UNIVERSIDADE FEDERAL DE UBERLÂNDIA

FACULDADE DE ENGENHARIA ELÉTRICA

PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

ALGORITMO DE ROTEAMENTO *ANYCAST* BASEADO EM ALGORITMOS GENÉTICOS PARA REDES TOLERANTES A ATRASOS E DESCONEXÕES

Éderson Rosa da Silva

Setembro

2010

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

FACULDADE DE ENGENHARIA ELÉTRICA

PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

ALGORITMO DE ROTEAMENTO *ANYCAST* BASEADO EM ALGORITMOS GENÉTICOS PARA REDES TOLERANTES A ATRASOS E DESCONEXÕES

Éderson Rosa da Silva

Tese apresentada por Éderson Rosa da Silva à Universidade Federal de Uberlândia para obtenção do título de Doutor em Ciências, avaliado em 23/09/2010 pela banca examinadora:

Paulo Roberto Guardieiro, Dr. (UFU) – Orientador

José Ferreira de Rezende, Dr. (UFRJ)

José Marcos Silva Nogueira, Dr. (UFMG)

Gilberto Arantes Carrijo, Dr. (UFU)

Keiji Yamanaka, Dr. (UFU)

Uberlândia (MG), setembro de 2010.

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da UFU - MG, Brasil

S586a Silva, Éderson Rosa da, 1984-
Algoritmo de roteamento *Anycast* baseado em algoritmos genéticos para redes tolerantes a atrasos e desconexões [manuscrito] / Éderson Rosa da Silva. - 2010.
163 f. : il.

Orientador: Paulo Roberto Guardieiro.

Tese (doutorado) – Universidade Federal de Uberlândia, Programa de Pós-Graduação em Engenharia Elétrica.

Inclui bibliografia.

1. Internet (Redes de computação) - Teses. 2. Redes de computação (Protocolos) - Teses. I. Guardieiro, Paulo Roberto, 1952- II. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

CDU: 681.3.02 INTERN

ALGORITMO DE ROTEAMENTO *ANYCAST* BASEADO EM ALGORITMOS GENÉTICOS PARA REDES TOLERANTES A ATRASOS E DESCONEXÕES

Éderson Rosa da Silva

Tese apresentada à Universidade Federal de Uberlândia para obtenção do título de Doutor em Ciências.

Prof. Paulo Roberto Guardieiro, Dr. Orientador	Prof. Alexandre Cardoso, Dr. Coordenador do curso de Pós-Graduação

Dedicatória

Dedico este trabalho a todas as pessoas presentes em minha vida e que tanto amo, cada uma com sua parcela de contribuição, pelo carinho e incentivo recíprocos em qualquer fase de minha vida.

Agradecimentos

Agradeço, em primeiro lugar, a Deus por me dar força, perseverança, teimosia e iluminar meu caminho, tornando possível a realização de mais essa etapa de minha vida.

Aos meus familiares por todo apoio, carinho e confiança irrestritos em todos os momentos. Especialmente, aos meus pais Ismael e Elza, por todas as lições e valores passados, que não foram poucos, e que sempre me auxiliaram e continuarão presentes comigo, além dos meus irmãos Eduardo e Éverton, pela paciência e incentivo.

Ao prof. Dr. Paulo Roberto Guardieiro por toda orientação e experiência concedida, pela dedicação e incentivo para realização deste trabalho, e pelo exemplo de profissionalismo e conduta.

Aos professores e colegas da Faculdade de Engenharia Elétrica da UFU (Universidade Federal de Uberlândia) pela colaboração no aprendizado.

Aos meus amigos pela amizade e companheirismo sempre presentes.

Por fim, meu agradecimento a CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) e a FAPEMIG (Fundação de Amparo à Pesquisa do Estado de Minas Gerais) pelo auxílio financeiro na realização deste trabalho.

Publicações

A seguir são apresentadas as publicações resultantes das pesquisas realizadas no decorrer deste trabalho:

- Periódicos

SILVA, E. R.; GUARDIEIRO, P. R. Application of genetic algorithm to optimization of anycast routing in delay tolerant networks. Accepted for publication. *Journal of Communications* (JCM, ISSN 1796-2021), special issue on delay tolerant networks, architecture, and applications. Not published.

SILVA, E. R.; GUARDIEIRO, P. R. Effect of buffer size on anycast routing in delay tolerant networks. *Revista Científica Telecomunicações* (*Inatel*), vol. 12, no. 2, pp. 53-60, Dec. 2009.

SILVA, E. R.; GUARDIEIRO, P. R. An efficient genetic algorithm for anycast routing in delay/disruption tolerant networks. *IEEE Communications Letters*, vol. 14, no. 4, pp. 315-317, Apr. 2010.

- Congressos

SILVA, E. R.; GUARDIEIRO, P. R. Roteamento *anycast* em redes tolerantes a atrasos e desconexões. Em *VI Conferência de Estudos em Engenharia Elétrica*, Uberlândia, 2008.

SILVA, E. R.; GUARDIEIRO, P. R. Anycast routing in delay tolerant networks using genetic algorithm for route decision. In *11th IEEE International Conference on Computer and Information Technology* (*ICCIT 2008*), Bangladesh, pp. 65-71, Dec. 2008.

SILVA, E. R.; GUARDIEIRO, P. R. Effect of buffer size on anycast routing using genetic algorithm in delay tolerant networks. In *International Workshop on Telecommunications* (*IWT 2009*), São Paulo, pp. 123-128, 2009.

- Capítulo de Livro

SILVA, E. R.; GUARDIEIRO, P. R. Application of genetic algorithm to optimization of anycast routing in delay and disruption tolerant networks. In: VERMA, D. C.; PIETRO LIO. *Biologically Inspired Networking and Sensing: Algorithms and Architecture*. IGI Global Publisher, 2010, no prelo.

- Pedido de Patente

UNIVERSIDADE FEDERAL DE UBERLÂNDIA. Silva, E. R.; Guardieiro, P. R. *Método de roteamento anycast utilizando algoritmos genéticos em redes tolerantes a atrasos e desconexões*. BR n. PI 0905542-8 A2, 16/12/2009.

Resumo

Silva, Éderson R. *Algoritmo de Roteamento Anycast Baseado em Algoritmos Genéticos para Redes Tolerantes a Atrasos e Desconexões*, Uberlândia, Faculdade de Engenharia Elétrica – UFU, 2010.

As redes tolerantes a atrasos e desconexões ou DTNs (*Delay and Disruption Tolerant Networks*) possuem o potencial de conectar dispositivos e áreas do mundo que não são servidas por redes tradicionais. O desenvolvimento dessas redes permite levar a revolução da informação tecnológica às populações dos países em desenvolvimento carentes de infraestrutura, especialmente nas regiões remotas e rurais. Neste cenário, as DTNs contribuem oferecendo uma arquitetura alternativa de redes de baixo custo, tolerante a enlaces intermitentes com atrasos variáveis e, possivelmente, longos. Um dos principais desafios que surge no projeto de redes com essas características é o roteamento, sendo este um tópico de grande interesse e importância na área das DTNs. Atualmente, a arquitetura DTN especificada pelo grupo de pesquisa DTNRG (*DTN Research Group*) oferece uma *framework* na qual uma variedade de protocolos de roteamento podem ser utilizados, mas não define nenhum protocolo de roteamento particular. Além disso, os nós DTN, provavelmente, terão que suportar diferentes estratégias de roteamento, a fim de operar eficientemente na enorme diversidade de ambientes em que o nó pode se encontrar. Assim, neste trabalho é proposto um algoritmo de roteamento para DTNs em cenários onde a topologia da rede pode ser conhecida ao longo do tempo. Mais precisamente, o roteamento visa a entrega *anycast*, por ser um serviço ainda pouco pesquisado e que possui diversas aplicações importantes nas DTNs. O algoritmo de roteamento *anycast* para DTNs, proposto neste trabalho, utiliza algoritmos genéticos ou GAs (*Genetic Algorithms*), que possuem a capacidade de resolver problemas

complexos com múltiplos objetivos. Para aumentar o desempenho do algoritmo baseado em GA proposto, algumas estratégias, como o conceito de subpopulação e a redução do número de soluções a ser avaliado pelo algoritmo, são utilizadas. Nesta tese, é apresentado todo o ambiente de simulação desenvolvido para modelagem das características das DTNs utilizando grafos evolutivos. A proposta de algoritmo de roteamento *anycast* baseado em GA é implementada e comparada com outras estratégias, e os resultados mostram uma significante melhora no objetivo de entregar mensagens visando otimizar as medidas de desempenho da rede. Como consequência, o tráfego de mensagens é adequadamente distribuído na rede e o esquema utilizado proporciona altas taxas de entrega e atrasos limitados. Desta forma, os estudos baseados em modelagem e simulação mostram que a proposta de algoritmo de roteamento *anycast* baseado em GA conduz a bons resultados na DTN modelada por grafos evolutivos.

Palavras-chave: DTNs, roteamento *anycast*, algoritmos genéticos.

Abstract

Silva, Éderson R. *Anycast Routing Algorithm Based on Genetic Algorithms for Delay and Disruption Tolerant Networks*, Uberlândia, Faculty of Electrical Engineering – UFU, 2010.

DTNs (Delay and Disruption Tolerant Networks) have the potential to interconnect devices and areas of the world that are underserved by traditional networks. The development of these networks can lead to the revolution of the technology information for the population in developing countries which lack infrastructure, especially in remote and rural regions. In this scenario, the DTNs can help offering an alternative network architecture with low cost, tolerant to intermittent connections and having variable and possibly long delays. One of the main challenges that arises in the design of networks with these characteristics is the routing, and this is a topic of great interest and importance of DTNs subject. Currently, the DTN architecture specified by the research group DTNRG (DTN Research Group) offers a framework where a variety of routing protocols can be used, but it does not define any particular routing protocol. Moreover, DTN nodes will likely have to support a number of different routing strategies in order to operate efficiently in the diversity of environments in which the node may find itself. Thus, in this work it is proposed a routing algorithm for DTNs in scenarios where the network topology may be known ahead of time. More specifically, the routing aims anycast delivery, because it is a service that has not been very well explored yet and with many important applications in DTNs. The proposed anycast routing algorithm for DTNs makes use of GAs (Genetic Algorithms), which have the ability to solve complex problems with multiple objectives. To improve the performance of the proposed algorithm based on GA, strategies as, concept of subpopulation and reduction of the number of solutions to be evaluated, are used. In this thesis, it is presented a complete simulation environment

developed for modeling the DTNs using evolving graphs. The proposed GA-based anycast routing algorithm is implemented and compared with other strategies, and results show a significant improvement in order to deliver messages to optimize the network performance metrics. As a result, the message traffic is properly distributed in the network and the proposed scheme provides high rates of delivery and limited delays. This way, studies based on modeling and simulation show that the proposed GA-based anycast routing algorithm leads to good results in the DTNs modeled by evolving graphs.

Keywords: DTNs, anycast routing, genetic algorithms.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

AA	*Application Agent*
AAA	*Authentication, Authorization, and Accounting*
ADSL	*Asymmetric Digital Subscriber Line*
ADUs	*Application Data Units*
AODV	*Ad-hoc On-demand Distance Vector*
BP	*Bundle Protocol*
BPA	*Bundle Protocol Agent*
BSR	*Bundle Status Report*
CLA	*Convergence Layer Adapters*
CM	*Current Membership*
CMD	*Current-Member Delivery*
DARPA	*Defense Advanced Research Projects Agency*
DNS	*Domain Name System*
DSR	*Dynamic Source Routing*
DTNs	*Delay and Disruption Tolerant Networks*
DTNRG	*DTN Research Group*
EaD	Ensino à Distância
ED	*Earliest Delivery*
EID	*Endpoint Identifier*
FIB	*Forwarding Information Base*
GAs	*Genetic Algorithms*

IANA	*Internet Assigned Numbers Authority*
IC	Intervalo de Confiança
IETF	*Internet Engineering Task Force*
INPI	Instituto Nacional da Propriedade Industrial
IP	*Internet Protocol*
IPN	*InterPlaNetary Internet*
IPNSIG	*IPN Special-Interest Group*
IRTF	*Internet Research Task Force*
JPL	*Jet Propulsion Laboratory*
LTP	*Licklider Transmission Protocol*
MANETs	*Mobile Ad Hoc Networks*
MDRA	*Maximum Delivery Rate for Anycast*
MED	*Minimum Expected Delay*
MEED	*Minimal Estimated Expected Delay*
MF	*Message Ferrying*
MORA	*Multi-Objective Robotic Assistance*
MPE	Mensagens por Enlace
MRG	*Minimum Reception Group*
MSB	*Most Significant Bit*
NS	*Network Simulator*
OLSR	*Optimized Link State Routing*
ONE	*Opportunistic Network Environment*
PDUs	*Protocol Data Units*
PE	Probabilidade de Entrega

PE_{min}	Probabilidade de Entrega mínima
PROPHET	*Probabilistic Routing Protocol using History of Encounters and Transitivity*
RFCs	*Request for Comments*
RIB	*Routing Information Base*
RTT	*Round-Trip Time*
SDNVs	*Self-Delimiting Numeric Values*
SI	Sistema Internacional
SP	*Shortest Path*
SSP	*Scheme Specific Part*
TCP	*Transmission Control Protocol*
TD	*Temporal Delivery*
TIM	*Temporal Interval Membership*
TM	*Temporal Membership*
TORA	*Temporally-Ordered Routing Algorithm*
TPM	*Temporal Point Membership*
URIs	*Uniform Resource Identifiers*
UTC	*Coordinated Universal Time*

Capítulo 1

INTRODUÇÃO

Há tempos que a Internet se tornou muito popular devido a sua flexibilidade, capacidade e robustez proporcionadas pelo sucesso dos protocolos utilizados (TCP/IP). Entretanto, à medida que a Internet vem se expandindo, surgem novos desafios e grupos de trabalho interessados em propor soluções. E, neste sentido, observa-se um crescente esforço para viabilizar a comunicação em redes cujos cenários envolvem atrasos e desconexões imprevisíveis, requerendo, desta forma, uma arquitetura de redes tolerante a atrasos e desconexões ou DTNs (*Delay and Disruption Tolerant Networks*) (Fall, 2003). Uma DTN pode ser definida como uma rede de redes regionais (Warthman, 2003). Com isso, as DTNs possuem o potencial de conectar dispositivos e áreas do mundo que não são servidas por redes tradicionais, possibilitando a comunicação por levar em conta a vantagem de conexões temporárias para retransmitir dados. Essas redes estão sendo investigadas para EaD (Educação à Distância), telecomunicações, serviços governamentais, ambientes de monitoramento, comunicação veicular e redes espaciais (Jones et al., 2007).

As pesquisas na área de DTNs têm recebido muita atenção nos últimos anos. O grupo de pesquisa DTNRG (*DTN Research Group*) da IRTF (*Internet Research Task Force*) está preocupado com a maneira de tratar princípios arquiteturais e de projeto de protocolos, decorrentes da necessidade de fornecer comunicações interoperáveis com e entre ambientes

desafiadores, ou seja, onde as técnicas tradicionais não são suficientes. Além disso, o DTNRG é, atualmente, o principal espaço aberto para trabalhos sobre a arquitetura e protocolos DTN.

Alguns aspectos da evolução das DTNs são apresentados em um livro (Farrell e Cahill, 2006) que trata desse assunto. É visto que redes tradicionais, como a Internet ou as MANETs (*Mobile Ad Hoc Networks*), geralmente são redes conectadas. Diferentemente, as DTNs podem ser desconectadas devido a vários fatores, como mobilidade, interferência, faixa de cobertura de rádio limitada e mecanismos de economia de energia, fazendo com que as DTNs representem um paradigma de rede diferente das redes tradicionais. Devido às características únicas das DTNs, os algoritmos de roteamento existentes, responsáveis por tomar decisões de caminhos através dos quais as mensagens geradas pelo nó fonte deverão ser repassadas a fim de alcançar o destino, que foram desenvolvidos para Internet e MANETs, irão falhar.

Para superar as desconexões de rede que frequentemente ocorrem nas DTNs, surge um novo paradigma de roteamento chamado *store-carry-forward*, como pode ser encontrado na revisão de muitos princípios da arquitetura DTN realizada por Fall e Farrell (2008). Neste paradigma, os nós armazenam dados enquanto a rede está desconectada, e repassam os dados para outros nós quando eles se conectam novamente. Pelo transporte de dados nos nós e pela transferência dos dados quando oportunidades de transmissão se tornam disponíveis, caminhos fim a fim são possíveis entre nós fontes e destinos ao longo do tempo.

Devido à diversidade de ambientes em que um nó pode se encontrar, diferentes estratégias e protocolos de roteamento DTN têm sido propostos. Um levantamento de tais esquemas aparece em uma revisão realizada por Zhang (2006). Apesar dos vários trabalhos presentes na literatura que lidam com o roteamento nas DTNs, muitos desafios ainda precisam ser resolvidos para o uso efetivo das DTNs.

Nesta tese é discutido o roteamento com entrega *anycast* em redes DTN, por ser um tópico de pesquisa com poucos trabalhos publicados e vários desafios em aberto, além de

fornecer um serviço que permite a um nó enviar mensagens a pelo menos um, e preferencialmente apenas um, dos membros de um grupo de nós de comunicação. Desta forma, o roteamento *anycast* pode ser aplicado nas DTNs em situações de catástrofes, nas quais pessoas procuram um médico ou um bombeiro sem conhecimento de seus identificadores ou localizações precisas. O *Anycast* também pode auxiliar em serviços como EaD e telemedicina, em regiões carentes de infraestrutura de comunicação, e na localização de um membro de um dado grupo que possa conter um determinado tipo de informação de interesse de outro membro. Em vista destes aspectos, será tratado o roteamento *anycast* em DTNs utilizando informações disponíveis sobre a topologia da rede, de modo a otimizar as medidas de desempenho da rede. Diante dos vários objetivos que o algoritmo de roteamento *anycast* deve atingir, além da grande complexidade da seleção de rotas em cenários tão restritos, é proposto um algoritmo de roteamento *anycast* baseado em algoritmos genéticos ou GAs (*Genetic Algorithms*), pois GAs são apropriados para otimização de problemas complexos e são algoritmos de busca baseados nos mecanismos de seleção e evolução natural (Goldberg, 1989). A proposta será para DTNs em cenários determinísticos, com sua viabilidade sendo estudada através de modelagem e simulação.

1.1. Motivação

As DTNs possuem o potencial de conectar dispositivos e áreas do mundo que não são servidas por redes tradicionais, possibilitando a comunicação por levar em conta a vantagem de conexões temporárias para transmitir dados. Assim sendo, o desenvolvimento dessas redes permite levar a revolução da informação tecnológica às populações dos países em desenvolvimento, pois em algumas áreas desses países, as pessoas não dispõem de nenhum tipo de comunicação confiável. Nestes casos, as redes DTNs surgem como uma solução

viável para auxiliar a EaD, telemedicina, situações de catástrofes, telecomunicações, serviços governamentais, ambientes de monitoramento e comunicação veicular.

Além disso, regiões carentes de infraestrutura, como é o caso de muitas regiões remotas e rurais em países em desenvolvimento, podem fazer uso de mensageiros móveis (carros, ônibus, caminhões, motocicletas, entre outros) para transportar mensagens entre essas regiões desconectadas, sendo uma solução atrativa para viabilizar a comunicação em tal cenário desafiador. Este cenário é típico de uma rede apresentando atrasos longos e frequentes desconexões, ou seja, possui as características de uma DTN. Por isso, a modelagem adequada do problema e a investigação das questões ainda em aberto nas DTNs podem possibilitar a comunicação no cenário descrito.

Uma questão em desenvolvimento nas DTNs é o roteamento, pois é preciso projetar protocolos capazes de superar os problemas dos atrasos extremamente longos e das frequentes desconexões. Diversos protocolos de roteamento foram especialmente projetados para DTNs. Apesar da maioria das propostas de roteamento realizarem entrega *unicast*, a utilização da entrega *anycast* pode ser vantajosa e mais apropriada para algumas situações, pois aproveita a oportunidade de transmitir para um, possivelmente o destino que apresentar as melhores condições de comunicação, dentre um grupo de possíveis nós. Embora seja um tópico de pesquisa com poucos trabalhos publicados (quando comparado ao *unicast*), é visto que o roteamento *anycast* (transmissão de um para qualquer um de um grupo) possui diversas aplicações em DTNs, além de permitir a comunicação em cenários nos quais muitas vezes o serviço *unicast* (transmissão de um para um) seria inviável.

Os algoritmos de roteamento que utilizam informações disponíveis a respeito da rede podem ser mais eficientes, pois permitem que as decisões de rotas não sejam tomadas ao acaso. Por isso, deve ser verificado se informações sobre a topologia da rede e características dos nós que a compõe podem auxiliar no roteamento. Além disso, a combinação do tráfego e

as restrições de armazenamento dos nós na rede, por serem fatores que tornam a seleção de rotas mais complexa, quando considerados pelo algoritmo de roteamento, podem produzir resultados mais satisfatórios. Neste contexto, são necessários novos algoritmos de roteamento *anycast* capazes de lidar com o dinamismo de uma DTN e acomodar o tráfego de maneira adequada utilizando informações disponíveis para prever as melhores rotas, sendo uma alternativa a questão ainda em aberto do roteamento em DTNs.

1.2. Objetivos e contribuições esperadas

Esta pesquisa consiste em, a partir do estudo sobre o estado atual das DTNs e do serviço *anycast*, desenvolver um algoritmo de roteamento *anycast* para realizar o roteamento em DTNs que possuam suas características representadas através de grafos. O algoritmo de roteamento deverá levar em consideração as restrições de tráfego na rede para efetuar decisões de seleção de rotas, visando aproveitar ao máximo as raras oportunidades de transmissão encontradas nas DTNs. Para atingir o objetivo proposto, as seguintes metas são executadas:

– Definição da técnica utilizada para modelar a DTN que permitirá representar os ambientes desafiadores da rede;

– Definição da semântica de serviço *anycast*, que devido às características das DTNs, deve ser diferente do *anycast* nas redes tradicionais;

– Apresentação de uma proposta de algoritmo de roteamento *anycast* em DTNs, que otimize as medidas de desempenho da rede na decisão de seleção de rotas;

– Ciente das condições acima, definir a ferramenta de avaliação mais adequada para analisar a solução proposta para o problema;

– Avaliação do algoritmo de roteamento proposto utilizando vários cenários e desafios, além de compará-lo com outras abordagens, a fim de demonstrar sua viabilidade.

Espera-se assim, apresentar, analisar e mostrar a viabilidade de uma proposta de algoritmo de roteamento *anycast* utilizando estratégias que ainda não foram aplicadas nas DTNs. Após todas as etapas citadas serem executadas, algumas questões para o roteamento *anycast* em DTNs devem ser esclarecidas, como a incorporação de restrições de armazenamento presentes na rede e otimização de medidas de desempenho do algoritmo de roteamento proposto.

1.3. Estrutura da tese

O Capítulo 2 apresenta as principais características e a arquitetura das redes DTN. Com base nessa arquitetura, a inserção de uma sobrecamada é descrita. Além disso, questões em aberto na arquitetura são citadas.

O Capítulo 3 contém um levantamento da literatura na área de roteamento em DTNs, sendo discutidas pesquisas relacionadas ao roteamento nas redes DTNs baseadas em diferentes abordagens. Além disso, como é visado o roteamento *anycast* nas DTNs em cenários determinísticos, o Capítulo 3 apresenta aproximações e considerações a respeito da utilização de grafos evolutivos para representar as características das DTNs, da semântica de entrega *anycast*, além do uso de algoritmos baseados no algoritmo de Dijkstra para cálculo de rotas *anycast* com o menor custo.

O Capítulo 4 descreve o algoritmo de roteamento *anycast* baseado em GA para redes DTN, proposto neste trabalho. São apresentadas as etapas de desenvolvimento do algoritmo proposto, bem como aproximações utilizadas visando obter o melhor desempenho do mesmo.

O Capítulo 5 apresenta detalhes do ambiente de simulação criado neste trabalho, descrevendo as principais características do mesmo. Em seguida, os parâmetros de simulação utilizados são definidos.

O Capítulo 6 analisa o algoritmo de roteamento proposto por meio de modelagem e simulação de cenários específicos e variados, e avalia a utilização de diferentes estratégias visando melhorar o desempenho da proposta de algoritmo de roteamento *anycast* baseado em GA.

O Capítulo 7 descreve as conclusões do trabalho, avaliando e discutindo as contribuições e resultados desta pesquisa. Por fim, possíveis direções para trabalhos futuros, também são destacadas.

Capítulo 2

REDES TOLERANTES A ATRASOS E DESCONEXÕES

2.1. Introdução

A arquitetura da Internet de comprovado sucesso, utilizada no mundo todo para interconectar os mais variados tipos de dispositivos de comunicação, em diferentes cenários e dando suporte a diversas aplicações, impõe algumas condições necessárias ao seu bom funcionamento. Como exemplos dessas condições, citam-se a existência de um caminho fim a fim entre a fonte e o destino durante uma sessão de comunicação, uma perda de pacotes fim a fim relativamente pequena, que todos os roteadores e estações finais suportem protocolos TCP/IP, que a seleção de uma única rota entre a fonte e destino seja suficiente para atingir desempenho aceitável, entre outros.

Entretanto, com a disseminação de novas tecnologias de comunicação e computação, surge também a necessidade de uma nova geração de redes de comunicação capaz de aumentar o alcance da comunicação e, possibilitar novos tipos de aplicações e serviços. Infelizmente, esses novos ambientes apresentam grandes desafios na necessidade de conectar dispositivos com diversas características, operando em ambientes incomuns.

Assim sendo, enviar dados sobre uma rede de computadores convencional pode ser difícil, especialmente se a rede sofre atrasos e desconexões significantes entre seus nós de comunicação. Esta visão foi inicialmente cultivada e promovida por um pequeno grupo de pessoas no JPL (*Jet Propulsion Laboratory*), que iniciaram o desenvolvimento da IPN (*InterPlaNetary Internet*) em 1998. A ideia básica da IPN é possibilitar a comunicação entre regiões remotas do espaço.

Eventualmente, surgiu o IPNSIG (*IPN Special-Interest Group*) que desenvolveu uma arquitetura para uma rede de larga escala e realizou alguns progressos. Todavia, o IPNSIG tinha um problema: a dificuldade de implementar uma rede interplanetária, por possuir custos elevados para ser criada.

Ao mesmo tempo, outros pesquisadores passaram a investigar como os conceitos da IPN poderiam ser utilizados em aplicações terrestres. Em particular, as redes de sensores sem fio acabaram por ter muito em comum com a IPN. Dado que experimentos em uma rede de sensores são muito mais fáceis de serem realizados (e baratos) que no caso do IPNSIG, a IRTF criou um novo grupo de pesquisa para examinar a área de redes tolerantes a atrasos e desconexões ou DTN. Esse grupo é denominado DTNRG, e é atualmente, o principal local para trabalhos sobre a arquitetura e protocolos DTN. O DTNRG documenta esses protocolos em RFCs (*Request for Comments*).

Em 2004, a DARPA (*Defense Advanced Research Projects Agency*) realizou uma chamada de trabalhos em DTNs. Assim, vários grupos estão trabalhando com o objetivo comum de desenvolver protocolos DTN. A Figura 2.1 (Farrell et al., 2006) ilustra esses grupos.

O restante do capítulo, fundamentalmente, descreve as posições do DTNRG por ser o único grupo aberto do diagrama da Figura 2.1. O DTNRG desenvolve dois protocolos

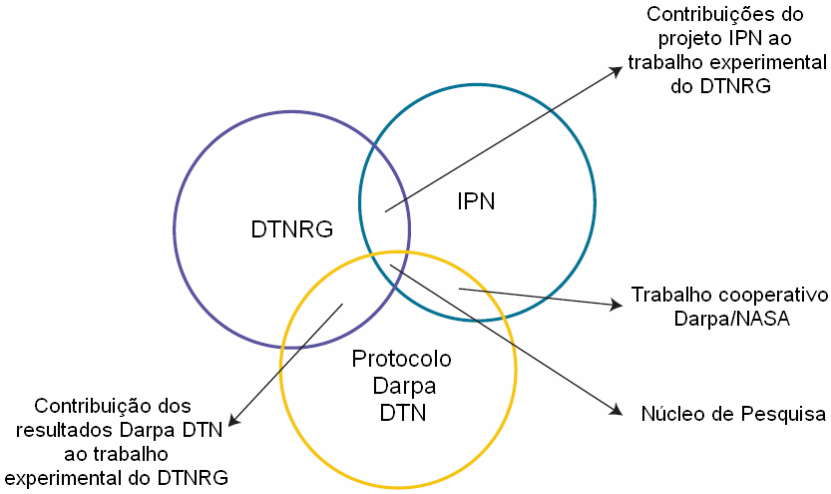

Figura 2.1. Diagrama de organizações que estão tentando resolver as questões das DTNs.

principais, o protocolo de agregação ou BP (*Bundle Protocol*) e o LTP (*Licklider Transmission Protocol*). Grande parte das informações contidas nesta tese está em conformidade com os RFCs 4838 (Cerf et al., 2007) e 5050 (Scoot e Burleigh, 2007) que representam a arquitetura DTN e o protocolo de agregação, respectivamente, desenvolvidos pelo DTNRG. O LTP é um protocolo ponto a ponto que pode servir como um protocolo da camada de convergência (responsável por possibilitar o envio e recebimento de agregados em favor de nós DTN, utilizando os serviços de algum protocolo da Internet nativo) subjacente ao BP. Além disso, o LTP é projetado para fornecer confiabilidade baseada em retransmissão em enlaces caracterizados por longos tempos de ida e volta para transmissão de mensagens e/ou frequentes interrupções na conectividade. Mais informações sobre o LTP podem ser encontradas em três RFCs: 5325 (Burleigh et al., 2008), 5326 (Ramadas et al., 2008) e 5327 (Farrell et al., 2008) tratando da motivação, especificação e extensões de segurança do LTP, respectivamente.

Portanto, este capítulo apresenta as características das DTNs. Inicialmente, a Seção 2.2 descreve a arquitetura DTN contida no RFC 4838. A Seção 2.3 apresenta brevemente o protocolo de agregação para DTNs, especificado no RFC 5050. As Seções 2.4 e 2.5 lidam sucintamente com os componentes conceituais de um nó DTN e as questões de segurança nas

DTNs, respectivamente. Finalmente, a Seção 2.6 apresenta as considerações finais do capítulo.

2.2. Arquitetura DTN

Esta seção apresenta uma descrição da maioria das características da arquitetura DTN resultante de decisões de projeto guiadas pelos princípios mencionados na seção anterior.

A arquitetura DTN discute muitos problemas de redes heterogêneas que devem operar em ambientes sujeitos a atrasos longos e conectividade fim a fim descontínua. Com isso, protocolos interativos que trocam informações, tais como o TCP, requerem o envio de várias mensagens fim a fim, seja para estabelecimento e encerramento de uma conexão, ou para o envio de dados. Os protocolos com essas características podem falhar completamente, quando sujeitos a RTTs (*Round-Trip Times*) de longa duração, isto é, em cenários nos quais o tempo gasto para se enviar uma mensagem de um nó fonte para um nó destino, e receber uma resposta desse nó destino, é muito grande. Por esta razão, os nós DTN conectados por uma rede caracterizada por atrasos longos e, possivelmente, altamente variáveis, comunicam-se utilizando sessões simples com o objetivo de minimizar trocas de mensagens, como por exemplo, para estabelecimento e encerramento de conexões, ou para o reconhecimento do envio dos dados. Por isso, qualquer reconhecimento a ser enviado pelo nó receptor é opcional. Com base nesses aspectos, uma única mensagem solicitando transferência de arquivo, por exemplo, pode incluir informação de autenticação (assim “agregando” esta informação em uma única mensagem).

Assim sendo, a arquitetura DTN define uma sobrecamada (*overlay*) denominada camada de agregação (*bundle layer*). A camada de agregação (camada comum a todos os nós que implementam a arquitetura DTN) localiza-se abaixo da camada de aplicação, e acima das demais (esses protocolos de camadas inferiores são específicos de cada região), como

ilustrado na Figura 2.2 (Oliveira et al., 2007). Esta disposição permite que as aplicações não precisem ser desenvolvidas levando em conta os problemas de atraso e desconexões, e torna a DTN independente das diversas redes regionais. Como nem todos os protocolos que operam abaixo da camada de agregação oferecem as mesmas funcionalidades, são necessárias camadas de convergência responsáveis por gerenciar os detalhes específicos dos protocolos através de uma interface com os protocolos das camadas inferiores e de uma interface consistente com a camada de agregação.

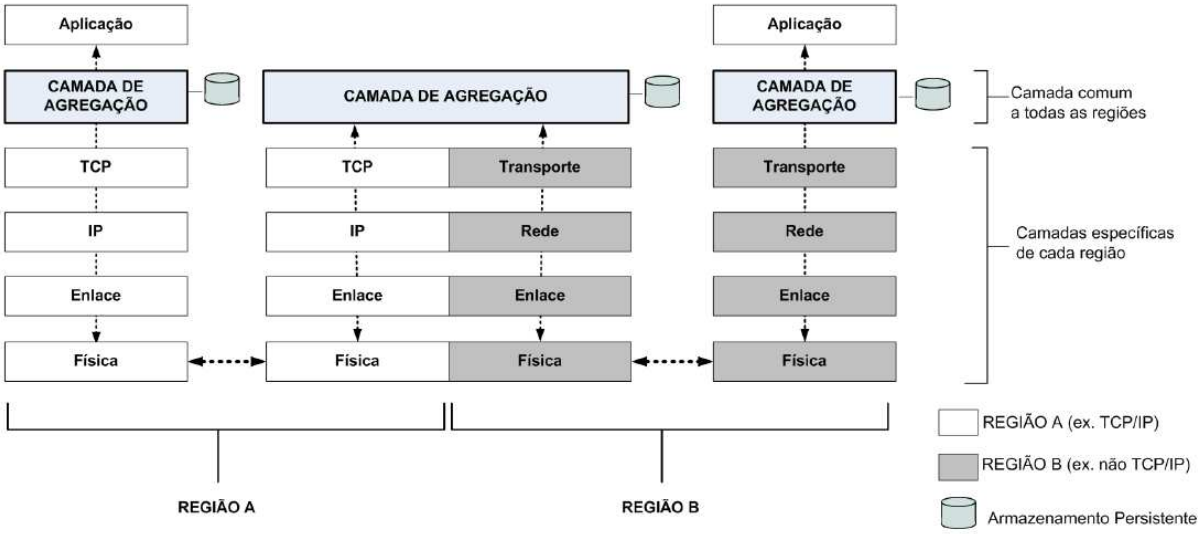

Figura 2.2. Camada de agregação.

Uma aplicação habilitada para DTN envia mensagens de comprimento arbitrário, também chamadas de unidades de dados da aplicação ou ADUs (*Application Data Units*). As ADUs são transformadas pela camada de agregação em uma ou mais unidades de dados de protocolo ou PDUs (*Protocol Data Units*) chamadas “agregados”, que são repassados pelos nós DTN. Os agregados possuem um formato definido contendo dois ou mais “blocos” de dados, que são descritos na Seção 2.2.3. Além disso, um agregado pode ou não ser fragmentado, isto é, ser dividido em tamanhos menores.

Na arquitetura DTN, a técnica de comutação de mensagens não estabelece nenhuma conexão com antecedência entre a origem e o destino, não existindo qualquer fase anterior ao

envio de dados. Quando uma mensagem precisa ser enviada, ela é armazenada e encaminhada nó a nó desde a origem até o destino.

A rede IP é baseada na operação "*store-and-forward*", existindo uma suposição que o "armazenamento" não irá persistir por mais que um modesto intervalo de tempo, na ordem dos atrasos de fila e de transmissão. Em contraste, a arquitetura DTN não espera que *edges* ou enlaces de rede estejam sempre disponíveis e confiáveis, e ao invés disso, espera que os nós possam escolher armazenar agregados por algum tempo. Tais fatores sugerem que os nós devem utilizar alguma forma de armazenamento persistente, podendo operar utilizando alguma combinação simples de repasse e uma operação tolerante a atraso de "*store-carry-forward*" (Fall e Farrell, 2008).

Para uma rede suportar eficientemente a arquitetura DTN algumas condições devem ser consideradas:

1. O armazenamento deve estar disponível e bem distribuído em toda a rede;

2. O armazenamento deve ser suficientemente persistente e robusto para armazenar agregados até que o repasse possa ocorrer.

