Multi-Priority Alternative Journey and Routing Protocol: um Algoritmo para Roteamento em Redes Tolerantes a Atrasos e Desconexões Previsíveis

Gabriel Argolo[1], Lúcia M. A. Drummond[1], Anna Dolejsi[1], Anand Subramanian[1],

[1]Instituto de Computação - Universidade Federal Fluminense
Rua Passo da Pátria 156 - Bloco E - 3º andar, São Domingos, CEP: 24210-240, Niterói - RJ

{grocha,lucia,annads,anand}@ic.uff.br

Abstract. *This work proposes the distributed algorithm Multi-Priority Alternative Journey and Routing Protocol (MP-AJRP) for routing in Delay-disruption Tolerant Networks, considering constraints such as links' bandwidth and nodes' buffer capacity. The MP-AJRP is an improved version of the AJRP algorithm and its goal is to deliver the largest number of messages to the destinations, considering alternative routes as well as multi-priorities for message scheduling. In order to evaluate the algorithm, several experiments were done using the traces produced with the contact history of the DieselNet nodes. The results showed that MP-AJRP delivered about 76% of messages and had a better performance when compared to certain algorithms that use message replication.*

Resumo. *Este trabalho propõe o algoritmo distribuído Multi-Priority Alternative Journey and Routing Protocol (MP-AJRP) para roteamento em Redes Tolerantes a Atrasos e Desconexões, considerando as restrições de largura de banda dos enlaces e de capacidade dos buffers dos nós. O MP-AJRP é uma versão aprimorada do algoritmo AJRP e tem como objetivo entregar o maior número de mensagens aos destinos considerando rotas alternativas e múltiplas prioridades para escalonamento de mensagens. Com o intuito de avaliar o algoritmo, foram realizados experimentos utilizando os traces com o histórico de contatos dos nós da rede DieselNet. Os resultados mostraram que o MP-AJRP entregou em torno de 76% das mensagens e obteve desempenho superior a determinados algoritmos que consideram a replicação de mensagens.*

1. Introdução

Redes Tolerantes a Atrasos e Desconexões (DTN - *Delay-disruption Tolerant Network*) possibilitam a transmissão de dados quando dispositivos móveis estão conectados intermitentemente. Neste tipo de rede, a comunicação entre os nós é feita diretamente ou através de nós intermediários que atuam como roteadores. A conectividade intermitente pode ser resultado da mobilidade dos nós, da potência de sinal ou até mesmo do gerenciamento de energia. DTNs são encontradas, por exemplo, em redes de sensores para monitoramento ecológico, na comunicação entre sistemas de satélites e em redes veiculares. Estas redes diferem da tradicional *Internet*, pois é assumido que esta última possui conectividade ininterrupta, além da baixa taxa de perda de pacotes e do baixo retardo de propagação.

Partindo deste pressuposto, os protocolos desenvolvidos para *Internet* cabeada são ineficazes para transmissão de dados em DTNs.

DTNs podem ser classificadas como previsíveis, também conhecidas como redes com contatos programados, imprevisíveis e probabilísticas. Na primeira, a variação da topologia ao longo do tempo é conhecida antecipadamente. As redes de satélites do tipo LEO (*Low Earth Orbit*) são um exemplo, onde as trajetórias dos satélites são previamente programadas. Na segunda, não se conhece de antemão as alterações que podem ocorrer na topologia como, por exemplo, nas redes *ad-hoc* onde os nós podem se mover arbitrariamente ao longo do tempo [Merugu et al. 2004, Oliveira e Duarte 2007]. Já nas probabilísticas, apesar de não conhecer *a priori* exatamente quando ocorrerão futuros contatos e a duração dos mesmos, é possível obter uma aproximação destes valores através da aplicação de métodos probabilísticos.

A variação da topologia da rede pode dificultar o roteamento em DTNs ao longo do tempo. A necessidade de rotear mensagens até os destinos, assim como assegurar uma comunicação eficiente, são desafios encontrados nestes tipos de rede. As desconexões frequentes causadas pelo deslocamento dos nós, o elevado tempo de permanência de mensagens nas filas e a possível inexistência de um caminho fim-a-fim são alguns dos problemas existentes.

O encaminhamento das mensagens até os destinos pode ser realizado através de diversos mecanismos como a duplicação das mensagens em nós intermediários, o encaminhamento das mensagens pelo primeiro enlace disponível ou a utilização de tabelas de roteamento [Oliveira e Duarte 2007]. Neste último caso, a construção das tabelas pode ser feita adotando-se uma abordagem centralizada, ou seja, cada nó dispõe previamente de toda informação relevante para o roteamento, ou distribuída, onde os nós não conhecem de antemão o estado global da rede. Apesar das tabelas poderem ser construídas de forma centralizada, esta abordagem apresenta inconvenientes. A necessidade de manter nos nós a informação global sobre o estado da rede torna-se difícil à medida que o tamanho da rede aumenta e também devido à intermitência dos canais de comunicação. Logo, a construção de tabelas de roteamento através de algoritmos distribuídos apresenta-se como uma solução mais apropriada.

