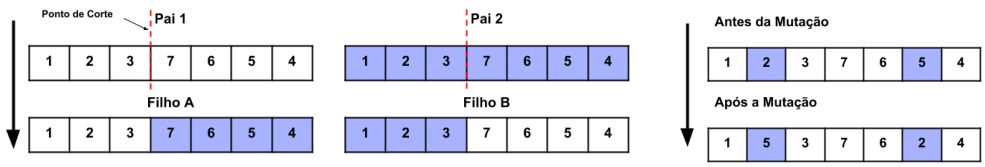
será expandido, desde que respeitando a janela de tempo (Δ_{tmax}) dos requisitantes no momento da expansão. Na Equação 4.7 ajusta-se uma janela de tempo com o horário para o atendimento para cada ponto do percurso, *N* = (*i*, *j*). O algoritmo $AS_{schedule}$, a partir da lista de adjacências de tempo de deslocamento (ver Tabela 4.1), gera uma fila de atendimento, tendo como prioridade o tempo de deslocamento do provedor e o tempo máximo para o atendimento.

Em relação ao A-Estrela na sua versão original, no $AS_{schedule}$ foi necessário adicionar uma janela de tempo com o horário para o atendimento Δ_{tmax} em cada ponto do percurso (Equação 4.7). Dessa forma, um provedor não pode chegar ao local da ocorrência após Δ_{tmax} de um nó *j*. Outra alteração realizada foi testar a viabilidade de atendimento entre todos os nós requisitantes.

4.1.3 Escalonamento com Algoritmo Genético (AG)

O Algoritmo Genético (denominado neste ponto de $AG_{schedule}$), a partir da matriz de adjacência de tempo de deslocamento (ver Tabela 4.1), gera uma fila de atendimento, tendo como prioridade o tempo de deslocamento do provedor, e o tempo máximo para o atendimento.

Primeiramente, define-se a quantidade de requisitantes que solicitaram atendimento conforme a ordem de chegada das mensagens ao provedor (ver Figura 4.5) formando o indivíduo (*R1*, *R2*, *R3*, *R4*, *R5*, *R6* e *R7*). Em seguida, cria-se uma população inicial, de forma aleatória. Para cada indivíduo dessa população, é avaliada a sua aptidão por meio da função objetivo expressa pela Equação 4.8. Os melhores indivíduos são preservados e inseridos em uma nova população. Os outros indivíduos da população passam pelo processo de seleção e, posteriormente, pelo cruzamento (Figura 4.8(a)), para gerar novos indivíduos, e, por fim, alguns destes indivíduos sofrem mutação (Figura 4.8(b)). Após a aplicação dos operadores genéticos, os novos indivíduos são inseridos na nova população. Com isso, todo o processo é repetido desde a etapa de *Avaliação dos Candidatos*. Tal fluxo persiste até que seja satisfeito o critério de parada por número máximo de gerações. Por fim, extrai-se o gene para a representação da melhor solução.



(a) Cruzamento (b) Mutação

Figura 4.8 – Operadores genéticos: cruzamento e mutação. Fonte: próprio autor.

A codificação do indivíduo é dada pela permutação de um conjunto finito de seus elementos (do tipo inteiro), neste caso, os requisitantes. O cromossomo que representa a rota de atendimento do provedor é composto de uma ordenação entre os requisitantes (conforme representado na Figura 4.5 o cromossomo é composto por uma identificação (tipo inteiro) do requisitante: *7, 2, 3, 1, 6, 4, 5*). A codificação define um caminho completo da origem ao destino, em que será mensurado o tempo de deslocamento para o atendimento. Uma vez que as soluções iniciais são geradas aleatoriamente, é necessário calcular a aptidão de cada indivíduo, representando seu *fitness*. O custo é associado ao caminho, de modo que, quanto menor o custo, melhor a solução encontrada.

A formalização matemática para o problema de escalonamento de recursos com o uso de algoritmos genéticos, requer a definição de um conjunto de parâmetros fundamentais: *N = {1,2,...,n}* que representam na forma de um grafo o conjunto com *n* nós distribuídos na área pós-desastre. O custo associado ao tempo de deslocamento entre os nós $i, j \subset N$ é dado por d_{ij} e, por fim, x_{ij} representa a ligação entre o nó *i* e o nó *j* (designadas como variáveis de decisão).

O modelo matemático do cálculo exato da função objetivo é apresentado na Equação 4.8, que realiza o ajuste do tempo de deslocamento para que o provedor de serviço percorra os distintos pontos uma única vez. A variável *pn* consiste no número de penalidades que ocorreram para cada possível solução na população, na qual é verificada sua factibilidade, ou seja, se é válida. A variável *m* representa um valor de penalidade. As Equações de 4.9 a 4.12 são as restrições da função objetivo.

$$min \sum_{i,j \in E}^{n} d_{ij} x_{ij} + pn.m \tag{4.8}$$

Sujeito às restrições:

$$\sum_{j,i=1}^{n} x_{ij} = n - 1 \qquad \forall i, j \in N \tag{4.9}$$

$$\sum_{i,j \in N}^{n} x_{ij} = 0 \qquad \forall i, j \in N \tag{4.10}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in N, i \neq j \tag{4.11}$$

$$\sum_{i,j \in N}^{n} d_{ij} \leq \Delta_{tmax_j} \qquad \forall i, j \in N \tag{4.12}$$

A Equação 4.9 estabelece as restrições para que o provedor passe uma única vez em cada nó requisitante, devendo partir para o próximo ponto até que todo o percurso seja realizado. A Equação 4.10 determina que não é possível ir de um nó

para ele mesmo. A Equação 4.11 representa confirmação de deslocamento x_{ij}, tal que $x_{ij} = 1$, se o provedor irá se deslocar do nó *i* para o nó *j*, ou $x_{ij} = 0$, caso contrário.

Na Equação 4.12, ajusta-se uma janela de tempo com o horário para o atendimento para cada ponto do percurso, $N = (i, j)$. Com isso, para cada nó *j* define-se uma janela de tempo Δ_{tmax}. Dessa maneira, um provedor não pode chegar ao local da ocorrência após Δ_{tmax} de um nó *j*. O principal objetivo a ser alcançado é a redução do custo de deslocamento, visando aumentar o número de requisitantes atendidos.

Na função objetivo apresentada na Equação 4.8 se define os pesos e as penalizações de acordo com os requisitos para o atendimento. Caso não exista uma solução válida que compõe o caminho, o mesmo é julgado inválido. Como resultado, este caminho receberá uma nota alta como penalização, diminuindo suas chances de ser selecionado na próxima geração. Na modelagem do problema adotou-se a seguinte premissa: quanto maior a nota, pior o caminho.

O fluxograma do escalonamento para encontrar uma solução por meio de algoritmo Genético é ilustrado na Figura 4.9.

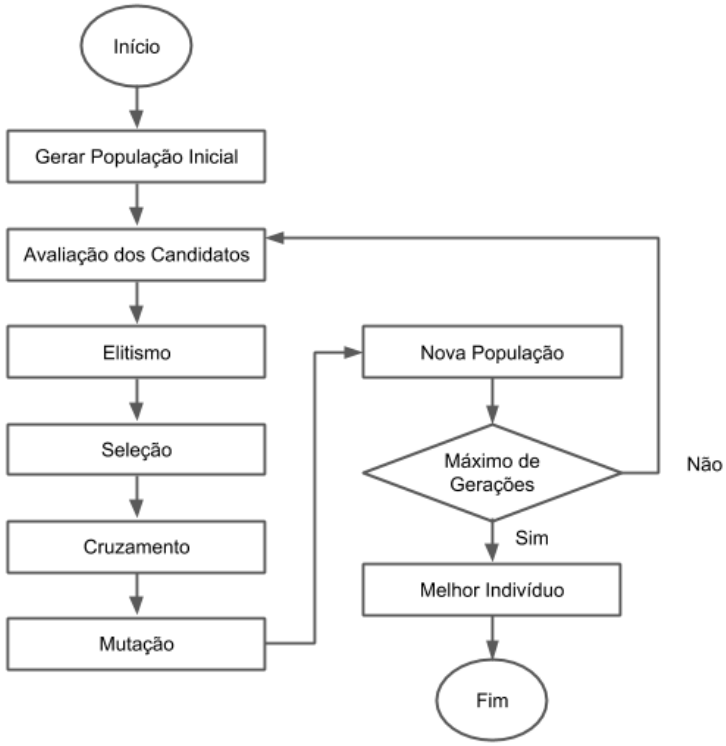


Figura 4.9 – Fluxograma de execução do algoritmo genético. Fonte: próprio autor.

Com o algoritmo $AG_{schedule}$ é possível obter o tempo de deslocamento de um provedor até os requisitantes, respeitando o tempo de atendimento Δ_{tmax}. Dessa forma, o $AG_{schedule}$ deverá evoluir suas soluções candidatas até chegar no seu critério de parada, que consiste no número de gerações.

Ao final do tempo estipulado pelo temporizador $\Delta_{schedule}$, o algoritmo $AG_{schedule}$ processa a fila de atendimento para que o provedor possa iniciar o deslocamento na direção do primeiro requisitante.

4.2 CONCLUSÕES DO CAPÍTULO

Neste capítulo, apresentou-se uma abordagem para escalonamento de recursos, que estende o protocolo de descoberta de serviços LADP proposto por Kniess et al. (2011), para cenários pós-desastres. Os mecanismos para escalonamento de recursos com base nos algoritmos Genético ($AG_{schedule}$) e A-Estrela ($AS_{schedule}$) foram apresentados. Ambos os algoritmos foram desenvolvidos a fim de facilitar a tratatividade do escalonamento de provedores de recursos em cenários de pós-desastres.

Para o gerenciamento da comunicação (troca de mensagens) entre provedores e requisitantes, foram desenvolvidos temporizadores. Por exemplo, o provedor, após receber a primeira mensagem de invocação de um requisitante, aciona o temporizador Δ_{invoc}. Ao receber uma mensagem de reconhecimento de confirmação, o provedor inicia o temporizador Δ_{ack_invoc}, que estabelece o tempo limite de espera pelas mensagens de reconhecimento da confirmação da invocação. $\Delta_{schedule}$ é a soma dos temporizadores Δ_{invoc} e Δ_{ack_invoc}, definindo, assim, o tempo de espera do provedor para recebimento das mensagens de invocação e confirmação enviadas pelos requisitantes, que antecede ao processamento da fila de atendimento.