Assim sendo, o armazenamento nos nós, na essência, representa um novo recurso que deve ser gerenciado e protegido. Muitas das pesquisas em DTN giram em torno da exploração destas questões.

2.2.1. *Endpoint* e vinculação

Aplicações utilizam nós DTN para enviar ou receber ADUs transportadas em agregados, podendo os mesmos serem membros de grupos denominados pontos de extremidade DTN ou *endpoints DTN*. Um *endpoint* DTN é consequentemente um conjunto de zero ou mais nós DTN, e um nó pode pertencer a mais de um *endpoint* (Figura 2.3). Considera-se que um agregado é entregue com sucesso para um *endpoint* DTN quando algum subconjunto mínimo

de nós no *endpoint* recebe o agregado sem erro. Este subconjunto é denominado de grupo de recepção mínimo ou MRG (*Minimum Reception Group*) do *endpoint*. O MRG de um *endpoint* pode referir-se a um nó (*unicast*), um nó de um grupo de nós (*anycast*), ou todos os nós de um grupo de nós (*multicast* e *broadcast*) (Figura 2.4).

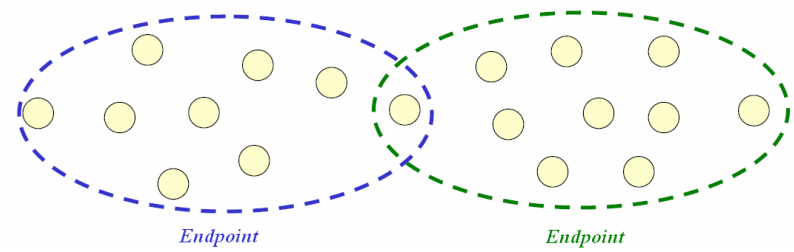

Figura 2.3. *Endpoint* DTN.

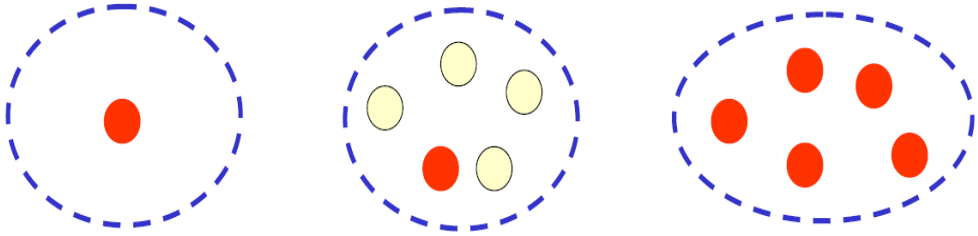

Figura 2.4. Semânticas de entrega DTN: *unicast*, *anycast* e *multicast*.

Uma estrutura de nomeação adequada é um aspecto fundamental de uma arquitetura de rede. Normalmente, nomes são *strings* de comprimento variável (útil para pessoas e organizações) e endereços são identificadores de tamanhos fixos (útil para operações de rede e roteamento). Além disso, os nós podem ter múltiplas interfaces de rede e serem móveis, requerendo flexibilidade adicional na maneira de nomear o nó. Assim, reconhecendo que os nós podem requerer identificadores múltiplos e mesmo múltiplos tipos de identificadores, uma estrutura de nomeação foi buscada para ser capaz de codificar nomes e endereços de "*namespace*" múltiplos diferentes.

A IETF (*Internet Engineering Task Force*) já havia tratado de sistemas de nomes generalizados na forma de URIs (*Uniform Resource Identifiers*). Embora sejam mais complexos do que o requerido pelas DTNs, possuem propriedades importantes, como um

conjunto de nomes de esquemas gerenciados globalmente pela IANA (*Internet Assigned Numbers Authority*), comprimento variável e obedecem a uma estrutura e semântica geral (Figura 2.5).

<nome do esquema> : <parte específica do esquema>

Figura 2.5. Formato URI.

Assim, um *endpoint* é identificado por um identificador de *endpoint* ou EID (*Endpoint Identifier*) que corresponde a um nome expresso sintaticamente como um URI. A sintaxe geral dos URIs é útil para registrar nomes de *endpoints* que formam as DTNs. Como nomes, os EIDs não são obrigatoriamente relacionados à organização topológica ou ao roteamento. Entretanto, tal relacionamento não é proibido e em alguns ambientes utilizando EIDs esta maneira pode ser vantajosa.

A manifestação do desejo da aplicação de receber ADUs destinadas para um nó DTN identificado por um EID particular é denominada de registro (*registration*) e, em geral, é mantida persistentemente pelo nó DTN. A tentativa de estabelecer um registro não tem garantia de sucesso. Por exemplo, uma aplicação pode requisitar registrar-se para receber ADUs pela especificação de um ID do *endpoint* que não é interpretada e nem disponível para o nó DTN que serve o pedido.

Uma característica fundamental nas DTNs é o suporte a vinculação tardia, o que significa interpretar a parte específica do esquema ou SSP (*Scheme Specific Part*) de um EID, com o objetivo de encaminhar uma mensagem em direção ao(s) seu(s) destinatário(s), em trânsito ou no destinatário. Neste caso, a vinculação não ocorre necessariamente no nó fonte, isto é, no momento em que o nó de origem envia uma mensagem para o destino, como é o caso da vinculação antecipada que ocorre na Internet, na qual se consulta um servidor DNS (*Domain Name System*) para fixar no *host* emissor o endereço IP do nó destino antes que o dado seja enviado. Assim, é importante prover uma maior flexibilidade nas vinculações em

uma rede com atrasos longos e frequentes desconexões, pois a localização associada a um nó móvel de destino pode ser diferente no momento que a mensagem chega ao destino, em relação ao momento que foi enviada pelo nó fonte. Neste cenário, a vinculação tardia pode ser vantajosa devido à possibilidade do tempo em trânsito de uma mensagem exceder o tempo de validade de uma vinculação, fazendo a vinculação no nó fonte impossível ou inválida.

As DTNs suportam tanto vinculação tardia como antecipada, dependendo do esquema de nomes sendo utilizado. Além disso, um único EID pode referir-se a um *endpoint* contendo mais que um nó DTN, sendo responsabilidade do projetista de esquema definir como interpretar a SSP de um EID.

2.2.2. Opções de entrega e registros administrativos

A arquitetura DTN permite a utilização de várias opções de entrega. Essas opções são determinadas pela aplicação que, ao enviar uma ADU, pode requisitar qualquer combinação das opções de entrega disponíveis, que são levadas juntamente com cada agregado produzido pela camada de agregação. Têm-se definidas oito opções de entrega, com a descrição de cada opção podendo ser encontrada no RFC 4838 (Cerf et al., 2007), sendo as quatro últimas concebidas para fins de diagnósticos:

- Pedido de transferência de custódia;
- Pedido de aceitação de custódia pelo nó fonte;
- Notificação de entrega do agregado;
- Notificação de reconhecimento do agregado pela aplicação;
- Notificação de agregado recebido;
- Notificação de aceitação de custódia;
- Notificação de repasse de agregado;

– Notificação de agregado apagado.

Se os procedimentos de segurança definidos na especificação do Protocolo de Segurança de Agregação (Symington et al., 2006) também estiverem habilitados, então três opções de entrega adicionais tornam-se disponíveis: solicitação de confidencialidade, solicitação de autenticação e solicitação de detecção de erro.

Já os registros administrativos são usados para notificar informações de estado ou condições de erros relacionados à camada de agregação. A arquitetura define dois tipos de registros administrativos: relatório sobre o estado do agregado ou BSR (*Bundle Status Report*) e sinalização de custódia. Esses registros são agregados que fornecem informação sobre a entrega de outros agregados, e são utilizados em conjunto com as opções de entrega.

A sinalização de custódia é definida atualmente para indicar os estados de uma transferência de custódia. Esses estados são enviados por um nó DTN para o EID do custódio atual incluído em um agregado. Custódios são os nós que aceitam transferência de custódia (Oliveira et al., 2007). A Seção 2.2.6 descreve a função da transferência de custódia.

Os BSRs (relatórios que informam o estado do agregado) definidos atualmente são: recepção do agregado, aceitação da custódia, repasse do agregado, agregado apagado, entrega do agregado e de reconhecimento pela aplicação.

2.2.3. Agregados

Os agregados transportados através de nós DTN obedecem a um protocolo de agregação BP padrão especificado no RFC 5050 (Scott e Burleigh, 2007).

Cada agregado deve ser uma sequência concatenada de pelo menos duas estruturas de blocos. O primeiro bloco na sequência deve ser um bloco de agregado primário (contém as informações básicas necessárias para encaminhar um agregado até o destino), e nenhum agregado pode ter mais que um bloco de agregado primário. Outros tipos de blocos adicionais

do protocolo de agregação podem seguir o bloco primário para suportar extensões do protocolo de agregação, como por exemplo, o Protocolo de Segurança de Agregação (Symington et al., 2007). Em mais um dos blocos na sequência pode estar um bloco de *payload* (Figura 2.6).

Figura 2.6. Estrutura de blocos.

Diversos campos que não possuem tamanho fixo utilizam a codificação denominada valores numéricos autodelimitantes ou SDNVs (*Self-Delimiting Numeric Values*). A codificação SDNV e cada bloco são descritos na Seção 2.3.1.

2.2.4. Roteamento e repasse

Um desafio comum às DTNs é o roteamento, pois devido aos problemas de atrasos longos e imprevisíveis, além das frequentes desconexões, os protocolos convencionais não funcionam bem em tais redes. Assim, têm-se muitas propostas de esquemas de roteamento diferentes, sendo esta área, uma área aberta para discussões, com o grupo de pesquisa DTNRG não definindo a opção por nenhum protocolo de roteamento particular.

O roteamento refere-se à execução de um (possivelmente distribuído) algoritmo para calcular caminhos de acordo com alguma função objetivo. O roteamento faz uso do estado de roteamento (RIB, ou base de informação de roteamento).

Já o repasse refere-se à ação de mover um agregado de um nó DTN para outro. O repasse faz uso do estado derivado do roteamento, e é mantido como estado de repasse (FIB, ou base de informação de repasse).

Devido à possibilidade dos nós em uma rede DTN serem interconectados utilizando mais de um tipo de tecnologia de rede subjacente, uma rede DTN pode ser descrita abstratamente

utilizando um multigrafo (um grafo onde vértices podem ser interconectados com mais de um *edge*). Quando caminhos de entrega através de um grafo DTN são perdidos ou intervalos de contatos (corresponde a uma ocasião favorável para os nós trocarem dados) e volumes (o conceito de volume é apresentado na próxima seção) não são conhecidos precisamente ao longo do tempo, o cálculo de roteamento torna-se especialmente desafiador.

Nesta tese, os termos *edges* e enlaces são utilizados ao se tratar de grafos de conexão (*edges* unem vértices) e da rede DTN (enlaces unem nós), respectivamente, ou seja, embora possuam o mesmo sentido, são utilizados de acordo com o contexto. Em um multigrafo, cada *edge* (ou enlace) representa uma conexão entre vértices (ou nós), possuindo capacidade e atraso de propagação variáveis ao longo de tempo que indicam as propriedades da conexão. Assim, as decisões de roteamento variam com o tempo. A Figura 2.7 mostra um grafo DTN com três *edges* (e_1, e_2 e e_3) entre os vértices A e B. O intervalo de tempo em que o contato está disponível em cada *edge* é indicado entre colchetes, por exemplo, para o *edge* e_1, dois contatos, contato1 e contato2, estão disponíveis entre os intervalos de tempo [5,10] e [50,60], respectivamente. Supondo que o objetivo seja minimizar o atraso de transferência da mensagem, ignorando os atrasos de transmissão e propagação no *edge*, pelo grafo, uma mensagem que chega em A no tempo 0, deve ser enviada a B via contato1 do *edge* e_1. Entretanto, se a mensagem chega em A no tempo 15, a melhor rota deve ser via contato3 do *edge* e_2.

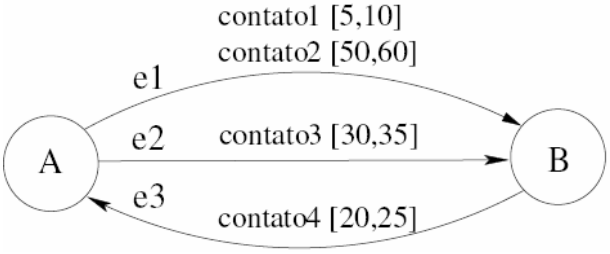

Figura 2.7. Exemplo de grafo DTN.

Devido à abrangência e complexidade do roteamento em DTNs, no Capítulo 3 são apresentados os tipos de contatos e de cenários possíveis em DTNs, assim como várias estratégias de roteamento.

2.2.5. Fragmentação e reagrupamento

Devido aos enlaces (*edges*) poderem variar entre capacidades positivas e zero, é possível descrever um período de tempo (intervalo) durante o qual a capacidade é estritamente positiva, podendo o atraso e a capacidade serem considerados constantes (Alonso e Fall, 2003). Este período de tempo é denominado “contato”. Além disso, o produto da capacidade pelo “contato” é conhecido como “volume” do contato.

Fragmentação e reagrupamento DTN são projetados para melhorar a eficiência da transferência de agregados assegurando-se que volumes de contatos sejam completamente utilizados e, evitando-se retransmissões de agregados parcialmente repassados.

Existem duas formas de fragmentação/remontagem DTN:

– A fragmentação pró-ativa é usada essencialmente quando volumes de contatos são conhecidos (ou previsíveis) com antecedência, permitindo que um nó DTN divida um bloco de dados da aplicação em múltiplos blocos menores. Neste caso, o(s) destinatário(s) final(is) deve(m) realizar o reagrupamento dos blocos de dados, formando o agregado original;

– Na fragmentação reativa o processo de fragmentação ocorre após uma tentativa de transmissão. Assim sendo, nós que compartilham um enlace em um grafo DTN podem fragmentar cooperativamente um agregado quando este for transferido apenas parcialmente. O emissor do salto ou *hop* anterior pode aprender, através de protocolos da camada de convergência, que somente uma parte do agregado foi entregue para o próximo salto. Por exigir certo nível de protocolos subjacentes, não é requerido que a capacidade de fragmentação reativa esteja disponível em implementações DTN.

Para suportar fragmentação DTN é requerida a capacidade de reagrupar fragmentos no destino.

2.2.6. Classes de prioridades, transferência de custódia e controles de congestionamento e de fluxo

O serviço mais básico fornecido pela camada de agregação é a entrega de mensagens *unicast* priorizadas (mas não garantidas) sem reconhecimento. A arquitetura DTN oferece capacidade de classes de prioridade (baixa (*bulk*), normal (*normal*) e a expressa (*expedited*)). A classe de prioridade de um agregado é requerida apenas para relacionar agregados do mesmo nó fonte. Dependendo da política de repasse/escalonamento de um nó DTN particular, a prioridade pode ou não ser forçada através de diferentes nós fontes. Isto significa que em alguns nós DTN, agregados expressos podem sempre ser enviados antes que qualquer agregado da classe baixa, independentemente da fonte.

Também são fornecidas duas opções para otimizar a confiabilidade da entrega: reconhecimentos fim a fim e transferência de custódia. Aplicações podem utilizar reconhecimentos fim a fim quando as mesmas desejarem implementar mecanismos de confiabilidade fim a fim.

A transmissão de agregados com a opção de pedido de transferência de custódia especificada envolve a transferência da responsabilidade para confiabilidade da entrega de um agregado da ADU entre diferentes nós DTN na rede. A transferência de custódia permite ao nó fonte delegar a responsabilidade de transmissão e recuperar seus recursos relacionados à retransmissão de forma relativamente breve após enviar um agregado. De acordo com a arquitetura DTN, nem todos os nós são requeridos para aceitar transferência de custódia, uma vez que não é um mecanismo salto a salto (*hop-by-hop*) verdadeiramente.

Quando uma aplicação solicita que uma ADU seja entregue com transferência de custódia, o pedido é consultivo. Além disso, em uma rede DTN onde um ou mais saltos custódio a custódio são estritamente unidirecionais, a impossibilidade de receber uma sinalização de custódia afeta o mecanismo de transferência de custódia, podendo resultar na expiração do agregado. Para ajudar a aperfeiçoar a informação, o nó na entrada de um salto unidirecional notifica a existência de um caminho unidirecional conhecido utilizando uma variante de uma notificação de estado do agregado.

De acordo com a especificação da arquitetura DTN no RFC 4838 (Cerf et al., 2007), define-se o controle de fluxo como um meio de garantir que a taxa média em que um nó emissor transmite dados para um nó receptor, não exceda a taxa média na qual o nó receptor está preparado para receber dados daquele emissor. Já o controle de congestionamento é definido como um meio de assegurar que a taxa agregada na qual todas as fontes de tráfego injetam dados dentro de uma rede, não exceda a taxa agregada máxima na qual a rede pode entregar dados para nós de destino ao longo do tempo.

Entretanto, a questão dos controles de congestionamento e de fluxo na camada de agregação é uma das questões que os autores do RFC 4838 (Cerf et al., 2007) ainda não atingiram um consenso completo. Além disso, o projeto e significado exato da transferência de custódia para entregas *multicast* e *anycast* ainda devem ser mais explorados.

A importância da transferência de custódia e o comportamento do congestionamento DTN, provavelmente, só serão completamente entendidos quando a arquitetura DTN for mais amplamente empregada e transportar cargas de tráfego significantes.

2.3. Protocolo de agregação

Nesta seção, são descritos os princípios fundamentais da versão 6 do protocolo de agregação em conformidade com o RFC 5050 (Scoot e Burleigh, 2007).

2.3.1. Formato do agregado e codificação SDNV

Cada agregado deve ser uma sequência concatenada de pelo menos duas estruturas de blocos. O primeiro bloco na sequência deve ser um bloco de agregado primário, e todo agregado contém apenas um bloco de agregado primário. Outros tipos de blocos adicionais do protocolo de agregação podem seguir o bloco primário para suportar extensões do protocolo de agregação. Em um dos blocos na sequência pode estar um bloco de *payload*. O último bloco na sequência deve ter a *flag* "último bloco" (na *flag* de controle de processamento de bloco) ajustado para um. Para cada outro bloco no agregado após o bloco primário, esta *flag* deve ser ajustada para zero.

Diversos campos que não possuem tamanho fixo utilizam a codificação denominada SDNV. O SDNV é um valor numérico codificado em N octetos, com o último octeto possuindo seu bit mais significativo ou MSB (*Most Significant Bit*) ajustado para zero e o MSB de todos os outros octetos no SDNV possuindo o valor um. O valor codificado em SDNV é um número binário sem sinal obtido pela concatenação de um bit indicando o fim ou a continuação do campo e sete bits de valores, como mostrado na Figura 2.8:

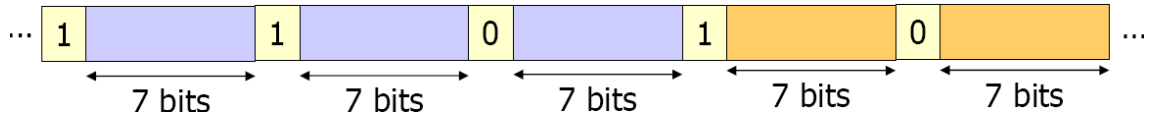

Figura 2.8. Codificação SDNV.

Exemplo do esquema de codificação SDNV:

0x4234: 01*00 0010 0***011 0100**

é codificado como:

{1 0000001} {1 *0000100*} {0 **0110100**}

= 10000001 *10000100* **00110100**

Assim, a codificação SDNV não é indicada para valores tipicamente maiores do que 56 bits, em função da sobrecarga gerada pela adição de um bit a cada sete bits. Em geral, acredita-se que a representação SDNV de valores numéricos em blocos de agregado produz o menor comprimento de bloco sem sacrificar a escalabilidade.

Os formatos dos dois blocos BP básicos, bloco primário e de *payload*, são mostrados nas Figuras 2.9 e 2.10, respectivamente.

<table>
<tr><td>Versão</td><td colspan="2">Bits de sinalização de controle de processamento (*)</td></tr>
<tr><td colspan="3">Comprimento do bloco (*)</td></tr>
<tr><td colspan="2">Desloc. do esquema do destino (*)</td><td>Desloc. da SSP do destino (*)</td></tr>
<tr><td colspan="2">Desloc. do esquema da fonte (*)</td><td>Desloc. da SSP da fonte (*)</td></tr>
<tr><td colspan="2">Desloc. do esquema do reportar-para (*)</td><td>Desloc. da SSP do reportar-para (*)</td></tr>
<tr><td colspan="2">Desloc. do esquema do custódio (*)</td><td>Desloc. da SSP do custódio (*)</td></tr>
<tr><td colspan="3">Tempo da estampa de tempo de criação (*)</td></tr>
<tr><td colspan="3">Número de sequência da estampa de tempo de criação (*)</td></tr>
<tr><td colspan="3">Tempo de vida (*)</td></tr>
<tr><td colspan="3">Comprimento do dicionário (*)</td></tr>
<tr><td colspan="3">Matriz (array) de bytes do dicionário (variável)</td></tr>
<tr><td colspan="3">Deslocamento do fragmento (*)</td></tr>
<tr><td colspan="3">Comprimento total da unidade de dados da aplicação (*)</td></tr>
</table>

Figura 2.9. Campos do bloco primário.

<table>
<tr><td>Tipo de bloco</td><td>Bits de sinalização de controle de processamento (*)</td><td>Comprimento do bloco (*)</td></tr>
<tr><td colspan="3">Carga útil (variável)</td></tr>
</table>

Figura 2.10. Campos do bloco de *payload*.

(*) indica que é um SDNV

Campos do bloco primário:

Versão: campo de um byte que indica a versão do protocolo de agregação que construiu esse bloco. O RFC 5050 (Scoot e Burleigh, 2007) descreve a versão 0x06 do protocolo de agregação.

O segundo campo é o de bits de sinalização de controle de processamento (Figura 2.11).

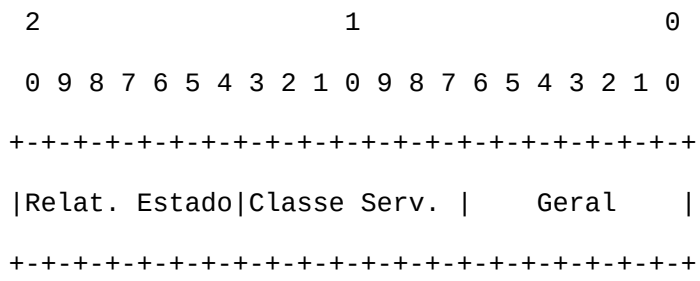

Figura 2.11. Bits de sinalização de controle de processamento.

Os bits das posições 0 a 6 caracterizam o agregado como:

0 – o agregado é um fragmento

1 – a unidade de dados da aplicação corresponde a um registro administrativo

2 – o agregado não deve ser fragmentado

3 – a transferência de custódia é solicitada

4 – o *endpoint* de destino contém apenas um nó

5 – a aplicação requer reconhecimento

6 – reservado para uso futuro

Os bits das posições 7 até 13 são utilizados para indicar a classe de serviço dos agregados. Os bits nas posições 7 e 8 constituem o campo de prioridade: 00 = baixa, 01 = normal, 10 = expressa e 11 são reservados para uso futuro. Os bits das posições 9 a 13 são reservados para uso futuro.

Os bits das posições 14 até 20 são *flags* de pedido de relatório de estado:

14 – solicitação de relatório de recepção do agregado

15 – solicitação de aceitação de custódia

16 – solicitação de encaminhamento de agregado

17 – solicitação de entrega do agregado

18 – solicitação de apagamento do agregado

19 e 20 – reservados para uso futuro

Comprimento do bloco: contém o comprimento do agregado de todos os campos restantes do bloco.

Deslocamento do esquema/SSP de destino: o campo deslocamento do esquema/SSP de destino contém o deslocamento dentro do *array* de bytes do dicionário do nome de esquema/SSP do ID de *endpoint* do destinatário do agregado, isto é, o *endpoint* contendo o(s) nó(s) para o(s) qual(is) o agregado deve ser entregue.

Deslocamento do esquema/SSP da fonte: o campo deslocamento do esquema/SSP da fonte contém o deslocamento dentro do *array* de bytes do dicionário do nome do esquema/SSP do ID de *endpoint* da fonte nominal do agregado.

Deslocamento do esquema/SSP *report-to*: o campo deslocamento do esquema/SSP *report-to* contém o deslocamento dentro do *array* de bytes do dicionário do nome de esquema/SSP do ID de *endpoint* para o qual os relatórios de estado, pertencentes ao repasse e entrega deste agregado, são transmitidos.

Deslocamento do esquema/SSP de custódio contém o deslocamento dentro do *array* bytes de dicionário do nome de esquema/SSP do ID de *endpoint* do custódio atual.

Estampa de tempo de criação: a estampa de tempo de criação é um par de SDNVs que, junto com o ID de *endpoint* da fonte e (se o agregado é um fragmento) o deslocamento do fragmento e comprimento do *payload*, servem para identificar o agregado. O primeiro SDNV da estampa de tempo é o tempo de criação do agregado, enquanto o segundo é o número de sequência da estampa de tempo de criação do agregado. O tempo de criação do agregado é o

tempo expresso em segundos desde o início do ano 2000, na escala UTC (*Coordinated Universal Time*).

Tempo de vida: indica o tempo no qual o *payload* do agregado não será mais útil, codificado como um número de segundos passados do tempo de criação.

Comprimento do dicionário: contém o comprimento do *array* de bytes do dicionário.

Dicionário: o campo de dicionário é um *array* de bytes formado pela concatenação de nomes de esquemas e SSPs terminados em zero de todos IDs de *endpoint* referenciados por qualquer campo em seu bloco primário junto com, potencialmente, outros IDs de *endpoint* referenciados pelos campos em outros blocos. Seu comprimento é dado pelo valor do campo comprimento do dicionário.

Deslocamento do fragmento: se as *flags* de controle de processamento do agregado de seu bloco primário indicam que o agregado é um fragmento, então o campo de deslocamento do fragmento indica o deslocamento a partir do início da unidade de dados da aplicação no qual os bytes compreendendo o *payload* deste agregado estão localizados. Caso contrário, o campo de deslocamento do fragmento é omitido do bloco.

Comprimento da unidade de dados da aplicação total: se as *flags* de controle de processamento do seu bloco primário indicar que o agregado é um fragmento, então o campo de comprimento da unidade de dados da aplicação indica o comprimento total da unidade de dados da aplicação original do qual seu *payload* de agregado é uma parte. Caso contrário, o campo de comprimento de unidade de dados da aplicação é omitido do bloco.

Bloco Canônico: todo bloco de agregado de tipo diferente do bloco de agregado primário inclui os seguintes campos, nesta ordem:

Código tipo de bloco: expresso como um inteiro binário sem sinal de oito bits. Código 1, para o tipo de bloco do agregado, indica que o bloco é um bloco de *payload* do agregado.

Códigos de tipo de bloco de 192 até 255 não são definidos no RFC 5050 (Scoot e Burleigh, 2007) e estão disponíveis para uso privado e/ou experimental. Todos outros valores de código de tipo de bloco são reservados para uso futuro.

Flags de controle de processamento de bloco: um inteiro sem sinal expresso como um SDNV. Os bits individuais deste inteiro são utilizados para invocar características de controle de processamento do bloco selecionado (Figura 2.12).

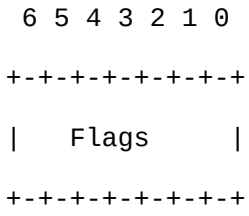

Figura 2.12. Bits de controle de processamento de bloco.

0 – O bloco deve ser replicado em todos os fragmentos

1 – Transmite um relatório de estado se o bloco não puder ser processado

2 – Elimina o agregado se o bloco não pude ser processado

3 – Último bloco

4 – Descarta o bloco se não puder ser processado

5 – Bloco foi encaminhado sem ter sido processado

6 – Bloco contém um campo *EID-reference*

Contagem de referência do EID do bloco e referência EID: está presente, se e somente se, as referências do bloco dos elementos do EID no dicionário do bloco primário, a *flag* ‘bloco contém um campo de EID de referência’ na *flag* de controle de processamento de bloco estiver ajustada para 1. Mais informações sobre esse campo podem ser encontradas no RFC 5050 (Scott e Burleigh, 2007).

Campo de comprimento de dados do bloco: contém o comprimento do agregado de todos os campos restantes do bloco, isto é, os campos de dados de tipo específico de bloco.

Campos de dados de tipo específico de bloco: formato e ordem são específicos do tipo.

2.4. Componentes conceituais de um nó DTN

Os nós DTN que obedecem ao protocolo de agregação possuem três componentes conceituais:

– Agente do protocolo de agregação ou BPA (*Bundle Protocol Agent*): é o componente do nó que oferece os serviços BP e executa os procedimentos do protocolo de agregação. É inteiramente um problema de implementação a maneira na qual estas operações são realizadas;

– Agente da aplicação ou AA (*Application Agent*): o AA de um nó é o componente do nó que utiliza os serviços BP para efetuar a comunicação para algum propósito. O AA por sua vez possui dois elementos, um elemento administrativo (constrói e requisita transmissão de registros administrativos, e aceita a entrega e processa qualquer sinal de custódia que o nó recebe) e um elemento específico da aplicação (constrói, requisita a transmissão de, aceita a entrega de, e processa ADUs específicas de cada aplicação);

– Adaptadores da camada de convergência ou CLAs (*Convergence Layer Adapters*): um CLA envia e recebe agregados em favor do BPA, utilizando os serviços de algum protocolo da Internet nativo. Assim, o sucesso da operação do BP fim a fim depende da operação de protocolos subjacentes em que é designada a camada de convergência. Uma grande variedade de protocolos pode servir para este propósito, desde que cada adaptador do protocolo da camada de convergência forneça um conjunto mínimo definido de serviços para o BPA. Assim, um CLA é designado a realizar o interfaceamento entre o protocolo de agregação comum e um protocolo subjacente específico e apropriado.

Espera-se que todo adaptador do protocolo da camada de convergência forneça os seguintes serviços para o BPA:

– Envie um agregado para todos os nós de agregação do MRG do *endpoint* identificado por um ID do *endpoint* que são alcançáveis via o protocolo da camada de convergência;

– Entregue para o BPA um agregado que foi enviado por um nó de agregação remoto via o protocolo da camada de convergência.

2.5. Segurança

As questões de segurança têm sido consideradas desde o início do projeto do protocolo de agregação. Sempre se assumiu que serviços de segurança são necessários na utilização do protocolo de agregação. Como resultado, a arquitetura de segurança do protocolo de agregação e os serviços de segurança disponíveis são especificados em um documento adicional, a especificação do Protocolo de Segurança de Agregação (Symington et al., 2007).

O protocolo de agregação está sendo projetado com a noção de que irá rodar sobre redes com recursos escassos. Por exemplo, a rede pode ter armazenamento restrito em nós *relay* (que realizam retransmissões), vazão e conectividade limitada, etc. Consequentemente, o protocolo de agregação deve garantir que somente aquelas entidades autorizadas a enviar agregados sobre tais ambientes restritos estão verdadeiramente autorizadas a fazer isso.

Existe uma série de questões abertas em segurança DTN. Um exemplo é a interação de fragmentação e a aplicação de mecanismos de criptografia que podem ser desafiadores, sendo uma preocupação e requerendo experiência de emprego adicional, antes que se possa escolher confiantemente entre várias restrições que poderiam melhorar estas situações problemáticas.

Enquanto o Protocolo de Segurança de Agregação define serviços de criptografia, não fornece (ainda) qualquer maneira de gerenciar as chaves requeridas. Pesquisas sobre este assunto estão apenas no começo e várias aproximações padrões regulares serão consideradas

antes que alguma solução seja escolhida. Naturalmente, qualquer solução precisa ser adequada para operação em ambientes DTN.

Outra área de segurança que merece mais estudo é um modelo para autorização de tráfego em DTNs que poderia ser análoga à forma como os problemas de AAA (*Authentication, Authorization, and Accounting*) são manipulados na Internet hoje. Novamente, o trabalho aqui está apenas começando, na tentativa de resolver o que sempre foi um grande problema de segurança em DTNs, como acontece na Internet: o problema do tráfego indesejado.

No contexto das DTNs, as questões de segurança podem seguir vários caminhos. Entretanto, ainda não se tem experiência suficiente para que boas decisões sejam tomadas.

2.6. Considerações finais

Neste capítulo, o conceito de redes tolerantes a atrasos e desconexões foi introduzido, juntamente com a definição de uma arquitetura DTN, sendo mostradas algumas características positivas que tornam as DTNs atrativas para prover comunicação em cenários desafiadores, como é o caso de regiões carentes de infraestrutura. Posteriormente, foram tratados o protocolo de agregação e questões de segurança nas DTNs. Além disso, para que as DTNs possam ser efetivamente implementadas, muitas questões em aberto, que foram citadas no decorrer do capítulo, precisam ser esclarecidas e desenvolvidas. Dentre elas, está o roteamento, que é o foco desta pesquisa. Por isso, no próximo capítulo é discutido o roteamento na rede descrita e detalhada neste capítulo, ilustrando-se alguns desafios de projeto do roteamento DTN.

Capítulo 3

ROTEAMENTO EM REDES TOLERANTES A ATRASOS E DESCONEXÕES

3.1. Introdução

Um dos componentes chaves da arquitetura DTN é o roteamento. Existem diferentes desafios nas DTNs, dependendo da natureza do ambiente de rede. Assim, nos últimos anos, pesquisadores vêm propondo protocolos de roteamento diferentes para os diversos ambientes encontrados nas DTNs. Com isso, as DTNs podem possuir vários tipos de contatos, e o cenário pode ser classificado em determinístico ou estocástico, dependendo do nível de informação a respeito da rede que pode ser utilizado pelo algoritmo de roteamento.

Este capítulo, inicialmente, descreve alguns desafios para o projeto de um algoritmo de roteamento em DTNs, seguidos dos diferentes tipos de contatos que podem ser encontrados nas DTNs. A Seção 3.2 lida com o estado atual da pesquisa sobre roteamento em DTNs, com um levantamento dos protocolos de roteamento classificados conforme o nível de informação utilizado. A Seção 3.3 mostra as características dos serviços *unicast*, *anycast* e *multicast* nas DTNs. Em seguida, a Seção 3.4 descreve as principais necessidades e aproximações do tema abordado, isto é, o roteamento *anycast* em cenários determinísticos, apresentando a utilização de grafos evolutivos para representação das DTNs, o modelo da rede, o *anycast* nas DTNs, os

objetivos de projeto do algoritmo de roteamento *anycast* e o problema do caminho mais curto. Por fim, a Seção 3.5 apresenta as considerações finais do capítulo.

3.1.1. Desafios de projeto do roteamento em DTNs

Nas redes mais tradicionais, protocolos de roteamento assumem que a rede possui nós constantemente conectados e visam encontrar a menor rota fim a fim entre um par fonte-destino dentro de um grafo de conexão. O procedimento da maioria dos protocolos é selecionar o caminho mais curto entre as rotas disponíveis, através da execução de um algoritmo de roteamento.

O crescimento do interesse em redes MANETs tem resultado em vários protocolos de roteamento novos, tais como: AODV (*Ad-hoc On-demand Distance Vector*) (Perkins et al., 2003), DSR (*Dynamic Source Routing*) (Johnson et al., 2001), TORA (*Temporally-Ordered Routing Algorithm*) (Park e Corson, 2001) e OLSR (*Optimized Link State Routing*) (OLSR, 2007), que tratam das questões de mobilidade e perda temporária de conectividade. Entretanto, esses protocolos ainda mantêm e contam com um grafo de conectividade para obter informação de conectividade e tomar decisões de roteamento, além de assumirem a existência de um caminho fim a fim completo (Spyropoulos et al., 2009). Desta forma, esses protocolos são apropriados para o roteamento convencional em MANETs, mas não para uma DTN com topologia alta e constantemente modificada pelas desconexões na rede.

Os protocolos de roteamento DTN são projetados para tratar e acomodar estas situações desafiadoras que são caracterizadas por frequentes desconexões de rede, alterações de topologia, alta perda de dados, atrasos longos e variáveis, limitação de energia devido ao uso de baterias, entre outros. Assim, protocolos de roteamento tradicionais não são suficientes para lidar com esses ambientes de rede altamente dinâmicos.

Além disso, para operação eficiente na enorme diversidade de ambientes em que um nó pode se encontrar, os nós DTN, provavelmente, terão que suportar diferentes estratégias de roteamento. Com isso, conclui-se que o projeto de protocolos de roteamento DTN é dependente do ambiente no qual os protocolos vão operar.