Embora algumas estratégias de encaminhamento valham da duplicação de mensagens na rede, esta aborgadem pode acarretar em uma utilização ineficaz de recursos quando aplicado em redes com largura de banda limitada e reduzida capacidade de armazenamento dos nós. Outras formas de encaminhamento necessitam do conhecimento prévio de toda a rede, apesar disto não ser simples de obter na prática [Peleg 2000]. Além disso, até nestes casos não há garantia da entrega das mensagens, mesmo havendo uma rota fim-a-fim, dado que as mensagens podem ser perdidas devido às limitações de capacidade de armazenamento dos nós e da largura de banda dos enlaces. Portanto, o projeto de algoritmos que utilizem mecanismos de encaminhamento sem duplicação de mensagens torna-se uma alternativa interessante.

Este trabalho propõe o algoritmo distribuído *Multi-Priority Alternative Journey and Routing Protocol* (MP-AJRP) para encaminhamento de mensagens em DTNs previsíveis cujo objetivo é entregar o maior número de mensagens aos destinos, executando um mecanismo de escolha de rotas alternativas para evitar o roteamento das mensagens

por nós com *buffer* saturado. O MP-AJRP é um aprimoramento do algoritmo proposto em [Argolo et al. 2009], pois, além da abordagem *first-in-first-out* (FIFO), considera outras duas políticas de seleção de mensagens do *buffer* nos nós pertencentes às rotas das mensagens. O encaminhamento das mensagens é feito sem duplicá-las em nós intermediários e cada nó conhece apenas os intervalos de disponibilidade para comunicação com seus vizinhos e a tabela de roteamento.

As principais contribuições deste trabalho são:

- Elaboração do algoritmo MP-AJRP para encaminhamento de mensagens até os destinos considerando jornadas alternativas e múltiplas políticas de seleção de mensagens no *buffer*; e
- Realização de experimentos para avaliar o desempenho dos algoritmos propostos e comparando os resultados obtidos com outras abordagens encontradas na literatura

O restante deste trabalho está organizado da seguinte forma. A Seção 2 apresenta trabalhos relacionados ao problema de roteamento em DTNs. A Seção 3 descreve o modelo adotado para a solução do problema. A Seção 4 descreve e analisa os algoritmos distribuídos propostos. Os resultados experimentais para análise do desempenho dos algoritmos são discutidos na Seção 5. A Seção 6 apresenta as considerações finais e propostas para trabalhos futuros.

2. Trabalhos relacionados

Nesta seção são abordados diversos trabalhos encontrados na literatura com relação a propostas de algoritmos para roteamento em DTNs.

Em [Bui-Xuan et al. 2003], três algoritmos centralizados denominados *foremost journey*, *shortest journey* e *fastest journey* foram desenvolvidos com o objetivo de encontrar, respectivamente, as jornadas mais cedo, ou seja, as jornadas onde o instante de tempo de chegada da mensagem nos nós de destino é o menor possível, as jornadas com menor número de saltos e as jornadas mais rápidas, isto é, as que apresentam as menores diferenças entre o instante de tempo de chegada da mensagem no destino e o instante de envio da mesma.

O algoritmo PROPHET é proposto em [Lindgren et al. 2004] para realizar o roteamento das mensagens utilizando como base a probabilidade que os nós possuem de encontrar uns aos outros. Esta probabilidade é obtida através de cálculos realizados levando em consideração o histórico de contatos dos mesmos ao longo do tempo. Nenhuma informação quanto às alterações futuras na topologia da rede é conhecida antecipadamente. Apesar de considerar restrições na capacidade de armazenamento dos nós, o algoritmo não leva em conta as limitações na largura de banda do enlaces. Um mecanismo de replicação de mensagens é utilizado, onde as cópias das mensagens são enviadas para os nós que possuem as maiores probabilidades de encontrar os respectivos destinos.

Em [Jain et al. 2004], os autores implementam algoritmos de roteamento que, baseado em oráculos, utilizam informações sobre o estado atual e futuro da rede como os contatos entre os nós ao longo do tempo, a demanda de mensagens e a ocupação dos *buffers*. A estratégia de encaminhamento utilizada envia as mensagens para os nós intermediários até atingir os destinos sem que as mensagens sejam duplicadas. Os experi-

mentos realizados com baixa e alta carga de mensagens na rede, mostram que os algoritmos que atingem o melhor desempenho com relação ao roteamento das mesmas são os que possuem mais informações acerca das características da rede. Os algoritmos também foram avaliados variando-se a capacidade de armazenamento dos nós e a largura de banda dos enlaces.

Chen em [Chen 2005] propõe o algoritmo SGRP para roteamento em redes de satélites com trajetórias previsíveis executado em duas etapas. Inicialmente, a coleta da informação de um grupo de satélites de baixa altitude (LEOS) é feita por um satélite de médio alcance (MEO). Em seguida, os diversos MEOS trocam as informações obtidas entre si e com a informação global disponível calculam as tabelas de roteamento e as redistribuem para os LEOS.