No requisitante, implementou-se os temporizadores: Δ_{wait}, que define o tempo que o requisitante aguardará a mensagem de confirmação da invocação enviada pelo provedor e posteriormente o tempo de espera referente a confirmação da inclusão do requisitante na fila de atendimento e; Δ_{attend}, tempo em que o requisitante aguardará pelo recebimento da mensagem de atendimento que será enviado pelo provedor.

No capítulo seguinte, são apresentados os experimentos realizados e a análise dos resultados obtidos com a abordagem para escalonamento de recursos desenvolvida.

5 EXPERIMENTOS E RESULTADOS

Neste capítulo, descreve-se a análise de desempenho realizada, bem como os resultados obtidos com a abordagem para escalonamento de recursos. Em suma, apresentam-se: (i) descrição do ambiente de configuração e as métricas utilizadas para a realização dos experimentos computacionais; (ii) análise de desempenho dos mecanismos de escalonamento, e; (iii) análise dos resultados obtidos nas simulações.

A implementação da abordagem para escalonamento foi realizada na linguagem C++. Para a modelagem dos cenários, semelhante a uma MANET 802.11 de saltos múltiplos (RILEY; HENDERSON, 2010), utilizou-se o simulador de redes *Network Simulator* (NS3) (NS-3, 2008) na versão 3.29. Nos cenários avaliados, a taxa de transmissão é de 2 Mbps, e o raio de alcance do sinal da antena é de 250 metros. No simulador NS3 utilizou-se uma distribuição uniforme do nós na área monitorada.

Com o propósito de simular a movimentação de humanos, utilizou-se o modelo de mobilidade de *Gauss-Markov* (CAMP; BOLENG; DAVIES, 2002). A velocidade de deslocamento dos nós varia entre 1,5 m/s e 6,0 m/s e os requisitantes são fixos. O modelo de *Gauss-Markov* abstrai as mudanças bruscas de direção e paradas repentinas dos modelos randômicos.

Nas simulações, usou-se o protocolo de roteamento *Optimized Link State Routing Protocol* (OLSR) (CLAUSEN; JAQCQUET, 2003), que é um protocolo de roteamento pró-ativo para MANETs de múltiplos saltos. No protocolo OLSR, os nós usam informações locais para rotear os pacotes, buscando reduzir o número de mensagens de controle para realizar o roteamento.

5.1 MÉTRICAS DE AVALIAÇÃO

O escalonamento de recursos foi desenvolvido com a meta de atender ao maior número de requisitantes dentro do menor período de tempo, levando em conta informações geográficas dos requisitantes e provedores, tempo de deslocamento do provedor até ao requisitante e o tempo máximo para o atendimento das requisições.

Alguns fatores, tais como, tamanho da área monitorada, quantidade de nós na rede, velocidade de deslocamento dos nós e número de requisitantes, podem afetar o desempenho do algoritmo. Considerando-se tais fatores, foram escolhidas as seguintes métricas para avaliação: consumo de energia, perda de pacotes, sobrecarga, tempo de processamento da fila e número de requisitantes atendidos:

- **Consumo de Energia (CE):** y = consumo_médio_energia - energia_inicial -> CE(%) = (energia_inicial x CE) / (y x 100) (KNIESS; LOQUES; ALBUQUERQUE, 2011).
- **Perda de Pacotes (PP):** perdas resultantes do número total de mensagens geradas pelo mecanismo de escalonamento / número total de mensagens transmitidas pelo mecanismo.
- **Sobrecarga (SB):** número de mensagens transmitidas pelo mecanismo de escalonamento / número total de mensagens transmitidas na rede.
- **Tempo de Processamento (TP):** tempo de processamento do algoritmo para a geração da fila de atendimento.
- **Requisitantes Atendidos (RA):** número de requisitantes atendidos pelo(s) provedor(es).

De acordo com (NASCIMENTO et al., 2017), a verificação da normalidade dos dados é uma das suposições utilizada para determinar que tipo de teste estatístico será empregado, e muitos dos procedimentos estatísticos são testes paramétricos, os quais requerem que os dados sejam retirados de uma população normalmente distribuída. O teste de *Shapiro-Wilk* (SHAPIRO; WILK, 1965), foi incorporado nesta dissertação, para uma analise estatística dos testes de normalidade realizados, a fim de, identificar se um conjunto de dados é normalmente distribuído. Os principais testes estatísticos empregados na análise dos dados basei-se em um modelo teórico que pressupõem a distribuição normal, como por exemplo o Teste t de *Student* (RAJU, 2005). Para os resultados não-normal, será aplicado o teste de *Wilcoxon* (SLUD; WEI, 1982), considerado um teste de hipóteses não paramétrico utilizado quando se deseja comparar duas amostras relacionadas, amostras emparelhadas ou medidas repetidas em uma única amostra para avaliar se os postos médios populacionais diferem. O teste de *Wilcoxon* é uma alternativa ao Teste t de *Student*, quando não se pode assumir que a população é normalmente distribuída.

Para obter-se os resultados estatísticos deste trabalho, o primeira etapa é analisar se os dados a serem comparados possui uma distribuição normal por meio do teste de *Shapiro-Wilk*. Caso os resultados comparados são normais, usa-se o Teste t de *Student*. Porém, caso um dos resultados for não-normal, será aplicado o teste de *Wilcoxon*. O valor-p é uma quantificação da probabilidade de se errar ao rejeitar H_0 e a mesma decorre da distribuição estatística adotada. Se o valor-p é menor que o nível de significância, conclui-se que o correto é rejeitar a hipótese de nulidade. Para todos os testes utilizou-se o valor-p = 0,05, de acordo com a seguinte parametrização: se p > 0,05 os dados não diferem de uma distribuição normal, ou seja, são normais; e, se p

< 0,05 os dados diferem de uma distribuição normal, neste caso, não são normais. A avaliação da normalidade da distribuição dos dados é importante para corroborar as conclusões sobre os resultados obtidos em cada cenário.

O ambiente de simulação, os experimentos computacionais e os resultados alcançados são apresentados nas seções seguintes.

5.2 CENÁRIOS DE SIMULAÇÃO E RESULTADOS

Nos experimentos, o tamanho da área monitorada é de 4km^2 (2000m x 2000m). O número de requisitantes varia em 100, 200, 300 e 400, e um número máximo de 450 nós (entre requisitantes e intermediários) foram distribuídos na rede. Utilizou-se um número de 1 a 2 provedores.

O tamanho do pacote da mensagem de invocação é de 32 bytes e contém as seguintes informações: localização geográfica do requisitante ($coord_x$, $coord_y$), tempo para o atendimento do pedido Δ_{tmax}, identificação da requisição $id_{Request}$, número de sequência $seq_{Request}$, identificador do requisitante $id_{Request}$, identificador do provedor $id_{Provider}$ e identificador do recebimento das mensagens id_{ACK}. Seguindo os padrões de consumo de rádio 802.11, o módulo de energia do NS-3 foi configurado com os seguintes valores: a energia inicial dos dispositivos é de 1000J, com os valores de potência de transmissão 55mW, potência de recepção 42mW e, no modo ocioso, 24mW. O tempo total de simulação é de 7200s.

O tempo máximo para o atendimento da requisição Δ_{tmax} foi estipulado em 15 minutos (900s). O intervalo de confiança apresentado nos resultados é de 95%. Cada experimento foi executado 30 vezes. A Tabela 5.1 sumariza os parâmetros (empíricos) do sistema usados nos experimentos.

Tabela 5.1 – Parâmetros do sistema.

Parâmetro	**Valor**
Terreno	4 km^2
Número máximo de nós	450
Número de nós provedores	1 e 2
Número de nós requisitantes	100, 200, 300, 400
Tempo de simulação	7200s
Alcance da antena	250m
Energia inicial dos nós	1000J
Δ_{tmax}	900s (15min)
Velocidade dos nós	5,4 km/h ->1,5 m/s 21,6 km/h ->6 m/s

Alguns parâmetros são modificados durante os experimentos. Estes são: (i) a matriz de tempo de deslocamento, que contém o tempo gasto pelo provedor para

chegar ao local de cada requisitante, a partir do provedor e seus pares adjacentes e; (ii) o tempo de atendimento (janela), que representa o tempo máximo para o atendimento (Δ_{tmax}). Este último, atua no problema como um limitador do espaço de busca das soluções, assegurando que o roteiro válido não ultrapasse o tempo fornecido pelo requisitante.

Para fins de comparações, nos cenários simulados, utilizou-se os algoritmos para escalonamento de recursos modelado por meio de um algoritmo Genético (Seção 4.1.3), e do algoritmo A-Estrela (Seção 4.1.2). Os mecanismos para escalonamento com base no Genético e A* também foram comparados com o protocolo LADP em sua versão original.

No algoritmo $AG_{schedule}$, o valor da população é 100, o número de gerações é 1000, e aplicou-se a técnica de elitismo. O valor da penalidade *m* é igual a 100, o método para mutação da ordem dos genes é o *SwapMutator* com valor de 1%, e o *Partially Mapped Crossover PMX* é de 90%. Para o algoritmo A-Estrela manteve-se as configurações conforme descritas na Seção 4.1.2. Os valores definidos no algoritmo $AG_{schedule}$ baseado na experiência e na observação dos testes.

Nas fases de seleção e invocação de serviços do protocolo LADP, incluindo-se o escalonamento, as mensagens são transmitidas em *unicast*, considerando-se que o endereço IP dos requisitantes e provedores é conhecido.

5.2.1 Análise dos Algoritmos de Escalonamento com Um Provedor

Os algoritmos para escalonamento de recursos foram avaliados em face das seguintes métricas: consumo de energia, sobrecarga, perda de pacotes, tempo de processamento e número de requisitantes atendidos. O tempo máximo para o atendimento Δ_{tmax} = 15min, e 1 (um) provedor está na área afetada pelo desastre. O provedor desloca-se com a velocidade de 6,0m/s (veículo com velocidade moderada devido as características do cenário) e os intermediários a 1,5m/s.

Visando avaliar o potencial dos algoritmos $AG_{schedule}$ e $AS_{schedule}$, em relação às métricas Tempo de Processamento (TP) e Requisitantes Atendidos (RA), ajustou-se o valor de $\Delta_{schedule}$ para 20 segundos a fim de aumentar a quantidade de mensagens de invocação recebidas pelo provedor. Através de simulações, constatou-se que dado este tempo, obtém-se um número alto de requisitantes na fila. Os experimentos que avaliam o uso dos temporizadores são apresentados a posteriori nesta seção.