3.1.2. Tipos de contatos

Um conceito importante que deve ser considerado na arquitetura DTN é o de contato. Um contato corresponde a uma ocasião favorável para os nós trocarem dados. De acordo com as características das DTNs, não é assumido que todos os nós são alcançáveis e possam ser contatados a qualquer instante. Os contatos são classificados da seguinte forma:

Contatos persistentes: são aqueles que estão disponíveis em qualquer instante. Um exemplo de contato persistente é o enlace de rede de acesso ADSL (*Asymmetric Digital Subscriber Line*) que fica sempre disponível.

Contatos sob demanda: são aqueles que requerem alguma ação para que sejam instanciados, mas que, se acionados, funcionam como contatos persistentes até o momento de serem encerrados. Um exemplo de contato sob demanda é o enlace de rede de acesso discado sob o ponto de vista do usuário.

Contatos programados: são aqueles nos quais uma agenda de contato pode ser estabelecida entre dois ou mais nós antes que ocorra a troca de informações. Neste caso, o horário e a duração de cada contato são estabelecidos previamente entre os nós comunicantes, exigindo uma sincronização de tempo na rede para que a troca de informações possa ocorrer com sucesso. Um exemplo de contato programado são as aplicações espaciais (Figura 3.1) (Oliveira et al., 2007), explorando-se a movimentação dos elementos da rede (planetas, satélites, naves espaciais etc.).

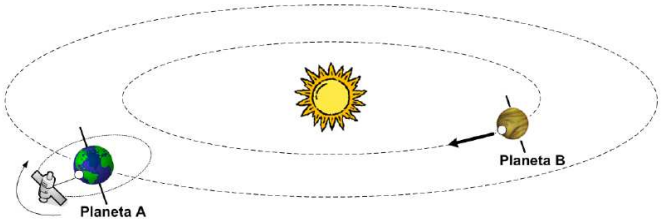

Figura 3.1. Exemplo de contato programado.

Contatos previsíveis: são aqueles em que os nós podem realizar previsões sobre o horário e a duração dos contatos baseados em históricos de contatos previamente realizados. Diferentemente dos contatos programados, os contatos previsíveis possuem certo grau de incerteza do contato. Um exemplo de DTN com contato previsível é uma rede rural esparsa (Figura 3.2) (Oliveira et al., 2007). Geralmente, nesse cenário, para superar os problemas ocasionados pela conectividade intermitente entre as duas regiões, ônibus públicos e motos são utilizados como mensageiros móveis, sendo responsáveis pelo armazenamento, transporte e entrega de dados entre as regiões. Como as visitas dos mensageiros móveis são variáveis, estando sujeita a falhas mecânicas e problemas ambientais (como estrada interditada), as regiões não possuem informações precisas sobre a próxima troca de dados. Além disso, as rotas seguidas pelos mensageiros podem ou não ser com finalidade de efetivar a comunicação.

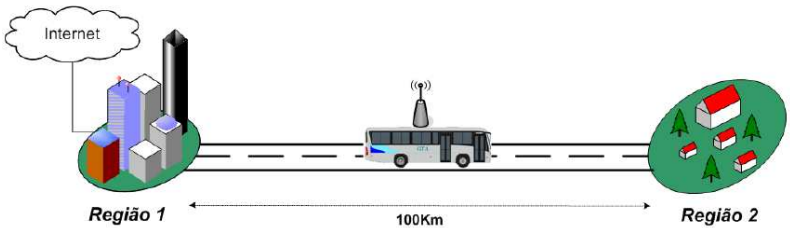

Figura 3.2. Exemplo de uma rede rural esparsa.

Contatos oportunistas: são aqueles decorrentes de encontros não previamente programados entre os nós. Esse tipo de contato tem como objetivo obter vantagens de contatos realizados totalmente ao acaso para realizar a comunicação entre nós que estejam, em alguns instantes, fora do alcance um do outro. Em alguns casos, em nenhum momento

existe um caminho inteiramente conectado entre um par fonte-destino, o que inviabilizaria a comunicação na Internet convencional. Entretanto, segundo o conceito de contato oportunista, é possível a comunicação sob tais condições. Para ilustrar o contato oportunista, suponha que Ana deseja encontrar-se com sua amiga Maria para entregar um dinheiro que estava lhe devendo. Porém, Ana não tem se comunicado com Maria. Certo dia, no shopping, Ana encontra João, o irmão de Maria. Sabendo que João é uma pessoa confiável, Ana entrega-lhe o dinheiro. Quando chega em casa, João encontra a irmã e entrega o dinheiro. Assim, apesar das duas amigas não terem se encontrado pessoalmente (conectividade intermitente), Ana (nó fonte) conseguiu fazer com que o dinheiro chegasse até Maria (nó de destino) através de uma pessoa intermediária (nó intermediário) que encontrou ao acaso no shopping (contato oportunista).

Desta forma, para os diferentes tipos de contatos, diferentes estratégias de roteamento devem ser empregadas, assim como, possivelmente, variados níveis de informações a respeito da rede estarão disponíveis e serão utilizados pelos algoritmos de roteamento.

3.2. Mecanismos de roteamento em DTNs

Devido aos diversos ambientes de rede desafiadores nos quais um nó pode se encontrar, além dos diferentes tipos de contatos que a rede pode possuir, o projeto do protocolo de roteamento será dependente do cenário para o qual é projetado. Basicamente, dividem-se os protocolos de roteamento nas DTNs em duas categorias: protocolos determinísticos ou estocásticos (Zhang, 2006). Protocolos determinísticos utilizam a informação que os nós obtêm a respeito da conectividade ou condições da rede para realizar decisões de repasse de forma eficiente. Já a outra categoria, protocolos estocásticos, lida com as maneiras que várias cópias de mensagens são disseminadas entre vários encaminhadores, sem o conhecimento da topologia da rede.

3.2.1. Cenário determinístico

Como visto anteriormente, o tempo durante o qual um enlace existe é chamado de contato. Quando informações sobre os contatos estão disponíveis, ou seja, tem-se conhecimento de quando ocorrem os contatos da rede, ou, em última instância, da topologia da rede em qualquer instante de tempo, diz-se que o cenário é determinístico.

Como mostrado na Seção 2.2.4 do Capítulo 2, uma rede DTN pode ser descrita abstratamente utilizando um multigrafo. Uma noção de grafo que captura a maioria das características de redes que variam com o tempo pode ser encontrada no trabalho de Ferreira (2004). A modelagem do tempo das redes obtidas pelos grafos dá origem a várias matrizes que podem ser utilizadas como funções objetivo nas estratégias de roteamento, tais como "menor tempo para alcançar um ou todos os destinos" ou "caminho com o menor número de saltos".

Para auxiliar no roteamento, informações a respeito da rede podem ser utilizadas. Conforme as informações disponíveis, estratégias diferentes são empregadas. Vários algoritmos de roteamento são apresentados por Jain et al. (2004) dependendo do tipo e quantidade de informação sobre as características da rede. Essas informações são classificadas e modeladas por oráculos que são simplesmente uma abstração capaz de responder a quaisquer perguntas sobre um assunto. São definidos quatro oráculos de conhecimento contendo alguma informação particular:

– Oráculo de resumo de contatos: capaz de dizer o tempo médio necessário até que um novo contato seja realizado entre dois nós, mas não o instante e duração exata dos contatos;

– Oráculo de contatos: fornece o instante de início e a duração de todos os contatos entre dois nós quaisquer da rede (equivalente ao grafo temporal completo da DTN);

– Oráculo de ocupação: informa, em qualquer instante de tempo, a ocupação dos *buffers* de transmissão em qualquer nó da rede;

– Oráculo de demanda de tráfego: capaz de fornecer a demanda de tráfego de todos os nós em qualquer instante de tempo.

Assim sendo, Jain et al. (2004) apresentam vários algoritmos de roteamento utilizando diferentes níveis de informação, ou seja, baseados em diferentes oráculos. É visto que, conforme a quantidade de informações utilizadas aumenta, a complexidade do algoritmo de roteamento também aumenta. Por isso, deve-se analisar a relação entre a complexidade e quantidade de dados adicionada por cada informação, e a melhora no desempenho provocado pela mesma. Desta forma, a importância do trabalho citado está na classificação do tipo de informação e no quanto cada uma destas informações pode melhorar o desempenho do algoritmo de roteamento.

Ainda no cenário determinístico, nós especiais podem ter a mobilidade controlada para aumentar o desempenho da comunicação. Em casos como balsas de mensagens ou MF (*Message Ferrying*), as balsas (que são dispositivos móveis que podem ter a mobilidade controlada) utilizam informações a respeito da localização dos nós para definirem suas rotas visando otimizar alguma medida de desempenho, como por exemplo, o atraso ou o número de mensagens entregues.

Devido às frequentes partições e longos atrasos nas DTNs, informação precisa e atualizada sobre o estado da rede pode não estar disponível. Assim, a disseminação do estado da rede é desafiadora em DTNs e pode envolver grandes atrasos, sendo importante analisar o efeito da informação atrasada no desempenho do roteamento.

3.2.2. Cenário estocástico

Em todas as aproximações sob o caso determinístico, um caminho fim a fim (possivelmente dependente do tempo) é determinado antes que as mensagens sejam transmitidas. Entretanto, em certos casos, a topologia de rede pode não ser conhecida ao longo do tempo. Neste caso, o comportamento da rede é aleatório e não conhecido.

Uma decisão simples de roteamento é repassar cópias de agregados para qualquer nó dentro de uma faixa de contatos sem utilizar qualquer informação. Essa é a categoria de roteamento epidêmico, cujos agregados recebidos em nós intermediários são repassados para todos ou parte dos nós vizinhos (exceto o nó que envia a mensagem) sem utilizar qualquer predição do enlace ou probabilidade de repasse do caminho. Um protocolo de roteamento epidêmico que utiliza inundação de mensagens foi proposto por Vahdat e Becker (2000), espalhando rapidamente cópias das mesmas mensagens nas porções conectadas da rede. Deste modo, o protocolo de roteamento epidêmico explora a mobilidade dos nós para distribuir mensagens através da rede. Entretanto, para tal esquema ser eficiente dois fatores são necessários: alta conectividade e capacidade de buffer suficiente para o armazenamento de mensagens.

Pelo lado oposto, tem-se o roteamento de entrega direta. Nesta aproximação, um nó fonte mantém a mensagem e entrega ao destinatário apenas quando ambos estão dentro do alcance de comunicação um do outro, ocasionando o mínimo *overhead*, por não distribuir cópias da mesma mensagem na rede. A desvantagem da entrega direta é o atraso, que tende a se tornar muito grande, pois é dependente do encontro entre os nós emissor e receptor, para que a mensagem seja entregue.

Assim sendo, muitos trabalhos já publicados são intermediários em relação aos dois extremos descritos acima, ou seja, colocam redundância na transmissão (ao contrário do roteamento com entrega direta) e limitam a quantidade de mensagens a serem repassadas para

os nós vizinhos (o roteamento epidêmico repassa para todos os vizinhos). Como exemplo, Spyropoulos et al. (2008) propõem um algoritmo de roteamento que, em uma fase inicial (*spray*), realiza uma inundação controlada de mensagens na rede. Após esta fase, inicia-se uma segunda fase (*wait*) na qual se realiza transmissão direta através dos nós que receberam cópia da mensagem na fase anterior, ou pode-se optar por reduzir o número de cópias deixadas pelos nós de uma unidade a cada transferência, ou ainda, o número pode ser reduzido pela metade a cada transferência. Neste esquema, qualquer oportunidade de repassar uma cópia disponível para um nó que não a tenha é sempre aproveitada. Spyropoulos et al. (2009) mostraram que um mecanismo para distinguir os melhores nós para se repassar as cópias das mensagens (baseado em uma função de utilidade) pode melhorar significativamente o desempenho em cenários heterogêneos, ou seja, considerou-se múltiplas classes de nós nessa última análise realizada.

Outra aproximação para o cenário estocástico é o repasse de mensagens baseado em estimativa, no qual nós intermediários estimam a chance, para cada enlace de saída, de eventualmente alcançar o destino. Baseados nessa estimativa, nós intermediários decidem se armazenam a mensagem e esperam por uma melhor oportunidade, ou para quais nós (e o momento) repassar. O PROPHET (*Probabilistic Routing Protocol using History of Encounters and Transitivity*) (Lindgren et al., 2003) é um exemplo de aproximação baseada em estimativa. Neste esquema, estima-se uma medida probabilística denominada previsibilidade da entrega, que estabelece o quão provável um nó será capaz de entregar uma mensagem para o destino, utilizando informação sobre contatos anteriores.

O PROPHET citado acima realiza decisões sobre repasse de mensagens baseada na probabilidade de entrega de cada vizinho. Existe também a possibilidade das decisões serem baseadas no desempenho fim a fim. Pode-se estimar o MEED (*Minimal Estimated Expected*

Delay) utilizando o histórico de contato observado. Três protocolos com essas características são revistos por Zhang (2006).

Existe também aproximação baseada em modelo, na qual dispositivos movem-se seguindo padrões conhecidos, tais como caminhar ao longo de uma rua ou dirigir na estrada. Uma vez que usuários descrevem seu padrão de movimento, nós intermediários tem uma estimativa mais precisa de quais nós se movem em direção ao destino com maior probabilidade.

Todas as aproximações discutidas até aqui deixam o nó móvel esperando passivamente pela rede para conectar-se novamente. Algumas pesquisas, entretanto, propõem aproximações que tentam limitar esses atrasos explorando e controlando a mobilidade do nó. Um exemplo de roteamento baseado no controle do movimento do nó é o esquema de MF descrito por Zhao et al. (2004). Neste esquema, a movimentação dos nós pode ser configurada para auxiliar a entrega de dados podendo o nó regular (ao contrário da balsa, não possui responsabilidade de transportar mensagens entre nós) deslocar-se até a rota realizada pela balsa para permitir o transporte das mensagens, ou então, o nó pode enviar um pedido de serviço para a balsa, fazendo com que a mesma se desloque em direção ao nó que requisitou o serviço. Além disso, os autores também analisam o problema de se utilizar múltiplas balsas (Zhao et al., 2005) para entregar dados em redes com nós estacionários, projetando as rotas das balsas de forma que o atraso de mensagens médio possa ser minimizado.

Outra proposta é o MORA (*Multi-Objective Robotic Assistance*) (Burns et al., 2005) que utiliza métodos de controle multiobjetivos de robôs para gerar movimentos capazes de otimizar múltiplas medidas de desempenho da rede. Além disso, as estratégias de controle são dependentes da qualidade da estimativa do estado da rede. Assim, conclui-se que o MORA tem mais a oferecer em cenários onde o movimento não é aleatório.

Há ainda o roteamento baseado em codificação. Para lidar com perdas de canal sem fio, técnicas de codificação por apagamento (*erasure coding*) e codificação de rede (*network coding*) foram propostas para redes ad hoc e DTNs. A ideia básica da codificação por apagamento é codificar uma mensagem original em um grande número de blocos de código. Suponha que a mensagem original contenha k_b blocos. Utilizando codificação por apagamento, a mensagem é codificada em n_b (n_b>k_b) blocos tal que se k_b ou mais dos n_b blocos são recebidos, a mensagem original pode ser decodificada. Aqui, $r_b = n_b/k_b$ é denominado de fator de replicação e determina o nível de redundância. Os autores de (Jain et al., 2005) assumem que a probabilidade P_{bi} de que a transmissão sobre o enlace *i* seja bem sucedida é conhecida. Dado que o fator de replicação é r_b, é estudado o seguinte problema de alocação: determinar uma fração ótima, x_{bi}, de blocos de código por apagamento que deverão ser enviados sobre o caminho *i*, tal que a probabilidade de sucesso na recepção seja maximizada. A vantagem da utilização da codificação por apagamento é a divisão da responsabilidade de encaminhamento das mensagens entre vários nós encaminhadores.

No caso da codificação de rede, ao invés do simples repasse de pacotes recebidos, nós intermediários podem combinar alguns dos pacotes já recebidos e enviá-los como um novo pacote. Por exemplo, supondo que existem três nós A, B e C. Os nós A e C querem trocar informações através do nó central B, como mostrado na Figura 3.3 (Oliveira et al., 2007). O nó A transmite o pacote *a'* para B, e o nó C transmite o pacote *c'* para B. O nó B realiza *broadcast* de *a'* XOR *c'*. Uma vez que o nó A tem o pacote *a'*, e o nó C o pacote *c'*, o nó A pode decodificar *c'* e o nó C pode decodificar *a'*. Neste exemplo, é visto que o número de transmissões é reduzido quando a codificação de rede é utilizada. A ideia básica de codificação de rede para geração de novos pacotes é utilizada por Widmer e Le Boudec (2005). Quando um pacote é recebido em um nó, d_p novos pacotes são gerados e realizam

broadcast para seus vizinhos, com o valor d_p sendo referido como fator de encaminhamento e dependente da densidade de nós.

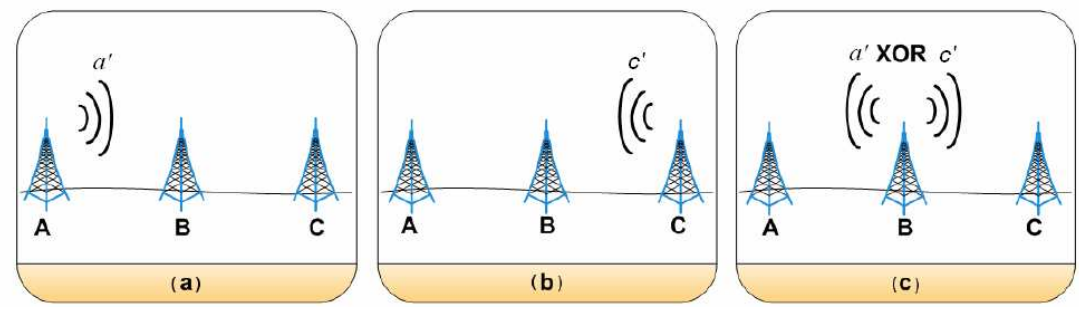

Figura 3.3. Exemplo de codificação de rede.

Diversos trabalhos relacionados com cada uma das aproximações anteriores podem ser encontrados no levantamento realizado por Zhang (2006).

3.3. Serviços *unicast*, *anycast* e *multicast*

Um EID pode referir-se a um *endpoint* contendo um ou mais nós DTN. Quando o grupo possui apenas um nó, a semântica da entrega é a *unicast*. Quando se refere a um grupo de tamanho maior que um, a semântica da entrega pode ser da variedade *anycast* ou *multicast* (*broadcast* é considerado ser de variedade *multicast*). No caso da semântica de entrega ser a *anycast*, um agregado é entregue para um nó entre um grupo de nós potenciais, e para semântica de entrega *multicast*, um agregado deve ser entregue para todos os nós potenciais do grupo.

Grande parte das pesquisas considera o *endpoint* contendo apenas um nó DTN. Além disso, a maioria dos algoritmos de roteamento citados na seção anterior realiza entrega *unicast*.

Para trabalhos envolvendo o serviço de entrega *anycast* ou *multicast*, alguns desafios adicionais são encontrados. Por exemplo, na tese de Zhao et al. (2006), para análise do

multicasting em DTNs são propostos três modelos de semântica *multicast*, pois em alguns casos para *multicasting* tradicional não fica claro quais nós serão os receptores da mensagem. A Figura 3.4 ilustra um exemplo de cenário que requer a definição de novas semânticas e também traz novas questões de projeto. Um nó fonte envia uma mensagem para um grupo no tempo *t*. *t*' é o tempo mais cedo que outros nós poderão, possivelmente, receber a mensagem de acordo com as limitações de topologia da rede. O nó A une-se ao grupo no tempo $t_1 < t$ e deixa-o no tempo t_2, $t < t_2 < t$'. O nó B une-se no tempo t_3, $t < t_3 < t$' e nunca deixa o grupo. Da perspectiva do *multicasting* tradicional, não fica claro quais nós deverão receber a mensagem, se A, B, ambos ou nenhum.

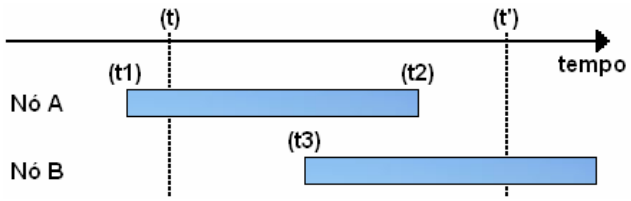

Figura 3.4. Exemplo de semântica *multicast* em DTNs.

Primeiramente, para solucionar o problema de se determinar os receptores da mensagem, definem-se os membros do grupo (podem mudar com o tempo conforme *endpoints* se unem ou deixam o grupo) e os receptores planejados (devem ser fixos para uma mensagem).

No trabalho de Zhao et al. (2006) são definidas três semânticas (Figura 3.5) de mensagens para definição dos receptores planejados de uma mensagem *multicast*:

– TM (*Temporal Membership*): a mensagem contém um intervalo de associação que especifica o período durante o qual os membros do grupo são definidos. Para uma mensagem com ID de grupo G e intervalo de associação $[t_1,t_2]$, os receptores planejados da mensagem consistem de *endpoints* que são membros do grupo G em qualquer instante durante o período $[t_1,t_2]$;

– TD (*Temporal Delivery*): a mensagem especifica um intervalo de associação e um intervalo de entrega. O intervalo de entrega indica o período de tempo durante o qual a

mensagem deverá ser entregue para o receptor planejado. O TM é um caso especial do TD, com intervalo de entrega ajustado para $[t_0,\infty]$, com t_0 sendo o tempo de geração da mensagem;

– CMD (*Current-Member Delivery*): quando a *flag* CMD está ajustada, os receptores da mensagem deverão ser membros do grupo no tempo da entrega da mensagem. Quando a *flag* CMD não está ajustada, o modelo CMD é reduzido para o TD.

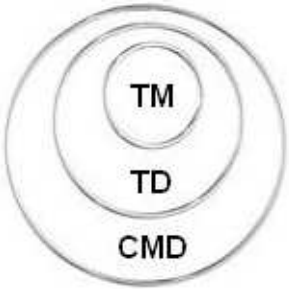

Figura 3.5. Modelos de semântica *multicast*.

Uma abstração semelhante dessas semânticas descritas para *multicast* pode ser utilizada para *anycast*. Depois de definidos os receptores planejados, fica a cargo do algoritmo de roteamento definir qual dos possíveis receptores deverá receber a mensagem visando otimizar alguma medida de desempenho da rede. Gong et al. (2006) definem três modelos de semântica *anycast* para DTNs (são vistos na Seção 3.4.3) baseados em um novo modelo e um algoritmo de roteamento correspondente para roteamento *anycast* é apresentado.

A necessidade de definição de novas semânticas *anycast* e *multicast* em DTNs, além de questões de projeto que podem surgir, tornam o roteamento ainda mais desafiador, com muitos trabalhos podendo explorar tal área.

3.4. Roteamento *anycast* em cenários determinísticos

Como na pesquisa aqui realizada é visado o roteamento *anycast* em cenários determinísticos, esta seção apresenta as aproximações e considerações a respeito da representação das principais características da DTN através de grafos evolutivos (Seção 3.4.1

e 3.4.2) e da semântica de entrega *anycast* (Seção 3.4.3) utilizada. Os objetivos do algoritmo de roteamento *anycast* e a aplicação de algoritmos que calculam o caminho com o menor custo são tratados nas Seções 3.4.4 e 3.4.5, respectivamente.

3.4.1. Representação da rede através de grafos evolutivos

Neste trabalho considera-se que o movimento e as conexões futuras são completamente conhecidos, isto é, a topologia da rede pode ser conhecida ao longo do tempo. Para permitir a análise do comportamento dessas redes dinâmicas ao longo do tempo, uma modelagem é realizada através de grafos evolutivos, que consistem da formalização de grafos no domínio do tempo. Em um artigo tutorial (Ferreira, 2004) é descrito um modelo combinatorial que captura a maioria das características de redes com comportamentos variantes ao longo do tempo. Para ilustrar a utilização de grafos evolutivos, têm-se a Figura 3.6 representando uma rede DTN em quatro intervalos de tempo distintos.

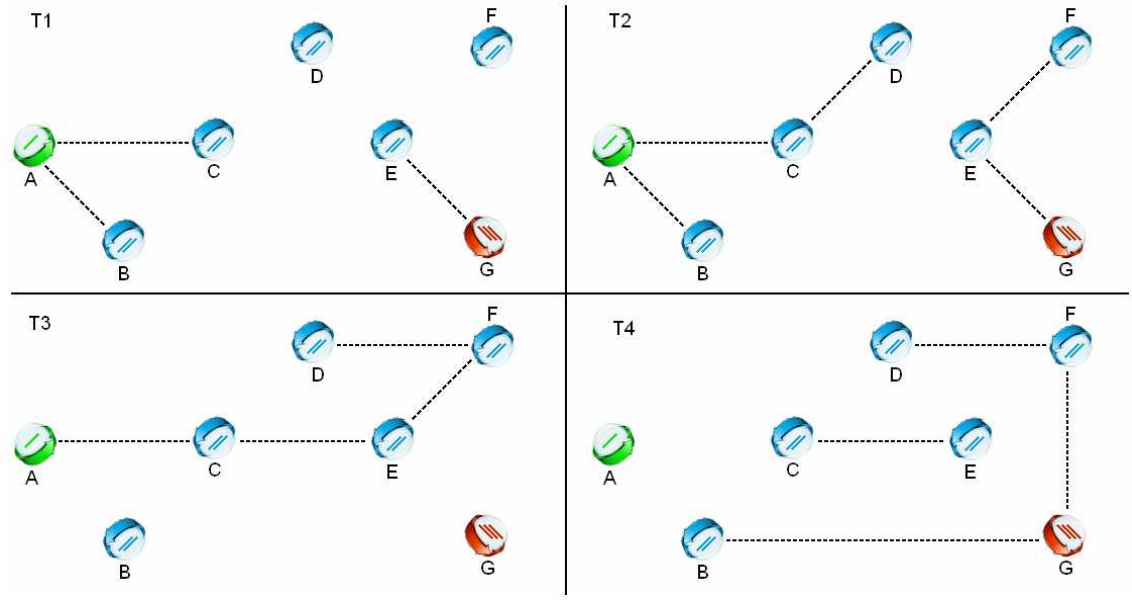

Figura 3.6. Representação de uma DTN em quatro intervalos de tempo distintos.

Supõe-se que uma mensagem deve ser enviada de A para G. Apesar de A e G não serem conectados em um único intervalo de tempo, A pode enviar mensagens para G utilizando caminhos compostos ao longo do tempo.

Um grafo evolutivo pode ser visto como uma abstração formal de redes dinâmicas, na qual cada nó e enlace têm uma lista de presença definida. Esta lista de presença de um enlace ou um nó indica o instante de tempo no qual o enlace ou nó está presente, além de outros parâmetros (por exemplo, custo do enlace) que caracterizam a conexão durante cada intervalo. Sucintamente, um grafo evolutivo é uma sequência indexada de subgrafos, onde o subgrafo associado a um dado índice corresponde à topologia da rede no intervalo de tempo indicado por aquele índice (Oliveira et al., 2007), como mostrado na Figura 3.7.

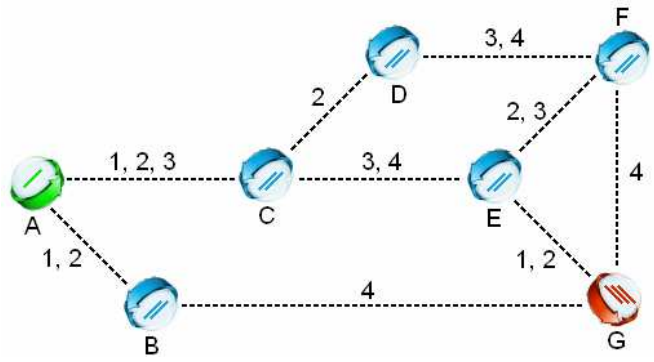

Figura 3.7. Grafo evolutivo representando a rede.

Utilizando a representação por grafo evolutivo fica fácil notar que a jornada, isto é, rota constituída de uma sequência de enlaces levando-se em conta os intervalos de tempo de existência dos enlaces, com o menor número de saltos é a A-B-G. Entretanto, o objetivo pode ser o caminho que entrega a mensagem mais cedo, no caso todos os três caminhos possíveis (A-B-G, A-C-E-F-G e A-C-D-F-G) entregam no intervalo de tempo 4. O objetivo pode ser encontrar a jornada que gasta o menor tempo para entregar a mensagem. Neste caso, devem ser levados em conta os tempos de transmissão e de propagação das mensagens sobre os enlaces. Considerando esses tempos desprezíveis, o caminho que gastaria o menor tempo dentro da rede, para entregar a mensagem, seria o A-C-E-F-G (a mensagem poderia ser

enviada entre os intervalos de tempo 3 e 4). Além disso, em uma jornada deve levar-se em consideração a restrição de que o próximo enlace nunca pode ser um enlace que só existiu em subgrafos passados. Por isso, o caminho A-C-E-G não é permitido, pois uma mensagem não pode ser enviada de C para E antes do intervalo de tempo 3. Por outro lado, o enlace E-G só existe durante os intervalos 1 e 2.

Uma característica importante da utilização do modelo de grafo evolutivo é permitir levar em consideração diferentes métricas como objetivo. Por exemplo, uma aplicação pode requerer que mensagens sejam entregues o mais rápido possível. Outra aplicação pode exigir que a mensagem seja entregue utilizando o menor número de saltos possível. O objetivo da aplicação pode ser ainda, minimizar o tempo gasto pela mensagem dentro da rede após ser injetada na mesma.

3.4.2. Representação de cada *edge* em um grafo DTN

O grafo DTN é um grafo G = (V, E), onde V do grafo representa os vértices (ou nós) e E representa os *edges* (ou enlaces). No trabalho de Jain et al. (2004), a DTN é representada através de um grafo no qual os nós são conectados por múltiplos enlaces, representando diferentes enlaces físicos (por exemplo: satélites, linhas telefônicas e mensageiros móveis). Como o objetivo é modelar o problema para viabilizar a comunicação entre regiões desconectadas que fazem uso de mensageiros móveis (carros, ônibus e caminhões), a representação que se mostrou mais adequada para representar as características das DTNs foi baseada no trabalho de Gong et al. (2006). Nela, os nós na rede são estacionários, com a conectividade entre os mesmos sendo realizada por dispositivos (mensageiros) móveis responsáveis por transportar e entregar as mensagens. Desta forma, um enlace entre dois nós *1* e *2* significa que existe algum dispositivo móvel movendo-se do nó inicial *1* para o nó terminal *2* (Figura 3.8).

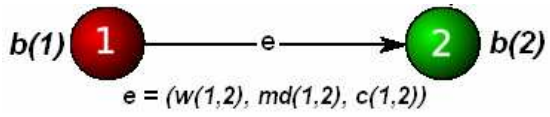

Figura 3.8. *Edge* em um grafo DTN.

A capacidade de armazenamento (*buffer*) dos nós *1* e *2* (*b*(*1*) e *b*(*2*)) e do dispositivo móvel (*c*(*1*,*2*)) é limitada. Supõe-se que cada dispositivo móvel que se desloca entre o mesmo nó inicial, por exemplo, *1*, e nó terminal, por exemplo, *2*, possui a mesma velocidade de deslocamento, ou seja, possui o mesmo atraso de movimento denotado por *md*(*1*,*2*). O tempo de partida dos dispositivos móveis em cada enlace, ou seja, o tempo no qual o dispositivo móvel deixa o nó *1* em direção ao nó *2*, é representado por *w*(*1*,*2*). Além disso, os nós *1* e *2* da rede são estacionários e geram mensagens. Por outro lado, dispositivos móveis deslocam-se de um nó para outro e não geram mensagens.

Desta forma, são utilizados os grafos evolutivos descritos na seção anterior com nós e enlaces possuindo as características descritas nesta seção, ou seja, com a capacidade do enlace (por exemplo, capacidade de armazenamento, atraso de propagação e tempo de partida) dependentes do tempo.

3.4.3. *Anycast* nas DTNs

O roteamento para entrega *anycast* é útil em situações nas quais um *host*, aplicação ou usuário deseja localizar um *host* que suporta um serviço particular. Entretanto, se vários servidores suportam o serviço, qualquer um desses servidores pode ser utilizado. O serviço *anycast* é adequado justamente para aproveitar essa oportunidade de transmitir para um, possivelmente o destino que apresentar as melhores condições de comunicação, dentre um grupo de possíveis nós, permitindo a comunicação em cenários nos quais muitas vezes o serviço *unicast* seria inviável. *Anycast* em DTNs pode ser utilizado em casos de descoberta de

recursos, com pessoas podendo procurar um serviço particular, como um médico ou bombeiro, sem conhecer o local ou ID específico. Aplica-se também em campos de batalhas quando um soldado pretende entregar uma mensagem particular a qualquer centro de comando, ou um determinado centro de comando pretende enviar uma mensagem a qualquer soldado de um batalhão. *Anycast* pode ainda ser utilizado na educação, na qual estudantes desejam obter um determinado arquivo de qualquer um dos membros de um grupo de estudantes.

Devido aos longos e/ou variáveis atrasos das DTNs, o *anycast* tradicional não deixa especificado de forma clara os nós que podem ser receptores de uma mensagem quando o ambiente apresenta os desafios de uma rede DTN. Por isso, como visto na Seção 3.3, a definição de novas semânticas de entrega *anycast* se torna necessária. A definição dos receptores planejados da mensagem, que deverão ser fixos (embora os membros do grupo de destino possam unir-se ou deixar o grupo), permite a especificação de novos modelos de semântica. Três modelos de semântica e exemplos de aplicações *anycast* em DTNs são apresentados por Gong et al. (2006). Para definição dos receptores planejados têm-se os três modelos de semântica *anycast* que se seguem:

– Modelo CM (*Current Membership*): o receptor da mensagem deverá ser membro do grupo de destino no tempo em que a mensagem for entregue;

– Modelo TIM (*Temporal Interval Membership*): a mensagem inclui um intervalo temporal que especifica o período durante o qual o receptor planejado deve ser um membro do grupo de destino;

– Modelo TPM (*Temporal Point Membership*): o receptor planejado ao menos deve ser um membro do grupo de destino em algum tempo durante um intervalo de associação.

Optou-se por utilizar os modelos CM, TIM e TPM, devido à simplicidade e serem três modelos de semântica distintos, permitindo a utilização de acordo com o contexto da

aplicação, embora os modelos TM, TD e CMD descritos na Seção 3.3 do Capítulo 3 para *multicast*, também possam ser utilizados. Assim sendo, depois de definidos os receptores planejados, fica a cargo do algoritmo de roteamento *anycast* definir qual dos possíveis receptores deverá receber a mensagem visando otimizar alguma medida de desempenho da rede. Para definição precisa de quais modelos de semânticas são mais apropriados e quantos são suficientes, experiência adicional com aplicações em redes DTN será necessária.

Gong et al. (2006) apresentam um algoritmo de roteamento *anycast* para DTNs baseado em estimação, ou seja, nós intermediários decidem entre armazenar a mensagem e esperar por uma melhor oportunidade, ou para quais nós repassar a mensagem. Nesta estratégia de roteamento, o algoritmo calcula o atraso mínimo esperado ou MED (*Minimum Expected Delay*) de cada nó ao nó destino através do algoritmo MED (Jain et al., 2004). E então, cada nó seleciona os nós cujos valores do MED são os menores como candidatos a entregar as mensagens.

Xiao et al. (2010) apontam que os algoritmos existentes, tais como o MED e o proposto por Gong et al. (2006), não são os ótimos em cenários com contatos probabilísticos, isto é, em cenários nos quais os nós encontram-se ou entram em contato seguindo algumas probabilidades. Em seguida, Xiao et al. (2010) definem uma nova medida MDRA (*Maximum Delivery Rate for Anycast*) para indicar a probabilidade máxima de sucesso de enviar uma única cópia de mensagem *anycast* de um nó para um grupo de nós de destino. Baseado nesta métrica, um algoritmo de roteamento *anycast* é proposto, com resultados baseados em simulação mostrando que o algoritmo proposto pode melhorar o desempenho quando comparado aos anteriores.

Uma aproximação de *anycast* visando aumento de desempenho em DTNs é proposta por Fazl-e-Hadi et al. (2007), onde a disponibilidade do enlace é oportunista e a aproximação utiliza a rede subjacente para conseguir informação topológica corrente. Diferentemente,

embora cada *edge* no grafo possua uma representação semelhante à utilizada por Gong et al. (2006), o algoritmo de roteamento *anycast* aqui desenvolvido é para o cenário determinístico, isto é, assume-se que a topologia da rede pode ser conhecida ao longo do tempo. Além disso, as restrições de armazenamento, bem como o tráfico dinâmico da rede são incorporados no presente trabalho.