O algoritmo Spray and Wait, elaborado em [Spyropoulos et al. 2005], considera um mecanismo de replicação que gera L cópias de cada mensagem e as distribui entre os contatos esperando que algum deles por ventura encontre o nó de destino. São analisadas algumas estratégias de priorização dos contatos que devem receber as cópias das mensagens, assim como realizada uma avaliação com relação ao número L de cópias a serem geradas considerando o tamanho da rede e a demanda de mensagens. O algoritmo não necessita de nenhuma informação prévia sobre a topologia da rede e não realiza nenhum tipo de verificação quanto à capacidade de *buffer* dos nós e largura de banda dos enlaces.

Outro algoritmo, denominado MaxProp, foi proposto em [Burgess et al. 2006] e utiliza informações de histórico de contatos para determinar a prioridade das mensagens a serem transmitidas. Um esquema de propagação de mensagens de controle para confirmação de recebimento também é implementado juntamente com uma política de replicação de mensagens e de exclusão de réplicas. As restrições de capacidade de *buffer* dos nós e largura de banda dos enlaces são consideradas, mas nenhuma informação sobre o estado da rede é conhecida antecipadamente.

O algoritmo Rapid foi desenvolvido e avaliado em [Balasubramanian et al. 2007] e tem o objetivo de rotear as mensagens até o destino por meio da replicação destas nos nós intermediários. Para evitar a sobrecarga de mensagens na rede implementou-se um mecanismo que determina se uma mensagem deve ser replicada ou removida em determinados nós intermediários. Este algoritmo não necessita de nenhuma informação prévia sobre o estado da rede, porém, utiliza informações relativas ao histórico desta para estimar novas alterações na topologia. Uma avaliação foi realizada utilizando os *traces* da rede veicular *DieselNet*, onde o Rapid foi comparado com os algoritmos MaxProp [Burgess et al. 2006], Spray and Wait [Spyropoulos et al. 2005] e PROPHET [Lindgren et al. 2004] utilizando distintas cargas de mensagens. Outro algoritmo, denominado Random, também foi implementado pelo autor com o intuito de avaliar o desempenho da entrega de mensagens quando aplicado um mecanismo de duplicação de mensagens de forma randômica entre os vizinhos. Foi verificado também o desempenho do Rapid comparado a uma solução ótima obtida através da formulação em programação linear elaborada também neste trabalho.

O algoritmo NECTAR proposto em [Oliveira e Albuquerque 2009] utiliza o conceito de índice de vizinhança, considerando que os nós movimentam-se de forma que existe certa probabilidade que vizinhos possam ser reencontrados. Políticas de escalo-

namento e descarte de mensagens são desenvolvidas e um mecanismo de replicação de mensagens é implementado. O algoritmo não considera para encaminhamento das mensagens nenhuma informação sobre o estado futuro da rede. Uma avaliação é realizada comparando os resultados do NECTAR com os obtidos pelos algoritmos Epidemic Routing [Vahdat e Becker 2000] e PROPHET [Lindgren et al. 2004] considerando tamanhos distintos de *buffer* e verificando a quantidade de mensagens entregues e o número de saltos realizados pelas mesmas.

Três algoritmos distribuídos para construção de tabelas de roteamento em DTNs previsíveis foram propostos em [Santos et al. 2008]. Em um dos algoritmos, o *Distributed Shortest Journey* (DSJ), cada nó calcula a tabela de roteamento dele para todos os outros nós considerando o menor número de saltos. No outro, denominado *Distributed Earliest Journey* (DEJ), as tabelas são construídas objetivando a chegada mais cedo da informação ao nó de destino. Por último, o algoritmo *Distributed Fastest Journey* (DFJ), realiza a construção das tabelas levando-se em consideração as jornadas mais rápidas para chegar até os destinos. Em todos os algoritmos a construção da tabela em cada nó é realizada à medida que os enlaces para os nós vizinhos tornam-se disponíveis e novas informações sobre o estado da rede são obtidas. As tabelas geradas mantêm, para cada nó de destino, uma lista ordenada dos intervalos de tempo de disponibilidade dos enlaces adjacentes. Cada intervalo é único na lista e determina qual vizinho deve receber a mensagem para posterior encaminhamento para cada nó de destino. Os algoritmos realizam um filtro nos instantes de tempo de disponibilidade dos enlaces adjacentes para evitar o envio desnecessário de mensagens de controle pelos canais de comunicação.

Em [Argolo et al. 2009], além do algoritmo AJRP, também é proposto um modelo de Programação Linear Inteira para DTNs baseado na abordagem *multi-commodities flow*. Esta formulação matemática difere das demais [Jain et al. 2004, Balasubramanian et al. 2007], pois a complexidade para encontrar a solução ótima é diretamente proporcional ao número de nós e não ao número de mensagens da rede. Uma comparação dos resultados do AJRP com a solução ótima foi realizada considerando as limitações de largura de banda dos enlaces e de capacidade de armazenamento dos nós. A avaliação mostra que o AJRP obteve um desempenho satisfatório, entregando cerca de 96% da demanda de mensagens quando atribuída baixa carga de mensagens na rede e cerca de 83% quando submetida a uma alta carga.