Conforme pode-se visualizar na Tabela 5.2, no cenário com 400 requisitantes, foi possível a adição de 399 requisitantes na fila de atendimento. Algumas mensagens de invocação podem ser perdidas na rede, e como os temporizadores não foram utilizados neste experimento, os requisitantes não realizam novas tentativas de envio.

Neste experimento, com 399 requisitantes aguardando fila, o $AG_{schedule}$ apresentou um tempo de processamento de 19,65s, enquanto que o $AS_{schedule}$ obteve um tempo de processamento de 2820,13s (cerca de 47 minutos). Ressalta-se, que o número de requisitantes na fila, não corresponde ao número de requisitantes atendidos. O número de requisitantes atendidos somente é obtido após o processamento da fila. O $AS_{schedule}$ é determinístico, e a medida que aumenta a quantidade de requisitantes na fila, o tempo de processamento aumenta exponencialmente, impactando negativamente no tempo de deslocamento do provedor, por conseguinte, diminui a quantidade de requisitantes atendidos. O LADP, não foi avaliado diante desta métrica por não formar fila de atendimento.

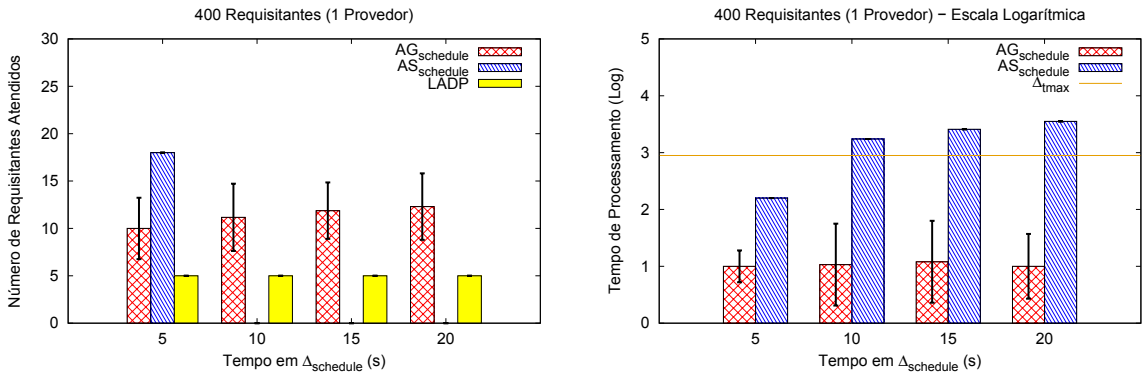
Tabela 5.2 – Tempo de processamento, $\Delta_{schedule}$ = 20s

		$AG_{schedule}$		$AS_{schedule}$	
Req	Fila (Qtde)	Processamento Média (segundos)	Desvio Padrão	Processamento (segundos)	Desvio Padrão
100	99	3,54	0,51	4,97	0,00
200	196	8,37	2,09	94,68	0,00
300	299	10,50	3,26	634,80	0,00
400	399	19,65	0,27	2820,13	0,00

Na Figura 5.1(a), apresenta-se o número de requisitantes atendidos em relação a uma variação no valor do temporizador $\Delta_{schedule}$. O temporizador variou de 5 a 20 segundos. O número de requisitantes foi ajustado em 400 para todos os cenários. Observa-se que o $AG_{schedule}$ possibilitou organizar a fila de atendimento aos requisitantes em todas as variações do temporizador $\Delta_{schedule}$. Por exemplo, para o cenário com $\Delta_{schedule}$ = 5s, o $AG_{schedule}$ obteve um número médio de 10 atendimentos, enquanto o $AS_{schedule}$, para o mesmo cenário, realizou em média 18 atendimentos. Por outro lado, a medida que o $\Delta_{schedule}$ varia para maior, também aumenta o tempo de processamento do $AS_{schedule}$, refletindo dessa forma no número de requisitantes atendidos. Neste experimento, verificou-se que o número de requisitantes na fila foi baixo devido ao fato do valor ajustado para $\Delta_{schedule}$ ser curto.

Na Figura 5.1(b), observa-se o tempo de processamento dos algoritmos - $AG_{schedule}$ e $AS_{schedule}$ em relação a variação no valor do temporizador $\Delta_{schedule}$ (5 a 20 segundos). Optou-se em utilizar escala logarítmica para a apresentação dos dados (*y = log x*) e o valor logarítmico de Δ_{tmax} = 2,95s como ponto de corte. Com $\Delta_{schedule}$ maior que 5 segundos, $AS_{schedule}$ obteve tempo de processamento acima do limitador Δ_{tmax}, refletindo dessa forma no número de requisitantes atendidos e consequentemente não gerando uma fila de atendimento.

Na Figura 5.2, apresenta-se uma análise do consumo de energia (CE) do nó provedor. Neste cenário, o número de nós totaliza 451 (450 requisitantes/intermediários e 1 provedor). O número de requisitantes variou de 100 a 400. Neste experi-



(a) 400 Requisitantes

(b) 400 Requisitantes

Figura 5.1 – Análise do número de requisitantes atendidos e tempo de processamento (Log), em função do temporizador $\Delta_{schedule}$, 1 Provedor.

mento, realizaram-se comparações entre os algoritmos de escalonamento, $AG_{schedule}$ e $AS_{schedule}$. O consumo de energia do provedor também foi medido para o protocolo LADP em sua versão original.

No cenário com 400 requisitantes e $\Delta_{schedule}$ = 20s (Figura 5.2), o consumo de energia do provedor foi aproximadamente 31,66% com o $AS_{schedule}$, 31,68% com o $AG_{schedule}$ e 31,61% com o LADP. Neste experimento, por tratar-se de uma distribuição não-normal, não possibilitou realizar testes paramétricos. Contudo, aplicou-se o teste de *Wilcoxon* e os resultados estatísticos demonstram-se equivalentes. O consumo de energia decorrente do processamento da fila, não foi significativo entre as duas abordagens neste cenário. Ressalta-se que os temporizadores não estão envolvidos neste experimento, e não existe o consumo de energia associado à transmissão de mensagens relativas aos temporizadores.

No gráfico reproduzido na Figura 5.3, apresenta-se o número de requisitantes atendidos (RA) pelo provedor. Neste experimento, o número de requisitantes variou entre 100 e 400, e o valor de $\Delta_{schedule}$ é de 20s. Avaliou-se os algoritmos $AS_{schedule}$, $AG_{schedule}$ e LADP.

O algoritmo $AS_{schedule}$ obteve melhores resultados no quesito número de atendimentos nos cenários entre 100 requisitantes (18 atendidos) e 200 requisitantes (19 atendidos). Já no cenário que envolveu 300 a 400 requisitantes, o $AS_{schedule}$ apresentou um número de atendimento inferior ao $AG_{schedule}$. Destaca-se, que no cenário com 200 nós, apesar do tempo de processamento do $AS_{schedule}$ ser maior (ver Tabela 5.2), o $AG_{schedule}$ atendeu a um número menor de requisitantes. Neste caso, o tempo de processamento não afetou no tempo de deslocamento do provedor até os requisitantes. Diferente do cenário com 400 nós, porque o tempo de processamento do $AS_{schedule}$

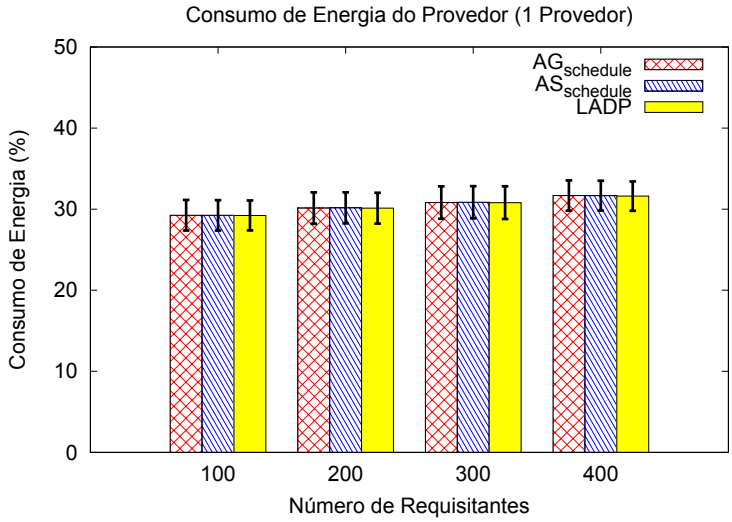


Figura 5.2 – Análise do consumo de energia do provedor, em função do número de requisitantes. $\Delta_{schedule}$ = 20s.

foi igual a 2820,13s (47min), valor acima de Δ_{tmax}, resultado que torna o algoritmo $AS_{schedule}$ inviável para a geração da fila de atendimento. O baixo número de atendimentos em todos os cenários e algoritmos, deve-se ao tempo de deslocamento do provedor até aos requisitantes, limitado a 15min.

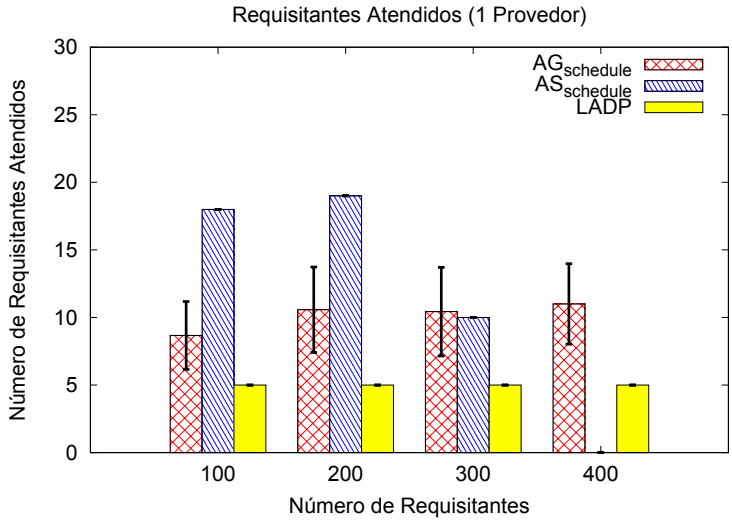


Figura 5.3 – Análise do número de atendimentos, em função do número de requisitantes. $\Delta_{schedule}$ = 20s

A Figura 5.4 representa a capacidade de convergência do $AG_{schedule}$. O eixo *y* representa o *fitness* que avalia a qualidade do indivíduo (requisitantes em fila) com relação ao custo do deslocamento do provedor até os requisitantes. O alto valor atribuído ao eixo *y* é decorrente das penalidades ocorridas em cada geração (ver Seção 4.1.3). E o eixo *x* representa o número de gerações.