Assim sendo, foi observado que os esquemas de roteamento *anycast* apresentados são utilizados em um cenário estocástico, e não incorporam restrições de armazenamento dos nós e dispositivos móveis da rede, nem o tráfego dinâmico da rede. Como resultado, surge a necessidade de desenvolver-se um algoritmo de roteamento *anycast* utilizado em um cenário determinístico, e que incorpore as restrições não consideradas pelos algoritmos de roteamento *anycast* anteriores.

Um cenário típico de aplicação *anycast* em redes DTN representadas por grafos envolve a conexão de regiões carentes de infraestrutura (por exemplo, áreas remotas e rurais). Essas regiões estacionárias podem fazer uso de carros, ônibus e caminhões que atravessam a região regular e periodicamente para atuarem como mensageiros móveis e permitir a comunicação de dados, sendo necessário que todos os envolvidos na transmissão possuam capacidade de transmitir e armazenar dados. Este cenário possui as características da representação de cada *edge* apresentadas na Seção 3.4.2. Um exemplo de aplicação do *anycast* em DTNs seria a busca por um serviço ou informações específicas nesse cenário citado. Deste modo, um usuário poderia solicitar a busca de um serviço ou informações específicas, e o *anycast* pode ser utilizado para encontrar usuários que forneçam o serviço ou contenham a informação requisitada, importando apenas o serviço e a informação, e não o ID ou localização precisa de onde está vindo o serviço e a informação. Assim, especificado o grupo de receptores planejados da mensagem (nós que suportam o serviço ou contém a informação requerida), a aplicação de um algoritmo de roteamento *anycast* pode otimizar a busca pela mensagem

dentre esse grupo de receptores planejados, fazendo o uso adequado das oportunidades de transmissão na rede.

O usuário do serviço descrito poderia estar requisitando serviço de um médico ou bombeiro, ou poderia ser o dono de uma fazenda trocando informações sobre o mercado agropecuário (Figura 3.9). Além disso, essas regiões estacionárias poderiam ser escolas trocando arquivos de disciplinas ou centros de comandos trocando informações em um campo de batalha. Outras aplicações importantes do serviço *anycast* em DTNs, utilizando nós estacionários e mensageiros móveis para viabilizar a comunicação, são citadas por Gong et al. (2006). Desta forma, observa-se que a modelagem utilizada para representação das DTNs e o serviço *anycast* possuem muitas aplicações importantes.

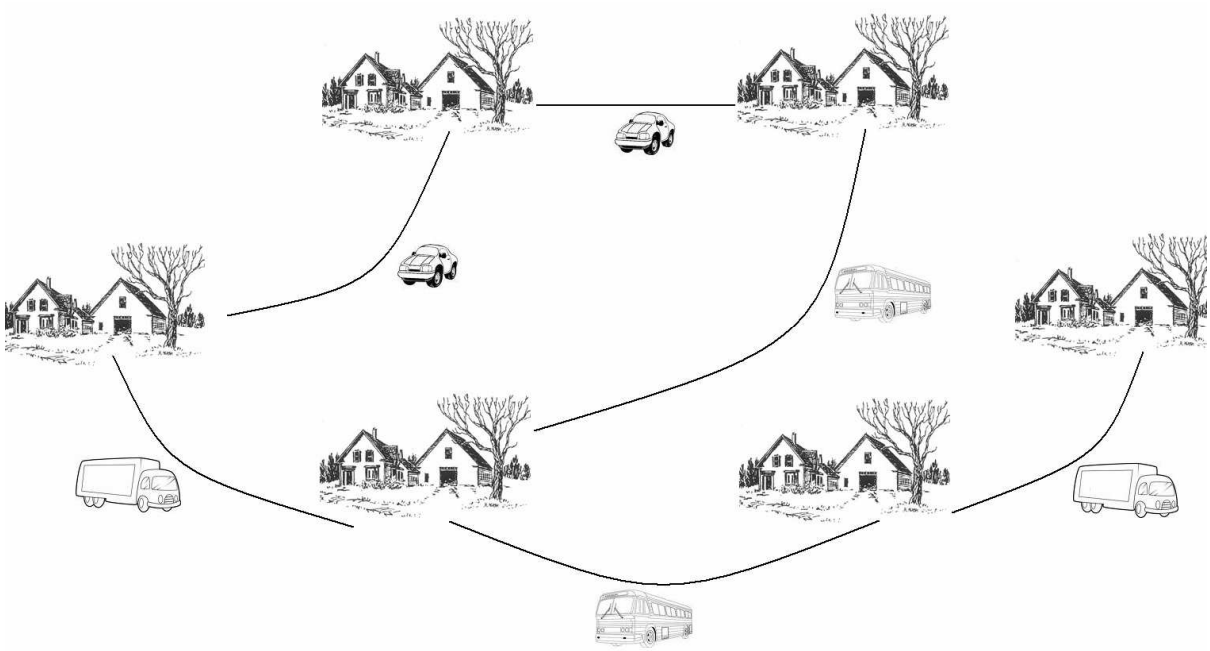

Figura 3.9. Exemplo de cenário típico de aplicação de *anycast* em DTNs.

3.4.4. Objetivos de projeto do algoritmo de roteamento *anycast*

O algoritmo de roteamento deve realizar o roteamento *anycast* em redes DTN utilizando as informações obtidas de grafos evolutivos. Além disso, deseja-se que o algoritmo seja capaz

de reagir ao comportamento dinâmico da rede, ou seja, deve acompanhar as constantes mudanças de topologia da rede.

Como os recursos em uma DTN são extremamente escassos, e a oportunidade de transmissão pode ser rara e muito disputada, uma característica essencial do algoritmo de roteamento será acomodar o tráfego na rede de modo que diferentes tráfegos interfiram o mínimo possível um no outro. Para que isso seja possível, o algoritmo deve ser capaz de direcionar o tráfego para rotas alternativas visando evitar que vários tráfegos disputem o mesmo contato. Além disso, o algoritmo de roteamento deve ser suficientemente robusto para não permitir que pequenas variações nas condições da rede alterem significativamente as chances das mensagens serem entregues aos seus destinos. Por fim, também é objetivo do algoritmo de roteamento *anycast* otimizar as medidas de desempenho da rede, visando fornecer um serviço requerido pelos usuários com qualidade.

3.4.5. O problema do roteamento *anycast* em cenários determinísticos

O algoritmo do menor caminho de Dijkstra (1959) pode ser utilizado para o cálculo dos menores caminhos, em termos de custo, entre nós de uma rede, quando os custos dos enlaces não variam com o tempo. Por outro lado, para representar as características de repasse de uma DTN, os custos dos enlaces são variantes ao longo do tempo, requerendo adaptações do algoritmo de Dijkstra original. Para que isso seja possível, é necessário modificar a função custo, que neste caso deverá levar em consideração as informações disponíveis sobre os custos dos enlaces (como o tempo de partida dos dispositivos móveis $w(i,j)$ e o atraso de movimento $md(i,j)$). A partir dos conceitos de grafo evolutivo podem ser construídos algoritmos com diferentes métricas como objetivo (número de saltos, tempo de entrega, tempo dentro da rede).

Alguns algoritmos de roteamento para DTNs em cenários determinísticos são apresentados por Jain et al. (2004), diferindo quanto à quantidade de informação utilizada por cada um. Como exemplo, o algoritmo atraso mínimo esperado ou MED (*Minimum Expected Delay*) utiliza o tempo médio necessário até que um contato seja realizado entre dois nós. Já o algoritmo entrega mais rápida ou ED (*Earliest Delivery*) se baseia nas informações sobre o início e a duração de todos os contatos, ou seja, utiliza informações de quando os enlaces estão disponíveis.

Como não se tem nenhum algoritmo de roteamento *anycast* projetado especificamente para DTNs em cenários determinísticos, são analisadas duas linhas:

– A primeira considera modificações dos algoritmos que calculam menores caminhos com base em alguma função custo (número de saltos ou atraso) para definição de rotas para o roteamento *anycast*:

- Algoritmo do menor caminho ou SP (*Shortest Path*): algoritmo de roteamento simples que utiliza informações sobre o grafo evolutivo para calcular a rota com o menor número de saltos entre o nó fonte e um dos receptores planejados, ou seja, a função custo será minimizar o número de saltos;
- Algoritmo de entrega mais rápida ou ED (*Earliest Delivery*): algoritmo de roteamento que utiliza informações sobre o tempo de partida dos dispositivos móveis $w(i,j)$ e o atraso de movimento $md(i,j)$ para calcular a rota na qual a mensagem será entregue o mais rápido possível.

– Uma vez que a utilização desses dois algoritmos pode não produzir resultados satisfatórios, nem atender os objetivos de projeto da seção anterior, outra aproximação, baseada em uma proposta de algoritmo mais complexa que leve em consideração as restrições de tráfego nos nós e enlaces do grafo, além da topologia dinâmica da rede, é apresentada e investigada:

- Proposta descrita no próximo capítulo.

3.5. Considerações finais

Neste capítulo, a questão do roteamento em DTNs foi tratada, e propostas de roteamento DTN existentes foram levantadas e discutidas. Além disso, observou-se que os nós DTN, possivelmente, irão suportar diversas estratégias e protocolos de roteamento a fim de operar eficientemente em uma vasta diversidade de ambientes no qual o nó pode encontrar-se. Em seguida, os serviços *unicast*, *anycast* e *multicast* em DTNs foram discutidos. Por fim, foram definidas as necessidades e aproximações utilizadas para o roteamento *anycast* em cenários determinísticos. Diante do estudo realizado, não foi encontrado nenhum algoritmo de roteamento *anycast* projetado especificamente para DTNs em cenários determinísticos. Assim sendo, como mencionado na seção anterior, optou-se por seguir duas abordagens para o projeto do algoritmo de roteamento *anycast*. A primeira segue uma linha mais simples e direta, realizando modificações em algoritmos que calculam caminhos com os menores custos. Já a segunda abordagem envolve o estudo e a proposta de um algoritmo de roteamento *anycast* mais complexo. Esta proposta de algoritmo de roteamento *anycast* para as DTNs é descrita no próximo capítulo.

Capítulo 4

PROPOSTA DE ALGORITMO DE ROTEAMENTO *ANYCAST* EM REDES TOLERANTES A ATRASOS E DESCONEXÕES UTILIZANDO ALGORITMOS GENÉTICOS

4.1. Introdução

Este capítulo destina-se ao detalhamento de uma proposta de algoritmo de roteamento *anycast* para DTNs que utiliza informações obtidas de grafos evolutivos. Grande parte dos problemas combinatoriais e de difícil solução, como é o caso do roteamento *anycast* em DTNs para seleção apropriada de rotas visando otimizar a entrega das mensagens, são do tipo NP-completo. Nestes casos, métodos heurísticos são aplicados objetivando reduzir o espaço de busca e, consequentemente, geram o mais rapidamente possível conjuntos aproximados de soluções. Como os algoritmos genéticos ou GAs (*Genetic Algorithms*) são apropriados para solução de problemas complexos, aplicou-se GAs para otimização do roteamento *anycast* em DTNs. Os GAs são um ramo dos algoritmos evolucionários (utilizam modelos computacionais dos processos naturais de evolução como ferramenta para solução de

problemas) e como tal podem ser definidos como algoritmos de busca baseados no mecanismo de seleção e evolução natural (Goldberg, 1989).

Nos GAs, populações de indivíduos são criadas de forma aleatória ou não e submetidas aos operadores genéticos que consistem em aproximações computacionais de fenômenos vistos na natureza, como a reprodução sexuada, a mutação genética, entre outros. Estes operadores genéticos, juntamente com a avaliação da aptidão de cada indivíduo, isto é, caracterização da qualidade de cada indivíduo como solução do problema, geram um processo de evolução natural.

É importante ressaltar que os GAs não constituem um algoritmo de busca da solução ótima de um problema, mas sim faz competir uma população de indivíduos e pelo processo de sobrevivência do mais apto, os melhores tendem a sobreviver. Assim sendo, os GAs, por realizar buscas direcionadas e inteligentes, são apropriados para solução de problemas com espaços de busca intratavelmente grandes, ou seja, que não podem ser resolvidos por técnicas tradicionais. Em vista disso, a proposta de algoritmo de roteamento *anycast* utiliza GAs.

Para explorar todas as vantagens de um GA, existem algumas questões básicas que precisam ser cuidadosamente discutidas (Davis, 1987), tais como:

– A definição de um método de codificação, ou seja, representar o problema em forma de cromossomos;

– A geração de uma população inicial;

– A definição de uma função de avaliação que possa avaliar a aptidão do cromossomo;

– A definição de operadores genéticos para reprodução;

– Ajuste dos parâmetros de controle.

Aproximações de roteamento baseadas em GA são citadas na Seção 4.2. Em seguida, na Seção 4.3 são especificadas não somente as questões supracitadas, mas também todas as características do GA utilizado para realizar o roteamento *anycast* em DTNs. O cálculo do

tempo utilizado pelo algoritmo de roteamento baseado em GA até a definição de rotas é mostrado na Seção 4.4. Visando analisar a eficiência das principais estratégias utilizadas na proposta, as aproximações de roteamento *anycast* baseadas em GA que serão analisadas são citadas na Seção 4.5. Por fim, a Seção 4.6 apresenta as considerações finais do capítulo.

4.2. Algoritmos de roteamento baseados em algoritmos genéticos

Na literatura, é encontrado um grande número de aproximações de roteamento baseadas em GAs. O problema do roteamento SP (*Shortest Path*) utilizando GAs é tratado em muitos trabalhos, com alguns utilizando cromossomos de tamanho variável (Munetomo et al., 1998; Ahn e Ramakrishna, 2002) e outros com tamanho constante (Inagaki et al., 1999; Chen e Sun, 2005). Na Seção 4.3.2 é visto o esquema de codificação utilizado, com cromossomos de tamanho constante.

Várias aproximações baseadas em GA para o problema de roteamento *multicast* podem ser encontradas (Leung et al., 1998; Xianwei et al., 2000; Zhang e Leung, 1999; Chen e Sun, 2005; Randaccio e Atzori, 2007). As principais diferenças nessas propostas estão nas várias representações para o cromossomo, diferentes objetivos para o roteamento, soluções de problemas baseados em alguma restrição, além das características da rede para a qual foram projetadas. Embora algumas ideias desses trabalhos possam ser utilizadas, o interesse aqui é o roteamento *anycast*, ou seja, selecionar o melhor nó possível entre os nós do grupo de receptores para fornecer um serviço requerido pelos usuários com qualidade, com as aproximações acima saindo fora do escopo deste trabalho.

A aplicação de GA para realizar o roteamento *anycast* em redes DTN, que seja do nosso conhecimento, é um trabalho inédito. Mais precisamente, o GA será responsável por realizar o roteamento para entrega *anycast*, e algumas aproximações dos trabalhos citados acima são utilizadas e descritas no decorrer deste capítulo.

4.3. Roteamento *anycast* em redes tolerantes a atrasos e desconexões baseado em algoritmos genéticos

O fluxograma geral do GA, utilizado para otimizar o desempenho do roteamento buscando a combinação apropriada de rotas e destinos de cada sessão *anycast*, é mostrado na Figura 4.1. Entende-se por sessão *anycast* a manifestação da intenção de cada nó fonte enviar mensagens para o respectivo grupo de receptores. As seções subsequentes descrevem cada passo do fluxograma.

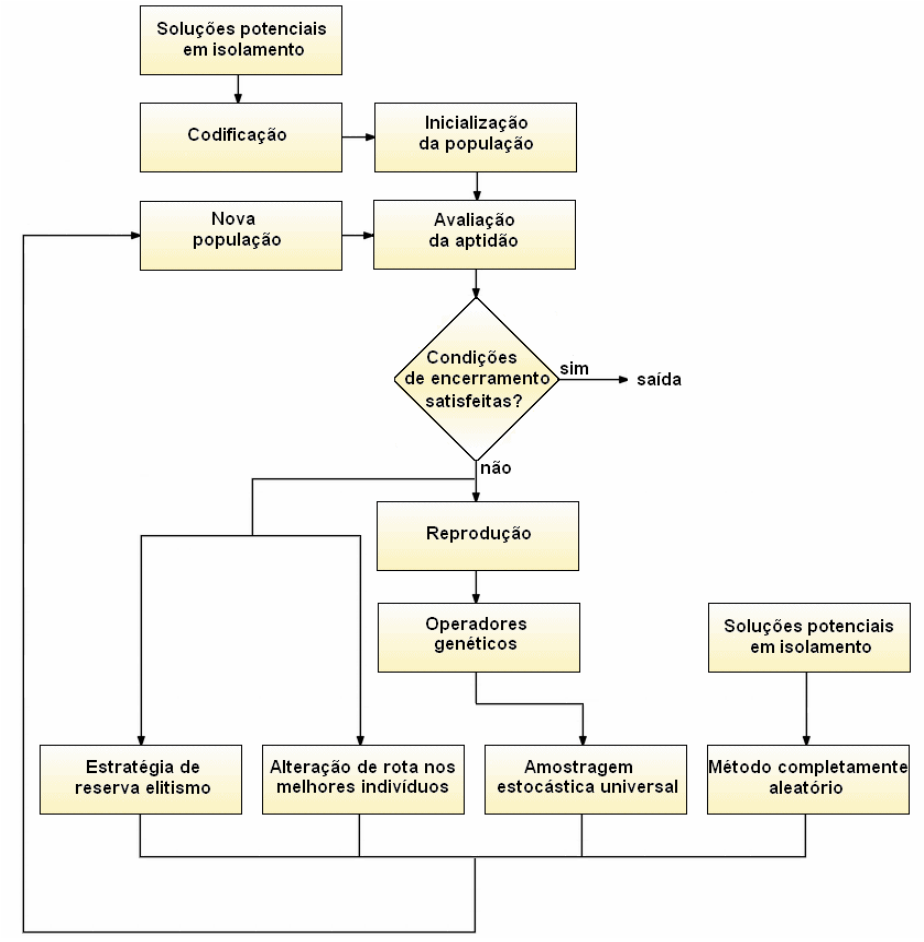

Figura 4.1. Fluxograma do GA utilizado para o roteamento.

4.3.1. Soluções potenciais em isolamento

O algoritmo do menor caminho de Dijkstra (1959) pode ser utilizado para o cálculo dos menores caminhos, em termos de custo, entre nós de uma rede. Inicialmente, escolheu-se um algoritmo de roteamento SP simples para cálculo dos menores caminhos entre pares de nós fonte e destino, considerando o número de saltos como função custo, ou seja, levou-se em consideração apenas o número de saltos, consequentemente desprezando-se as características temporais dos enlaces na rede.

O conjunto de soluções potenciais para o roteamento *anycast*, calculadas de forma isolada utilizando o algoritmo de Dijkstra do menor caminho SP com base no número de saltos, é definido como soluções potenciais em isolamento. Assim sendo, soluções potenciais para o roteamento *anycast* são consideradas em isolamento através de caminhos *unicast* conectando cada par fonte-destino de uma sessão. Para uma rede com *z* sessões *anycast*, um conjunto de nós fonte *anycast* S_i (com $i = 1,2,\ldots,z$) pode enviar mensagens para um conjunto de grupos de receptores de destino $K_{i,j}$, com *j* representando o número de receptores planejados *l* de cada sessão *anycast z*:

$$S_i = [s_1, s_2, s_3, ..., s_z]$$

$$K_{i,j} = \begin{bmatrix} k_{1,1} & k_{1,2} & \ldots & k_{1,l_1} \\ k_{2,1} & k_{2,2} & \ldots & k_{2,l_2} \\ \vdots & \vdots & \vdots & \vdots \\ k_{z,1} & k_{z,2} & \ldots & k_{z,l_z} \end{bmatrix}$$

Para demonstrar como as soluções potenciais em isolamento são obtidas, suponha a rede simples mostrada na Figura 4.2 com uma sessão *anycast*. A rede possui nove nós: n_1 até n_9. O nó fonte n_1 deseja enviar mensagens a qualquer um dos três possíveis receptores n_5, n_8 e n_9 (receptores planejados). Analisando a rede, a menor rota seria n_1-n_4-n_5. A próxima rota poderia ser n_1-n_2-n_4-n_5, e assim, sucessivamente, aplica-se o algoritmo do menor caminho até

encontrar-se o número de soluções potenciais em isolamento desejado. Desta forma, as soluções potenciais em isolamento formam a matriz de rotas F_i para a sessão *anycast* da rede em questão.

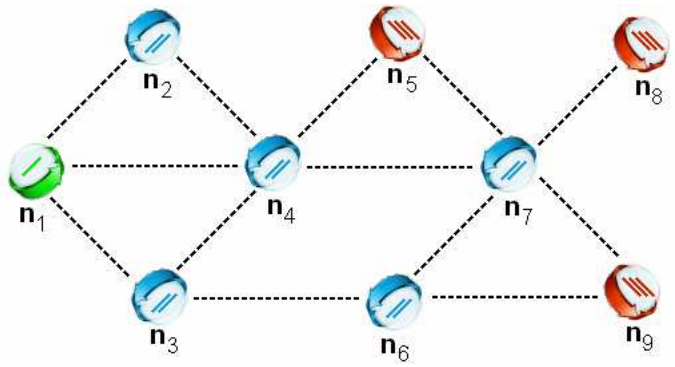

Figura 4.2. Rede para cálculo de soluções potenciais em isolamento.

$$F_i = \begin{bmatrix} n_1 & n_4 & n_5 & 0 & 0 & \cdots & 0 \\ n_1 & n_2 & n_4 & n_5 & 0 & \cdots & 0 \\ n_1 & n_3 & n_4 & n_5 & 0 & \cdots & 0 \\ n_1 & n_3 & n_6 & n_9 & 0 & \cdots & 0 \\ n_1 & n_4 & n_7 & n_8 & 0 & \cdots & 0 \\ n_1 & n_4 & n_7 & n_9 & 0 & \cdots & 0 \\ n_1 & n_2 & n_4 & n_7 & n_8 & \cdots & 0 \\ n_1 & n_2 & n_4 & n_7 & n_9 & \cdots & 0 \\ n_1 & n_3 & n_4 & n_7 & n_8 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & 0 \\ n_1 & \cdots & \cdots & \cdots & \cdots & \cdots & n_9 \end{bmatrix}$$

Dado que as soluções potenciais em isolamento são fornecidas como entrada para o algoritmo de roteamento, a combinação de todas as possibilidades das mesmas representa o espaço de busca. Para se ter uma dimensão do espaço de busca, supondo uma rede com dez sessões *anycast*, e cada uma dessas sessões tendo em média dez soluções potenciais em isolamento, o espaço de busca seria 10^{10}. Aumentando-se para 20 a quantidade de sessões *anycast* e permanecendo com o mesmo número de soluções potenciais em isolamento, o número de combinações passaria a 10^{20}. Assim sendo, o número de combinações possíveis é

dado pelo número médio de soluções potenciais em isolamento das sessões *anycast* elevado pelo número total de sessões *anycast*. Em vista disso, a complexidade de se encontrar a solução com a combinação ótima cresce com o número de sessões e soluções potenciais em isolamento, necessitando de uma ferramenta de busca e otimização eficiente para a solução de tal problema complexo. Como os GAs são apropriados para busca de soluções em espaços complexos, a proposta de algoritmo de roteamento *anycast* faz uso de GAs.

Dependendo do número de nós, enlaces e sessões o número de soluções potenciais pode aumentar muito. As reticências na matriz F_i indicam duas maneiras de limitar o tamanho da matriz: pelo número de saltos (colunas) ou pelo número de soluções encontradas (linhas). Como cada sessão *anycast* possui uma matriz de soluções potenciais em isolamento, é analisada a proposta de limitar-se esse número de soluções para cada sessão *anycast*, que são os menores caminhos possíveis entre o par nó fonte e um dos possíveis destinos *anycast* de cada sessão, calculados utilizando o algoritmo de Dijkstra do menor caminho. Atenção especial foi dada a essa limitação do número de soluções potenciais em isolamento, pois se busca um número de soluções potenciais suficiente para que boas soluções não sejam perdidas. Por outro lado, o número deve ser reduzido visando diminuir o espaço de busca do algoritmo de roteamento *anycast* baseado em GA, evitando assim, que o algoritmo encontre soluções levando grandes quantidades de tempo.

Essas soluções potenciais são utilizadas pelo algoritmo de roteamento baseado em GA. Randaccio e Atzori (2007) propõem a redução do número de soluções para serem avaliadas, com uma grande melhora no tempo de processamento. Entretanto, no presente trabalho, o método de obtenção de soluções potenciais em isolamento e de limitação das mesmas é totalmente diferente. Assim sendo, limitou-se o número de soluções potenciais para cada sessão *anycast*, que são os menores caminhos possíveis entre o par nó fonte e um dos possíveis destinos *anycast* de cada sessão, baseado no número de nós da rede (15% do

número de nós) e calculados utilizando o algoritmo de Dijkstra do menor caminho. Para uma rede com 40 nós, seriam consideradas seis soluções potenciais em isolamento, e na rede da Figura 4.2 só seriam consideradas as duas primeiras linhas da matriz F_i (15% de nove nós, aproxima-se para o número inteiro imediatamente acima do valor encontrado). No Capítulo 6, é visto que essa limitação produz resultados com as características desejadas (soluções boas não são perdidas e o algoritmo de roteamento *anycast* baseado em GA encontra soluções em um tempo reduzido). A presença de zeros na matriz é para fazer com que todas as linhas da matriz tenham o mesmo tamanho. Veremos, na próxima seção, que essa representação será útil para a codificação do cromossomo na proposta de algoritmo de roteamento *anycast* baseado em GA.

4.3.2. Método de codificação do cromossomo e formação da população inicial

Nesta etapa é necessária a definição de um método para codificar soluções potenciais em cromossomos. Essa codificação em forma de cromossomo é fundamental por traduzir a informação do problema, ou seja, quanto mais adequada for ao problema, maior a qualidade dos resultados obtidos. Para o problema do roteamento, o cromossomo é formado pela combinação de rotas de cada sessão *anycast* que irá representar uma solução possível, e seu comprimento é proporcional ao número de sessões *anycast*. Para formação de um indivíduo da população, seleciona-se uma rota para a primeira sessão, outra para a segunda, até a sessão *z*, utilizando para isso as soluções potenciais em isolamento descritas na seção anterior. O esquema de representação utilizado é mostrado na Figura 4.3 (baseado na associação de cada gene, isto é, unidade básica com posição específica no cromossomo, a cada nó da rota entre os nós fonte e destino). Neste esquema s_i (com $i = 1,2,\cdots,z$) representa os nós fonte, n_x (com

$x = m, n, \ldots, q$) os nós intermediários e $k_{i,j}$ (com $i = 1,2,\cdots,z$ e j = número de receptores planejados, sendo $1 \leq j \leq l$) os destinos escolhidos. sp_1 até sp_z são as posições iniciais de cada sessão, que estão contidas em um vetor. Para manter essas posições constantes para todos os indivíduos, alguns genes são preenchidos com valores zero. Essa adição de redundância é para evitar o surgimento de indivíduos indesejáveis, ou seja, são criados apenas indivíduos regulares. Com isso, evita-se a necessidade de utilização de funções de reparo e de verificação para testar se os indivíduos são válidos.

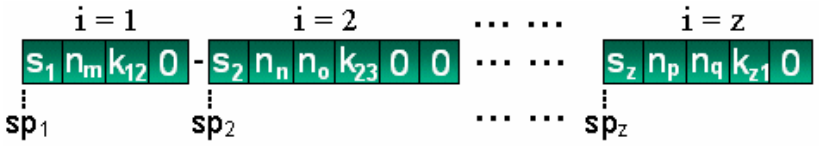

Figura 4.3. Esquema de codificação do cromossomo.

O passo seguinte mostrado no fluxograma do GA utilizado é a criação da população inicial. Para isso, selecionam-se aleatoriamente rotas em isolamento para cada sessão, ou seja, para cada conjunto de rotas disponíveis para as sessões *anycast*, escolhe-se uma rota aleatoriamente de cada conjunto, da primeira sessão até a última, formando assim um indivíduo da população. O mesmo procedimento é repetidamente executado até formar todos os indivíduos da população.

4.3.3. Método de avaliação dos indivíduos da população

Para avaliação da aptidão dos indivíduos da população são utilizadas funções de aptidão que devem representar precisamente a qualidade do cromossomo na população. Consequentemente, a definição da função de aptidão é bastante crítica (Hue, 1997). O algoritmo de roteamento sendo projetado considera DTNs em cenários determinísticos, ou seja, que os contatos são previsíveis. Por isso, utiliza-se fragmentação pró-ativa na fonte, isto

é, blocos de dados podem ser fragmentados apenas no nó fonte. Como o objetivo é a busca de combinações de rotas *anycast* visando à otimização do desempenho da entrega de mensagens, para avaliação da aptidão dos indivíduos da população, são utilizadas medidas de desempenho da rede. Mais precisamente, são consideradas duas medidas de desempenho que fazem uso das informações provenientes da representação de cada *edge* no grafo DTN (Seção 3.4.2):

1) Probabilidade de entrega (*PE*) das mensagens:

$$PE = \sum_{a=1}^{z} m_k(a) / m_s(a, f) \qquad f = 1,2,3,\ldots,N_f \tag{4.1}$$

com

$$m_k(a) = \sum_{f=1}^{N_f} m_s(a, f) \qquad a = 1,2,3,\ldots,z \tag{4.2}$$

sujeito à

$$\sum_{(i,j)\subset Fa} x_{ij} m_s(a, f) < c(i, j) \tag{4.3}$$

$$\sum_{(i,j)\subset Fa} x_{ij} [md(i, j) + w(i, j)] < t_{total} \tag{4.4}$$

onde

- $m_k(a)$ número estimado de mensagens *anycast* recebidas por um membro do grupo *anycast* de cada sessão *anycast z*;
- $m_s(a,f)$ número de mensagens transmitidas por cada nó fonte s_a, com $a = 1,2,3,\ldots,z$. f indica os fragmentos (considera-se que apenas o nó fonte possui a capacidade de fragmentar mensagens);
- z número de sessões *anycast*;
- N_f número total de fragmentos;
- F_a matriz contendo soluções potenciais em isolamento;
- $c(i,j)$ capacidade de armazenamento dos dispositivos móveis;
- $md(i,j)$ atraso de movimento dos dispositivos móveis;
- $w(i,j)$ tempo de partida dos dispositivos móveis;
- t_{total} tempo total simulado;

x_{ij} 1 se o enlace (i,j) é selecionado, caso contrário é 0.

Assim sendo, a *PE* é a proporção do número estimado de mensagens *anycast* recebidas $m_k(z)$ (restritas pela capacidade de armazenamento dos dispositivos móveis eq. (4.3) e pelo tempo total simulado eq. (4.4)) pelo número de mensagens transmitidas por cada nó fonte *anycast* $m_s(a,f)$. Desta forma, a primeira restrição busca evitar que as mensagens repassadas excedam a capacidade dos dispositivos móveis, e a segunda restrição faz com que as mensagens sejam entregues antes da expiração do tempo total simulado (funcionando como um tempo de vida).

2) Para o cálculo do atraso é definido o atraso médio ponderado ou atraso total (D) para todas as mensagens entregues, refletindo o tempo médio gasto entre o instante que as mensagens são geradas por cada nó fonte até serem entregues ao receptor de destino:

$$D = \sum_{f=1}^{N_f} \left(w_f \cdot D_z(f) \right) \Big/ \sum_{f=1}^{N_f} w_f \tag{4.5}$$

onde N_f é o número total de fragmentos, w_f é o peso de cada fragmento proporcional ao número de fragmentos f e $D_z(f)$ é o atraso de cada fragmento, ou seja, é a soma do atraso médio $d(i,j)$ de cada salto formando a rota fonte-destino:

$$D_z(f) = \sum\nolimits_{s_z \le i \le k_{z-1}, s_{z+1} \le j \le k_z} d(i,j) \tag{4.6}$$

onde s_z representa o nó fonte de cada sessão z e k_z o receptor do grupo de destino. O atraso médio $d(i,j)$ é a soma dos tempos de partida dos dispositivos móveis $w(i,j)$ e do atraso de movimento $md(i,j)$:

$$d(i,j) = w(i,j) + md(i,j) \tag{4.7}$$

Deste modo, a aptidão do indivíduo é determinada de acordo com a *PE* da eq. (4.1) e o atraso total (*D*) da eq. (4.5). Além disso, uma característica do algoritmo de roteamento *anycast* baseado em GA é o cálculo de rotas com uma probabilidade de entrega mínima ($PE_{\min}$) acima de um *threshold* e com o menor atraso. A Figura 4.4 mostra um fluxograma descrevendo como o GA coloca os indivíduos em ordem de aptidão. Primeiramente, os indivíduos da população da geração corrente são colocados em ordem decrescente de probabilidade de entrega. Depois são verificados os indivíduos que possuem uma probabilidade de entrega maior que um valor mínimo $PE_{\min}$. Os indivíduos que possuem probabilidade de entrega maior que $PE_{\min}$ são dispostos em ordem crescente de atraso e, juntamente com os que não possuem probabilidade de entrega maior do que $PE_{\min}$, formam a população colocada em ordem de aptidão.

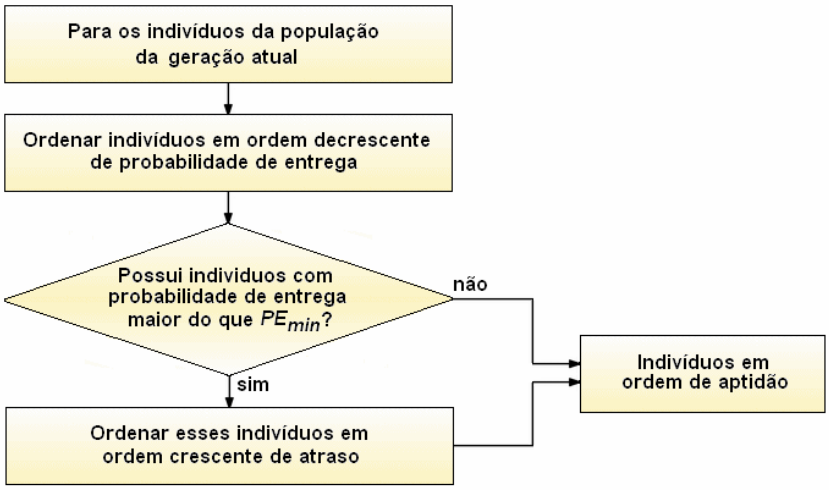

Figura 4.4. Ordenamento da população em ordem de aptidão.

As informações utilizadas pelo algoritmo de roteamento *anycast* baseado em GA, isto é, $c(i,j)$, $d(i,j)$ e $md(i,j)$, são trocadas entre os nós da rede através do esquema de roteamento epidêmico (Vahdat e Becker, 2000). Tanto Xiao et al. (2010), como Demmer e Fall (2007), também utilizam o esquema de roteamento epidêmico para distribuir informações de roteamento entre os nós da rede.

4.3.4. Reprodução

A condição de terminação do GA utilizada é o número de gerações (cada uma das repetições do *loop* principal do fluxograma da Figura 4.1). Caso essa condição não seja satisfeita, a população da próxima geração (ou nova população) deve ser formada. Assim sendo, o GA armazena os indivíduos com os melhores valores de aptidão através da estratégia de reserva elitismo e inicia o processo de reprodução. Normalmente, deve-se escolher um método de seleção que privilegie os indivíduos com valores de aptidão altos, sem desprezar completamente os indivíduos com valores de aptidão baixos, pois indivíduos com valores de aptidão ruins podem ter características genéticas favoráveis à criação de um indivíduo que seja a melhor solução para o problema. Em vista disso, para seleção dos indivíduos para reprodução é utilizada a técnica de seleção por torneio aos pares. Nela, dois indivíduos da população são escolhidos aleatoriamente, e o que tiver o melhor valor de aptidão será utilizado para reprodução, e assim sucessivamente, até formar todos os indivíduos utilizados para reprodução.

Os operadores genéticos utilizados são os de cruzamento e de mutação. Em alguns casos, pode ser que a aplicação dos operadores genéticos gere indivíduos ilegais, tornando necessários esquemas de verificação de indivíduos (Ahn e Ramakrishna, 2002; Oh et al., 2006). Como o algoritmo proposto utiliza o vetor com as posições iniciais de cada rota, representadas por $sp_{1\text{-}z}$, para realizar o cruzamento e a mutação, apenas indivíduos regulares são gerados, tornando desnecessária a utilização de funções de verificação e/ou reparo. Para ilustrar o funcionamento dos operadores genéticos são considerados os indivíduos X1 e X2 mostrados na Figura 4.5.