3. Modelo

DTNs previsíveis podem considerar várias informações conhecidas antecipadamente, tais como: a disponibilidade de contato entre os nós da rede ao longo do tempo; as demandas de mensagens de cada nó; e a capacidade de armazenamento destes. No entanto, a obtenção a *priori* de todo este conhecimento é praticamente inviável devido ao tamanho da rede ou até mesmo a própria dinâmica de geração das mensagens.

O problema de roteamento em DTNs consiste em entregar as mensagens aos destinos de acordo com uma métrica pré-determinada, levando-se em conta as restrições de disponibilidade e capacidade dos enlaces para transmissão das mensagens, assim como as limitações de armazenamento destas pelos nós da rede. As desconexões presentes nestas redes podem implicar na inexistência de uma rota fim-a-fim entre a origem e o destino.

O presente trabalho utiliza o modelo para DTNs proposto em [Argolo et al. 2009],

que considera que apenas os períodos de disponibilidade dos enlaces adjacentes são conhecidos previamente por cada nó. Para contornar as limitações de conectividade o modelo utiliza o conceito de jornada, que são definidas como rotas construídas considerando os instantes de tempo de existência dos enlaces. Desta forma, os nós intermediários armazenam em seus *buffers* as mensagens recebidas e as encaminham em momentos adequados, evitando que estas sejam roteadas para enlaces que estavam disponíveis apenas no passado. A Figura 1 ilustra uma rede onde as jornadas válidas para envio de mensagens do nó A para o nó C, devem considerar o nó B como intermediário. Ou seja, o nó A deverá enviar as mensagens para B entre os instantes 1 e 10, o nó B armazenará as mensagens em seu *buffer* e poderá encaminhá-las para o nó C a partir do instante 20.

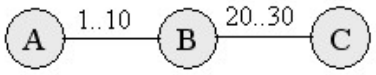

Figura 1. Exemplo de jornada

O roteamento das mensagens na rede segue uma métrica específica cujo objetivo é mensurar a qualidade da solução obtida. Exemplos de métricas que podem ser adotadas são: tempo médio ou máximo para entrega das mensagens; percentual de mensagens entregues dentro de um prazo pré-estabelecido; e número máximo de mensagens entregues [Ramanathan et al. 2007, Balasubramanian et al. 2007]. O presente trabalho optou por explorar este último critério, pois este representa um dos objetivos mais básicos e importantes com relação à entrega de mensagens em uma DTN.

Uma arquitetura para DTN com contatos programados ou previsíveis deve considerar sincronismo de tempo entre os nós da rede, conforme descrito na *Request For Comments* [Cerf 2007] publicada no *Internet Engineering Task Force* (IETF). No modelo em questão, foi definido que cada período de tempo é denominado pulso. Portanto, a cada pulso p os nós podem criar, receber, processar e enviar mensagens. Considera-se que a comunicação entre nós adjacentes é realizada dentro de um pulso, ou seja, o envio e recebimento de mensagens de controle e de aplicação não podem ultrapassar o término do pulso em que estas tiveram sua transmissão iniciada. Assume-se ainda que a fragmentação das mensagens não é admitida e que os canais de comunicação obedecem a ordem FIFO (*first-in-first-out*). Ressalta-se que os nós da rede usam esta informação para controle síncrono do tempo e verificação de quando cada enlace adjacente está ativo para comunicação.

DTNs podem apresentar propriedades cíclicas, isto é, após o último pulso a contagem é reiniciada e a execução retorna para o primeiro pulso, onde volta-se novamente a topologia de rede inicial. Para efeito de avaliação admitiu-se apenas a execução do primeiro ciclo e, por conseguinte, as mensagens não entregues neste período são descartadas.

O modelo estabelece que uma DTN previsível pode ser representada por um grafo orientado e subdividido em pulsos, isto é, períodos de tempo de mesma dimensão. Cada nó da rede possui identificação distinta e é representado por vértices em diferentes pulsos. Cada enlace é simbolizado por dois arcos de capacidade finita, onde um deles conecta um vértice v_1 no pulso t_x ao v_2 em t_{x+1}, enquanto o outro arco conecta o vértice v_2 no pulso t_x

ao v_1 em t_{x+1}. No caso do *buffer*, os vértices de origem e destino referenciam o mesmo nó da rede. Considera-se ainda que a largura de banda disponível é independente para cada um dos contatos, isto é, não é admitido o compartilhamento dos canais de comunicação estabelecidos simultaneamente com nós vizinhos.

A Figura 2 mostra um exemplo do modelo proposto. O grafo ilustrado contém 4 nós (a, b, c e d) indexados pelos instantes de tempo e seu ciclo de execução ocorre a cada 4 pulsos. Observa-se que entre os pulsos 1 e 2 os enlaces entre os nós a e b e entre os nós c e d estão disponíveis. No entanto, entre os instantes 2 e 3 apenas a comunicação entre a e b permanece disponível. Como pode ser notado, os vértices a e d não possuem conectividade, mas o d é alcançável por a através do vértice intermediário b. Para tanto, basta que, no pulso 1, a encaminhe a mensagem para b, que por sua vez deverá armazenar a mensagem no pulso 2 e reencaminhá-la para d no pulso 3. Verifica-se que a recíproca não é verdadeira, pois não existe jornada partindo de d até a.