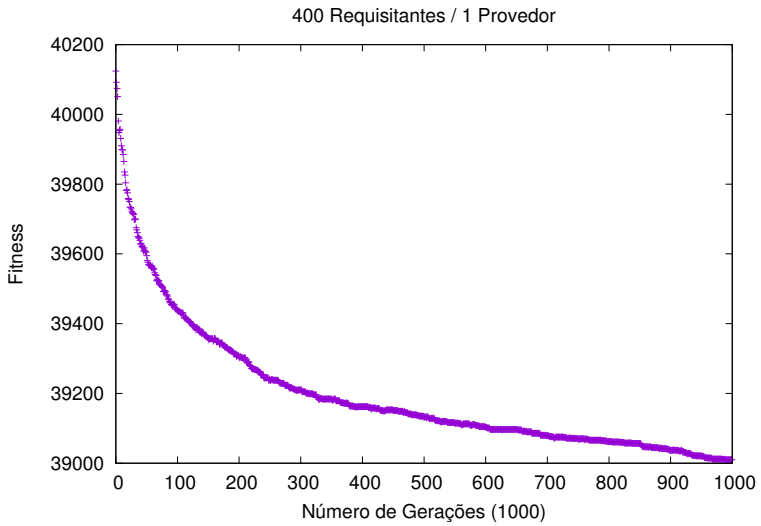


Figura 5.4 – Avaliação da convergência (*fitness*), em função do número de gerações (400 requisitantes). $\Delta_{schedule}$ = 20s

Analisando-se a métrica *fitness* no gráfico de convergência, nota-se que o $AG_{schedule}$ convergiu de forma rápida nas gerações de 400 a 500. Entretanto, não convergiu ao atingir o máximo de 1000 gerações, porque quando se aumenta o número de variáveis (requisitantes em fila) aumenta também o seu espaço de busca. Em uma análise qualitativa, podemos observar que o algoritmo genético ainda esta em fase de convergência, dessa forma, com o aumento do número de gerações pode-se obter um melhor resultado. Porém, aumentar o número de gerações também aumenta o tempo de processamento, e pode afetar no número de atendimento dos requisitantes.

5.2.1.1 Análise dos Temporizadores com Um Provedor

Nos experimentos apresentados a seguir, foram aplicados os temporizadores definidos na Capítulo 4 (Seção 4.1) para a troca de mensagens entre requisitantes e provedor, visando avaliar o desempenho dos temporizadores na recuperação de possíveis falhas como, por exemplo, perda de mensagens. Utilizou-se os mesmos cenários de configuração dos experimentos apresentados anteriormente. A diferença está no valor associado ao temporizador $\Delta_{schedule}$, torna-se dinâmico, e pode variar de acordo com as condições de tráfego da rede.

No gráfico reproduzido na Figura 5.5, apresenta-se o número de requisitantes atendidos (RA) pelo provedor. Com base nos resultados obtidos, verificou-se que algoritmo $AS_{schedule}$ alcançou um número maior de atendimentos comparado ao $AG_{schedule}$ e o LADP nos cenários com 100, 200, 300 e 400 requisitantes.

Analisando os resultados, observou-se que o valor associado ao temporizador $\Delta_{schedule}$ foi em média de 1051ms, tempo que não permitiu o recebimento de um maior

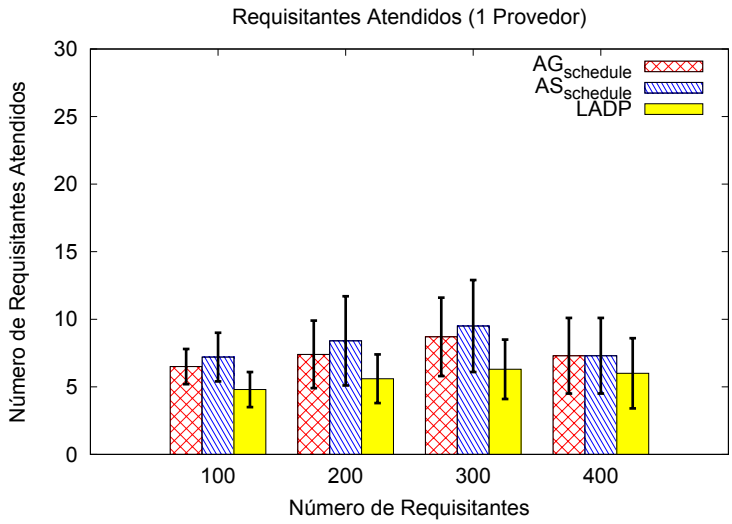


Figura 5.5 – Análise do número de atendimentos, em função do número de requisitantes, 1 Provedor.

número de mensagens enviadas por requisitantes. Por exemplo, no cenário com 300 requisitantes, o $AG_{schedule}$ atendeu em média 8,7 requisitantes, enquanto o $AS_{schedule}$ atendeu 9,5 requisitantes. Os algoritmos de escalonamento encontraram soluções, levando-se em consideração a soma do tempo de deslocamento do provedor até os requisitantes, e o tempo de atendimento estipulado em Δ_{tmax} pelos requisitantes. O LADP atendeu a um número de 6 requisitantes. Este comportamento deve-se ao fato do LADP atender ao requisitante, cuja a mensagem de reconhecimento da confirmação for a primeira a chegar no provedor.

O experimento conforme sistematizado no gráfico da Figura 5.6, demonstra o tempo médio de processamento (TP) dos algoritmos de escalonamento para a geração da fila de atendimento (Q). Por exemplo o cenário com 300 requisitantes, o algoritmo $AS_{schedule}$ apresentou um tempo médio de processamento de 0,05s, quando comparado com o $AG_{schedule}$ com o tempo de processamento de 0,70s. Este comportamento justifica-se pelo fato do baixo número de requisitantes que ficaram na fila para posterior processamento (média de 16,2 requisitantes). O experimento não possibilitou realizar testes paramétricos, por tratar-se de uma distribuição não-normal. Nenhum efeito significativo foi observado.

Tomando por base esses experimentos, conclui-se que o valor do temporizador $\Delta_{schedule}$ varia de acordo com as condições da rede. No gráfico apresentado na Figura 5.10, demonstra-se que a sobrecarga da abordagem para o escalonamento de recursos ficou abaixo de 2%, e em decorrência deste fato, o processo de comunicação entre requisitantes e provedores ocorreu rapidamente. Por outro lado, no cenário alvo, o objetivo é atender ao maior número de vítimas.

Visando resolver o problema do valor de $\Delta_{schedule}$ pequeno, sugere-se 2 (duas)

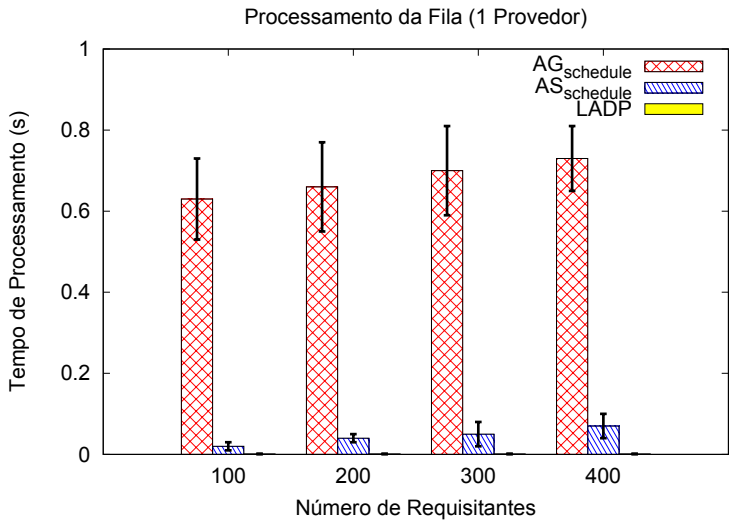


Figura 5.6 – Análise do tempo de processamento, em função do número de requisitantes, 1 Provedor.

estratégias: na primeira, aumenta-se o valor da distância do requisitante mais distante para o qual o provedor enviou as mensagens de resposta por um número *n*. Por exemplo, se $\Delta_{schedule}$ expirar e o provedor verificar que apenas 50% dos nós para os quais ele enviou mensagem de resposta estão na fila, o provedor pode duplicar o valor de $\Delta_{schedule}$ para esperar pelas mensagens dos requisitantes faltantes; na segunda, o provedor gera a fila quando $\Delta_{schedule}$ expirar e, a medida que se encaminhar na direção do primeiro requisitante da fila, vai processando novas mensagens de descoberta de serviços recebidas, analisando a viabilidade de realizar novos atendimentos. Caso seja possível, avisa os requisitantes já escalonados sobre o novo tempo para atendimento.

Na Figura 5.7, apresentam-se os resultados do consumo de energia (CE) dos dispositivos da MANET em um cenário com 451 nós (formado por intermediários, requisitantes e 1 provedor). O consumo de energia foi avaliado em função do número de requisitantes que variou de 100 a 400 nós.

Por exemplo, o consumo médio de energia com 200 nós requisitantes foi de 16,04% com a utilização do $AG_{schedule}$. Com o algoritmo $AS_{schedule}$ o consumo médio de energia foi de 16,04%, enquanto o LADP obteve um consumo médio de energia de 16,02%. Segundo análise estatística realizada, os dados seguem uma distribuição normal (Rejeite H_0). No teste *t* de *Student*, demonstra-se que não existe diferença estatística entre as médias do consumo de energia entre os algoritmos $AG_{schedule}$, $AS_{schedule}$ e LADP. Ressalta-se que, na fase de invocação de serviços, considerando-se o escalonamento de recursos, as mensagens são enviadas em *unicast*, uma vez que provedores e requisitantes já são conhecidos, fato que resulta em baixa sobrecarga e consumo de energia.