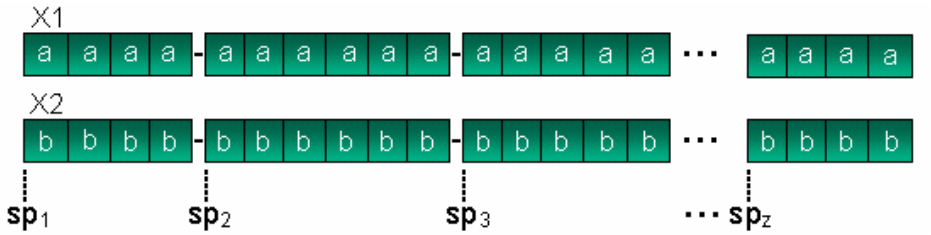

Figura 4.5. Indivíduos X1 e X2.

O cruzamento de um ponto é realizado escolhendo-se de forma aleatória a posição inicial de uma das rotas representada por sp_2, sp_3, até sp_z, que será o ponto de corte (constitui uma posição entre dois genes de um cromossomo). Depois de selecionados dois indivíduos pelo método de seleção, escolhe-se uma posição aleatória, por exemplo, sp_3, que indicará a posição a partir da qual o material genético dos indivíduos será trocado. A Figura 4.6 mostra os indivíduos Y1 e Y2 resultantes do cruzamento dos indivíduos X1 e X2 no ponto sp_3, com os indivíduos X1 e X2 podendo ser denominados de pais. Assim, depois de sorteado o ponto de corte sp_3, separa-se os indivíduos X1 e X2 em duas partes: uma à esquerda do ponto de corte e outra à direita. O primeiro filho Y1 é composto através da concatenação da parte esquerda do primeiro indivíduo X1 com a parte direita do segundo pai X2. Para o segundo filho ocorre o processo inverso, ou seja, compõe-se o filho Y2 através da concatenação da metade esquerda do segundo indivíduo X2 com a metade à direita do primeiro indivíduo X1.

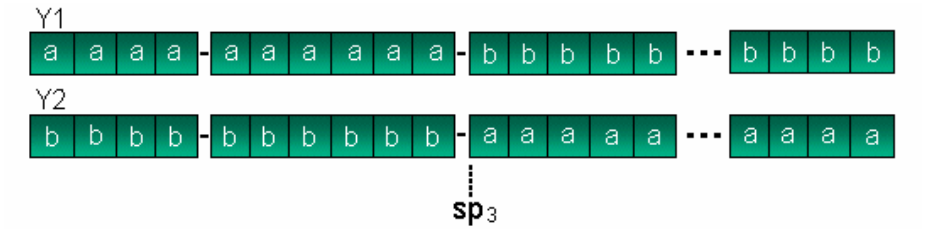

Figura 4.6. Cruzamento de um ponto.

Há também o cruzamento de dois pontos, onde são escolhidas duas posições aleatórias, que compreendem as posições entre as quais o material genético dos indivíduos será trocado. A Figura 4.7 mostra os indivíduos Y1’ e Y2’ resultantes do cruzamento dos indivíduos X1 e

X2 com os pontos de cruzamento sendo sp_2 e sp_3. Assim, Y1' será formado pela parte do indivíduo X1 fora dos pontos de corte e pela parte do indivíduo X2 entre os pontos de corte. Já o indivíduo Y2' será formado pelas partes restantes.

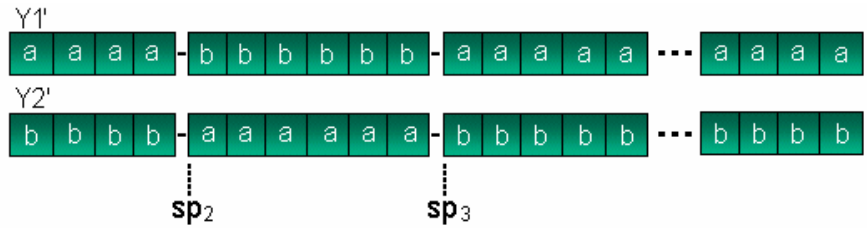

Figura 4.7. Cruzamento de dois pontos.

O operador de mutação é essencial para auxiliar a continuidade da diversidade genética da população. A mutação é realizada alterando-se de forma aleatória uma das rotas do indivíduo. Para isso, seleciona-se uma rota aleatoriamente dentro do conjunto de soluções potenciais em isolamento, associada à sessão *anycast* em questão, para substituir a rota compreendida entre as posições sp_z e sp_{z+1}, com z representando a sessão para a qual a rota será substituída. A Figura 4.8 ilustra X' resultante da mutação da segunda rota (z = 2) do indivíduo X1. Para delimitar a mutação são utilizados os vetores de posição inicial sp_2 e sp_3.

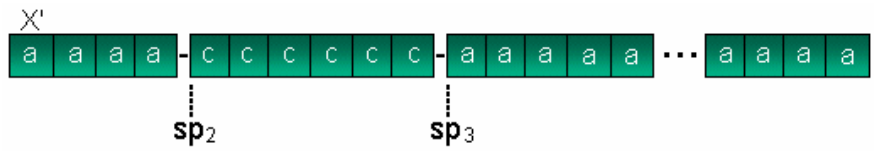

Figura 4.8. Mutação.

Nesta seção os operadores genéticos foram exemplificados utilizando uma representação genérica através de letras. Evidentemente, a rota de cada sessão *anycast* é representada pelos nós através dos quais a mensagem será encaminhada. Por isso, um exemplo de aplicação dos operadores genéticos, com os cromossomos sendo representados pelas rotas de cada sessão, é apresentado no Apêndice A.

Após os operadores genéticos serem aplicados, utiliza-se o conceito de subpopulação (população da próxima geração formada por um misto de subpopulações) para formar a população da próxima geração.

4.3.5. Formação da população da próxima geração

O GA proposto utiliza o conceito de subpopulação, proposto por Schaffer (1985), com a população da próxima geração sendo um misto de outras quatro subpopulações (Figura 4.9).

Figura 4.9. Conceito de subpopulação para formação da população da próxima geração.

1) A primeira subpopulação é formada pela seleção de cromossomos com os melhores valores de aptidão. Nessa estratégia, os cromossomos com as melhores aptidões são preservados e levados ao longo das gerações, como em (Tamaki et al., 1996). Este procedimento é denominado de elitismo. Como visto na Seção 4.3.3, os indivíduos são colocados em ordem de aptidão. Deste modo, supondo que a população tenha *ni* indivíduos em ordem de aptidão, o melhor indivíduo será o indivíduo da posição 1, o segundo melhor, o da posição 2, e assim sucessivamente, até chegar ao indivíduo *ni*. Na prática o elitismo resulta numa busca mais agressiva e, geralmente, é bastante efetiva. No entanto, existe o perigo de uma convergência prematura para mínimos locais.

2) A substituição de uma rota nos cromossomos com os melhores valores de aptidão forma a segunda subpopulação. A estratégia desta subpopulação é permitir que os melhores indivíduos não sejam perdidos, assim como no elitismo, juntamente com a alteração de uma das rotas nos cromossomos a fim de buscar indivíduos bem próximos aos melhores já encontrados. Para ilustrar a estratégia, supõe-se que um dos melhores indivíduos da população X1 é mostrado na Figura 4.10. Após a substituição de uma rota no cromossomo deste indivíduo, seria produzido um indivíduo X1' bem semelhante ao anterior, com exceção da rota de uma sessão (no exemplo, a rota da terceira sessão foi alterada).

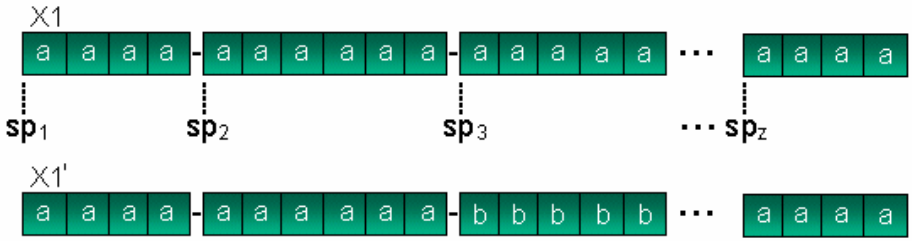

Figura 4.10. Substituição de uma rota em um cromossomo.

3) Uma amostragem estocástica universal é realizada, como sugerido em (Baker, 1985), com esse método também sendo conhecido como método da roleta. Esta subpopulação utiliza os indivíduos resultantes da reprodução. Cada indivíduo tem uma probabilidade de ser selecionado proporcional à sua aptidão. Para visualizar este método considere um círculo dividido em algumas regiões (tamanho da população), onde a área de cada região é proporcional à aptidão do indivíduo (Figura 4.11). Coloca-se sobre este círculo uma "roleta" com cursores igualmente espaçados. Após um giro da roleta a posição dos cursores indica os indivíduos selecionados. Evidentemente, os indivíduos cujas regiões possuem maior área terão maior probabilidade de serem selecionados várias vezes. Como consequência, a seleção de indivíduos pode conter várias cópias de um mesmo indivíduo enquanto outros podem desaparecer.

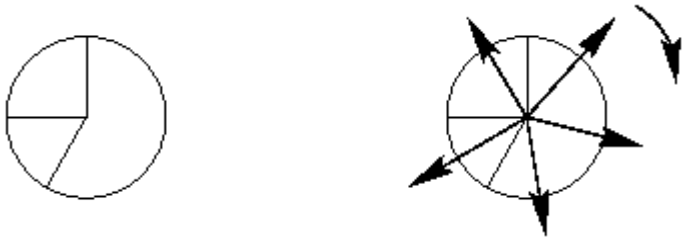

Figura 4.11. Método da roleta.

4) A última subpopulação é formada por um método completamente aleatório, ou seja, indivíduos são gerados de forma aleatória. Esse mesmo método foi utilizado para gerar a população inicial, e faz uso das soluções potenciais em isolamento. O maior motivo da utilização desta subpopulação é a manutenção da diversidade da população.

No procedimento de seleção de indivíduos de um GA existem dois fatores competindo: pressão de seleção (*selection pressure*) e a diversidade da população. A pressão de seleção representa a intensidade com a qual melhores indivíduos são favorecidos. Assim, quando a pressão de seleção é alta, significa que a chance de se escolher indivíduos mais aptos é muito grande. Um aumento na pressão de seleção diminui a diversidade de população e vice-versa. O método de amostragem estocástica universal aumenta a pressão de seleção, o que pode causar uma convergência prematura do GA. Para diminuir a pressão de seleção, o método completamente aleatório é utilizado. Todavia, os melhores cromossomos da geração atual podem ser perdidos. Este problema é resolvido incluindo-se a utilização do elitismo. Por fim, a estratégia de substituição de uma rota em um cromossomo é adicionada ao processo de seleção com a finalidade de auxiliar na busca da solução ótima global.

Desta forma, foi utilizado o conceito de subpopulação pelo mesmo se mostrar eficiente e utilizar diferentes valores de pressão de seleção, como também pode ser visto nos trabalhos (Lo e Chang, 2000; Schaffer, 1985).

Assim, quatro subpopulações formam a população da próxima geração. Entretanto, a população da próxima geração terá quantidades diferentes de indivíduos formados por cada subpopulação (Tabela 4.1).

Tabela 4.1. Distribuição dos indivíduos para formação da população da próxima geração.

	Subpopulação	Quantidade de indivíduos (%)
1	Estratégia de reserva elitismo	10%
2	Substituição de um elemento do cromossomo	10%
3	Amostragem estocástica universal	20%
4	Método completamente aleatório	60%

Os valores da Tabela 4.1 indicam que a estratégia utilizada foi manter a diversidade da população através da última subpopulação, método completamente aleatório, representando 60% da formação da população da próxima geração. Para justificar a utilização dessa distribuição, observou-se que as três primeiras subpopulações, representando 40% da formação da população da próxima geração, tendem a produzir uma população com pouca diversidade, o que poderia causar uma convergência prematura do GA. Por isso, para aumentar a diversidade da população, o método completamente aleatório foi utilizado. A análise do algoritmo de roteamento *anycast* baseado em GA proposto é realizada no Capítulo 6, sendo mostrado que essa distribuição das subpopulações produz um bom desempenho.

Após a população da próxima geração ser formada, a aptidão de cada indivíduo é avaliada e ordenada como descrito anteriormente. As etapas de reprodução, aplicação de operadores genéticos, formação das quatro subpopulações para formação da população da próxima geração, e avaliação da mesma, são realizadas até que a condição de encerramento seja satisfeita. A condição de encerramento do algoritmo utilizada foi o número de gerações. Entretanto, outras condições de encerramento podem ser utilizadas, como por exemplo, o tempo de partida do dispositivo móvel $w(i,j)$ que forma o primeiro *hop* de cada sessão *anycast*.

4.3.6. Definição dos parâmetros de controle

Nesta seção são vistos os ajustes dos parâmetros de controle do GA. Conforme dito anteriormente, o algoritmo de roteamento *anycast* baseado em GA, proposto neste trabalho, é controlado pelo número de gerações. Esse número de gerações é feito proporcional ao número de sessões *anycast* presentes na rede, pois como o cromossomo possui tamanho proporcional ao número de sessões *anycast*, à medida que se aumenta o número de sessões, aumenta-se o tamanho do cromossomo também, e consequentemente, a complexidade do problema. Com isso, um número maior de gerações é necessário.

Um tamanho de população que garanta a obtenção de uma solução ótima em um tempo adequado é um tópico de pesquisa intensa (Harik et al., 1999; Ahn e Ramakrishna, 2002). Grandes populações normalmente resultam em soluções melhores, mas ao custo do aumento computacional. Por outro lado, se a população é muito pequena, provavelmente o GA não irá encontrar soluções de alta qualidade. Além disso, quando o número de genes torna-se grande, ou seja, têm-se várias sessões *anycast* e/ou as rotas de cada sessão possuem vários saltos, uma população maior será necessária para alcançar uma solução particular de qualidade.

Um estudo realizado por Jong (1975) sugere a utilização de uma probabilidade de cruzamento p_c alta e uma probabilidade de mutação p_m baixa (inversamente proporcional ao tamanho da população). Sorteia-se um número aleatoriamente entre zero e um. Se ele for menor que a probabilidade p_c, então o operador de cruzamento atua sobre os indivíduos em questão. O mesmo ocorre com o operador de mutação, para cada indivíduo, considerando a probabilidade de mutação p_m.

4.4. Tempo de convergência do algoritmo de roteamento *anycast* baseado em algoritmos genéticos

Como a proposta de algoritmo de roteamento *anycast* baseado em GA é controlada pelo número de gerações, esse número deve ser adequado, de forma a permitir a obtenção de rotas ótimas por parte do algoritmo e procurando utilizar o mínimo tempo possível. Para avaliar o tempo que o algoritmo de roteamento *anycast* baseado em GA leva até encontrar a melhor combinação de rotas, define-se o tempo médio de simulação até o algoritmo convergir (t_{sim}). Uma vez que a proposta de algoritmo de roteamento *anycast* baseado em GA é controlada pelo número de gerações, o tempo médio de simulação até o algoritmo convergir t_{sim} é calculado através da seguinte equação:

$$t_{sim} = t_{total} \cdot g_{convergir} \big/ g_{total} \tag{4.8}$$

onde t_{total} é o tempo total gasto pelo algoritmo até atingir a condição de terminação do GA, ou seja, o número de gerações total (g_{total}). $g_{convergir}$ é o número total de gerações que o algoritmo leva até obter rotas boas e/ou ótimas (geração na qual houve a última mudança no melhor indivíduo encontrado pelo GA).

A medida t_{sim} tem um papel importante, pois irá indicar se o algoritmo de roteamento *anycast* baseado em GA obtém rotas boas e/ou ótimas em intervalos de tempo adequados.

4.5. Aproximações de roteamento *anycast* baseadas em GA

Para análise das principais estratégias utilizadas na proposta, o desempenho de três aproximações baseadas em GA é comparado. Primeiramente, uma aproximação baseada em GA (GA1) que considera apenas a estratégia de reserva elitismo é analisada. Em seguida, uma aproximação baseada em GA (GA2), que utiliza a estratégia de reserva elitismo e um número limitado de soluções potenciais em isolamento, é estudada. Por fim, uma aproximação

baseada em GA (GA3), que utiliza o conceito de subpopulação e um número limitado de soluções potenciais em isolamento, é proposta e analisada. As aproximações baseadas em GA são sumarizadas na Tabela 4.2.

Tabela 4.2. Aproximações baseadas em GA.

Algoritmo	Número limitado de soluções potenciais em isolamento	Estratégia de reserva elitismo	Conceito de subpopulação
GA1	Não	Sim	Não
GA2	Sim	Sim	Não
GA3	Sim	Sim	Sim

4.6. Considerações finais

Neste capítulo, foram especificadas todas as características do GA proposto para realizar o roteamento *anycast* em DTNs. O algoritmo irá buscar a melhor combinação de rotas de cada sessão *anycast* utilizando o menor número de gerações possível. Para isso, aproximações visando aumentar a eficiência do algoritmo foram descritas. O próximo capítulo descreve o ambiente de simulação utilizado para avaliação da proposta de algoritmo de roteamento *anycast* baseado em GA.

Capítulo 5

PROJETO DO AMBIENTE DE SIMULAÇÃO PARA O ROTEAMENTO *ANYCAST* EM REDES TOLERANTES A ATRASOS E DESCONEXÕES

5.1. Introdução

O roteamento em DTNs é um dos principais componentes da arquitetura DTN. Diferentes tipos de DTN requerem diferentes protocolos de roteamento. Neste capítulo, é especificado todo o ambiente de rede para avaliação da proposta de algoritmo de roteamento *anycast* baseado em GA. A utilização de simulação para análise de desempenho da proposta de roteamento *anycast* é discutida na Seção 5.2, assim como alguns simuladores de rede existentes são brevemente citados. O ambiente de simulação construído para implementar as características das DTNs é descrito na Seção 5.3. As medidas de desempenho e os parâmetros do ambiente de simulação utilizados são mostrados na Seção 5.4. Finalmente, a Seção 5.5 apresenta as considerações finais do capítulo.

5.2. Utilização de simulação para avaliação da proposta de algoritmo de roteamento *anycast* baseado em GA

A ferramenta utilizada para implementação e avaliação da proposta de roteamento *anycast* em DTNs foi a simulação, que tem emergido como uma ferramenta popular para avaliação do desempenho de algoritmos e protocolos complexos em redes de computadores. Esta seção descreve o método utilizado para definição das características do simulador.

5.2.1. Simulação

Uma técnica usual para avaliação do desempenho de redes é a simulação, que tem sua utilização motivada por algumas razões:

1. Se a rede apresentada não for disponível para medição, a simulação provê uma maneira conveniente de predizer o seu desempenho;

2. Mesmo se a rede está disponível para medição, a simulação pode ainda ser recomendada por permitir a avaliação do desempenho sob uma imensa variedade de cargas de trabalho e condições de rede;

3. Simulações permitem uma análise de desempenho para comparação de diversas arquiteturas alternativas sob condições de rede idênticas e repetidas;

4. Simulações podem incorporar mais detalhes do que a modelagem analítica; assim, frequentemente são produzidos resultados mais próximos da realidade;

5. Pesquisadores frequentemente usam resultados de simulações mais detalhadas para validar resultados analíticos. Uma “equiparação” entre resultados analíticos e resultados de simulação dá mais confiança ao usuário.

Diante da dificuldade de implementar uma rede DTN em larga escala com as características descritas na Seção 3.4.2 do Capítulo 3, e a complexidade de se realizar uma

modelagem analítica detalhada de todos os componentes da rede, a ferramenta que se mostrou mais adequada para análise da proposta foi a simulação.

A fim de investigar e validar a proposta de algoritmo de roteamento com entrega *anycast* baseado em GA, diferentes cenários precisam ser testados utilizando simulação, de modo que a taxa de entrega das mensagens, o atraso, a distribuição das mesmas na rede e outras métricas necessárias possam ser avaliadas de maneira adequada. Além disso, os algoritmos de roteamento SP e ED são implementados e comparados com a proposta de algoritmo de roteamento baseado em GA visando demonstrar sua viabilidade. Para decisão de escolha do simulador de rede, os requisitos de projeto apresentados no Capítulo 3, como representação da rede utilizando grafos evolutivos e as características das DTNs, foram considerados. Desta forma o simulador deve ser capaz de:

1) Suportar a arquitetura de rede DTN;

2) Representar a rede utilizando grafos evolutivos;

3) Permitir modificações para ser apropriado às necessidades de projeto do algoritmo de roteamento *anycast*;

4) Monitorar medidas de desempenho da rede.

5.2.2. Simuladores investigados

No decorrer da pesquisa, simuladores foram investigados para verificar o quão apropriados eram para testar ambientes que suportassem as necessidades de projeto do algoritmo de roteamento *anycast* em DTNs. O NS-2 (NS-2, 2008) e o OPNET (OPNET, 2008), simuladores muito populares, permitem a simulação de MANETs. Esses simuladores suportam vários protocolos de roteamento. Entretanto, eles não suportam comunicação tolerante a atraso e desconexões com as características do modelo de rede sendo analisado (Seções 3.4.1 e 3.4.2).

O NS-3 (NS-3, 2008), com projeto iniciado em 2006, está focado em melhorar a arquitetura de núcleo, integrar software, modelos e componentes do NS-2. O simulador foi projetado com a intenção de suportar uma arquitetura de redes tolerantes a atrasos e desconexões. Entretanto, no início de 2008, o simulador não tinha qualquer versão disponível, e na própria *homepage*, no link de dúvidas frequentes, usuários que estavam procurando por um simulador para pesquisa eram aconselhados a não utilizá-lo naquele momento.

Outro simulador analisado foi o simulador de ambiente oportunista ONE (*Opportunistic Network Environment*) (ONE, 2008). O ONE é capaz de simular o roteamento de mensagens entre nós com vários algoritmos de roteamento DTN. Entretanto, o ONE foi construído para ambientes oportunistas, ou seja, não suporta a modelagem da rede através de grafos evolutivos com as características buscadas.

5.2.3. Projeto do ambiente de simulação

Diante da pesquisa realizada na seção anterior, não foram encontrados simuladores de redes que atendessem a todos os requerimentos para implementação e análise do algoritmo de roteamento *anycast* sendo projetado. Assim, fica claro que é necessário um simulador para roteamento *anycast* em DTNs determinísticas representadas por grafos evolutivos.

Consequentemente foi desenvolvido um simulador orientado por eventos (*event-driven*) especificamente para ambientes tolerantes a atrasos com utilização de grafos evolutivos. A simulação orientada por eventos induz eventos em um tempo arbitrário, ou seja, o intervalo de tempo entre dois eventos não é fixo, como é o caso da simulação orientada pelo tempo (*time-driven*). Mais informações sobre simulação orientada por evento e orientada pelo tempo podem ser encontradas em (Issariyakul e Hossain, 2009). O simulador trabalha com troca de mensagens ao invés de pacotes como nos simuladores tradicionais. O suporte a algoritmos de roteamento é integrado junto com um gerador de topologia e de tráfego. Além disso, o

simulador fornece possibilidades de utilização de sub-rotinas para controle do gerador de topologia, geração de tráfego, definição de nós e quantidade de nós, variação de vários parâmetros da rede, visualização da topologia utilizada e análise dos dados.

5.3. Construção do ambiente de simulação

Nesta seção, o modelo e o projeto do ambiente de simulação são discutidos. O projeto do simulador leva em consideração questões importantes em DTNs, como tempo e espaço, assim como facilidade na geração de grafos e definição de medidas de desempenho da rede.

5.3.1. Gerador de topologia de Waxman

Primeiramente, foi buscado um gerador de topologia para representação dos grafos evolutivos que permita, através de um ajuste apropriado, a obtenção de um grafo resultante com as características desejadas. Nesse sentido, o gerador de grafo aleatório proposto por Waxman (1988), mostrou-se promissor para geração da topologia. O gerador primeiramente distribui um número N de nós sobre uma área quadrada. Tanto o número de nós, como a área sobre a qual os nós são distribuídos, podem ser escolhidos. Um enlace entre dois nós n_i e n_j quaisquer é adicionado seguindo a função densidade de probabilidade $P(n_i,n_j)$.

$$P(n_i, n_j) = \alpha \cdot \exp(-dist(n_i, n_j) / \beta \cdot L) \tag{5.1}$$

Onde $dist(n_i,n_j)$ é a distância cartesiana entre os nós n_i e n_j, L é a maior distância possível entre dois nós quaisquer, e os parâmetros α e β são números reais no intervalo de zero a um. α indica a densidade de enlaces curtos em relação a enlaces com distâncias maiores entre os nós, e β representa a densidade de enlaces no grafo gerado. O ajuste destes parâmetros permite a geração de um grafo resultante com as características desejadas.

Para simular o comportamento das DTNs, a representação mostrada na Seção 3.4.2 é utilizada. Desta forma, é assumido que toda comunicação entre os nós é realizada através de dispositivos móveis, ou seja, cada *edge* gerado pelo gerador de Waxman é substituído por um dispositivo móvel que explora a mobilidade para possibilitar a comunicação.

Com isso, a utilização de diferentes parâmetros no gerador de topologia irá impactar diretamente nas características da rede obtida. Por exemplo, com valores pequenos de α a rede apresenta enlaces com distâncias menores entre os nós, ou seja, dispositivos móveis que realizam a comunicação entre nós da rede terão que se deslocar por uma distância menor, consequentemente tendo uma influência menor do atraso de movimento $md(i,j)$ na rede do que se fossem utilizados valores grandes de α. Já quando se utiliza um β maior, ou seja, uma densidade de enlace maior, a rede apresenta maiores oportunidades de transmissão. Desse modo, os ajustes dos parâmetros do gerador de topologia influenciam diretamente na representação das características da rede.

5.3.2. Contagem do tempo e espaço

Nesta seção, é descrito como o tempo e espaço são interpretados no simulador. O modelo de tempo utilizado é linear, ou seja, estende-se na direção oposta ao passado até o tempo final da simulação que pode ser definido no simulador. O tempo inicial no simulador é o instante em que ainda não aconteceu nada na rede, ou seja, nenhum nó está requerendo a transmissão de mensagens. Antes do tempo inicial, assume-se que a topologia da rede já está gerada. A partir do instante que se começa a definir os parâmetros de tráfego e os nós emissores e receptores, o tempo começa a ser contado. O tempo final pode ser definido no programa e será o instante no qual são adquiridos os últimos dados para serem analisados a respeito do desempenho da rede.

A unidade de tempo não é limitada pelo simulador a nenhuma unidade, como segundos, minutos ou horas. Unidades do SI (Sistema Internacional) são comumente utilizadas. Por isso, para contagem do tempo a unidade utilizada é o segundo.

Embora o mundo real seja em três dimensões, a comunicação em redes e a mobilidade dos dispositivos podem ser interpretadas em duas dimensões. O espaço é uma área fechada definida e limitada no simulador. Essa área é quadrada e todos os nós e dispositivos móveis são distribuídos na mesma, de acordo com o grafo gerado pelo gerador de topologia de Waxman. Apesar da unidade de comprimento não ser limitada pelo simulador, é utilizado o metro, unidade do SI.

5.3.3. Parâmetros da simulação

Os parâmetros para geração do tráfego podem ser definidos pelo usuário. Depois de gerado o grafo aleatório utilizando o gerador de topologia de Waxman, o usuário define a quantidade de sessões *anycast* que a rede irá possuir durante a simulação. De acordo com a representação das características de cada enlace no grafo gerado (Capítulo 3 Seção 3.4.2), define-se a capacidade de armazenamento (número de mensagens) de cada nó i da rede $b(i)$, e dos enlaces (dispositivos móveis que ligam os nós i e j) $c(i,j)$ que formam o grafo. Para geração da capacidade de armazenamento estabelece-se uma quantidade mínima e máxima de mensagens, e então, gera-se um número aleatório nesse intervalo para cada nó e enlace. Para cada sessão *anycast*, define-se aleatoriamente um nó fonte e o respectivo grupo de receptores planejados. O nó fonte de cada sessão pode enviar um número limitado de mensagens, definido aleatoriamente dentro de um intervalo, para o grupo de destino. Para ilustrar mais detalhes a respeito da geração da topologia de rede e do tráfego inicial, um exemplo é apresentado no Apêndice B.

Para cada um dos dispositivos móveis da rede, o tempo de partida $w(i,j)$ em cada enlace entre os nós i e j é representado por um número com distribuição de Poisson com intervalo de tempo médio selecionado aleatoriamente entre dois intervalos que podem ser definidos em cada simulação. O atraso de movimento $md(i,j)$ em cada enlace é um número selecionado aleatoriamente entre dois intervalos, que é posteriormente, multiplicado pela distância entre os nós $dist(i,j)$.

Conforme visto na Seção 2.2.5 do Capítulo 2, a capacidade de fragmentação reativa não é requerida para estar disponível em toda implementação DTN. Entretanto, a capacidade de reagrupar fragmentos no destino é requerida a fim de suportar fragmentação. Como o algoritmo de roteamento sendo projetado considera DTNs em cenários determinísticos, ou seja, que os contatos são previsíveis, utiliza-se fragmentação pró-ativa na fonte, isto é, blocos de dados podem ser fragmentados apenas no nó fonte e remontados no destino o qual deve, obrigatoriamente, ter essa capacidade de remontagem. Além disso, fragmentos de uma mesma sessão são encaminhados através de uma única rota.

5.3.4. Avaliação dos algoritmos de roteamento

O simulador permite facilmente a inclusão de algoritmos de roteamento para serem testados, bastando adicionar módulos de roteamento apropriadamente integrados ao ambiente de simulação. Para o roteamento *anycast* foram adicionadas as aproximações baseadas em GA (GA1, GA2 e GA3) descritas na Seção 4.5 do Capítulo 4, e os algoritmos de roteamento SP e ED, descritos na Seção 3.4.5 do Capítulo 3.

Para avaliação da proposta de algoritmo de roteamento na rede, uma série de técnicas e métodos de avaliação podem ser empregados. Devido às DTNs serem orientadas a mensagens, será testada a taxa de mensagens entregues, o atraso de entrega das mensagens e a distribuição das mesmas na rede. Baseado principalmente nesses parâmetros, para avaliação

da proposta de algoritmo de roteamento *anycast* em DTNs, o simulador suporta o monitoramento das seguintes medidas de desempenho:

– Probabilidade de entrega (*PE*) das mensagens: é a proporção do número total de mensagens *anycast* únicas recebidas por qualquer membro do grupo de receptores planejados *anycast* pelo número total de mensagens transmitidas pelo nó fonte *anycast*. Esta estatística irá refletir a eficiência do algoritmo de roteamento *anycast* em entregar mensagens a pelo menos um membro do grupo *anycast* de destino;

– Atraso médio ponderado ou atraso total (*D*) definido na eq. (4.5): média ponderada do atraso de todas as mensagens entregues a um membro de destino. Esta estatística irá indicar o tempo médio que as mensagens efetivamente entregues (dadas pela *PE* das mensagens) gastam, desde o instante que são geradas, até alcançarem os receptores de destino;

– Número total de saltos $h(i)$ de cada sessão *anycast z*: é o número de saltos utilizados nas rotas definidas pelos algoritmos de roteamento. Esta estatística irá mostrar se o algoritmo de roteamento será capaz de distribuir as mensagens ao longo da rede de modo a evitar que determinados enlaces (cada salto representa a passagem por um enlace) sejam saturados ou funcionem como gargalos. Além disso, define-se uma medida de desempenho que calcula o número médio de mensagens por enlace (*MPE*):

$$MPE = \sum_{i=1}^{z} m_k(i) \Big/ \sum_{i=1}^{z} h(i) \tag{5.2}$$

onde $m_k(i)$ é o número total de mensagens *anycast* únicas recebidas por um membro do grupo de receptores *anycast* para cada sessão *anycast*, e $h(i)$ é o número total de saltos de cada sessão *anycast*;

– O tempo que o algoritmo de roteamento gasta até encontrar as rotas para as sessões *anycast*. Essa estatística irá indicar se o algoritmo de roteamento leva um tempo apropriado para definição de rotas, ou seja, se o algoritmo obtém bons resultados em um tempo aceitável.

Assim sendo, o simulador irá monitorar as principais informações a respeito das mensagens presentes na rede. Com essas informações será possível avaliar o comportamento da proposta de algoritmo de roteamento *anycast* baseado em GA.

5.4. Ambiente de simulação

A avaliação dos algoritmos de roteamento *anycast* é realizada através de modelagem e simulação. A seguir são apresentados os parâmetros de simulação utilizados.

5.4.1. Ajuste dos parâmetros da rede e do algoritmo genético

Nas simulações realizadas, a menos que sejam especificados outros valores, os parâmetros utilizados são mostrados na Tabela 5.1. Ajustou-se α para um valor igual (0,4) ao utilizado por Gong et al. (2006). Já β possui um valor ligeiramente maior do que o encontrado nos experimentos de Gong et al. (2006) (0,25 contra 0,2), porém menor do que o utilizado por Randaccio e Atzori (2007) (0,4), ou seja, uma densidade de enlace pequena. Na área da grade foi distribuído um total de 40 nós. Os parâmetros $w(i,j)$ e $md(i,j)$ seguem os valores utilizados no trabalho de Gong et al. (2006), como pode ser visto na Tabela 5.1. O número de receptores planejados l de cada sessão *anycast* pode variar entre 2 e 5, com a fonte de cada sessão *anycast* podendo enviar um número de mensagens $m_s(z)$ entre 300 e 500. Gong et al. (2006)

Tabela 5.1. Parâmetros iniciais utilizados na simulação.

Parâmetro	Descrição	Valor
α	Indica a densidade de enlaces curtos em relação a enlaces com distâncias maiores entre os nós	0,4
β	Densidade de enlaces	0,25
N	Número total de nós	40
Área	Área da grade onde são distribuídos os N nós	1300 metros x 1300 metros
$w(i,j)$	Tempo de partida dos dispositivos móveis entre os nós i e j	Números aleatórios com distribuição de Poisson possuindo intervalo médio de tempo selecionado aleatoriamente entre 600 e 6000 segundos
$md(i,j)$	Atraso de movimento dos dispositivos móveis entre os nós i e j	Número aleatório escolhido entre 60 e 600 segundos
$b(i)$	Capacidade de armazenamento do nó i	Pode variar para cada nó de 600 a 1000 mensagens
$c(i,j)$	Capacidade de armazenamento do dispositivo móvel entre o nó i e j	Pode variar de 500 a 800 para cada enlace
l	Número de receptores planejados de cada sessão *anycast*	Cada sessão *anycast* pode ter entre 2 e 5 receptores planejados
z	Número de sessões *anycast* de uma simulação	Variável
$m_s(z)$	Número de mensagens que cada nó fonte de uma sessão pode enviar	Cada nó fonte de uma sessão *anycast* pode enviar entre 300 e 500 mensagens
$PE_{\min}$	Probabilidade de entrega mínima requerida	96%
n_pop	Tamanho da população de indivíduos do GA	50
n_sol_pot	Número de soluções potenciais em isolamento	15% do número total de nós N
p_c	Probabilidade de aplicar-se o operador genético de cruzamento	0,8
p_m	Probabilidade de aplicar-se o operador genético de mutação	0,03
n_ger	Número de gerações	Variável e dependente do número de sessões
t_{total}	Tempo total simulado	43200 segundos (≈ 12 horas)

utilizaram um cenário mais simples e menos desafiador ($l = 2$ e $m_s(z) = 200$), o que justifica a utilização de $b(i)$ e $c(i,j)$ menores do que os apresentados na Tabela 5.1. A $PE_{\min}$ utilizada pelos algoritmos de roteamento baseados em GAs serve como um parâmetro base e é ajustado

de acordo com os requisitos de diferentes aplicações, com o valor utilizado ficando entre os valores comumente encontrados na literatura (Sudhaakar et al., 2009; Wang e Wu, 2007; Yoon et al., 2007). Para uma população de 50 indivíduos, ajustaram-se p_c e p_m para valores ligeiramente superiores aos sugeridos por Jong (1975) visando aumentar a diversidade da população. Entretanto, como o conceito de subpopulação está sendo utilizado, a influência desses parâmetros é reduzida, assim como pode ser visto em (Lo e Chang, 2000). Para simular diferentes condições de tráfego variou-se o número de sessões *anycast*. Como o tamanho do cromossomo e o número de combinações que o GA deve buscar são dependentes do número de sessões, fez-se o número de gerações variável e dependente do número de sessões *anycast*.