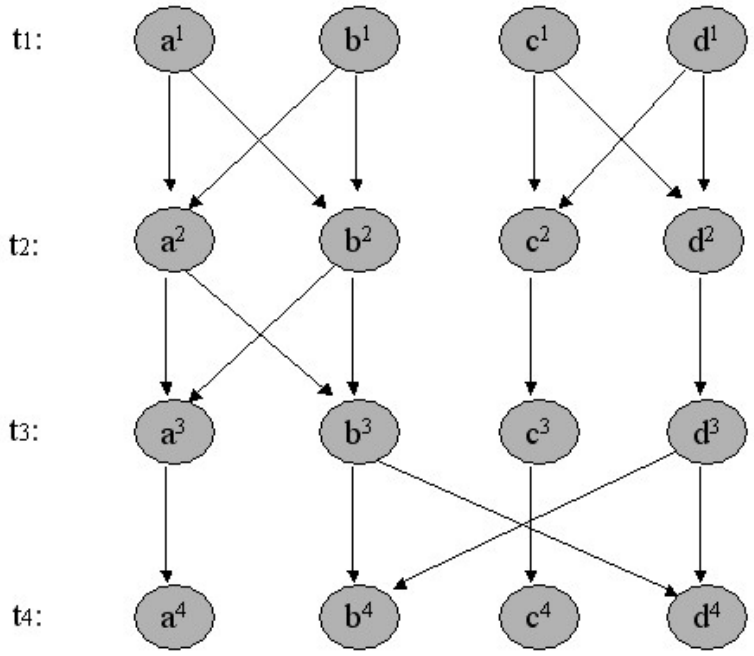

Figura 2. Exemplo de representação de uma DTN

4. O algoritmo MP-AJRP

Esta seção apresenta o algoritmo distribuído *Multi-Priority Alternative Journey Routing Protocol* (MP-AJRP) que tem como objetivo realizar o roteamento de mensagens em DTNs considerando as restrições de largura de banda dos enlaces e de capacidade de armazenamento dos nós. A estratégia de roteamento distribuído proposta é composta de duas etapas. Primeiramente, executa-se um algoritmo para construção de tabelas de roteamento em cada nó da rede, mapeando as jornadas viáveis até os destinos. A partir destas tabelas, realiza-se o envio e recebimento das mensagens empregando um mecanismo de escolha de jornadas alternativas evitando, assim, a sobrecarga de mensagens em determinados nós da rede.

Conforme mencionado na Seção 3, um ciclo é dividido em instantes de tempo denominados pulsos. Em cada pulso, quando necessário, os nós enviam mensagens de

controle e aplicação para os vizinhos que estão com enlace ativo e processam as mensagens recebidas. Assume-se que a transmissão é realizada dentro de um pulso e que os tempos de processamento das mensagens são desprezíveis. Classificam-se as mensagens como referentes à aplicação, ou seja, advindas de uma camada de *software* de nível superior ao roteamento, ou como mensagens de controle, que servem como auxílio para os algoritmos executarem o roteamento. Define-se ainda que mensagens são perdidas apenas quando o nó receptor atingiu sua capacidade máxima de armazenamento ou quando as mesmas não alcançaram o destino até o último pulso t_f, ou seja, permaneceram no *buffer* de algum nó intermediário.

Uma vez construídas as tabelas de roteamento, cada nó tem conhecimento dos momentos adequados para envio de mensagens de aplicação para os demais nós da rede. No entanto, à medida que mais mensagens são geradas, tanto a utilização dos canais de comunicação como a dos *buffers* dos nós intermediários aumentam, congestionando as jornadas e provocando o atraso na entrega ou até a perda de mensagens.

Em virtude disto, o MP-AJRP faz uso de mensagens de controle com o intuito de balancear a carga de mensagens entre as jornadas disponíveis para cada nó da rede. Isto é feito através do envio do percentual de ocupação do *buffer* dos nós para seus vizinhos. A partir desta informação, os nós escolhem como preferencial uma jornada que passa pelo vizinho com maior disponibilidade de armazenamento de mensagens no momento. Isto posto, as mensagens de aplicação são encaminhadas para jornadas que apresentam menos congestionamento, prevenindo eventuais gargalos na rede. Além disso, as tabelas de roteamento utilizadas consideram os menores números de saltos que a mensagem de aplicação realiza até chegar ao destino, reduzindo a utilização de recursos da rede e, consequentemente, a perda de mensagens devido às limitações de armazenamento dos nós e à capacidade de trasmissão dos enlaces da rede.

Para avaliação do MP-AJRP foi empregada uma estrutura de dados para mapear os instantes de tempo de geração das mensagens pelos nós. Embora o algoritmo não faça uso desta informação para tomada de decisões de roteamento, ou seja, os nós não detêm conhecimento *a priori* da lista de demandas futuras, esta informação é necessária para simulação da geração das mensagens nos momentos pré-estabelecidos nos experimentos.

Os dados utilizados pelo Algoritmo 1, assim como as principais variáveis e suas inicializações, estão descritos a seguir.