No gráfico da Figura 5.8, apresenta-se uma análise individual do consumo de

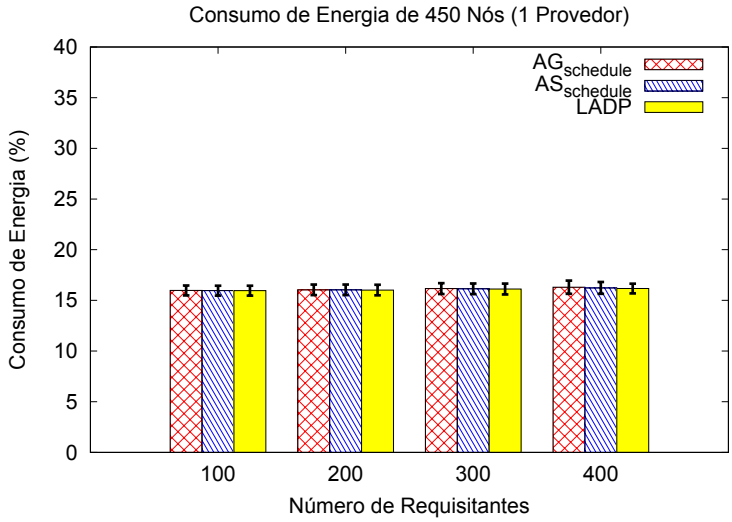


Figura 5.7 – Análise do consumo de energia, em função do número de requisitantes, 1 Provedor.

energia (CE) do nó provedor, considerando-se o mesmo cenário de configuração da Figura 5.7. No cenário com 300 requisitantes, o consumo de energia do provedor foi de 17,7% com o $AG_{schedule}$, 17,6% com o $AS_{schedule}$ e de 17,5% com o LADP. Segundo análise estatística realizada, os dados não seguem uma distribuição normal. No teste *Wilcoxon*, demonstra-se que não existe diferença estatística entre as médias do consumo de energia dos algoritmos $AG_{schedule}$, $AS_{schedule}$ e LADP.

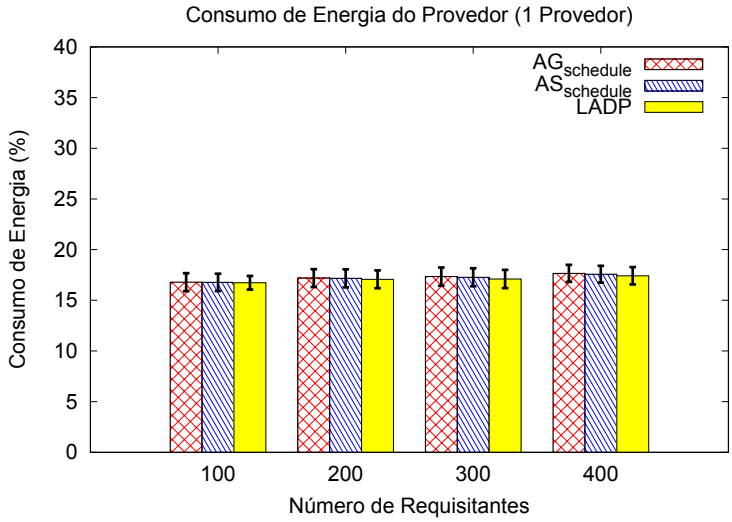


Figura 5.8 – Análise do consumo de energia do provedor, em função do número de requisitantes, 1 Provedor.

Na Figura 5.9, ilustram-se os resultados relacionados com a perda de pacotes (PP). Na especificação do algoritmo de escalonamento, implementou-se nos nós requisitantes os temporizadores Δ_{wait} e Δ_{attend} (Seção 4.1), que estabelecem um número fixo de tentativas diante de mensagens perdidas. Estatisticamente, os dados não

seguem uma distribuição normal (Rejeite H_0). No teste *Wilcoxon*, demonstra-se que não existe diferença estatística na perda de pacotes entre os algoritmos $AG_{schedule}$, $AS_{schedule}$ e LADP.

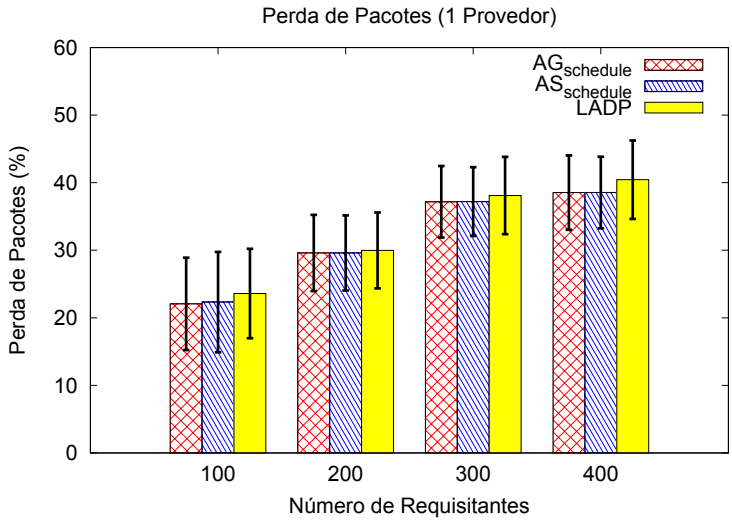


Figura 5.9 – Análise da perda de pacotes, em função do número de requisitantes, 1 Provedor.

O LADP se restringe ao atendimento da primeira mensagem de invocação recebida. Neste caso, os requisitantes que não tiveram seus pedidos atendidos devem reiniciar o processo de descoberta de serviços gerando um número maior de mensagens na rede. Um exemplo pode ser visualizado no cenário de 400 requisitantes, no qual o LADP apresentou uma perda de 40%, para os algoritmos $AG_{schedule}$ e o $AS_{schedule}$, o percentual foi acima 38%. Nos cenários com 100 e 200 requisitantes, o percentual de perdas do $AG_{schedule}$ e do $AS_{schedule}$ ficou abaixo de 30%.

No experimento reproduzido na Figura 5.10, avaliou-se a sobrecarga (SB) decorrente das mensagens geradas pela abordagem para o escalonamento de recursos, para diferentes números de requisitantes.

Para os cenários mais densos, por exemplo com 300 requisitantes, a sobrecarga resultante com o $AG_{schedule}$ e o $AS_{schedule}$ foi de 1,5%. Conforme descrito no experimento representado pela Figura 5.10, o uso dos temporizadores não afetou negativamente a sobrecarga de mensagens na rede. Os dados apresentados não seguem uma distribuição normal (Rejeite H_0). No teste *Wilcoxon*, demonstra-se que não existe diferença estatística na sobrecarga da rede com a utilização dos algoritmos $AG_{schedule}$, $AS_{schedule}$ e LADP

Na Figura 5.11, apresenta-se uma análise da capacidade de convergência do algoritmo $AG_{schedule}$ em função do número de requisitantes (300 requisitantes). O valor atribuído ao eixo *y* corresponde ao custo de deslocamento do provedor somado

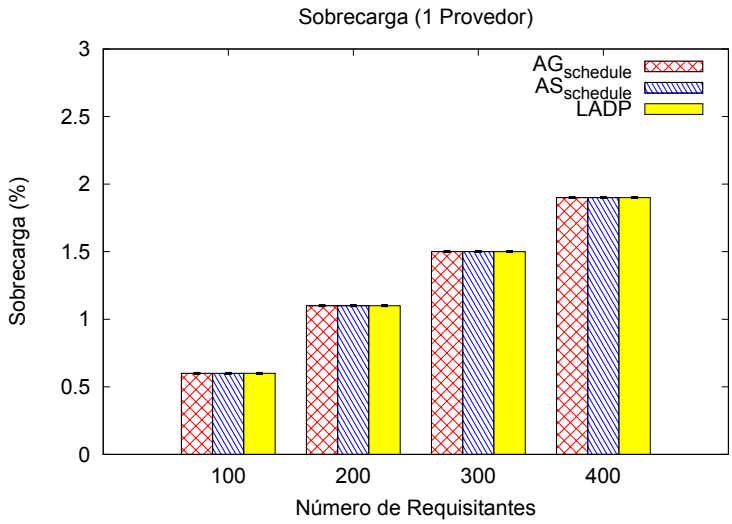
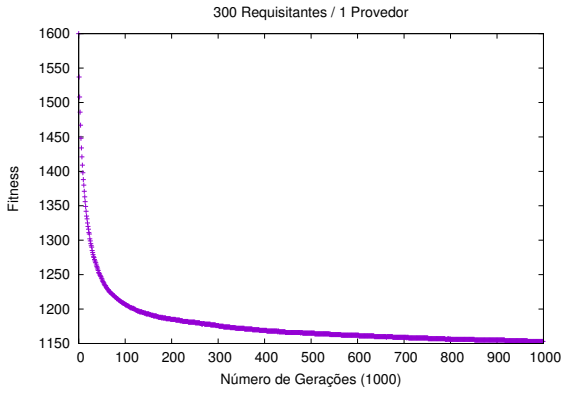


Figura 5.10 – Análise da sobrecarga, em função do número de requisitantes, 1 Provedor.

as penalidades ocorridas em cada geração (ver Seção 4.1.3). E o eixo representa *x* o número de gerações.



(a) 300 Requisitantes

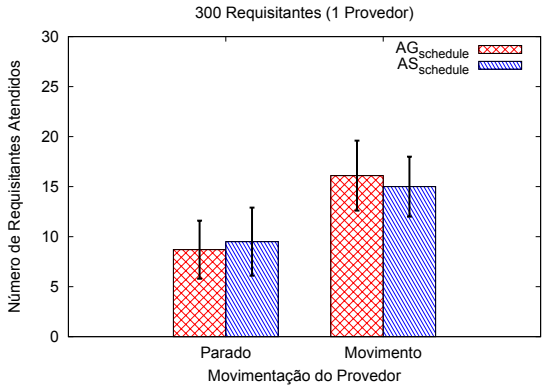
Figura 5.11 – Avaliação da convergência (*fitness*), em função do número de gerações do $AG_{schedule}$.