5.4.2. Geração dos resultados e medidas de desempenho

O simulador é projetado para gerar as medidas de desempenho necessárias para analisar os algoritmos de roteamento *anycast*. A simulação é cessada assim que o tempo total de simulação é atingido. São geradas várias matrizes contendo medidas de desempenho da rede, como *PE*, atraso total (*D*), número médio de saltos e de gerações utilizadas pelos algoritmos de roteamento *anycast* baseados em GA. Além disso, com exceção da Seção 6.2 do próximo capítulo, os resultados são gerados através da média de dez execuções com diferentes topologias de rede e parâmetros de geração aleatórios, ou seja, diversas sementes de simulação. Por fim, foi calculado o IC (Intervalo de Confiança) para a *PE* média das mensagens quando a variância populacional é desconhecida. Para isso, foi utilizada a distribuição t de *Student*, desenvolvida por William Sealy Gosset, que é uma distribuição de probabilidade estatística.

5.5. Considerações finais

Este capítulo, inicialmente, justificou e analisou a utilização de simulação para modelagem da rede e avaliação da proposta de roteamento *anycast* em DTNs. Depois de um levantamento dos principais simuladores de rede existentes foi descrito detalhadamente o ambiente de simulação, desenvolvido neste trabalho, para implementar as características das DTNs. Por fim, os parâmetros utilizados na simulação, bem como a forma de geração dos resultados, foram apresentados.

No capítulo seguinte, a proposta de algoritmo de roteamento *anycast* baseado em GA é avaliada através da modelagem e simulação descrita neste capítulo.

Capítulo 6

AVALIAÇÃO DA PROPOSTA DE ALGORITMO DE ROTEAMENTO *ANYCAST* BASEADO EM ALGORITMOS GENÉTICOS

6.1. Introdução

Neste capítulo analisa-se por meio de modelagem e simulação o desempenho do algoritmo de roteamento *anycast* baseado em GA, proposto neste trabalho. São avaliadas diferentes configurações para o GA a fim de verificar-se a eficiência de cada estratégia. Além disso, o desempenho do algoritmo é comparado com o algoritmo SP que calcula o menor número de saltos na rede para realizar o roteamento, e com o algoritmo ED que calcula rotas com o menor atraso de entrega. Basicamente, os algoritmos de roteamento são comparados em termos da probabilidade de entrega, atraso de entrega, distribuição do tráfego na rede e tempo até a definição de rotas.

Este capítulo está organizado da seguinte maneira: a Seção 6.2 mostra resultados da utilização de um simples GA no roteamento *anycast* em DTNs. O efeito da capacidade de armazenamento da rede nos algoritmos de roteamento *anycast* é analisado na Seção 6.3. A avaliação da utilização do conceito de subpopulação é apresentada na Seção 6.4. A proposta

de algoritmo de roteamento *anycast* baseado em GA é avaliada sob diferentes densidades de enlace e áreas para distribuição dos nós na Seção 6.5. A eficiência da utilização de um número limitado de soluções potenciais e do conceito de subpopulação na proposta de algoritmo de roteamento *anycast* baseado em GA é estudada na Seção 6.6. Para analisar a escalabilidade da proposta, na Seção 6.7, o comportamento dos algoritmos é investigado em cenários utilizando diferentes números de nós na rede. Na Seção 6.8, analisa-se o tempo de execução dos algoritmos baseados em GA. Por fim, na Seção 6.9 apresentam-se as considerações finais deste capítulo.

6.2. Análise inicial da utilização de algoritmos genéticos para roteamento *anycast* em DTNs

Neste estudo, por meio de modelagem e simulação, pretende-se avaliar a utilização de GAs para efetuar o roteamento *anycast* em DTNs sem a utilização das estratégias de limitação do número de soluções potenciais em isolamento e do conceito de subpopulação (GA1). Para análise inicial, o algoritmo de roteamento *anycast* baseado em GA (GA1) e o algoritmo SP são avaliados sob diferentes números de sessões. Foram utilizados os mesmos parâmetros de rede apresentados na Tabela 5.1, para quantidades diferentes de sessões *anycast* e $PE_{\min}$ de 80%. Vale ressaltar que os resultados são calculados com base em execuções com diferentes topologias de rede, o que justifica as variações encontradas nos gráficos a seguir.

Primeiramente, simula-se uma DTN com 4 sessões *anycast*. As Figuras 6.1.a e 6.1.b ilustram a *PE* e o atraso total (*D*), respectivamente, obtidos pelos algoritmos. A Figura 6.1 mostra que a partir de 80 gerações o algoritmo baseado em GA obtém um atraso menor ou igual ao algoritmo SP e com *PE* acima de $PE_{\min}$ (80%). A diferença entre os algoritmos GA1 e o GA1 entrada SP é que o segundo algoritmo possui a rota do menor caminho como entrada.

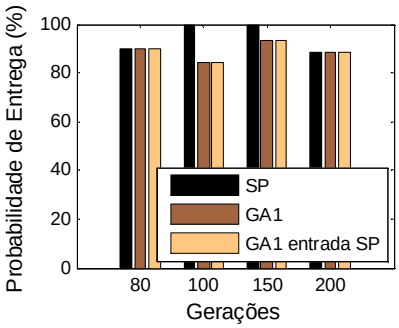

Figura 6.1.a. Probabilidade de entrega (*PE*) para 4 sessões *anycast* com os algoritmos baseados em GA utilizando um número variado de gerações.

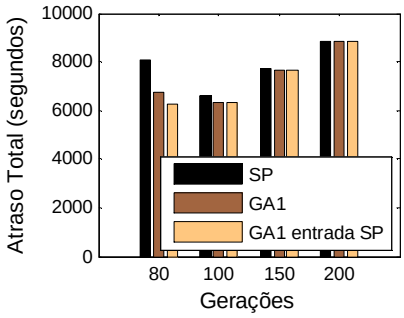

Figura 6.1.b. Atraso total (*D*) para 4 sessões *anycast* com os algoritmos baseados em GA utilizando um número variado de gerações.

Os resultados para 10 sessões *anycast* são mostrados nas Figuras 6.2.a e 6.2.b. Agora o algoritmo necessita de um número maior de gerações para convergir devido ao aumento do número de sessões, ou seja, a diversidade e complexidade do problema são maiores. Pode-se observar que apenas com mais de 200 gerações os algoritmos baseados em GA atingiram um desempenho superior ao algoritmo SP. Nota-se que na execução que os algoritmos baseados em GA utilizaram 250 e 300 gerações, o algoritmo SP não atingiu a $PE_{\min}$ (Figura 6.2.a).

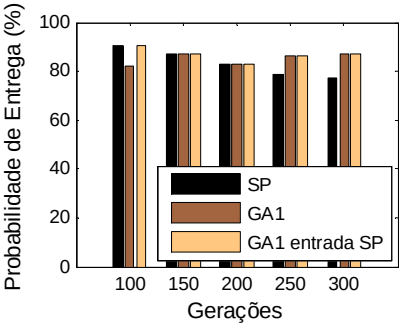

Figura 6.2.a. Probabilidade de entrega (*PE*) para 10 sessões *anycast* com os algoritmos baseados em GA utilizando um número variado de gerações.

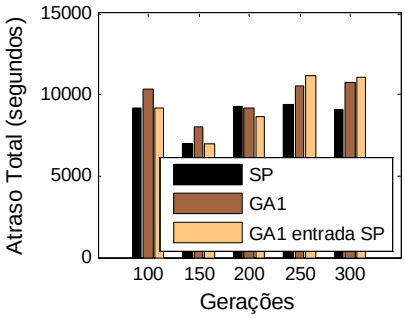

Figura 6.2.b. Atraso total (*D*) para 10 sessões *anycast* com os algoritmos baseados em GA utilizando um número variado de gerações.

Os resultados para 20 sessões *anycast* são mostrados nas Figuras 6.3.a e 6.3.b. O algoritmo baseado em GA não consegue convergir utilizando o número de gerações especificado, e o algoritmo SP encontra resultados melhores ou iguais aos do algoritmo GA1.

Isto pode ser explicado pelo fato de que o algoritmo baseado em GA aumenta o número de gerações necessárias para o algoritmo convergir quando a diversidade do problema é grande.

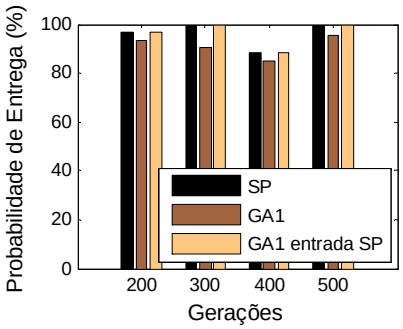

Figura 6.3.a. Probabilidade de entrega (*PE*) para 20 sessões *anycast* com os algoritmos baseados em GA utilizando um número variado de gerações.

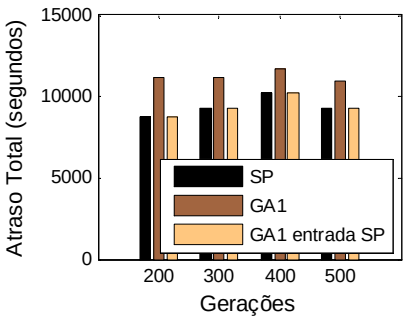

Figura 6.3.b. Atraso total (*D*) para 20 sessões *anycast* com os algoritmos baseados em GA utilizando um número variado de gerações.

Desta forma, o algoritmo de roteamento *anycast* baseado em GA pode produzir resultados superiores aos do algoritmo SP, mas precisa ser desenvolvido utilizando técnicas diferentes para convergir mais rapidamente. Os resultados, conclusões e mais alguns detalhes sobre os experimentos desta seção podem ser encontrados em (Silva e Guardieiro, 2008). Nas Seções 6.3 e 6.4, são vistas algumas técnicas utilizadas para tornar o algoritmo de roteamento *anycast* baseado em GA apropriado para as redes DTNs, em termos de tempo de convergência e otimização de medidas de desempenho.

6.3. Efeito da capacidade de armazenamento dos dispositivos móveis no roteamento *anycast* nas DTNs

Neste estudo, por meio de modelagem e simulação, pretende-se avaliar o efeito da capacidade de armazenamento (número de mensagens) dos dispositivos móveis $c(i,j)$ no desempenho dos algoritmos de roteamento *anycast* nas DTNs.

Cada dispositivo móvel pode armazenar uma quantidade limitada de mensagens. Evidentemente, quando essa quantidade é reduzida, o roteamento se torna mais desafiador e os resultados são mais dependentes das decisões de rota. Por isso, esta seção investiga o comportamento do algoritmo SP e do algoritmo de roteamento baseado em GA com um número de soluções potenciais limitado (GA2), visando reduzir o número de gerações necessárias até o algoritmo convergir. São analisados dois cenários: o primeiro estuda a influência do número de mensagens que cada nó fonte *anycast* pode enviar e, o segundo, o comportamento dos algoritmos de roteamento *anycast* quando os dispositivos móveis possuem diferentes capacidades de armazenamento.

Inicialmente, foram utilizados os mesmos parâmetros de rede apresentados na Tabela 5.1, para 12 sessões *anycast*, variando-se o número de mensagens $m_k(z)$ que a fonte de cada sessão *anycast* pode enviar: 200 a 500, 300 a 500 e de 400 a 500 mensagens. Além disso, a capacidade de armazenamento dos dispositivos móveis $c(i,j)$ foi ajustada para variar entre 400 e 700 mensagens. O algoritmo GA2 utiliza 200 gerações como condição de terminação e busca rotas com uma $PE_{\min}$ de 90%.

A Figura 6.4 mostra que ambos os algoritmos atingem boas taxas de *PE*, com um leve declínio à medida que se aumenta o número de mensagens enviadas $m_k(z)$ pela fonte de cada sessão *anycast*. Esse declínio é explicado pelo fato de que com o aumento do número de mensagens enviadas, mantendo-se a capacidade dos nós e dispositivos móveis da rede, o roteamento se torna mais desafiador, pois a capacidade de armazenamento da rede se torna menor quando comparada à quantidade de mensagens enviadas.

Analisando os resultados do atraso total (*D*) mostrados na Figura 6.5, pode-se notar que o algoritmo GA2 sempre entrega mensagens com atrasos menores que o algoritmo SP. Assim, nota-se que o algoritmo GA2 está desempenhando corretamente a função para o qual foi projetado: encontra rotas com *PE* acima da $PE_{\min}$ = 90% e com o menor atraso. O atraso tende

a aumentar quando o número de mensagens enviadas cresce devido a um número maior de mensagens competirem por uma oportunidade de transmissão.

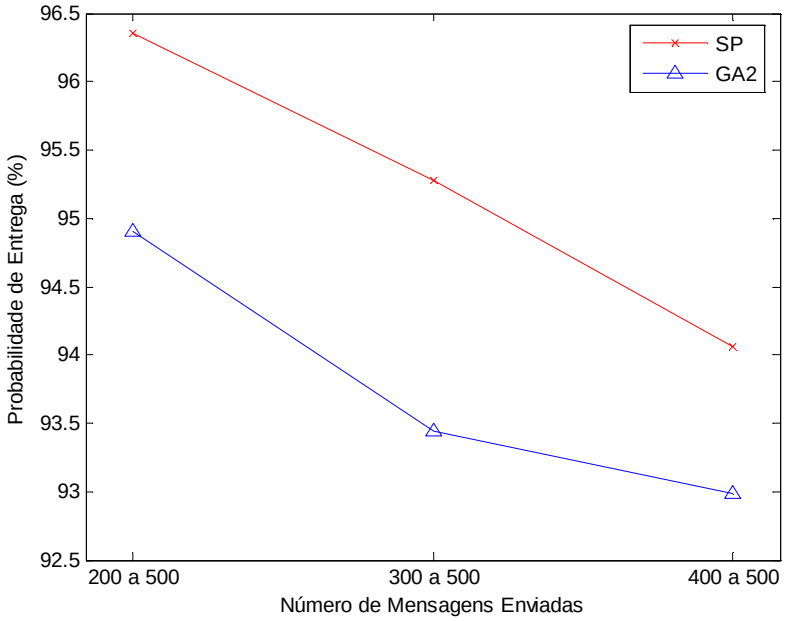

Figura 6.4. Probabilidade de entrega (*PE*) obtida pelos algoritmos de roteamento SP e GA2 variando-se o número de mensagens enviadas $m_k(z)$ pela fonte de cada sessão *anycast*.

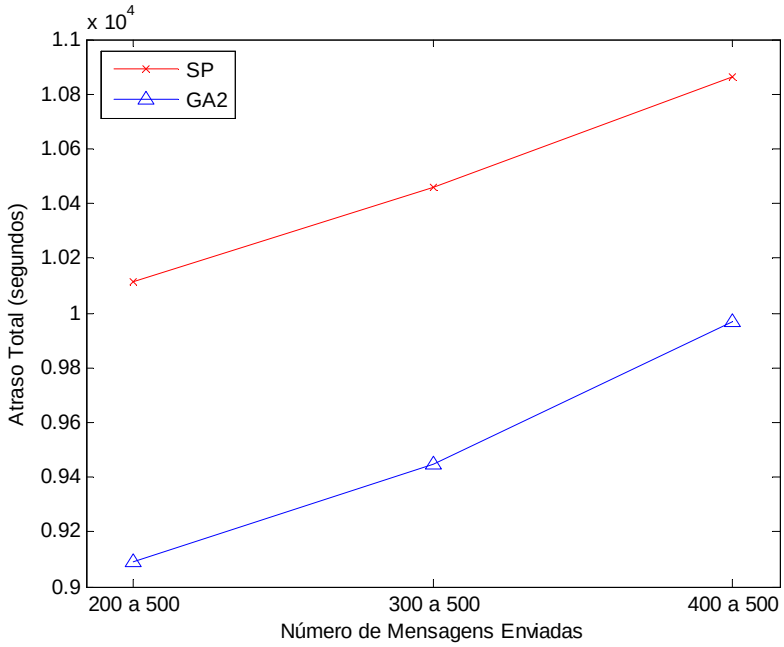

Figura 6.5. Atraso total (*D*) obtido pelos algoritmos de roteamento SP e GA2, variando-se o número de mensagens enviadas $m_k(z)$ pela fonte de cada sessão *anycast*.

No próximo cenário, o número de mensagens enviadas $m_k(z)$ pela fonte de cada sessão *anycast* foi fixado para estar entre 400 e 500 mensagens. A capacidade de armazenamento dos dispositivos móveis $c(i,j)$ foi variada de 400 a 700, 400 a 500 e de 300 a 500 mensagens. Os demais parâmetros possuem os mesmos valores dos apresentados no cenário anterior. A Figura 6.6 mostra a *PE* atingida pelos algoritmos de roteamento *anycast*. A *PE* de ambos os algoritmos decresce à medida que a capacidade de armazenamento dos dispositivos móveis $c(i,j)$ diminui, pois com essa redução de capacidade, a competição por uma oportunidade de transmissão é maior. Além disso, para o cenário mais desafiador ($c(i,j)$ variando entre 300 e 500 mensagens), a *PE* obtida pelo algoritmo SP diminui consideravelmente, enquanto o algoritmo GA2 consegue manter a *PE* acima da $PE_{\min}$. Assim, quando os recursos são mais escassos, o algoritmo GA2 mostrou-se mais robusto que o algoritmo SP.

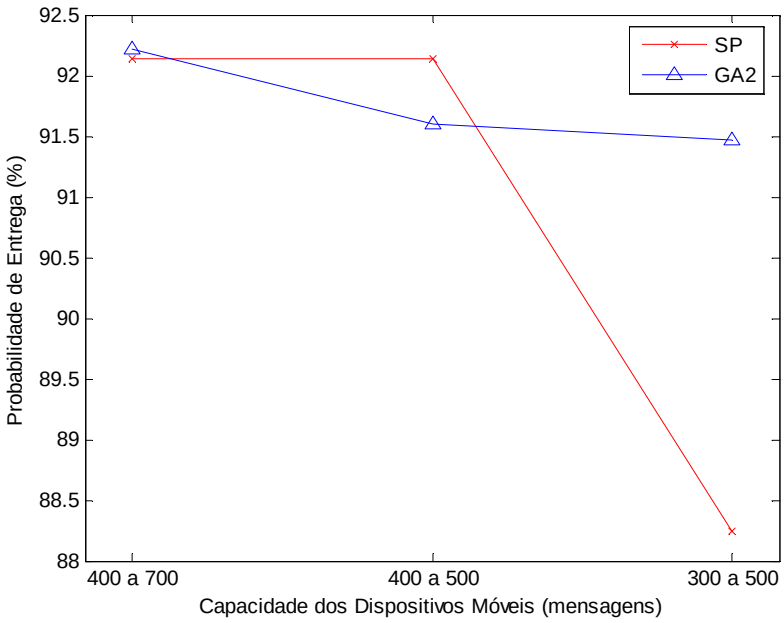

Figura 6.6. Probabilidade de entrega (*PE*) obtida pelos algoritmos de roteamento SP e GA2, variando-se a capacidade de armazenamento dos dispositivos móveis $c(i,j)$.

Os resultados para o atraso total (*D*) são mostrados na Figura 6.7. Assim como para o primeiro cenário, as rotas encontradas pelo algoritmo GA2 entregam as mensagens com um atraso menor em relação ao algoritmo SP. É importante notar que os resultados obtidos neste

segundo cenário, para $c(i,j)$ variando de 400 a 700 mensagens, não são idênticos ao do primeiro cenário devido ambos serem a média de dez execuções com sementes aleatórias e topologias de rede diferentes. Além disso, para ambos os cenários, os resultados apresentaram o mesmo comportamento.

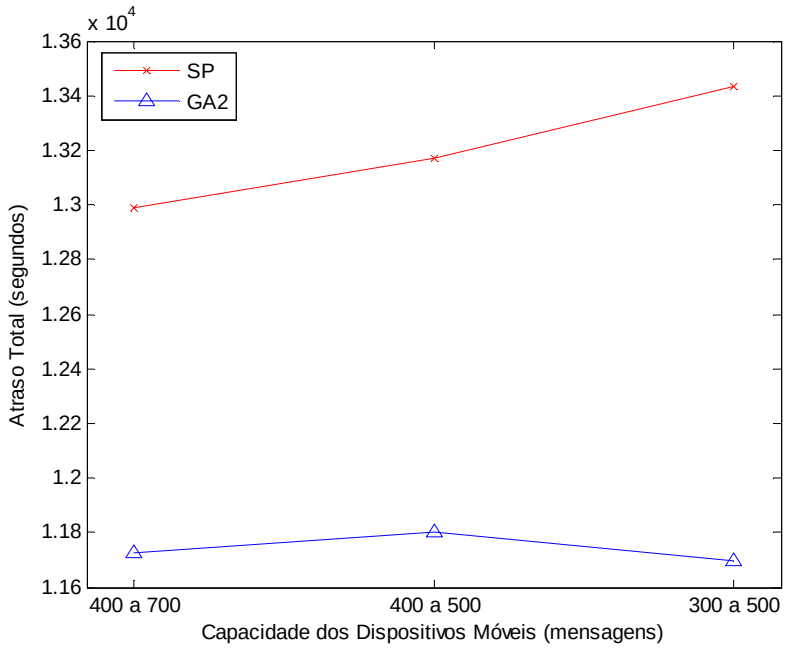

Figura 6.7. Atraso total (D) obtido pelos algoritmos de roteamento SP e GA2, variando-se a capacidade de armazenamento dos dispositivos móveis $c(i,j)$.

Os resultados desta seção sugerem que quando os recursos da rede são mais escassos, o melhoramento obtido pelo algoritmo GA2 é maior se comparado ao algoritmo SP. Esse comportamento se deve ao fato do algoritmo SP considerar apenas o número de saltos para decisão de rotas. Assim, observa-se que com menos recursos, algoritmos de roteamento mais complexos são necessários para que se possa atingir um bom desempenho. Além disso, os resultados obtidos pelo algoritmo GA2 podem ser explicados pela característica principal do algoritmo baseado em GA: busca pela combinação de rotas acima de uma probabilidade de entrega mínima e possuindo o menor atraso. Os resultados, conclusões e mais alguns detalhes sobre os experimentos desta seção podem ser encontrados em (Silva e Guardieiro, 2009a). Por fim, a influência da utilização de um número limitado de soluções potenciais em isolamento

pelo algoritmo GA2 pode ser observada no número de gerações requeridas pelo algoritmo até convergir. Como citado anteriormente, nesta seção o número de gerações foi limitado a 200, sendo que na média, o algoritmo GA2 utilizou 117,8 dessas gerações até encontrar os resultados apresentados. Comparando esses resultados com os do algoritmo GA1 na seção anterior, fica evidenciado que a utilização de um número limitado de soluções potenciais em isolamento pode ser uma boa aproximação para melhorar o tempo de convergência do algoritmo baseado em GA.

6.4. Utilização do conceito de subpopulação para otimização do desempenho do algoritmo de roteamento *anycast* baseado em algoritmos genéticos

Neste estudo, por meio de modelagem e simulação, pretende-se avaliar a influência da utilização do conceito de subpopulação no desempenho do algoritmo de roteamento *anycast* baseado em GA, proposto neste trabalho. Neste caso, a população da próxima geração é um misto de quatro subpopulações (estratégia de reserva elitismo, substituição de um elemento do cromossomo, amostragem estocástica universal e método completamente aleatório), como descrito na Seção 4.3.5 do Capítulo 4, com este método visando à utilização de diferentes valores de pressão de seleção. Para avaliar o desempenho do algoritmo proposto, foram analisados dois algoritmos baseados em GA: o primeiro utiliza o conceito de subpopulação (GA3), e o segundo utiliza apenas elitismo (GA2). Além disso, ambos os algoritmos utilizam um número de soluções potenciais em isolamento limitado. Os algoritmos são avaliados utilizando quatro quantidades diferentes de sessões *anycast z*: 4, 10, 16 e 20. O número de gerações utilizadas pelos algoritmos de roteamento *anycast* baseados em GA foi variável e

dependente do número de sessões. Foram utilizados os mesmos parâmetros da Tabela 5.1, com exceção dos seguintes parâmetros:

- β = 0,2;
- $c(i,j)$ variando entre 500 e 1000 mensagens;
- $m_k(z)$ variando entre 200 e 500 mensagens;
- PE_{min} de 95%.

A Figura 6.8 mostra a probabilidade de entrega obtida pelos algoritmos. O intervalo de confiança de 95% é inserido no gráfico utilizando uma escala de 1:5. A probabilidade de entrega de todos os algoritmos diminui quando o número de sessões na rede aumenta, ou seja, quando o tráfego cresce. Isto se deve ao fato de que com um tráfego maior, os nós precisam esperar mais tempo por uma oportunidade de transmissão. O algoritmo SP atinge os piores resultados e o GA3 sempre obtêm resultados melhores que GA2. Além disso, a diferença na probabilidade de entrega entre os algoritmos tende a crescer quando o número de sessões aumenta. Isto indica que o algoritmo GA3 é mais robusto que os algoritmos GA2 e SP.

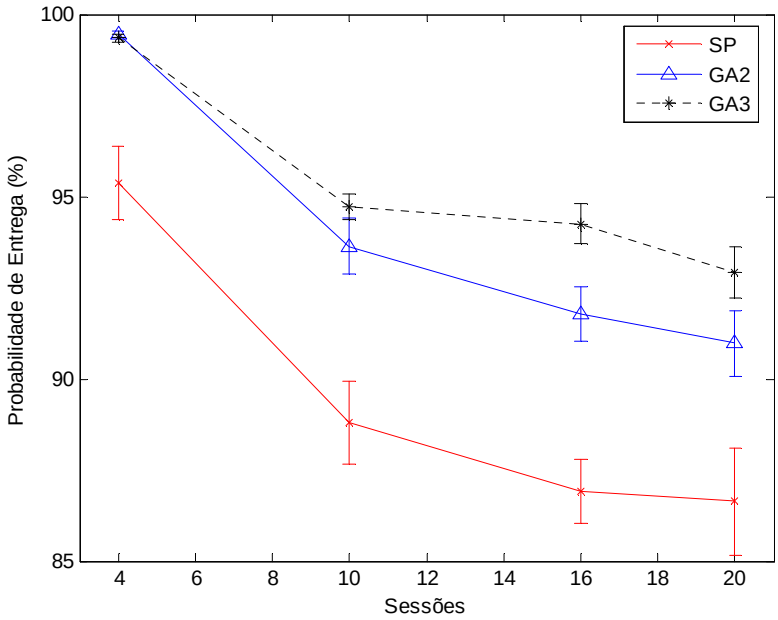

Figura 6.8. Probabilidade de entrega (*PE*) obtida pelos algoritmos de roteamento SP, GA2 e GA3, utilizando diferentes números de sessões e intervalo de confiança de 95%.

O atraso total (*D*) para os algoritmos de roteamento é apresentado na Tabela 6.1. Para o cenário desta seção, notou-se que a influência do número de sessões no atraso de entrega obtido é menos intenso do que na probabilidade de entrega.

Tabela 6.1. Atraso total (*D*) obtido pelos algoritmos de roteamento SP, GA2 e GA3, utilizando diferentes números de sessões.

Algoritmo	**4 sessões**	**10 sessões**	**16 sessões**	**20 sessões**
SP	20057 s	20059 s	20129 s	19202 s
GA2	18486 s	21936 s	21283 s	20197 s
GA3	18134 s	21349 s	21258 s	21104 s

O número médio de saltos utilizado por cada algoritmo de roteamento é mostrado na Figura 6.9. Os algoritmos baseados em GA distribuem o tráfego melhor que o algoritmo SP, isto é, utilizam um número de saltos maior. A busca de rotas alternativas para uma boa distribuição do tráfego na rede é uma característica desejável para um algoritmo de roteamento, com os algoritmos GA3 e GA2 obtendo resultados similares.

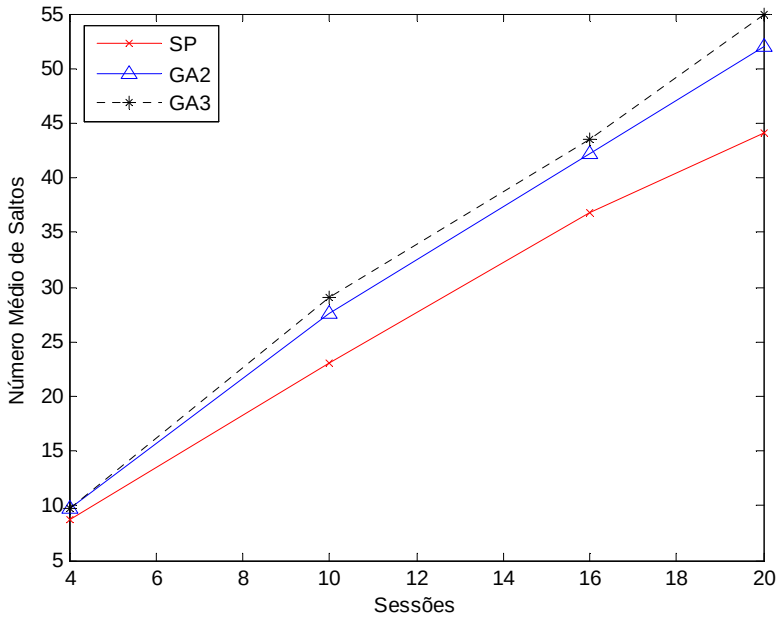

Figura 6.9. Número médio de saltos utilizados pelos algoritmos de roteamento SP, GA2 e GA3, para diferentes números de sessões.

Os resultados apresentados até aqui mostraram um desempenho melhor do algoritmo GA3 em relação ao GA2. Agora, é analisado o número médio de gerações e o tempo gasto por

cada algoritmo para atingir os resultados mostrados nesta seção. O número médio de gerações utilizadas pelos algoritmos baseados em GA é mostrado na Figura 6.10. O número de gerações requerido por ambos os algoritmos é bastante similar. Além disso, o número médio de gerações necessárias cresce para ambos quando se aumenta o número de sessões. Isto é devido ao fato de que com mais sessões, o comprimento do cromossomo é maior, e um número maior de combinações é avaliado pelos algoritmos baseados em GA.

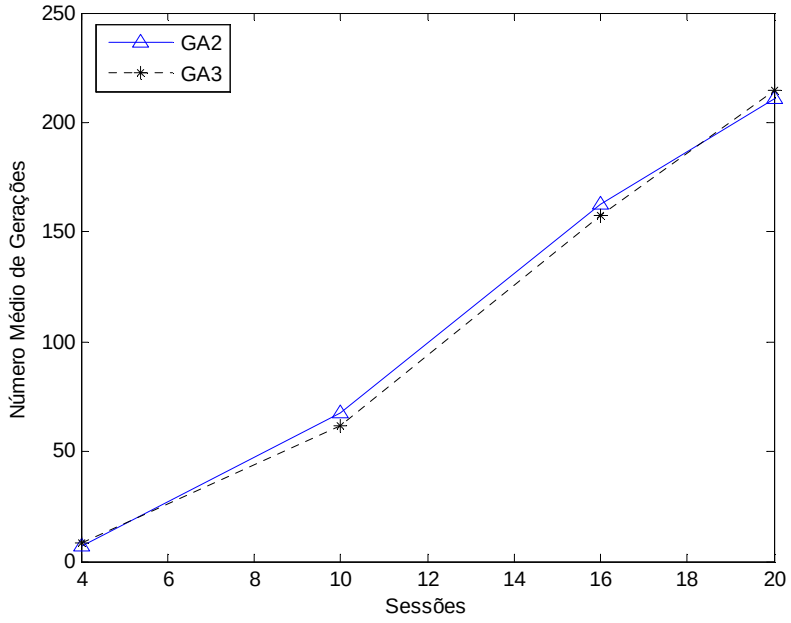

Figura 6.10. Número médio de gerações para obtenção de rotas boas e/ou ótimas pelos algoritmos de roteamento GA2 e GA3, utilizando diferentes números de sessões.

O tempo médio de simulação até o algoritmo convergir (t_{sim}) é o tempo total gasto pelos algoritmos até a obtenção de boas rotas. O tempo médio de simulação até o algoritmo convergir (t_{sim}) utilizado pelos algoritmos baseados em GA, calculado de acordo com a eq. (4.8), é ilustrado na Figura 6.11. A figura mostra que o tempo médio de simulação até o algoritmo convergir (t_{sim}) segue o mesmo comportamento do número de gerações.

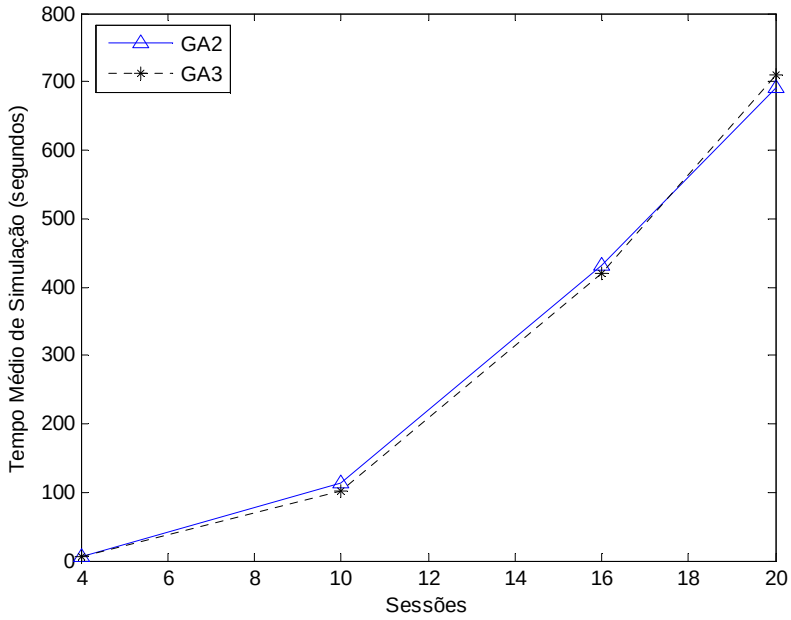

Figura 6.11. Tempo médio de simulação até o algoritmo convergir (t_{sim}) para obtenção de rotas boas e/ou ótimas pelos algoritmos de roteamento GA2 e GA3, utilizando diferentes números de sessões.

Analisando os resultados desta seção, o algoritmo que utiliza o conceito de subpopulação (GA3) apresentou uma probabilidade de entrega mais robusta do que quando não se utilizou esta estratégia (GA2) com um tempo similar até o algoritmo encontrar a rota. Com base no tempo médio de simulação até o algoritmo convergir (t_{sim}), observa-se que este tempo é, normalmente, menor que o tempo de partida dos dispositivos móveis $w(i,j)$, ou seja, os algoritmos baseados em GA encontram rotas em tempos inferiores ao tempo de partida dos dispositivos móveis $w(i,j)$ em cada enlace. Desta forma, para os cenários simulados (4, 10, 16 e 20 sessões), o tempo médio de simulação gasto até o algoritmo convergir (t_{sim}) é aceitável, e mostra que a utilização do conceito de subpopulação, juntamente com a limitação do número de soluções potenciais em isolamento, escalam bem com o número de sessões, ou seja, mantém o desempenho mesmo com o aumento das sessões, e são robustos para os cenários simulados.

6.5. Comportamento dos algoritmos de roteamento utilizando diferentes topologias de rede e densidades de enlace

Neste estudo, por meio de modelagem e simulação, pretende-se avaliar a influência da utilização de diferentes topologias de rede no algoritmo de roteamento *anycast* baseado em GA, proposto neste trabalho. Para isso, diferentes topologias de rede são geradas, variando-se a área sobre a qual os nós são dispostos e a densidade de enlace na rede, ou seja, variou-se *Área* e β. Foram utilizados os mesmos parâmetros da Tabela 5.1, com exceção dos seguintes parâmetros:

- $PE_{\min}$ de 92%;
- z: foram consideradas 16 sessões *anycast*.

Neste cenário de simulação, foi variada a área sobre a qual os nós são distribuídos na rede: 1300 m x 1300 m, 2600 m x 2600 m e 5200 m x 5200 m. O número de gerações dos algoritmos de roteamento *anycast* baseados em GA foi limitado a 400.

A *PE* obtida pelos algoritmos de roteamento é apresentada na Figura 6.12. O intervalo de confiança de 95% é inserido no gráfico utilizando uma escala de 1:5. Quando a área sobre a qual os nós são distribuídos aumenta, a distância entre os nós também aumenta, consequentemente, o atraso de movimento $md(i,j)$ (que é proporcional à distância) cresce. O algoritmo GA3 utiliza quatro subpopulações para formação da população da próxima geração. Já o algoritmo GA2 utiliza apenas elitismo para evitar o descarte de indivíduos bons. O GA3 obteve uma *PE* mais consistente do que o GA2 por apresentar uma variação menor. Ambos os algoritmos apresentam desempenho bem superior ao algoritmo SP e, portanto, o GA3 mostrou-se mais robusto em termos de *PE* quando a área sobre a qual os nós são distribuídos é aumentada.