O algoritmo MP-AJRP é síncrono e sua apresentação é feita através de dois eventos que, por sua vez, demandam uma ação específica. O primeiro, refere-se à contagem do tempo, onde, a cada pulso p, mensagens de aplicação podem ser recebidas pelos nós e armazenadas na variável $buffer_i$ (linhas 5 a 12). Mensagens também podem ser geradas neste momento de acordo com a variável $demandList_i$ (linhas 13 a 17). O envio destas, e de outras que estiverem no *buffer* do nó, é feito caso o enlace com o vizinho apropriado esteja disponível e sua capacidade ainda não tenha sido esgotada (linhas 22 a 26). As funções $enqueue(message, pol)$ (linha 8 e 15) e $dequeue(j, pol)$ (linha 22) são empregadas, respectivamente, para incluir uma mensagem no *buffer* do nó e para extrair uma mensagem que tenha um destino tal que a jornada para este passe pelo vizinho j, ambas utilizando uma determinada política pol. Esta política pode ser a FIFO, onde a escolha da mensagem no *buffer* é realizada seguindo a ordem em que foram inseridas, a HOP,

Dados

N	=	conjunto de nós da rede
M	=	conjunto de enlaces da rede
R	=	conjunto de jornadas para cada destino
Y	=	conjunto de intervalos de tempo
pol	=	política de seleção das mensagens do *buffer* - FIFO, HOP ou MEET
$Neig_i$	=	conjunto de vizinhos do nó i
$myRank_i$	=	identificação do nó i

Variáveis

p	=	0
SET_i	=	$nulo$ (conjunto de mensagens enviadas pelo nó i)
$message$	=	$nulo$
$setSize$	=	0
$buffer_i$	=	$nulo$
$bufferUsage_i$	=	0
$bufferSize_i$	=	tamanho do *buffer* do nó i
$status_i[j]$	=	0, $\forall\, j \in Neig_i$
$vecInter_i[j]$	=	(vizID, capacidade, up[], down[]), $\forall\, n_j \in Neig_i$ (vetor com informações sobre o enlace entre os nós i e j)
$vecUsage_i[j]$	=	$nulo$, $\forall\, j \in Neig_i$
$routeTable_i[j][r][y]$	=	$nulo$, $\forall\, j \in N$, $r \in R$ e $y \in Y$
$demandList_i[j][y]$	=	conjunto de mensagens para envio, tal que $j \in N$ e $y \in Y$

onde as mensagens são ordenadas priorizando as que são destinadas a nós que demandem o menor número de saltos, e a MEET, onde a ordenação das mensagens é feita beneficiando aquelas destinadas a nós que possuem o menor número de intervalos de tempo de disponibilidade durante sua jornada. Estas informações são obtidas através da tabela de roteamento gerada pelo algoritmo descrito em [Santos et al. 2008].

O segundo evento trata das mensagens de controle recebidas. A ação tomada neste caso é verificar para cada destino da tabela de roteamento, representada pela variável $routeTable_i$, qual jornada está menos congestionada e considerar esta como preferencial para o roteamento das próximas mensagens (linhas 36 a 43).

4.1. Complexidade do algoritmo

O algoritmo MP-AJRP termina sua execução após processamento do último pulso $t_f \in T$, isto é, a complexidade de tempo é da ordem de $O(T)$. A quantidade total de pulsos (T) depende da DTN em questão. Ou seja, de acordo com os tempos de disponibilidade dos enlaces da rede será identificada a quantidade de pulsos de execução.

Com relação a quantidade de mensagens de controle enviadas pelo MP-AJRP, considerando que cada enlace $e \in M$ possua y_e ativações, e que cada período iniciado por um y possua p_{ey} pulsos, o número de mensagens de controle enviado é $\sum_{e=1}^{|M|} \sum_{y=1}^{|y_e|} p_{ey}$.

A tabela de roteamento mapeia os Y intervalos de cada conjunto R de jornadas possíveis para cada um dos $N-1$ nós de destino. Assim sendo, a complexidade de

```
1  Algorithm MP − AJRP(verInter, routeTable, demList)
2  INPUT:
3    p > 0, SET_j ∈ MSG_i(p), origem_i(SET_j) = (n_i, n_j);
4  ACTION:
5  forall message ∈ SET_j do
6      if (message.destId ≠ myRank_i) then
7          if (bufferUsage_i < bufferSize_i) then
8              enqueue(message, pol);
9              bufferUsage_i ← bufferUsage_i + 1;
10         end
11     end
12 end
13 forall message ∈ demandList_i[j][y], j ∈ N, p = y, y ∈ Y do
14     if (bufferUsage_i < bufferSize_i) then
15         enqueue(message, pol);
16         bufferUsage_i ← bufferUsage_i + 1;
17     end
18 end
19 forall active (i, j) in p, j ∈ Neig_i do
20     SET_i[k] ← nulo, ∀ k < setSize;
21     setSize ← 0;
22     while ((message = dequeue(j, pol)) ≠ nulo) AND
       (setSize ≤ vecInter_i(j).capacidade) do
23         bufferUsage_i ← bufferUsage_i − 1;
24         setSize ← setSize + 1;
25         SET_i[setSize] ← message;
26     end
27     SEND SET_i to n_j;
28     percent ← (bufferUsage_i/bufferSize_i) ∗ 100;
29     SEND Control(percent) to n_j;
30 end
31 INPUT:
32   p > 0, Control_i(percent), origem_i = n_j;
33 ACTION:
34 vecUsage_i[j] ← percent;
35 choice ← vecUsage_i[0];
36 forall k ∈ N do
37     forall r ∈ R, y ∈ Y, y > p do
38         if vecUsage_i[j] < choice then
39             choice ← vecUsage_i[j];
40         end
41     end
42     routeTable_i[k].choice ← choice;
43 end
```