Analisando o gráfico de convergência (Figura 5.11), constata-se que o $AG_{schedule}$ não convergiu totalmente ao atingir o máximo de 1000 gerações disponíveis no cenário avaliado, por exemplo com. A rápida convergência do $AG_{schedule}$ se deve ao fato de que o domínio considerado desta função é relativamente pequeno para o tamanho da população utilizada (100 indivíduos) e, assim, é possível varrer o espaço de busca como um todo. Porém, a partir da geração 500 a convergência é lenta devido a quantidade de elementos da população gerada. Em uma análise qualitativa, o algoritmo genético continua em fase de convergência, e com o aumento do número de gerações pode-se obter um melhor resultado. Porém, conforme descrito anteriormente, pode afetar no número de atendimento dos requisitantes.

5.2.1.2 *Análise dos Algoritmos de Escalonamento, Provedor em Movimento*

Nesta seção, analisa-se a métrica, número de requisitantes atendidos para um cenário com o provedor em movimento, ou seja, ele já processou a fila e está a caminho do primeiro requisitante da fila para realizar o atendimento. Neste cenário, 300 (trezentos) requisitantes e 1 (um) provedor estão na área afetada. A verificação de novas mensagens que chegam ao provedor é realizada durante o seu deslocamento para o primeiro requisitante da fila de atendimento. Para possibilitar medir o consumo de energia nos nós da rede, após o primeiro atendimento, realiza-se uma nova distribuição dos requisitantes na área afetada e reinicia-se todo o processo de descoberta de serviços.

Na Figura 5.12, observa-se que o número de requisitantes na fila de atendimento com o provedor em movimento foi de 16 com o $AG_{schedule}$ e 15 com o $AS_{schedule}$. Comparando-se ao cenário no qual o provedor não trata novas requisições durante o deslocamento obteve-se 8 requisitantes com o $AG_{schedule}$ e 9 com o $AS_{schedule}$. Com o provedor em movimento foi igualado o número de atendimentos dos dois algoritmos, porque o processo permitiu a recuperação de uma mensagem perdida durante o recebimento da invocação. Neste experimento, por tratar-se de uma distribuição não-normal, não possibilitou realizar testes paramétricos. No teste de *Wilcoxon*, demonstra-se mesma distribuição (estatística=18000, p=0,333). A baixa quantidade de requisitantes que entraram em fila de processamento resultou em uma baixa taxa de atendimento, isso ocorreu devido a quantidade de mensagens de confirmação da invocação que o provedor recebeu dos requisitantes.



(a) 300 Requisitantes

Figura 5.12 – Análise do número de requisitantes atendidos, em função da movimentação do provedor, 1 Provedor.

Conclui-se, a partir dos resultados apresentados na Figura 5.12, que permitir que provedor processe novamente a fila durante o deslocamento na direção do primeiro requisitante, contribui para amenizar o fato que o temporizador $\Delta_{schedule}$ pode

ser curto, como consequência das condições da rede. Neste contexto, foi possível aumentar o número de requisitantes atendidos, sem prejudicar os atendimentos previamente agendados. Motivo da baixa taxa de atendimento ocorreu porque as novas mensagens recebidas pelo provedor já em movimento, não foram aceitas para a integração na fila de atendimento.

5.2.2 Análise dos Algoritmos de Escalonamento com Dois Provedores

Os algoritmos para escalonamento de recursos foram avaliados em face da presença de 2 (dois) provedores disponíveis no cenário. O aumento do número de provedores, demanda poder computacional, por esse motivo, limitou-se a somente 2 (dois) provedores. Avaliou-se as métricas: consumo de energia, sobrecarga e número de requisitantes atendidos dentro do tempo estipulado Δ_{tmax} = 15 minutos. Para essa análise, aplicou-se os temporizadores definidos na Capítulo 4 (Seção 4.1) para a troca de mensagens entre requisitantes e provedor. Os provedores deslocam-se com a velocidade de 6,0m/s e os intermediários a 1,5m/s. Um número máximo de 452 nós foram utilizados, considerando 400 nós requisitantes, 50 nós intermediários e 2 provedores.

Na Tabela 5.3, o valor de $\Delta_{schedule}$ foi em média de 2585ms para o cenário com 400 requisitantes. Ressalta-se que o valor de $\Delta_{schedule}$ varia de acordo com as condições da rede. Conforme apresentado anteriormente, diante de um valor de $\Delta_{schedule}$ baixo, o número de requisitantes em fila tende a ser baixo. Por exemplo, para o cenário com 2 provedores e 400 requisitantes, somente 24,45 requisitantes em média aguardaram na fila.

Tabela 5.3 – Desvio padrão do tempo de processamento e atendimentos (400 requisitantes)

		$AG_{schedule}$				$AS_{schedule}$			
Provedor (Qtde)	Fila (Qtde)	Processamento (segundos)	DV.P	Atendidos (Qtde)	DV.P	Processamento (segundos)	DV.P	Atendidos (Qtde)	DV.P
1	29,93	0,84	0,09	6,14	3,15	0,11	0,13	7,20	2,95
2	24,45	0,47	0,07	11,30	4,22	0,06	0,08	12,20	4,44

Entretanto, conforme mostrado na Figura 5.13, o número de atendimentos é maior com 2 (dois) provedores. Por exemplo, no cenário contendo 2 (dois) provedores e fazendo uso do $AS_{schedule}$, atendeu-se na média 12,2 requisitantes, enquanto no cenário com apenas 1 (um) provedor e com o $AS_{schedule}$, a média de atendimento foi de 7,2 requisitantes. No mesmo cenário, fazendo uso do $AG_{schedule}$, cada provedor atendeu na média 11,3 requisitantes, enquanto no cenário com apenas 1 (um) provedor e com o $AG_{schedule}$, a média de atendimento foi de 6,2 requisitantes. Neste experimento, por tratar-se de uma distribuição não-normal, não possibilitou realizar testes paramétricos. No teste de *Wilcoxon*, demonstra-se mesma distribuição (estatística=15000, p=0,203). Como a sobrecarga (Figura 5.15) com 2 (dois) provedores não é significa-

tiva, conclui-se que a adição de mais provedores beneficia o número de atendimentos no cenário pós-desastre.

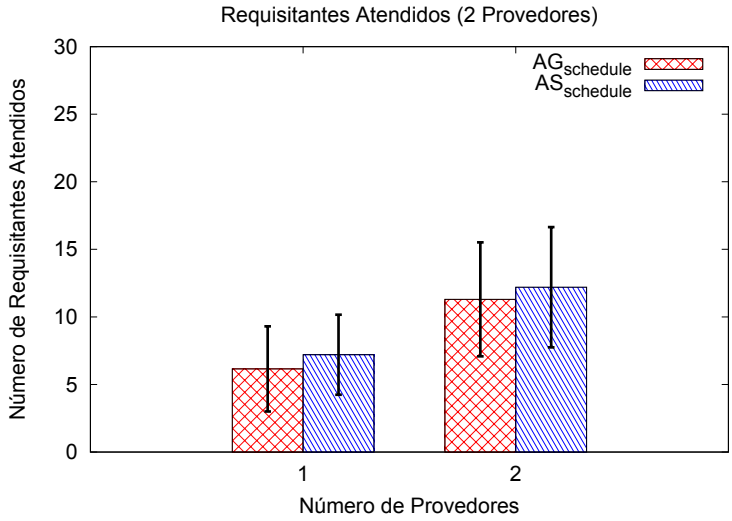


Figura 5.13 – Análise do número de atendimentos, em função do número de provedores, 400 requisitantes e 2 Provedores.

Na Figura 5.14, apresenta-se o consumo individual de energia (CE) dos nós provedores. No cenário com 400 requisitantes, o consumo de energia dos provedores foi de 15,25% com o $AG_{schedule}$, 15,23% como o $AS_{schedule}$. Os resultados comprovam que a distribuição de mais provedores na área afetada pode contribuir para reduzir o consumo individual de energia dos provedores. Segundo análise estatística realizada, os dados seguem uma distribuição normal. No teste *t* de *Student*, demonstra-se que existe uma diferença estatística entre as médias do consumo de energia entre os algoritmos $AG_{schedule}$ e $AS_{schedule}$ (Rejeite H_0).

No gráfico da Figura 5.15, avaliou-se a sobrecarga (SB) de pacotes gerados pelos algoritmos em relação ao número de provedores. Constatou-se que a sobrecarga de ambos os algoritmos foi baixa. Neste experimento, o tamanho da amostra é muito pequeno para aproximação normal.

Por exemplo, no cenário com 2 provedores e com o uso do algoritmo $AG_{schedule}$, a sobrecarga foi de 2,22%. A adição de mais um provedor aumentou a sobrecarga na rede, como consequência do aumento da quantidade de mensagens trocadas entre requisitantes e provedores. Este resultado demonstra que o uso de mais provedores beneficia o número de atendimento, porém, pode afetar na sobrecarga da rede. Isto porque, ambos os provedores (se aptos) enviam mensagem de respostas aos requisitantes.

Contudo, no trabalho de Kniess et al. (2011) um mecanismo de agregação de respostas foi proposto para reduzir as mensagens de resposta na rede. Os resulta-

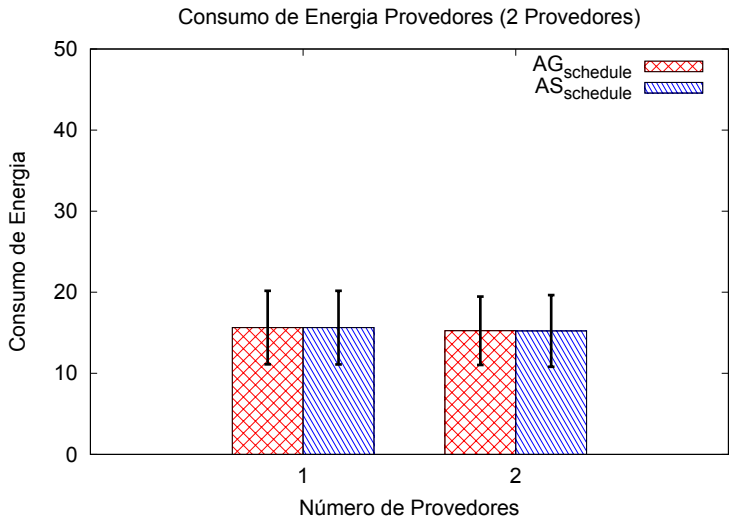


Figura 5.14 – Análise do consumo de energia do provedor, em função do número de requisitantes, 400 requisitantes e 2 Provedores.