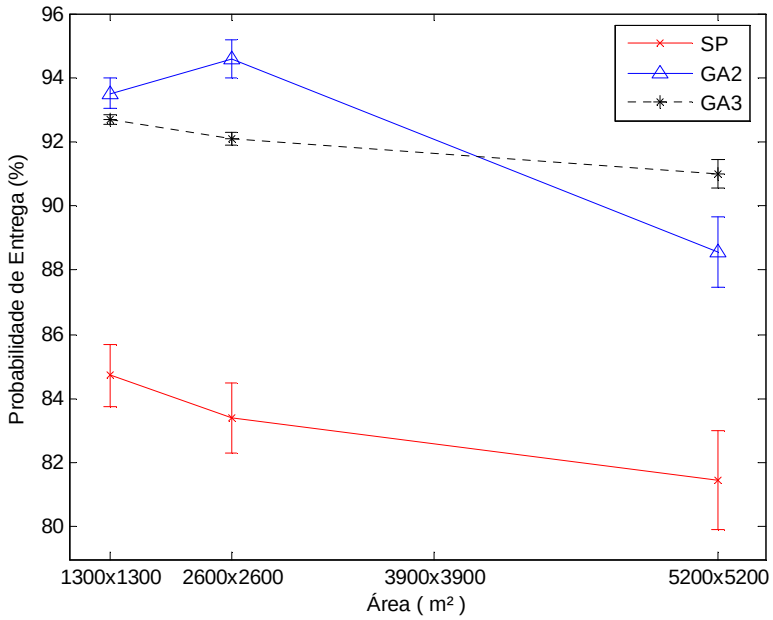

Figura 6.12. Probabilidade de entrega (*PE*) obtida pelos algoritmos de roteamento SP, GA2 e GA3, utilizando diferentes áreas para distribuição dos nós na DTN e intervalo de confiança de 95%.

A Figura 6.13.a mostra o número médio de mensagens por enlace (*MPE*). Como o algoritmo SP calcula rotas com o menor número de saltos o número médio de mensagens por enlace (*MPE*) é alto. Já os algoritmos baseados em GA distribuem melhor as mensagens. O número médio de mensagens por enlace *(MPE*) é decrescente conforme se aumenta a área, pois quando o atraso de movimento *md*(*i*,*j*) dos dispositivos móveis aumenta, o tempo de espera por uma oportunidade de transmissão é maior. Como consequência, os algoritmos baseados em GA precisam evitar que um grande número de mensagens trafegue pelo mesmo enlace, isto é, que busquem rotas alternativas. Desta forma, um número maior de enlaces é utilizado, e se mais enlaces são utilizados, o número médio de mensagens por enlace (*MPE*) diminui.

O atraso total (*D*) é apresentado na Figura 6.13.b. O atraso aumenta para todos os algoritmos devido ao crescimento do atraso de movimento *md*(*i*,*j*). Os algoritmos GA3 e SP apresentam os melhores resultados para o atraso. Entretanto, este atraso é obtido para a *PE* mostrada na Figura 6.12, ou seja, apenas o algoritmo GA3 consegue bons valores de atraso

mantendo também boas taxas de *PE*, que são características buscadas para o algoritmo de roteamento *anycast*.

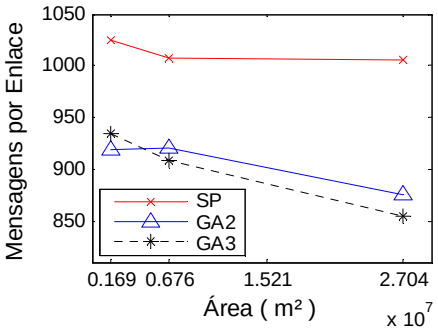

Figura 6.13.a. Número médio de mensagens por enlace (*MPE*) obtido pelos algoritmos de roteamento SP, GA2 e GA3, utilizando diferentes áreas para distribuição dos nós na DTN.

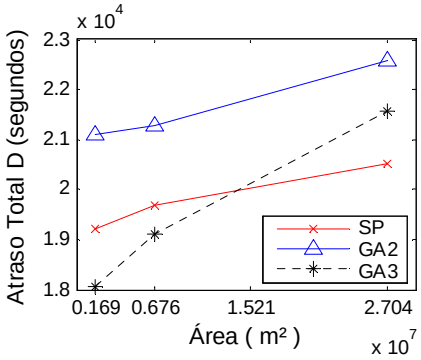

Figura 6.13.b. Atraso total (*D*) obtido pelos algoritmos de roteamento SP, GA2 e GA3, utilizando diferentes áreas para distribuição dos nós na DTN.

No próximo cenário de simulação variou-se a densidade de enlaces β na rede: 0,20, 0,25 e 0,30 utilizando uma área de 1300m x 1300 m. A Figura 6.14 mostra a *PE* obtida pelos algoritmos de roteamento. O intervalo de confiança de 95% é inserido no gráfico utilizando uma escala de 1:2, e um comportamento semelhante ao da Figura 6.12 pode ser observado, ou seja, o algoritmo GA3 se mostra mais robusto (menor intervalo de confiança) quando os recursos da rede são mais escassos (quando a densidade de enlaces β é baixa).

Quando a densidade de enlaces β é aumentada, o número médio de mensagens por enlace *(MPE*) também aumenta (Figura 6.15.a). Isto se deve ao fato de que com mais opções de enlaces, as mensagens podem ser encaminhadas através de rotas contendo um número menor de saltos, pois há uma disponibilidade maior de rotas para repassar mensagens. Como resultado, o número médio de mensagens por enlace *(MPE*) aumenta. Novamente, o algoritmo GA3 tende a utilizar um número médio de mensagens por enlace *(MPE*) menor do que os demais algoritmos utilizados para comparação nesse cenário, ou seja, distribui melhor as mensagens na rede.

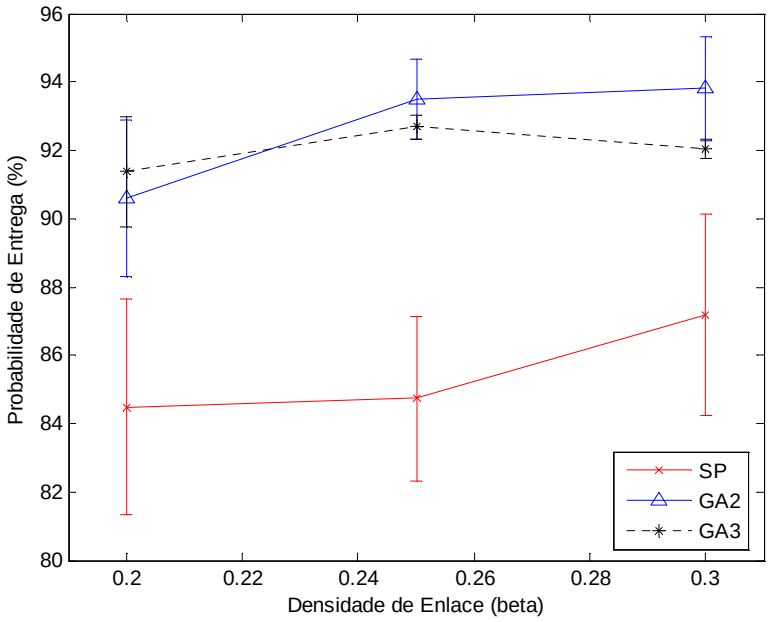

Figura 6.14. Probabilidade de entrega (*PE*) obtida pelos algoritmos de roteamento SP, GA2 e GA3, utilizando diferentes densidades de enlace e intervalo de confiança de 95%.

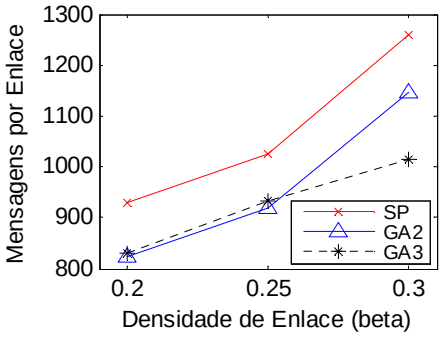

Figura 6.15.a. Número médio de mensagens por enlace (*MPE*) obtido pelos algoritmos de roteamento SP, GA2 e GA3, utilizando diferentes densidades de enlace.

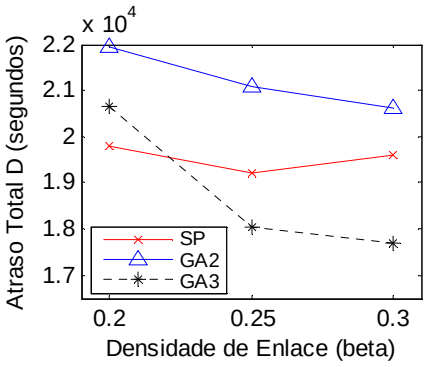

Figura 6.15.b. Atraso total (*D*) obtido pelos algoritmos de roteamento SP, GA2 e GA3, utilizando diferentes densidades de enlace.

O atraso total (*D*) obtido pelo algoritmo GA3 diminui mais que o dos outros algoritmos (Figura 6.15.b), mantendo uma *PE* estável, pois o algoritmo GA3 faz uma utilização eficiente do aumento da densidade de enlaces β, isto é, o algoritmo GA3 é mais eficiente na busca da combinação de rotas que otimiza as medidas de desempenho da rede do que o algoritmo GA2.

Analisando os resultados desta seção, quando se variou a área sobre a qual os nós são distribuídos e a densidade de enlaces, o algoritmo GA3, que utiliza o conceito de

subpopulação e um número limitado de soluções potenciais em isolamento, mostrou escalar bem com as mudanças, ou seja, conseguiu manter uma *PE* próxima da PE_{min}, mesmo com as variações da área sobre a qual os nós são distribuídos e diferentes densidades de enlaces utilizadas. Além disso, quando se aumenta os recursos da rede (por exemplo, diminuindo a área e/ou aumentando a densidade de enlaces), o algoritmo GA3 apresentou a maior variação no atraso. Isto indica que o algoritmo GA3 explora de uma maneira mais eficiente os recursos da rede do que os algoritmos GA2 e SP. Desta forma, mais uma vez, a utilização do algoritmo de roteamento *anycast* baseado em GA, do conceito de subpopulação, juntamente com a limitação do número de soluções potenciais em isolamento, permitiu obter maior eficiência.

6.6. Influência da limitação do número de soluções potenciais e do conceito de subpopulação na proposta de algoritmo de roteamento *anycast* baseado em GA

Neste estudo, por meio de modelagem e simulação, pretende-se avaliar a influência da limitação do número de soluções potenciais em isolamento e do conceito de subpopulação no algoritmo de roteamento *anycast* baseado em GA, proposto neste trabalho. Para isso, diferentes topologias de rede são geradas, variando-se o número de sessões *anycast z* da rede. Foram utilizados os mesmos parâmetros da Tabela 5.1, com exceção da PE_{min} ser ajustada para 94%. Cinco algoritmos (SP, ED, GA1, GA2 e GA3) são avaliados utilizando quatro quantidades diferentes de sessões *anycast z*: 4, 10, 16 e 20. O número de gerações utilizadas pelos algoritmos de roteamento *anycast* baseados em GA foi variável e dependente do número de sessões.

A *PE* obtida pelos algoritmos de roteamento é apresentada na Figura 6.16. O intervalo de confiança de 95% é inserido no gráfico utilizando uma escala de 1:2. À medida que se

aumenta o número de sessões a *PE* diminui, pois com mais tráfego os nós precisam esperar mais tempo por uma oportunidade de transmitir. Os algoritmos ED e SP apresentam os piores resultados, com o algoritmo ED possuindo desempenho ligeiramente superior ao algoritmo SP. Além disso, a diferença de desempenho entre os algoritmos baseados em GA e os algoritmos ED e SP tende a crescer quando o número de sessões é maior, ou seja, quando o roteamento é mais desafiador. Isso se deve ao fato dos algoritmos ED e SP não considerarem a combinação do tráfego para decisão de rotas, nem as restrições de armazenamento dos dispositivos móveis responsáveis por transportar as mensagens. O GA2 é o algoritmo baseado em GA que apresenta o pior resultado para *PE*. Os algoritmos GA1 e GA3 atingem resultados similares e são mais robustos que as outras aproximações (intervalo de confiança menor).

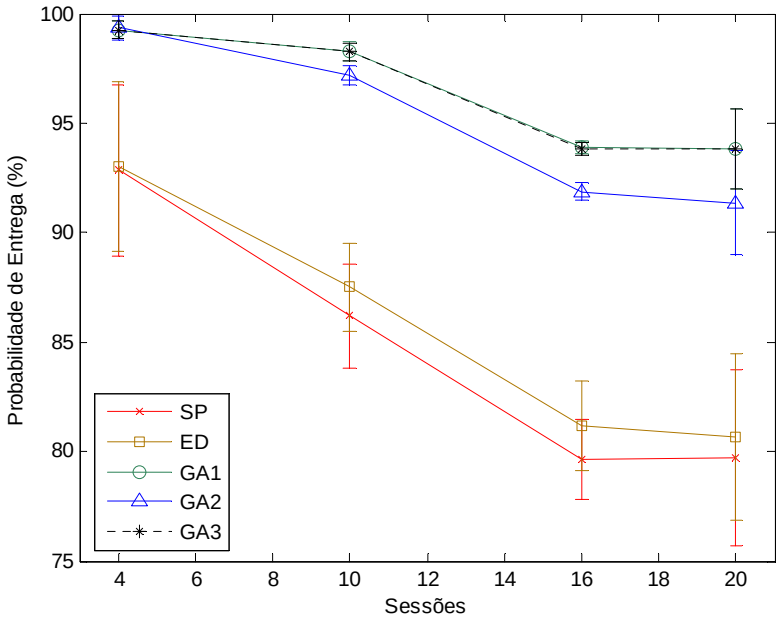

Figura 6.16. Probabilidade de entrega (*PE*) obtida pelos algoritmos de roteamento SP, ED, GA1, GA2 e GA3, utilizando diferentes números de sessões e intervalo de confiança de 95%.

A Tabela 6.2 apresenta o atraso total (*D*) para os algoritmos de roteamento. O número de sessões altera o comportamento dos algoritmos de roteamento, entretanto, sua influência é mais intensa na *PE* (o atraso total (*D*) não apresenta grandes variações), principalmente para os algoritmos SP e ED. Em alguns cenários, os algoritmos SP e ED apresentaram um atraso

total (*D*) ligeiramente menor do que os algoritmos baseado em GA. Entretanto, vale ressaltar que esses resultados são para a *PE* mostrada na Figura 6.16, ou seja, os algoritmos de roteamento baseados em GA conseguem entregar um número maior de mensagens, sem prejudicar o atraso. Alguns resultados, conclusões e detalhes sobre os experimentos desta seção podem ser encontrados em (Silva e Guardieiro, 2010).

Tabela 6.2. Atraso total (*D*) obtido pelos algoritmos de roteamento SP, ED, GA1, GA2 e GA3, utilizando diferentes números de sessões.

Sessões	**SP**	**ED**	**GA1**	**GA2**	**GA3**
4 sessões	16111 s	15991 s	16846 s	17199 s	16847 s
10 sessões	20834 s	20120 s	19779 s	20458 s	19871 s
16 sessões	19119 s	19087 s	19841 s	20540 s	20515 s
20 sessões	19848 s	19180 s	21386 s	20891 s	20798 s

A Figura 6.17 mostra o número médio de saltos utilizado por cada algoritmo de roteamento *anycast*. Por calcular caminhos com o menor número de saltos, o algoritmo SP é o que obtém rotas contendo menos saltos. Já as rotas obtidas pelo algoritmo ED possuem mais saltos do que as encontradas pelo algoritmo SP, mas inferiores aos resultados dos algoritmos baseados em GA. O fato dos algoritmos de roteamento baseados em GA encontrarem rotas que utilizam um número maior de saltos indica que o tráfego na rede, para produzir a *PE* e o atraso total (*D*) mostrados nesta seção, é melhor distribuído pelos algoritmos. Corroborando com essa ideia, a diferença entre o número médio de saltos utilizado pelos algoritmos baseados em GA e os algoritmos SP e ED tende a aumentar quando o número de sessões é maior. A justificativa é que quando o tráfego aumenta, o tempo e a concorrência por uma oportunidade de transmitir são altos. Com isso, os algoritmos de roteamento baseados em GA procuram evitar que um grande número de mensagens passe através de um mesmo enlace, isto é, procura rotas alternativas para distribuir melhor o tráfego na rede. Como consequência, essas rotas alternativas, normalmente, contém um número maior de saltos fazendo com que o número médio de saltos cresça.

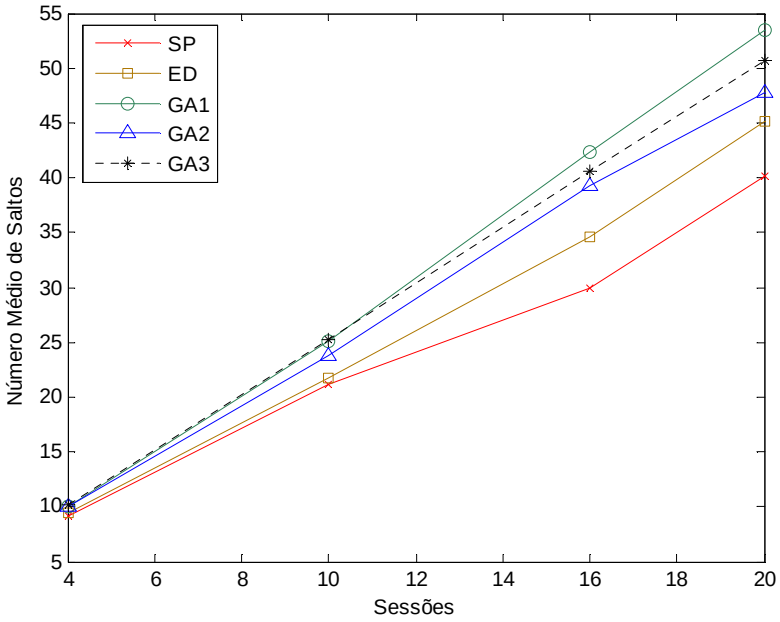

Figura 6.17. Número médio de saltos para os algoritmos de roteamento *anycast* utilizando diferentes números de sessões.

Até aqui os algoritmos GA1 e GA3 apresentaram desempenho bastante similar. A Figura 6.18 mostra o número médio de gerações que os algoritmos baseados em GA requerem até convergir. Este número cresce para todos os algoritmos, pois com o aumento do número de sessões, o tamanho do cromossomo é maior e aumenta-se o número de combinações para serem avaliadas pelos algoritmos. Os algoritmos GA2 e GA3 utilizam praticamente o mesmo número de gerações. Por outro lado, o número médio de gerações que o GA1 leva para convergir cresce significativamente.

O tempo médio de simulação até o algoritmo convergir (t_{sim}) para os algoritmos baseados em GA é ilustrado na Figura 6.19 e segue o comportamento do número médio de gerações. Como consequência, o algoritmo GA1 gasta um longo tempo até convergir. Para o cenário simulado mais desafiador (20 sessões *anycast*), o algoritmo GA3 gasta em média 786,3 segundos, contra 2063,8 segundos para o algoritmo GA1 convergir. Se compararmos estes valores com o tempo de partida dos dispositivos móveis $w(i,j)$, este tempo mostra que o

algoritmo GA3 é uma solução melhor, com exceção de situações nas quais não se tenha restrição de tempo.

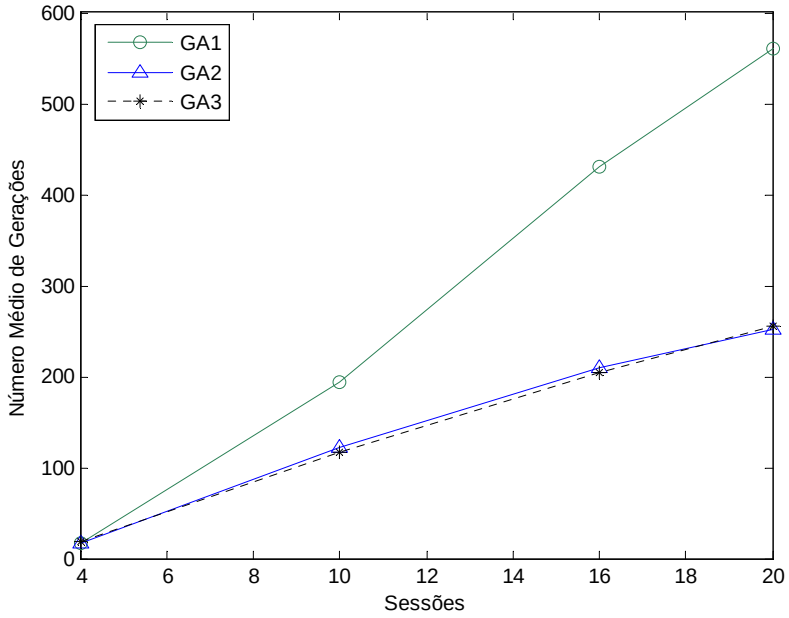

Figura 6.18. Número médio de gerações para os algoritmos de roteamento *anycast* baseados em GA utilizando diferentes números de sessões.

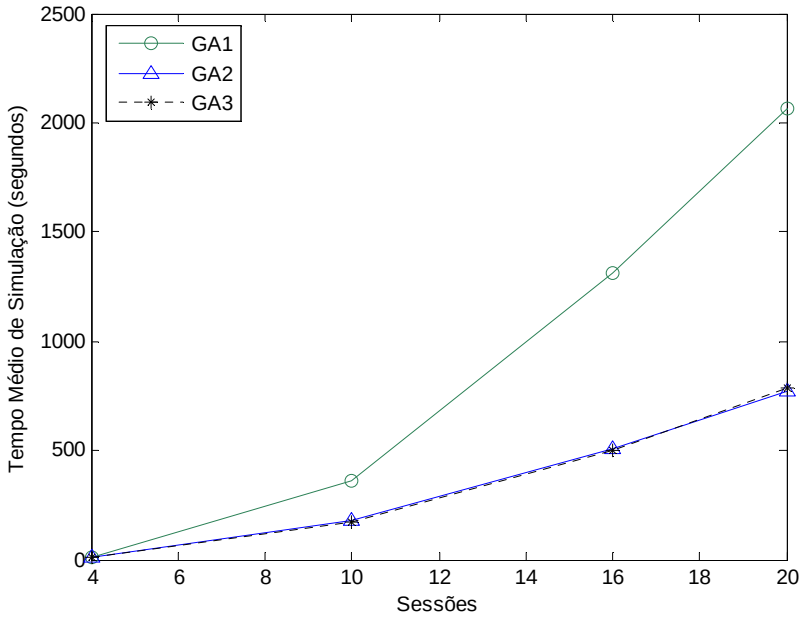

Figura 6.19. Tempo médio de simulação até o algoritmo convergir (t_{sim}) para obtenção de rotas boas e/ou ótimas pelos algoritmos de roteamento *anycast* baseados em GA, utilizando diferentes números de sessões.

Os resultados mostram que os algoritmos GA1 e GA3 apresentam o melhor desempenho no objetivo de entregar mensagens. Além disso, o algoritmo GA3 obtém estes resultados utilizando um número de gerações menor que o algoritmo GA1. Desta forma, quando se limita o número de soluções potenciais em isolamento, os algoritmos baseados em GA convergem mais rapidamente, e quando o conceito de subpopulação é utilizado, os algoritmos baseados em GA buscam soluções mais eficientemente.

6.7. Análise de escalabilidade dos algoritmos de roteamento *anycast*

Neste estudo, por meio de modelagem e simulação, pretende-se avaliar a escalabilidade dos algoritmos de roteamento *anycast* SP, ED e GA3. Para isso, diferentes topologias de rede são geradas, variando-se o número de nós presentes na rede. Foram utilizados os mesmos parâmetros da Tabela 5.1, com exceção da $PE_{\min}$ ser ajustada para 94% e o número de sessões *anycast* foi fixado em 16. Três algoritmos (SP, ED e GA3) são avaliados utilizando quatro quantidades diferentes de nós: 30, 40, 50 e 60. O número de gerações para o algoritmo GA3 foi fixado em 300.

A *PE* obtida pelos algoritmos de roteamento é apresentada na Figura 6.20. O intervalo de confiança de 95% é inserido no gráfico utilizando uma escala de 1:2. À medida que se altera o número de nós na rede, a *PE* obtida pelos algoritmos sofre variações. Entretanto, essas variações são pequenas para o algoritmo GA3, enquanto para os algoritmos SP e ED, os valores da *PE* são bem mais instáveis. Isso se deve ao fato dos algoritmos ED e SP não considerarem a combinação do tráfego para decisão de rotas, nem as restrições de armazenamento dos dispositivos móveis responsáveis por transportar as mensagens. Além disso, o algoritmo GA3 atinge resultados mais robustos que as outras aproximações (intervalo de confiança menor). O desempenho da *PE* indica que o algoritmo de roteamento *anycast*

baseado em GA escala melhor com o número de nós do que os algoritmos SP e ED, além de obter resultados com uma *PE* acima da PE_{min} definida.

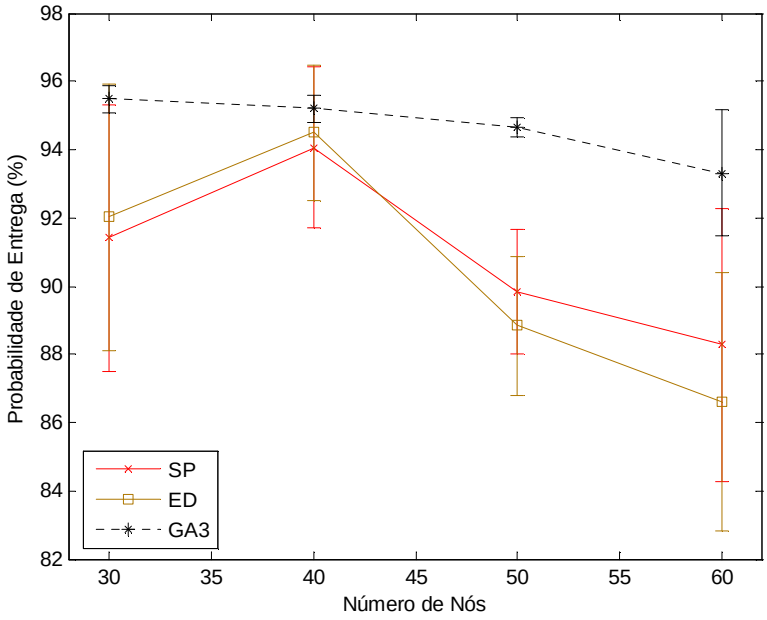

Figura 6.20. Probabilidade de entrega (*PE*) obtida pelos algoritmos de roteamento SP, ED e GA3, utilizando diferentes números de nós na rede e intervalo de confiança de 95%.

A Figura 6.21 mostra que o atraso total (*D*) não apresenta grandes variações para os algoritmos de roteamento SP, ED e GA3. Com isso, mostra-se que o algoritmo GA3, no cenário simulado, consegue satisfazer a PE_{min} sem prejudicar o atraso total (*D*), que é próximo aos resultados obtidos pelos algoritmos SP e ED.

A Figura 6.22 mostra o número médio de saltos utilizado por cada algoritmo de roteamento *anycast* sendo analisado nesta seção. Mais uma vez, por calcular caminhos com o menor número de saltos, o algoritmo SP é o que obtém rotas contendo menos saltos. Já as rotas obtidas pelo algoritmo ED possuem mais saltos do que as encontradas pelo algoritmo SP, mas inferior aos resultados do algoritmo GA3. Isso mostra que o GA3 procura evitar um grande número de mensagens passando através de um mesmo enlace, isto é, procura rotas alternativas para distribuir melhor o tráfego na rede, ocasionando um número maior de saltos. Além disso, o número médio de saltos, obtido pelos algoritmos de roteamento *anycast*, possui

comportamento semelhante, apresentando ligeiras variações à medida que o número de nós na rede cresce.

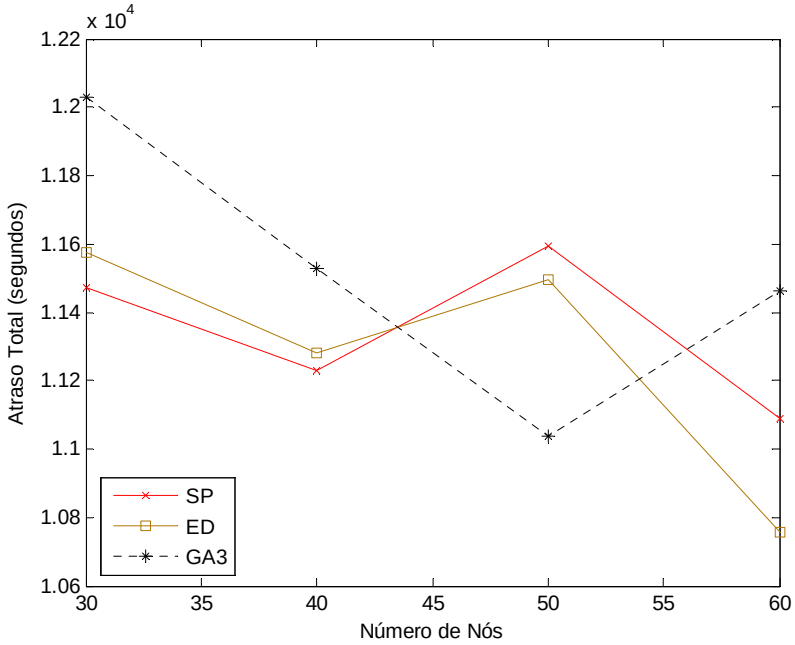

Figura 6.21. Atraso total (*D*) obtido pelos algoritmos de roteamento SP, ED e GA3, utilizando diferentes números de nós na rede.

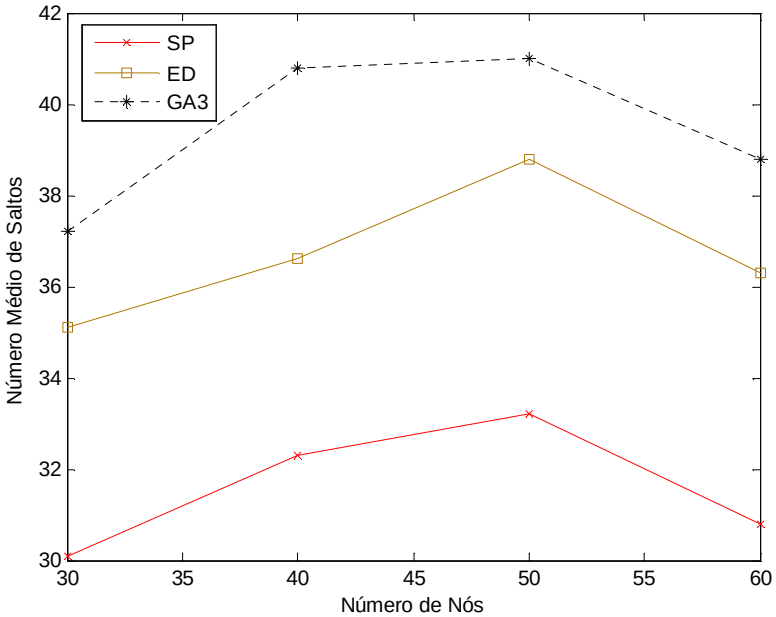

Figura 6.22. Número médio de saltos para os algoritmos de roteamento *anycast* utilizando diferentes números de nós na rede.

Analisando os resultados desta seção, quando se variou o número de nós na rede, observa-se que a proposta de algoritmo roteamento *anycast* baseado em GA reage melhor às mudanças do que os algoritmos SP e ED. Os resultados são justificados pelo fato da proposta baseada em GA ser mais complexa, buscar a melhor combinação de rotas e levar em consideração as restrições de armazenamento dos dispositivos móveis na rede. Assim como nas seções anteriores, a proposta baseada em GA produziu resultados superiores aos encontrados pelos algoritmos SP e ED, mostrando mais uma vez ser mais apropriada para o cenário em questão.

6.8. Complexidade do algoritmo de roteamento baseado em GA

Além do tempo médio de simulação até o algoritmo convergir (t_{sim}), que foi calculado para os algoritmos de roteamento baseados em GA, pode-se verificar como o tempo de execução cresce conforme o volume de dados oferecido como entrada para os algoritmos aumenta. Para isso, utilizou-se a notação O que define os tempos de execução em termos de funções fundamentais, eliminando constantes e outras funções. Os cálculos são realizados visando o tempo de pior caso, pois se o algoritmo for aceitável neste caso, também será para todos os outros que podem ser enfrentados.

A complexidade dos algoritmos baseados em GA O(*GA*) pode ser dada pela expressão:

$$O(GA) = O\left(n\_ger \cdot n\_pop \cdot z \cdot N^2\right)$$

No cálculo anterior, *n_ger* representa o número de gerações, *n_pop* o tamanho da população e *z* o número de sessões *anycast*. Além disso, considerou-se uma matriz *NxN* como pior caso, ou seja, *N* soluções potenciais em isolamento contendo todos os *N* nós da rede.

Analisando-se o tempo de execução encontrado, fica evidenciado que esse tempo é dependente das dimensões da matriz de soluções potenciais em isolamento, o que justifica a

redução no tempo de convergência dos algoritmos baseados em GA que utilizam um número de soluções potenciais em isolamento limitado. Por outro lado, ao limitarem-se essas soluções potenciais, algumas rotas que poderiam fazer parte de soluções melhores são perdidas. Pelos resultados encontrados na Seção 6.6, observa-se que a limitação do número de soluções potenciais em isolamento reduz o tempo de execução do algoritmo consideravelmente (veja Figura 6.19), com resultados muito próximos aos encontrados quando o número de soluções potenciais é ilimitado. Assim sendo, a limitação do número de soluções potenciais em isolamento é uma boa aproximação para diminuir o tempo médio de execução até os algoritmos de roteamento baseados em GA convergirem.

A complexidade do algoritmo de Dijkstra O(*Dijk*) depende apenas da quantidade de nós presentes na rede (Randaccio e Atzori, 2007):

$$O(Dijk) = O(N^3)$$

Comparando a complexidade do algoritmo de Dijkstra O(*Dijk*) com a dos algoritmos baseados em GA O(*GA*), observa-se que esta última é maior, sendo dependente do número de gerações e sessões *anycast*, tamanho da população e quantidade de soluções potenciais em isolamento. Além disso, a influência da quantidade de nós é maior no algoritmo de Dijkstra. Uma análise dessa influência foi realizada na seção anterior.

Visando demonstrar a eficiência da utilização do conceito de subpopulação, uma análise operacional é realizada em termos do número de operações executadas pelos algoritmos. A análise do número de operações é detalhada na Tabela 6.3, notando-se que a utilização do conceito de subpopulação reduz o número de operações. Assim sendo, apesar do algoritmo que utiliza o conceito de subpopulação possuir grande parte da população criada aleatoriamente (60%), isto é, incentivando a diversidade da população e consequentemente podendo requerer um número maior de gerações até convergir, a utilização do conceito de subpopulação necessita de um número menor de operações até o algoritmo convergir. Isto

justifica a obtenção de resultados melhores por parte do algoritmo GA3, utilizando um tempo médio de simulação até o algoritmo convergir (t_{sim}), como definido na eq. (4.8), próximo ao algoritmo GA2. Portanto, se por um lado a diversidade da população aumenta (evitando mínimos locais) e o algoritmo pode requerer mais gerações até convergir, por outro o número de operações a serem realizadas diminui.

Tabela 6.3. Número de operações dos algoritmos de roteamento baseados em GA utilizando ou não o conceito de subpopulação.