Algorithm 1: Algoritmo MP-AJRP

armazenamento desta estrutura de dados é de $O(NRY)$.

5. Resultados experimentais

A avaliação dos algoritmos foi realizada através de simulações utilizando um ambiente com 5 computadores conectados em LAN. Cada máquina possui processador Intel Pentium IV 2.8GHz e memória RAM de 512MB, rodando o sistema operacional Ubuntu V3. A implementação foi feita em linguagem C utilizando a biblioteca MPI para troca de mensagens. Cada nó das instâncias de rede utilizadas nos testes foi simulado por um processo MPI e alocado em uma das máquinas da LAN.

Nesta seção são apresentados os resultados com relação ao encaminhamento das mensagens até os destinos, considerando as restrições de largura de banda dos enlaces e da capacidade de armazenamento dos nós.

Experimentos foram realizados com o MP-AJRP utilizando os *traces* gerados pela rede veicular *DieselNet* entre 14 de fevereiro e 15 de maio de 2007, onde foram consideradas as mesmas capacidades de *buffer* dos nós, larguras de banda dos enlaces e demandas de mensagens utilizadas em [Balasubramanian et al. 2007]. A Tabela 1 apresenta estas informações. Para evitar a geração de mensagens para destinos que nunca seriam alcançados foram considerados como potenciais destinos apenas os ônibus circulantes no dia corrente. Definiu-se como carga de mensagem a quantidade de mensagens geradas a cada hora para todos os destinos disponíveis no *trace* do dia corrente.

Primeiramente, foi avaliado apenas o algoritmo MP-AJRP utilizando três políticas distintas para seleção de mensagens no *buffer*, a saber: FIFO, HOP e MEET. Na primeira, a escolha da mensagem no *buffer* é realizada seguindo a ordem em que foram inseridas. Na segunda, as mensagens são ordenadas priorizando as que são destinadas a nós que demandem o menor número de saltos. Na última, a ordenação das mensagens é feita beneficiando aquelas destinadas a nós que possuem o menor número de intervalos de tempo de disponibilidade durante sua jornada.

Nas execuções do MP-AJRP foram utilizadas as tabelas de roteamento geradas pelo algoritmo DSJ, mantendo até três jornadas por intervalo $(R = 3)$ para cada destino.

Capacidade de armazenamento de cada ônibus	40GB
Quantidade total de ônibus	40
Quantidade média de ônibus por dia	20
Hora de início da avaliação para cada dia de execução	8 horas
Hora de término da avaliação para cada dia de execução	19 horas
Hora de início da geração de mensagens	8 horas
Hora de término da geração de mensagens	15 horas
Tempo entre a geração das cargas de mensagens	1 hora
Tempo de cada pulso	1 segundo

Tabela 1. Descrição do ambiente de teste com os *traces* da DieselNet

A Figura 3 ilustra a porcentagem de entrega de mensagens para cada um dos destinos. Como pode ser observado, a seleção através do menor número de saltos conseguiu entregar, para todas as cargas de mensagem avaliadas, o maior número de mensagens aos destinos, seguido pela política MEET e por último a FIFO. Observa-se também que estas duas últimas políticas tiveram desempenhos muito próximos. A explicação para o melhor resultado obtido pela política HOP é exatamente a priorização das mensagens reduzindo

a utilização dos recursos da rede. Ou seja, primeiro envia-se as mensagens cujos destinos são os próprios vizinhos. Em seguida, as mensagens cujos destinos são os vizinhos dos vizinhos são selecionadas, e assim sucessivamente.