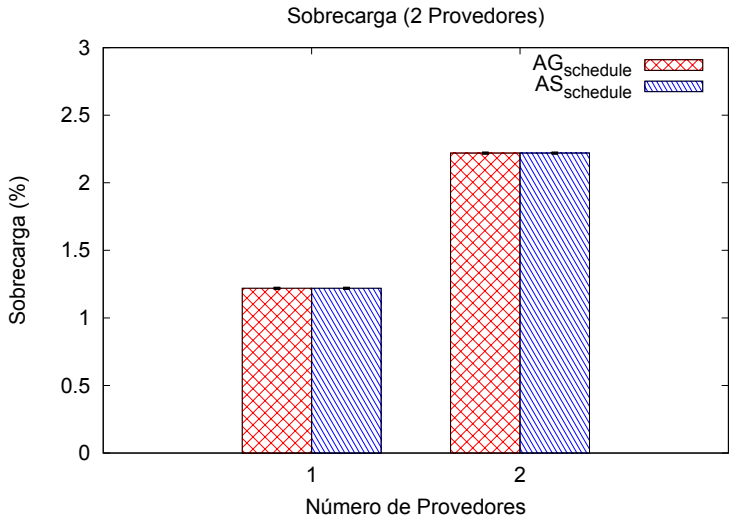


Figura 5.15 – Análise da sobrecarga, em função do número de requisitantes, 400 requisitantes e 2 Provedores.

dos apresentados em Kniess et al. (2011), mostram que o mecanismo de agregação reduz significativamente as mensagens de resposta sem impactar no processo de descoberta. O mecanismo de agregação do LADP não foi implementado no contexto deste trabalho de dissertação.

5.3 CONCLUSÕES DO CAPÍTULO

A abordagem para escalonamento de recursos é composta pelos algoritmos $AG_{schedule}$ e $AS_{schedule}$ e por temporizadores para a sincronização da comunicação entre requisitantes e provedores.

Com base na análise dos resultados, conclui-se que o tempo definido em $\Delta_{schedule}$ é ajustável de acordo com as condições da rede, resultando em um valor baixo em alguns cenários. Neste sentido, neste capítulo, foram apresentadas duas soluções para este problema: (i) aumentar o valor da distância do requisitante mais distante para o qual o provedor enviou as mensagens de resposta por um número *n*; (ii) permitir que o provedor processe novas mensagens de descoberta de serviços recebidas durante o seu deslocamento na direção do primeiro requisitante. Na Subseção 5.2.1.2 deste capítulo, a segunda proposta foi avaliada, e um número maior de requisitantes puderam ser atendidos, sem prejudicar os atendimentos previamente agendados.

Tomando por base os resultados obtidos com os mecanismos de escalonamento $AG_{schedule}$ e $AS_{schedule}$, verificou-se que o desempenho de ambas está relacionado ao número de requisitantes que deve ser processado. Nos cenários mais densos, por exemplo, com 300 nós, o $AG_{schedule}$ apresentou melhor desempenho em relação ao número de atendimentos. Neste caso, o tempo de processamento do $AS_{schedule}$ para encontrar uma solução viável, reduz expressivamente o número de atendimentos possíveis.

Por fim, avaliou-se cenários com 2 (dois) provedores. O consumo de energia foi melhor distribuído entre os provedores neste cenário. Entretanto, observou-se uma sobrecarga maior que nós cenários com 1 (um) provedor. Atribui-se a este resultado, o fato que ambos os provedores enviaram mensagens de respostas para os requisitantes. Se os dois provedores são aptos para o atendimento, enviam mensagens de respostas para um mesmo requisitante. Este problema pode ser solucionado com a utilização do mecanismo de agregação de respostas proposto no protocolo LADP na sua versão original. O mecanismo de agregação não foi implementado no escopo deste trabalho.

6 CONCLUSÃO

Neste trabalho de dissertação, apresentou-se a especificação, a implementação e a avaliação de desempenho de uma abordagem para escalonamento de recursos que estende o protocolo de descoberta de serviços *Location Aware Discovery Protocol* (LADP) (KNIESS; LOQUES; ALBUQUERQUE, 2011), que em sua versão original não contempla o escalonamento dos provedores. A solução para o escalonamento foi desenvolvida no contexto de MANETs de múltiplos saltos, assistidas por equipes de resgate em áreas afetadas por um desastre natural ou não natural.

Com base em pesquisas realizadas na literatura da área (Capítulo 3), verificou-se que os protocolos de descoberta de serviços desenvolvidos para MANTEs e para atuarem em cenários de emergência, não contemplam estratégias para o escalonamento dos provedores de recursos.

Tendo em vista essa lacuna nos protocolos para descoberta de serviços, foram desenvolvidos os seguintes mecanismos: (i) escalonamento de recursos com base em algoritmos genéticos, $AG_{schedule}$; (ii) escalonamento de recursos com base no algoritmo A-Estrela (A*), $AS_{schedule}$ e; (iii) temporizadores para o gerenciamento da comunicação entre requisitantes e provedores. A abordagem para o escalonamento de recursos, foi especificada com as seguintes características: os nós da rede são cientes de sua localização geográfica, os provedores de recursos são móveis e sujeitos ao tempo máximo para atendimento de um pedido, à velocidade de deslocamento e à informação de localização para atender a uma requisição.

Neste contexto, foram desenvolvidos temporizadores, com a finalidade de gerenciar o envio e o recebimento de mensagens entre provedores e requisitantes. Por exemplo, o provedor após receber a primeira mensagem de invocação de um requisitante, aciona o temporizador Δ_{invoc}, que em conjunto com o temporizador Δ_{ack_invoc} definem o tempo total em que o provedor espera para receber as mensagens de confirmação da invocação do requisitante ($\Delta_{schedule}$). Os temporizadores possibilitam que os provedores e requisitantes recebam confirmações de mensagens anteriormente enviadas.

A análise de desempenho realizada com a aplicação dos temporizadores, mostrou que o valor de $\Delta_{schedule}$ é bastante dependente das características da rede, podendo resultar em um tempo curto, e que afeta negativamente no número de mensagens dos requisitantes enviadas ao(s) provedor(es) para a formação da fila de atendimento. Duas estratégias foram propostas como alternativas para esta limitação no uso dos temporizadores. Aumentar o valor da distância do requisitante mais distante

para o qual o provedor enviou as mensagens de resposta por um número *n*, se $\Delta_{schedule}$ expirar e o provedor verificar um baixo percentual de requisitantes na fila em relação ao número de respostas enviadas. Na segunda opção, o provedor gera a fila quando $\Delta_{schedule}$ expirar e, a medida que se desloca na direção do primeiro requisitante da fila, vai processando novas mensagens de descoberta de serviços recebidas, analisando a viabilidade de realizar novos atendimentos. Permitir que o provedor analise novamente a fila durante o deslocamento, contribuiu para para aumentar o número de requisitantes atendidos, sem prejudicar os atendimentos previamente agendados.

Os mecanismos para escalonamento de recursos, $AG_{schedule}$ e $AS_{schedule}$ possibilitaram que o provedor organize uma fila de atendimento e realize um número de atendimento mais adequado, quando comparado ao protocolo LADP em sua versão original.

Os resultados mostraram que o $AS_{schedule}$ apresentou melhores resultados quando o seu espaço de busca para solução é pequeno, porém, a medida que aumenta o número de requisitantes na fila de atendimento, o algoritmo requer maior tempo de processamento, impactando diretamente no tempo de deslocamento do provedor para atendimento aos requisitantes.

Conclui-se que para aplicações complexas $AG_{schedule}$ é uma abordagem recomendada, como por exemplo, para os cenários onde o número de requisitantes em fila é acima de 300 nós. Por outro lado, quando o número de requisitantes na fila for baixo, ou seja, para instâncias pequenas, o uso do $AS_{schedule}$ é indicado, uma vez que atende a um número maior de requisitantes e apresenta, neste caso, um custo menor associado ao processamento da fila.

Também cita-se como contribuição desta dissertação, a implementação da abordagem para o escalonamento de recursos, composta pelos algoritmos $AS_{schedule}$ e $AG_{schedule}$, e dos temporizadores no simulador de redes Network Simulator (NS3). Através desta implementação, novas pesquisas podem ser realizadas pela comunidade acadêmica.

6.1 TRABALHOS FUTUROS

A pesquisa desenvolvida neste trabalho de dissertação apresentou uma solução para o estado da arte na área de escalonamento de recursos para descoberta de serviços operando em MANETs. Contudo, existem alguns tópicos sobre os quais sugere-se a possibilidade de pesquisas futuras.

A partir dos resultados apresentados no Capítulo 5, sugere-se uma investigação em relação às estratégias para ajustar de forma adequada os temporizadores. Outros experimentos necessitam ser realizados, visando confirmar a acurácia das al-

ternativas propostas, como por exemplo, verificar a sensibilidade do parâmetro penalidade utilizada utilizada na função *fitness* na implementação do algoritmo genético, cujo algoritmo probabilístico tem condições de melhora. Mais Gerações seriam necessárias para uma eventual convergência do algoritmo genético, e podem dessa forma, melhorar os resultados. No algoritmo A-Estrela, testes com plano de corte podem ser aplicados a fim de diminuir o tempo de processamento.

Outro tópico que ainda pode ser explorado diz respeito à inclusão de um tempo de parada em cada requisitante. Esse tempo de parada determinaria o tempo que o provedor ficará no local de atendimento, antes de prosseguir para o próximo atendimento. Experimentos com essa variável poderiam ser realizados para verificar a eficiência do deslocamento do provedor até as vítimas.

Alguns aspectos não foram tratados no escopo deste trabalho, e que podem ser fontes interessantes de pesquisa, por exemplo, a adição de obstáculos durante o percurso que o provedor irá realizar para o atendimento às vítimas.

6.2 OUTRAS CONTRIBUIÇÕES DA DISSERTAÇÃO

Os resultados obtidos com a abordagem para o escalonamento de recursos foram publicadas nas seguintes conferências:

- PETRI, M.; KNIESS, J.; PARPINELLI, R. S. Escalonamento de recursos em descoberta de serviços para MANETs em cenários de emergência. XLIX Simpósio Brasileiro de Pesquisa Operacional, Blumenau, SC, BR, 8 2017;
- PETRI, M.; KNIESS, J.; PARPINELLI, R. S. Scheduling services for MANETs using genetic algorithms. 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery, Guilin, China, 7 2017.
- PETRI, M.; KNIESS, J.; PARPINELLI, R. S. Resource Scheduling for Mobility Scenarios with Time Constraints. CLEI 2018, XLIV Latin American Computing Conference, São Paulo, SP, BR, 10 2018;

  Afora as publicações mencionadas, um artigo a ser submetido para publicação em uma revista internacional está em fase final de desenvolvimento.