Conceito de subpop. / Operações	Não	Sim
Estratégia de reserva elitismo	N/A	$0{,}10 \cdot n\_ger \cdot n\_pop$
Substituição de um elemento do cromossomo	N/A	$0{,}10 \cdot n\_ger \cdot n\_pop$
Método completamente aleatório	N/A	$0{,}60 \cdot n\_ger \cdot n\_pop$
Cruzamento e mutação	$n\_ger \cdot (z-1) \cdot n\_pop + n\_ger \cdot n\_pop$	$0{,}20 \cdot n\_ger \cdot (z-1) \cdot n\_pop + 0{,}20 \cdot n\_ger \cdot n\_pop$
Total	$n\_ger \cdot (z-1) \cdot n\_pop + n\_ger \cdot n\_pop$	$0{,}20 \cdot n\_ger \cdot (z-1) \cdot n\_pop + n\_ger \cdot n\_pop$

onde: *n_ger* representa o número de gerações,
n_pop representa o tamanho da população,
z indica o número de sessões *anycast* e
N/A significa não aplicável

6.9. Considerações finais

Neste capítulo, por meio de modelagem e simulação, o algoritmo de roteamento *anycast* baseado em GA proposto foi analisado. Foram avaliados diferentes cenários e estratégias para melhorar o desempenho do algoritmo proposto. No próximo capítulo, uma análise geral e conclusões são realizadas a respeito de toda a pesquisa.

Capítulo 7

CONCLUSÃO E DESENVOLVIMENTOS FUTUROS

7.1. Introdução

Inicialmente, neste capítulo, uma revisão da tese é apresentada, mostrando como o projeto, desenvolvimento e avaliação da proposta de algoritmo de roteamento *anycast* baseado em GA foram realizados e como se conseguiu um bom desempenho utilizando essas aproximações. Na Seção 7.3 as contribuições resultantes dessa pesquisa são identificadas. Em seguida, uma análise crítica dos resultados obtidos no capítulo anterior é apresentada na Seção 7.4. Por fim, desenvolvimentos futuros são discutidos na Seção 7.5 e as considerações finais do trabalho são apresentadas na Seção 7.6.

7.2. Revisão da tese

No Capítulo 1, a hipótese e o propósito deste trabalho são definidos, com os principais objetivos sendo traçados.

A definição, as características e os principais conceitos da arquitetura DTN (rede para a qual a pesquisa é direcionada) são apresentados no Capítulo 2.

O roteamento nas redes DTN é tratado no Capítulo 3, sendo realizada uma revisão da literatura. O direcionamento e a originalidade da pesquisa são fixados, além de serem citados os trabalhos diretamente relacionados à pesquisa.

No Capítulo 4, a proposta de algoritmo de roteamento *anycast* baseado em GA foi definida. As aproximações utilizadas na proposta são descritas na Seção 4.3, destacando-se as técnicas empregadas para otimizar o desempenho do algoritmo de roteamento *anycast* baseado em GA.

No Capítulo 5, o projeto e considerações do ambiente de simulação implementado para avaliação da proposta de algoritmo de roteamento *anycast* são apresentados. Através da descrição da geração da topologia de rede (Seção 5.3.1), dos procedimentos para aquisição das medidas de desempenho (Seção 5.3.4) e dos ajustes dos parâmetros de simulação (Seção 5.4.1), o Capítulo 5 detalha o ambiente de simulação implementado.

No Capítulo 6, os resultados produzidos pela simulação são analisados. Várias estratégias e cenários são investigados. Pelos resultados fornecidos na Seção 6.2, viu-se que a utilização de um algoritmo de roteamento *anycast* baseado em GA simples necessita de muitas gerações para obtenção de bons resultados. Duas técnicas visando melhorar o desempenho da proposta, limitação do número de soluções potenciais em isolamento e conceito de subpopulação, são apresentadas nas Seções 6.3 e 6.4, respectivamente, ocasionando grande melhora na eficiência da proposta de algoritmo de roteamento *anycast* baseado em GA. Ambientes de simulação testando o comportamento da proposta de algoritmo de roteamento em diferentes cenários são encontrados nas Seções 6.5, 6.6 e 6.7. Os resultados apresentados no Capítulo 6 mostram que a utilização da proposta de algoritmo de roteamento *anycast* baseado em GA realiza significativos melhoramentos no roteamento *anycast* em DTNs determinísticas, por buscar a melhor combinação do tráfego *anycast*, quando comparado com algoritmos que calculam rotas com os menores custos levando-se em

consideração apenas o número de saltos, ou informações sobre a disponibilidade de cada enlace na rede.

7.3. Contribuições da tese

Através do trabalho descrito nesta tese, foi apresentada e provada a viabilidade de uma proposta de algoritmo de roteamento *anycast* baseado em GA para redes DTNs em cenários determinísticos. Para isso, realizou-se uma modelagem e simulação por meio de uma ferramenta aqui desenvolvida para representar as características das DTNs e avaliar o algoritmo proposto. Por fim, após a realização de extensivas simulações, mostrou-se a eficiência da proposta e de algumas estratégias para melhorar o desempenho do GA utilizado.

7.3.1. Proposta de algoritmo de roteamento *anycast* baseado em GA

O roteamento DTN é um tópico de intensa pesquisa e em aberto, sendo observada uma grande diversidade de propostas, dependendo dos cenários e desafios sendo considerados por cada aproximação. Depois de um amplo levantamento bibliográfico foi constatada a necessidade de um algoritmo de roteamento *anycast* para DTNs em cenários determinísticos, pois é uma área com muitas aplicações importantes e um tópico de pesquisa ainda pouco abordado. Além disso, a proposta de algoritmo de roteamento *anycast* é baseada em GA, justamente pelas DTNs serem toleráveis aos atrasos provocados pelo tempo requerido para operação de um GA até convergir para soluções boas e/ou ótimas. A aplicação de GAs para otimização do roteamento, através da busca apropriada da combinação do tráfego *anycast*, é algo novo em DTNs. Para aumentar a eficiência do algoritmo, a limitação do número de soluções potenciais em isolamento como entrada dos GAs e a utilização do conceito de subpopulação, se mostraram técnicas úteis e, que seja do nosso conhecimento, só vistas em

outros trabalhos separadamente, ou seja, nunca foram aplicadas ao mesmo tempo em um GA e em DTNs.

7.3.2. Desenvolvimento de uma ferramenta de avaliação

Foram investigadas técnicas e ferramentas que representassem e permitissem a análise das características desafiadoras das DTNs em cenários determinísticos. A solução proposta foi modelada para viabilizar a comunicação entre regiões desconectadas e carentes de infraestrutura, através da utilização de dispositivos móveis que se deslocam entre as regiões e permitem o transporte das mensagens. Devido à dificuldade de implementação de uma rede DTN real em grande escala, viu-se que a melhor ferramenta para avaliação da proposta, no momento, seria a simulação. Para a modelagem realizada, observou-se que a representação utilizando grafos evolutivos era apropriada, mas requerendo alguns ajustes (características de cada *edge* no grafo) para implementação das características desafiadoras das DTNs. Além disso, o foco foi no roteamento *anycast*. Como não foi encontrada nenhuma ferramenta de simulação disponível que possibilitasse a análise buscada, todo um ambiente de simulação foi projetado e desenvolvido, permitindo a avaliação completa da proposta de algoritmo a ser analisada.

7.3.3. Avaliação da proposta de algoritmo de roteamento *anycast* baseado em GA

Para demonstrar a viabilidade e eficiência da proposta, extensivas simulações foram realizadas. Foi visto que um GA, sem a utilização de técnicas para melhorar o desempenho do algoritmo, necessita de um número muito grande de gerações até convergir, o que poderia tornar a proposta inviável. Para solucionar tal problema, a limitação do número de soluções

potenciais em isolamento e o conceito de subpopulação foram analisados e mostraram melhorar o desempenho do GA. Além disso, diversos cenários e desafios foram testados, com a proposta sendo comparada a dois algoritmos, SP e ED, que calculam rotas com os menores custos levando-se em consideração apenas o número de saltos e informações sobre a disponibilidade de cada enlace na rede, respectivamente. Os resultados da simulação foram analisados, sendo mostrada a viabilidade da proposta de algoritmo de roteamento *anycast* baseado em GA.

7.4. Análise geral dos resultados

Diante da avaliação realizada no capítulo anterior para redes DTN em cenários determinísticos modeladas por grafos evolutivos, a utilização de um algoritmo de roteamento *anycast* baseado em GA, somada à limitação do número de soluções potenciais em isolamento para serem combinadas pelo GA e ao conceito de subpopulação, apresentaram bom desempenho, possuindo um mérito significativo.

Quando comparado ao algoritmo SP do menor caminho com base no número de saltos e ao algoritmo ED que calcula a rota que entrega as mensagens no menor tempo possível, os algoritmos baseados em GA apresentaram desempenho bastante superior, e essa superioridade se intensifica à medida que os recursos da rede se tornam mais escassos. Essa diferença de desempenho deve-se ao fato dos algoritmos SP e ED considerarem apenas o número de saltos e informações sobre disponibilidade do enlace, respectivamente, para realizar o roteamento, não utilizando quaisquer outras informações disponíveis, tais como, capacidade dos nós e dispositivos móveis da rede, para decisão de rotas. Diferentemente, a proposta de algoritmo de roteamento *anycast* baseado em GA analisa a combinação do tráfego na rede.

A limitação do número de soluções potenciais em isolamento para serem combinadas pelo algoritmo de roteamento *anycast* baseado em GA, proposto neste trabalho, mostrou-se

útil na tarefa de diminuir o tempo de convergência do algoritmo, produzindo resultados extremamente próximos aos obtidos quando se utilizou um número ilimitado de soluções potenciais em isolamento. Com isso, a limitação do número de soluções potenciais em isolamento é sempre indicada, com exceção de situações nas quais não se tenha restrição de tempo, pois quando não se limita o número de soluções potenciais em isolamento, soluções ótimas podem ser encontradas, mas utilizando tempos muito elevados.

Quando o conceito de subpopulação é empregado, o algoritmo de roteamento *anycast* apresenta desempenho superior do que quando o conceito não é utilizado, no que diz respeito às medidas de desempenho da rede, sem sacrificar o tempo de convergência do algoritmo. Desta forma, o conceito de subpopulação deve ser sempre empregado.

Por fim, a proposta de algoritmo de roteamento *anycast* baseado em GA utilizando as duas técnicas citadas acima teve o desempenho analisado sob diferentes topologias e condições de rede. Como resultado, o algoritmo proposto mostrou-se robusto às variações consideradas no Capítulo 6 e utiliza de maneira eficiente recursos que possam tornar-se disponíveis na rede. Além disso, foi visto que o tempo médio que o algoritmo proposto leva até a definição das rotas é apropriado, pois, normalmente, é menor que o tempo de partida dos dispositivos móveis considerado. Assim sendo, conclui-se que a proposta de algoritmo de roteamento baseado em GA é uma boa aproximação para o problema do roteamento *anycast* nas DTNs em cenários determinísticos.

7.5. Desenvolvimentos futuros

As direções para desenvolvimentos futuros que seguem, representam possíveis extensões desta pesquisa que valeriam a pena uma investigação mais aprofundada:

- Como conceito, a proposta de algoritmo de roteamento *anycast* baseado em GA para DTNs demonstrou com sucesso a sua utilidade. No entanto, os resultados obtidos na pesquisa

são a partir de simulações, que possuem suas particularidades. Assim sendo, uma extensão futura deste trabalho é a criação de um cenário real utilizando dispositivos móveis (mensageiros) visando testar a proposta em ensaios mais realistas. Como a arquitetura DTN e o suporte ao protocolo de agregação são relativamente recentes, poucas propostas em cenários reais empregando a arquitetura DTN foram testadas satisfatoriamente. Por isso, grande parte dos esquemas de roteamento propostos são analisados através de simulação, com o emprego desses esquemas no mundo real podendo contribuir e solidificar os conceitos investigados através de simulações;

- Possivelmente, o suporte a certa quantidade de esquemas de roteamento será necessário para que um nó opere eficientemente na diversidade de ambientes no qual pode se encontrar em uma DTN. Assim sendo, uma extensão futura para a pesquisa aqui realizada seria a investigação da utilização de esquemas de roteamento baseados em GA em outros ambientes, tais como no cenário probabilístico, e também visando outros serviços de entrega, como o *unicast* e o *multicast*.

7.6. Considerações finais

Como uma nova tecnologia emergente, as redes DTN possuem o potencial de prover comunicação a dispositivos e áreas que não são servidas por redes tradicionais. Tais ambientes compreendem regiões carentes de infraestrutura, como é o caso de muitas regiões remotas e rurais. Neste cenário, o serviço *anycast* é útil em diversas aplicações, tais como em descoberta de recursos, educação à distância, campos de batalhas, entre outros. Entretanto, existem muitas dificuldades e desafios a enfrentar antes que as DTNs sejam efetivamente implementadas, com o roteamento eficaz sendo um tópico importante de pesquisa. Por isso, analisou-se a utilização de mensageiros móveis e do algoritmo de roteamento *anycast* baseado em GA proposto para auxiliar a comunicação em tais ambientes desafiadores. A avaliação de

desempenho realizada mostrou que a proposta, através do emprego adequado dos GAs, pode produzir bons resultados e auxiliar no problema do roteamento *anycast* em cenários determinísticos. O estudo mostrou que a proposta baseada em GA é apropriada, podendo melhorar o roteamento *anycast* e solucionar algumas das dificuldades encontradas até então.

REFERÊNCIAS BIBLIOGRÁFICAS

(Ahn e Ramakrishna, 2002) AHN, C. W.; RAMAKRISHNA, R. S. A genetic algorithm for shortest path routing problem and the sizing of populations. *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 6, pp. 566-579, Dec. 2002.

(Alonso e Fall, 2003) ALONSO, J.; FALL, K. *A linear programming formulation of flows over time with piecewise constant capacity and transit times*. Intel Research, Technical Report IRB-TR-03-007, June 2003.

(Baker, 1985) BAKER, J. E. Adaptive selection methods for genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pp. 101-111, 1985.

(Burleigh et al., 2007) BURLEIGH, S.; RAMADAS, M.; FARRELL, S. *Licklider transmission protocol – motivation*. IETF RFC 5325, informational, Sept. 2008.

(Burns et al., 2005) BURNS, B.; BROCK, O.; LEVINE, B. N. MV routing and capacity building in disruption tolerant networks. In *Proceedings of the IEEE INFOCON*, vol. 1, pp. 398-408, Mar. 2005.

(Cerf et al., 2007) CERF, V. et al. *Delay-tolerant networking architecture*. IETF RFC 4838, informational, Apr. 2007.

(Chen e Sun, 2005) CHEN, H.; SUN, B. Multicast routing optimization algorithm with bandwidth delay constraint based on GA. *Journal of Communication and Computer*, vol. 2, pp. 63-67, May 2005.

(Davis, 1987) DAVIS, L. *Genetic algorithms and simulated annealing*. Los Altos, California, Morgan Kaufmann, 1987.

(Demmer e Fall, 2007) DEMMER, M.; FALL, K. DTLSR: Delay tolerant routing for developing regions. In *Proceedings of the 2007 Workshop on Networked Systems for Developing Regions*, pp. 1-6, 2007.

(Dijkstra, 1959) DIJKSTRA, E. W. A note on two problem in connexion with graphs. *Numerische Mathematik*, vol. 1, pp. 269-271, 1959.

(Fall, 2003) FALL, K. A delay-tolerant network architecture for challenged internets. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 27-34, 2003.

(Fall e Farrell, 2008) FALL, K.; FARRELL, S. DTN: An architectural retrospective. *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 5, pp. 828-836, June 2008.

(Farrell e Cahill, 2006) FARRELL, S.; CAHILL, V. *Delay- and disruption- tolerant networking*. Norwood MA, Artech House, 2006.

(Farrell et al., 2006) FARRELL, S.; CAHILL, V.; GERAGHTY, D.; HUMPHEREYS, I.; MCDONALD, P. When TCP breaks: Delay- and disruption- tolerant networking. *IEEE Computing Internet*, vol. 10, no. 4, pp. 72-78, July/Aug. 2006.

(Farrell et al., 2008) FARRELL, S.; RAMADAS, M.; BURLEIGH, S. *Licklider transmission protocol – security extensions*. IETF RFC 5327, experimental, Sept. 2008.

(Fazl-e-Hadi et al., 2007) FAZL-E-HADI; SHAH, N.; SYED, A. H.; YASIN, M. Adaptive anycast: A new anycast protocol for performance improvement in delay tolerant networks. In *IEEE International Conference on Integration Technology* (*ICIT*), pp. 185-189, Mar. 2007.

(Ferreira, 2004) FERREIRA, A. Building a reference combinatorial model for MANETs. *IEEE Network*, vol. 18, no. 5, pp. 24-29, Sept./Oct. 2004.

(Goldberg, 1989) GOLDBERG, D. E. *Genetic algorithms in search, optimization and machine learning*. Massachusetts, Addison-Wesley, 1989.

(Gong et al., 2006) GONG, Y.; XIONG, Y.; ZHANG, Q.; ZHANG, Z.; WANG, W.; XU, Z. Anycast routing in delay tolerant networks. In *IEEE Global Telecommunications Conference 2006*, pp. 1-5, 2006.

(Harik et al., 1999) HARIK, G.; CANTÚ-PAZ, E.; GOLDBERG, D. E.; MILLER, B. L. The Gambler's ruin problem, genetic algorithm, and sizing of populations. *Evolutionary Computation*, vol. 7, no. 3, pp. 231-253, 1999.

(Hue, 1997) HUE, X. *Genetic algorithms for optimization: Background and applications*. Edinburgh Parallel Computing Centre, University of Edinburgh, Edinburgh, Scotland, Ver. 1.0, Feb. 1997.

(Inagaki et al., 1999) INAGAKI, J.; HASEYAMA, M.; KITAJIMA, H. A genetic algorithm for determining multiple routes and its applications. In *Proceedings of the 1999 IEEE International Symposium on Circuits and Systems*, vol. 6, pp. 137-140, 1999.

(Issariyakul e Hossain, 2009) ISSARIYAKUL, T.; HOSSAIN, E. *Introduction to network simulator NS2*. Copyrighted Material, Springer, 2009.

(Jain et al., 2004) JAIN, S.; FALL, K.; PATRA, R. Routing in a delay tolerant network. In *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 145-158, 2004.

(Jain et al., 2005) JAIN, S.; DEMMER, M.; PATRA, R.; FALL, K. Using redundancy to cope with failures in a delay tolerant network. *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 109-120, 2005.

(Johnson et al., 2001) JOHNSON, D. B.; MALTZ, D. A.; BROCH, J. DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks. In: *Ad Hoc Networking*. Addison-Wesley Longman Publishing Co., Massachusetts, pp. 139-172, 2001.

(Jones et al., 2007) JONES, E. P. C.; LI, L.; SCHMIDTKE, J. K.; WARD, P. A. S. Practical routing in delay-tolerant networks. *IEEE Transactions on Mobile Computing*, vol. 6, no. 8, pp. 943-959, Aug. 2007.

(Jong, 1975) JONG, K. A. D. *An analysis of the behavior of a class of genetic adaptive systems*. PhD dissertation, University of Michigan, 1975.

(Leung et al., 1998) LEUNG, Y.; LI, G.; XU, Z. B. A genetic algorithm for multiple destination routing problems. *IEEE Transactions on Evolutionary Computation*, vol. 2, no. 4, pp. 150-161, Nov. 1998.

(Lindgren et al., 2003) LINDGREN, A.; DORIA, A.; SCHELÉN, O. Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, no. 3, pp. 19-20, July 2003.

(Lo e Chang, 2000) LO, C. C.; CHANG, W. H. A multiobjective hybrid genetic algorithm for the capacitated multipoint network design problem. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*: *Cybernetics*, vol. 30, no. 3, pp. 461 – 470, June 2000.

(Munetomo et al., 1998) MUNETOMO, M.; TAKAI, Y.; SATO, Y. A migration scheme for the genetic adaptive routing algorithm. In *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 3, pp. 2774-2779, 1998.

(NS-2, 2008) The Network Simulator - NS-2. Disponível em: <http://www.isi.edu/nsnam/ns/>. Acesso em junho de 2008.

(NS-3, 2008) The NS-3 Network Simulator. Disponível em: <http://www.nsnam.org/>. Acesso em julho de 2008.

(Oh et al., 2006) OH, S.; AHN, C. W.; RAMAKRISHNA, R. S. A genetic-inspired multicast routing optimization algorithm with bandwidth and end-to-end delay constraints. In: Lectures Notes in Computer Science, *Neural Information Processing*, Springer, vol. 4234, pp. 807-816, 2006.

(Oliveira et al., 2007) OLIVEIRA, C. T.; MOREIRA, M. D. D.; RUBINSTEIN, M. G.; COSTA, L. H. M. K.; DUARTE, O. C. M. B. Redes tolerantes a atrasos e desconexões. Em *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos* (*SBRC'07*), pp. 203-256, maio de 2007.

(OLSR, 2007) OLSR website – An ad hoc wireless mesh routing daemon. Disponível em: <http://www.olsr.org/>. Acesso em dezembro de 2008.

(ONE, 2008) The Opportunistic Network Environment Simulator. Disponível em: <http://www.netlab.tkk.fi/tutkimus/dtn/theone/>. Acesso em dezembro de 2008.

(OPNET, 2007) OPNET Technologies. Disponível em: <http://www.opnet.com/>. Acesso em dezembro de 2008.

(Park e Corson, 2001) PARK, V.; CORSON, S. Temporally-ordered routing algorithm (TORA) version 1. IETF Internet Draft (draft-ietf-manet-tora-spec-04), July 2001.

(Perkins et al., 2003) PERKINS, C.; BELDING-ROYER, E.; DAS, S. *Ad-hoc on-demand distance vector (AODV) routing*. IETF RFC 3561, experimental, July 2003.

(Ramadas et al., 2008) RAMADAS, M.; BURLEIGH, S.; FARREL, S. *Licklider transmission protocol – specification*. IETF RFC 5326, experimental, Sept. 2008.

(Randaccio e Atzori, 2007) RANDACCIO, L. S.; ATZORI, L. Group multicast routing problem: A genetic algorithms based approach. *Computer Networks*, vol. 51, no. 14, pp. 3989-4004, 2007.

(Schaffer, 1985) SCHAFFER, J. D. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pp. 93-100, 1985.

(Scott e Burleigh, 2007) SCOTT, K.; BURLEIGH, S. *Bundle protocol specification*. IETF RFC 5050, experimental, Nov. 2007.

(Silva e Guardieiro, 2008) SILVA, E. R.; GUARDIEIRO, P. R. Anycast routing in delay tolerant networks using genetic algorithm for route decision. In *11th IEEE International Conference on Computer and Information Technology* (*ICCIT 2008*), Bangladesh, pp. 65-71, Dec. 2008.

(Silva e Guardieiro, 2009a) SILVA, E. R.; GUARDIEIRO, P. R. Effect of buffer size on anycast routing in delay tolerant networks. *Revista Científica Telecomunicações* (*Inatel*), vol. 12, no. 2, pp. 53-60, Dec. 2009.

(Silva e Guardieiro, 2009b) UNIVERSIDADE FEDERAL DE UBERLÂNDIA. Silva, E. R.; Guardieiro, P. R. *Método de roteamento anycast utilizando algoritmos genéticos em redes tolerantes a atrasos e desconexões*. BR n. PI 0905542-8 A2, 16/12/2009.

(Silva e Guardieiro, 2010) SILVA, E. R.; GUARDIEIRO, P. R. An efficient genetic algorithm for anycast routing in delay/disruption tolerant networks. *IEEE Communications Letters*, vol. 14, no. 4, pp. 315-317, Apr. 2010.

(Spyropoulos et al., 2008) SPYROPOULOS, T.; PSOUNIS, K.; RAGHAVENDRA, C. S. Efficient routing in intermittently connected mobile networks: The multiple-copy case. *IEEE/ACM Transactions on Networking*, vol. 16, no. 1, pp. 77-90, Feb. 2008.

(Spyropoulos et al., 2009) SPYROPOULOS, T.; TURLETTI, T.; OBRACZKA, K. Routing in delay-tolerant networks comprising heterogeneous node populations. *IEEE Transactions on Mobile Computing*, vol. 8, no. 8, pp. 1132-1147, Ago. 2009.

(Sudhaakar et al., 2009) SUDHAAKAR, R. S.; YOON, S.; ZHAO, J.; QIAO, C. A novel Qos-aware MAC scheme using optimal retransmission for wireless networks. *IEEE Transactions on Wireless Communications*, vol. 8, no. 5, pp. 2230-2235, May 2009.

(Symington et al., 2006) SYMINGTON, S.; FARRELL, S.; WEISS, H. Bundle security protocol specification. Internet-Draft (draft-irtf-dtnrg-bundle-security-02), Oct. 2006.

(Symington et al., 2007) SYMINGTON, S.; FARRELL, S.; WEISS, H.; LOVELL, P. Bundle security protocol specification. Internet-Draft (draft-irtf-dtnrg-bundle-security-04), Sept. 2007.

(Tamaki et al., 1996) TAMAKI, H.; KITA, H.; KOBAYASHI, S. Multi-objective optimization by genetic algorithms: A review. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pp. 517-522, May 1996.

(Vahdat e Becker, 2000) VAHDAT, A.; BECKER, D. *Epidemic routing for partially connected ad hoc networks*. Technical Report CS-200006, Department of Computer Science, Duke University, Durham, 2000.

(Wang e Wu, 2007) WANG, Y.; WU, H. Delay/Fault-Tolerant mobile sensor network (DFT-MSN): A new paradigm for pervasive information gathering. *IEEE Transactions on Mobile Computing*, vol. 6, no. 9, pp. 1021-1034, Sept. 2007.

(Warthman, 2003) WARTHMAN, F. *Delay-tolerant networks (DTNs): A tutorial v1.1*. Technical Report, Warthman Associates, Mar. 2003.

(Waxman, 1988) WAXMAN, B. M. Routing of multipoint connection. *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617 – 1622, Dec. 1988.

(Widmer e Le Boudec, 2005) WIDMER, J.; LE BOUDEC, J. Y. Network coding for efficient communication in extreme networks. In *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, pp. 284-291, 2005.

(Xianwei et al., 2000) XIANWEI, Z.; CHANGJIA, C.; GANG, Z. A genetic algorithm for multicasting routing problem. In *Proceedings of the International Conference Communication Technology*, pp. 1248-1253, 2000.

(Xiao et al., 2010) XIAO, M.; HUANG, L.; LIU, A.; CHEN, W. Anycast routing in probabilistically contacted delay tolerant networks. In *2010 International Conference on Communications and Mobile Computing (CMC)*, pp. 442-446, 2010.

(Yoon et al., 2007) YOON, S.; QIAO, C.; SUDHAAKAR, R. S.; LI, J.; TALTY, T. QoMOR: A QoS-aware MAC protocol using optimal retransmission for wireless intra-vehicular sensor networks. In *Mobile Networking for Vehicular Environments*, pp. 121-126, 2007.

(Zhang, 2006) ZHANG, Z. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: Overview and challenges. *IEEE Communications Surveys & Tutorials*, vol. 8, no. 1, pp. 24-37, First Quarter 2006.

(Zhang e Leung, 1999) ZHANG, Q.; LEUNG, Y. W. An orthogonal genetic algorithm for multimedia multicast routing. *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 1, pp. 53-62, Apr. 1999.

(Zhao et al., 2004) ZHAO, W.; AMMAR, M.; ZEGURA, E. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 187-198, 2004.

(Zhao et al., 2005) ZHAO, W.; AMMAR, M.; ZEGURA, E. Controlling the mobility of multiple data transport ferries in a delay-tolerant network. In *Proceedings of the IEEE INFOCOM 2005, 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, pp. 1407-1412, Mar. 2005.

(Zhao et al., 2006) ZHAO, W.; AMMAR, M.; ZEGURA, E. *Routing and network design in delay tolerant networks*. PhD Dissertation, Georgia Institute of Technology, 2006.

Apêndice A – EXEMPLO DE APLICAÇÃO DOS OPERADORES GENÉTICOS

Os operadores genéticos utilizados são: cruzamento de um ponto e mutação. Ambos os operadores utilizam o vetor de posições iniciais de cada sessão, denotado por sp_i, como ponto de corte. Para uma rede possuindo quatro sessões *anycast*, consideram-se X1 e X2 da Figura A.1 como os indivíduos sobre os quais serão aplicados os operadores genéticos para geração dos descendentes.

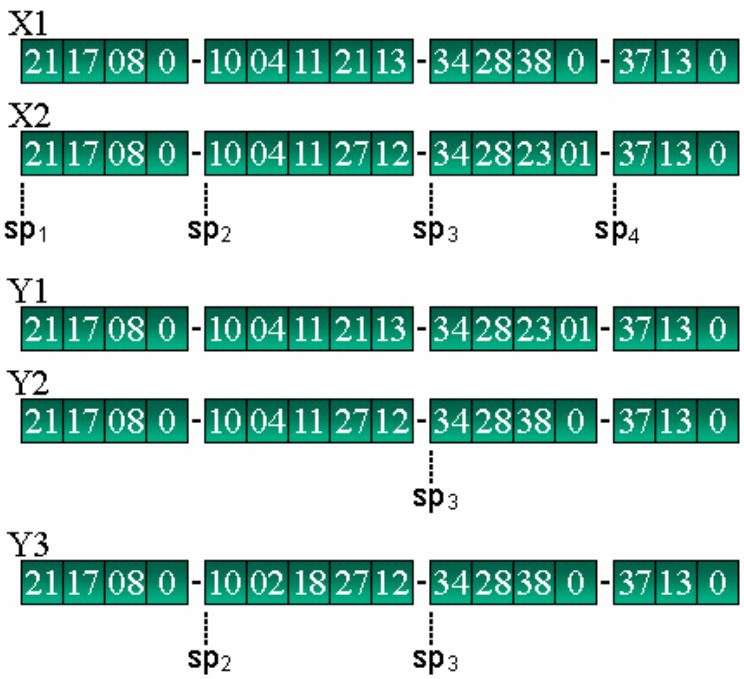

Figura A.1. Exemplo de funcionamento dos operadores genéticos.

Para o exemplo da Figura A.1 tem-se o vetor de posições iniciais de cada sessão sp_i, a seguir:

$$sp_i = [1 \quad 5 \quad 10 \quad 14]$$

O cruzamento de um ponto é realizado da seguinte forma: gera-se aleatoriamente um número entre um e o número de sessões *anycast* da rede (no caso quatro). Com esse número gerado, consulta-se o vetor de posições iniciais de cada sessão sp_i, para determinar a posição a partir da qual o material genético dos indivíduos (rotas de cada sessão) será trocado. Y1 e Y2 da Figura A.1 são os indivíduos gerados a partir do cruzamento entre X1 e X2, com o ponto de cruzamento sendo a terceira sessão (i = 3), ou seja, $sp_3 = 10$.

No caso da mutação, um número aleatório entre um e o número de sessões *anycast* da rede (no caso quatro), também é gerado. Com esse número, verifica-se o vetor de posições iniciais de cada sessão sp_i, para delimitar as posições entre as quais o material genético dos indivíduos será alterado. Y3 da Figura A.1 é o indivíduo gerado da mutação do indivíduo X1, com a sessão gerada aleatoriamente sendo a segunda. Assim, utilizam-se as soluções potenciais em isolamento para escolher, de forma aleatória, uma rota que substituirá a anterior. No caso a segunda sessão é delimitada por sp_2 e sp_3.

Apêndice B – EXEMPLO DE EXECUÇÃO DE UMA SIMULAÇÃO

O modo de geração da topologia e tráfego inicial de uma rede, além dos resultados obtidos por algoritmos de roteamento, utilizando o ambiente de simulação criado, é ilustrado no exemplo a seguir.

– A Figura B.1 mostra uma topologia de rede gerada pelo gerador de topologia de Waxman com os parâmetros para geração da topologia inicial possuindo os seguintes valores:

- Quantidade de nós: 40;
- Área da grade na qual são distribuídos os nós: 1300 m x 1300 m;
- α = 0,4 e β = 0,25.

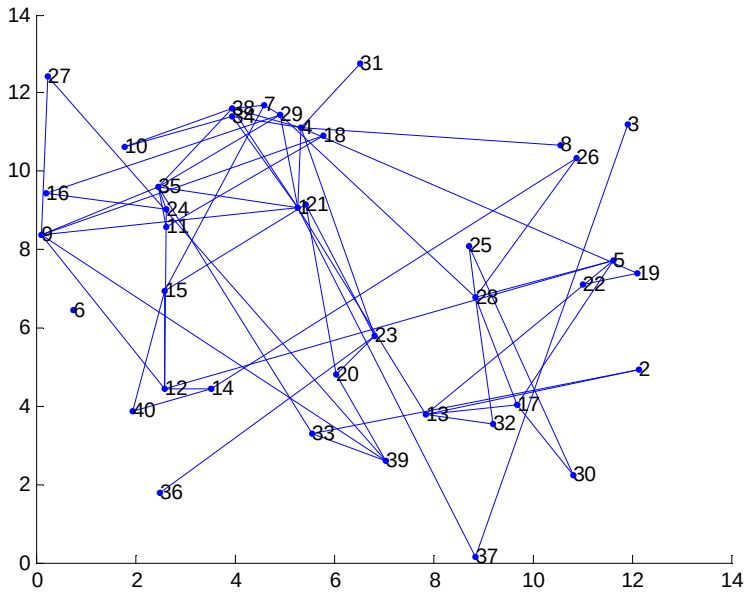

Figura B.1. Topologia de rede.

– Após a topologia de rede ser gerada, o tráfego inicial definido pelo usuário é criado. A Tabela B.1 mostra o tráfego inicial para uma rede com quatro sessões *anycast*.

Tabela B.1. Tráfego inicial.

	Nó fonte	Receptores planejados l	Mensagens enviadas pela fonte	Início da sessão (segundos)
Sessão 1	21	[8; 9]	380	1700
Sessão 2	10	[12; 13]	380	2150
Sessão 3	34	[38; 39; 40; 1; 2]	370	2700
Sessão 4	37	[12; 13; 14]	300	4300

Para cada sessão *anycast*, define-se aleatoriamente uma fonte dentre todos os 40 nós da rede, como pode ser visto na segunda coluna da Tabela B.1. Em seguida, o grupo de receptores planejados de cada sessão é gerado. Para isso, define-se uma quantidade mínima l_{min} e máxima l_{max} de receptores planejados. Para cada sessão *anycast*, gera-se de forma aleatória o número de receptores planejados l dentro do intervalo $l_{min} \leq l \leq l_{max}$. No exemplo anterior, l_{min} foi ajustado para 2 e l_{max} para 5. O primeiro nó de destino é escolhido aleatoriamente dentre todos os nós da rede, com exceção do nó fonte da sessão *anycast* em questão. Os demais nós são os nós na sequência com exceção do nó fonte, até atingir o número de receptores planejados, como encontrado na terceira coluna da Tabela B.1. As fontes de cada sessão *anycast* podem enviar uma quantidade limitada de mensagens dentro de um intervalo. No exemplo anterior, as fontes enviam mensagens dentro do intervalo de 300 a 400 mensagens, como pode ser visto na quarta coluna da Tabela B.1. Na quinta coluna é apresentado o tempo no qual a sessão é iniciada, ou seja, o momento a partir do qual a fonte da respectiva sessão irá enviar as mensagens ao grupo de receptores planejados.

– Juntamente com o tráfego inicial definido pelo usuário, as características de cada nó e enlace da topologia de rede são definidas. Neste caso, inclui-se a definição do tempo de partida dos dispositivos móveis $w(i,j)$ em cada enlace, do atraso de movimento $md(i,j)$, além da capacidade de armazenamento dos nós $b(i)$ e dispositivos móveis $c(i,j)$.

– A próxima etapa envolve o emprego dos algoritmos de roteamento para cálculo de rotas. Após a definição de rotas, os resultados são gerados. A Tabela B.2 apresenta as rotas e medidas de desempenho obtidas pelos algoritmos de roteamento SP e GA3.

Tabela B.2. Exemplo de medidas de desempenho.

	Rota para sessão 1	Rota para sessão 2	Rota para sessão 3	Rota para sessão 4	*DP*(%)	*D*(s)	Saltos
SP	21-17-8	10-4-11-21-13	34-2-38	37-13	77	11398	9
GA3	21-17-8	10-4-11-27-12	34-28-23-1	37-13	91	10450	10

O algoritmo de roteamento baseado em GA (GA3) obteve um desempenho superior ao algoritmo SP. Estes resultados se devem ao modo de funcionamento do algoritmo GA3, que busca a melhor combinação de rotas e, por isso, evita que várias rotas passem por um mesmo nó ou enlace. Neste sentido, pode-se observar que as rotas para as sessões 2 e 4 obtidas pelo algoritmo SP possuem o mesmo receptor de destino (nó 13). Já o algoritmo GA3 busca evitar a passagem de várias mensagens pelo mesmo nó ou enlace, o que poderia prejudicar o desempenho do algoritmo de roteamento.