REFERÊNCIAS

ARENELLA, G.; SANTIS, F. de; MALANDRINO, D. Beeadhocservicediscovery: A MANET service discovery algorithm based on bee colonies. **Informatics in Control, Automation and Robotics (ICINCO), 2014 11th International Conference on**, Vienna, Austria, v. 1, p. 244–251, 2014.

BECCENERI, J. C. Meta-heurísticas e otimização combinatória: Aplicações em problemas ambientais. **INPE, Sao José dos Campos**, 2008.

BITAM, S.; BATOUCHE, M. C. Bees' system based topology discovery for mobile ad hoc network. **Information Technology Journal**, v. 5, p. 465–469, 2008. ISSN 1812-5638.

CAMP, T.; BOLENG, J.; DAVIES, V. A survey of mobility models for ad hoc network research. **Wireless communications and mobile computing**, Wiley Online Library, Colorado, U.S.A, v. 2, n. 5, p. 483–502, 2002.

CARVALHO, A. et al. Inteligência artificial–uma abordagem de aprendizado de máquina. **Rio de Janeiro: LTC**, 2011.

CHEN, Z.-G.; GE, Z.-H.; ZHAO, M. Congestion aware scheduling algorithm for manet. In: IEEE. **Wireless Communications, Networking and Mobile Computing, 2006. WiCOM 2006. International Conference on**. Wuhan, China, 2006. p. 1–5.

CHUN, B.-G.; BAKER, M. Evaluation of packet scheduling algorithms in mobile ad hoc networks. **ACM SIGMOBILE Mobile Computing and Communications Review**, ACM, New York, NY, USA, v. 6, n. 3, p. 36–49, 2002.

CLAUSEN, T.; JAQCQUET, P. Optimized link state routing (olsr) rfc 3626. **IETF Networking Group (October 2003)**, RFC Editor, n. 3626, p. 1–75, October 2003. Disponível em: <https://www.rfc-editor.org/rfc/pdfrfc/rfc3626.txt.pdf>.

CORMEN, T. H. et al. Algoritmos: teoria e prática. **Edição Americana. Editora Campus**, Rio de Janeiro, RJ, BR, v. 2, 2002.

DANIELSSON, P.-E. Euclidean distance mapping. **Computer Graphics and image processing**, Elsevier, v. 14, n. 3, p. 227–248, 1980.

ENZAI, N. I. M.; RAIS, S. S.; DARUS, R. An overview of scheduling algorithms in mobile ad-hoc networks. In: IEEE. **Computer Applications and Industrial Electronics (ICCAIE), 2010 International Conference on**. Kuala Lumpur, Malásia, 2010. p. 120–124.

FATTAH, H.; LEUNG, C. An overview of scheduling algorithms in wireless multimedia networks. **IEEE wireless communications**, IEEE, v. 9, n. 5, p. 76–83, December 2002. ISSN 1558-0687.

GLOVER, F. W.; KOCHENBERGER, G. A. **Handbook of metaheuristics**. New York, USA: Springer Science & Business Media, 2006. v. 57.

HART, P. E.; NILSSON, N. J.; RAPHAEL, B. A formal basis for the heuristic determination of minimum cost paths. **IEEE transactions on Systems Science and Cybernetics**, IEEE, v. 4, n. 2, p. 100–107, July 1968. ISSN 2168-2887.

HOLLAND, J. H. et al. **Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence**. London, England: MIT press, 1992.

JAMES, K.; KEITH, R. **Redes de Computadores e a Internet: Uma abordagem top-down**. São Paulo, SP: Persion Education do Brasil, 2013. v. 6.

JANG, H.-C.; LIEN, Y.-N.; TSAI, T.-C. Rescue information system for earthquake disasters based on manet emergency communication platform. In: ACM. **Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly**. Leipzig, Alemanha, 2009. p. 623–627.

JUSZCZYK, L.; DUSTDAR, S. A middleware for service-oriented communication in mobile disaster response environments. **Proceedings of the 6th international workshop on Middleware for pervasive and ad-hoc computing**, Leuven, Bélgica, p. 37–42, 2008.

KARABOGA, D.; AKAY, B. A survey: algorithms simulating bee swarm intelligence. **Artificial intelligence review**, Springer, Kluwer Academic Publishers Norwell, MA, USA, v. 31, n. 1-4, p. 61, 2009.

KNIESS, J.; LOQUES, O.; ALBUQUERQUE, C. V. N. de. **Descoberta de Serviço em Redes Ad Hoc Móveis**. Tese (Doutorado) — Universidade Federal Fluminense (UFF), Niterói, Rio de Janeiro, Julho 2011.

LABIOD, H. **Wireless ad hoc and Sensor Networks**. Hoboken, USA: John Wiley & Sons, 2010. v. 6.

LINDEN, R. **Algoritmos genéticos (2a ediçao)**. Rio de Janeiro, BR: Brasport, 2008.

MALLAH, R. A.; QUINTERO, A. A light-weight service discovery protocol for ad hoc networks. **Journal of Computer Science**, CiteSeerx, Montreal, Canadá, v. 5, n. 4, p. 330–337, 2009. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/similar?doi=10.1.1.165.8044&type=ab>.

MARIN-PERIANU, R.; HARTEL, P. H.; SCHOLTEN, J. **A Classification of Service Discovery Protocols**. Enschede, Holanda: Tech. rep., Centre for Telematics and Information Technology, University of Twente, 2005.

MIAN, A. N.; BALDONI, R.; BERALDI, R. A survey of service discovery protocols in multihop mobile ad hoc networks. **IEEE Pervasive Computing**, Piscataway, NJ, USA, v. 8, n. 1, p. 66–74, 2009.

MICHALEWICZ, Z. Evolution strategies and other methods. In: **Genetic algorithms+ data structures= evolution programs**. Berlin, Heidelberg: Springer, 1996. p. 159–177.

MITCHELL, T. M. et al. Machine learning. 1997. **Burr Ridge, IL: McGraw-Hill Science/Engineering/Math**, Ithaca, NY, v. 45, n. 37, p. 870–877, March 1997.

NASCIMENTO, D. da C. et al. Testes de normalidade em análises estatísticas: Uma orientação para praticantes em ciências da saúde e atividade física. **Revista Mackenzie de Educação Física e Esporte**, v. 14, n. 2, 2017.

NS-3 development team. **Ns-3 network simulator**. 2008. Acessado em 01-05-2017. Disponível em: <https://www.nsnam.org/>.

PARK, J. C. et al. Distributed semantic service discovery for MANET. **Ubiquitous Intelligence and Computing, 2013 IEEE 10th International Conference on and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC)**, Vietri sul Mere, Itália, p. 515–520, 2013.

PARPINELLI, R. S.; LOPES, H. S. New inspirations in swarm intelligence: a survey. **International Journal of Bio-Inspired Computation**, Inderscience Publishers Ltd, v. 3, n. 1, p. 1–16, 2011.

PEARL, J. **Intelligent Search Strategies for Computer Problem Solving**. [S.l.]: Addison-Wesley: Reading MA, 1984.

PRIM, R. C. Shortest connection networks and some generalizations. **The Bell System Technical Journal**, Nokia Bell Labs, v. 36, n. 6, p. 1389–1401, 1957.

RAJU, T. N. William sealy gosset and william a. silverman: two “students” of science. **Pediatrics**, Am Acad Pediatrics, v. 116, n. 3, p. 732–735, 2005.

RILEY, G. F.; HENDERSON, T. R. The ns-3 network simulator. **Modeling and tools for network simulation**, Springer, Berlin, Heidelberg, p. 15–34, 2010.

ROY, M.; KAR, P.; MUKHERJEE, N. An enhanced service framework in manets for application in emergency services. **Proceedings of international conference on intelligent network and computing**, Kuala Lumpur, Malaysia, 2010.

RUELA, J. C. Algumas análises sobre mecanismos para prover qualidade de serviço em redes multimídia. Instituto Nacional de Telecomunicações, Santa Rita do Sapucaí, MG, 2006.

SERHANI, M. A.; GADALLAH, Y. A service discovery protocol for emergency response operations using mobile ad hoc networks. Barcelona, Espanha, p. 280–285, 2010.

SHAPIRO, S. S.; WILK, M. B. An analysis of variance test for normality (complete samples). **Biometrika**, JSTOR, v. 52, n. 3/4, p. 591–611, 1965.

SHUKLA, A.; PANDEY, H. M.; MEHROTRA, D. Comparative review of selection techniques in genetic algorithm. In: IEEE. **Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE), 2015 International Conference on**. Noida, Índia, 2015. p. 515–519.

SIDDARTH, E. C.; SEETHARAMAN, K. A cluster based qos-aware service discovery architecture for mobile ad hoc networks. In: IEEE. **Computational Intelligence & Computing Research (ICCIC), 2012 IEEE International Conference on**. Coimbatore, Índia, 2012. p. 1–6.

SINGH, G.; KUMAR, N.; VERMA, A. K. Ant colony algorithms in manets: A review. **Journal of Network and Computer Applications**, Elsevier, v. 35, n. 6, p. 1964–1972, 2012.

SLUD, E.; WEI, L. Two-sample repeated significance tests based on the modified wilcoxon statistic. **Journal of the American Statistical Association**, Taylor & Francis Group, v. 77, n. 380, p. 862–868, 1982.

TÁVORA, R. C. M. et al. Grupos de visitação na aman= um estudo de caso do problema do caixeiro viajante. [sn], 2011.

TEODOROVIC, D. et al. Bee colony optimization: principles and applications. **Neural Network Applications in Electrical Engineering, 2006. NEUREL 2006. 8th Seminar on**, Belgrade e Montenegro, Sérvia, p. 151–156, 2006.

WALDO, J. The jini architecture for network-centric computing. **Communications of the ACM**, ACM, New York, NY, USA, v. 42, n. 7, p. 76–82, 1999.

YIN, J.; YANG, X. Elqs: An energy-efficient and load-balanced queue scheduling algorithm for mobile ad hoc networks. In: IEEE. **Communications and Mobile Computing, 2009. CMC'09. WRI International Conference on**. Yunnan, China, 2009. v. 2, p. 121–126.