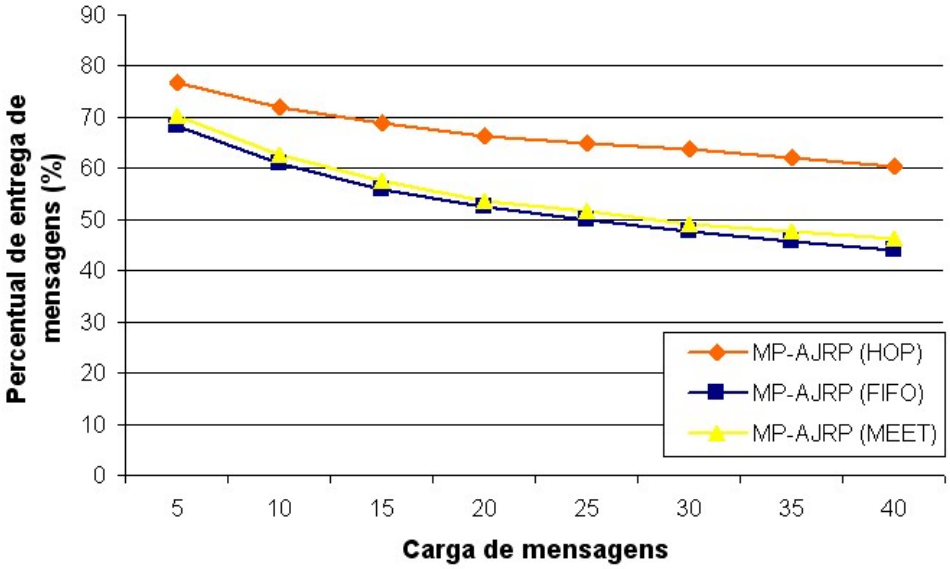

Figura 3. Entrega de mensagens variando o algoritmo de seleção de roteamento

Os resultados obtidos pelo MP-AJRP utilizando a política HOP foram comparados com os obtidos pelos algoritmos RAPID, MaxProp, Spray and Wait, Random e PROPHET, que são algoritmos que foram propostos nos últimos anos com o intuito de otimizar o roteamento de mensagens em DTNs e foram recentemente implementados e avaliados em [Balasubramanian et al. 2007].

A Figura 4 demonstra que o MP-AJRP, mesmo sem realizar nenhum tipo de replicação de mensagens, ou seja, apenas encaminhando cada mensagem a um nó intermediário até alcançar o destino, obteve desempenho superior ao algoritmo Random para todas as cargas de mensagem consideradas. Além disso, observa-se que a diferença na porcentagem de entrega de mensagens pelo MP-AJRP e pelo algoritmo Spray and Wait é relativamente pequena, sendo o MP-AJRP melhor com as cargas de 5 e 40 mensagens, e pior nas demais. Nota-se, contudo, que a curva apresentada pelo algoritmo Spray and Wait apresenta tendência de queda mais acentuada a partir da carga de 35 mensagens, enquanto que a tendência do MP-AJRP é de decréscimo mais suave. Ressalta-se que o algoritmo PROPHET não aparece neste gráfico devido ao seu desempenho muito inferior comparado ao algoritmo Random, conforme já descrito em [Balasubramanian et al. 2007].

Como pode ser observado, o MP-AJRP não conseguiu superar os algoritmos Rapid e MaxProp para as instâncias utilizadas. Um dos motivos para isso é que estes dois algoritmos fazem duplicação de mensagens e usam mecanismos para gerenciamento das réplicas. Por demandar mais recursos da rede, estas estratégias obtiveram melhor desempenho para a topologia de rede avaliada, favorecendo o percentual de entrega das mensagens. No entanto, os resultados apontam que com algum tipo de replicação, menos onerosa do que as empregadas em algoritmos da literatura, o MP-AJRP poderia obter melhores resultados frente aos algoritmos avaliados neste trabalho.

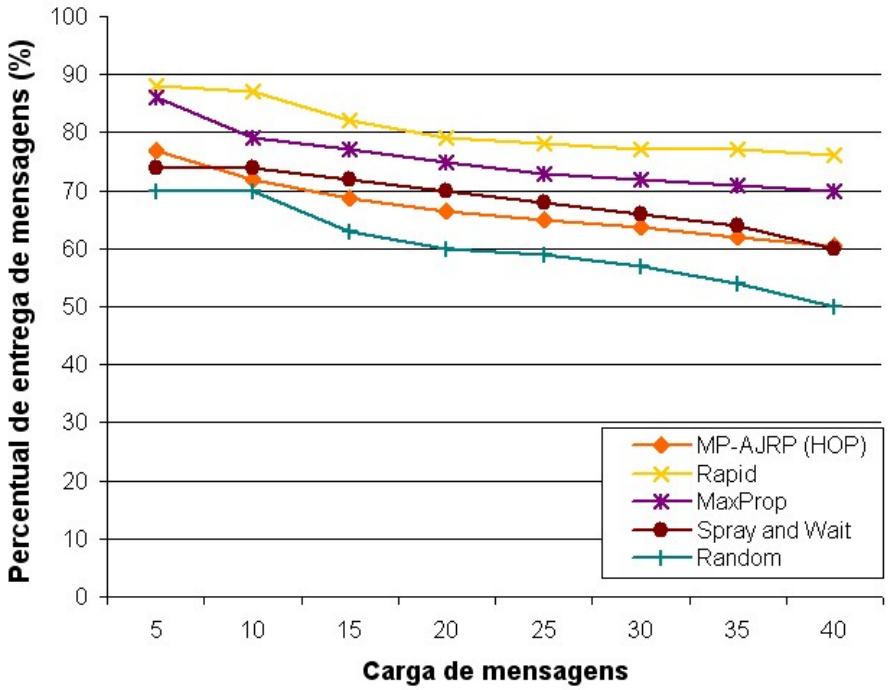

Figura 4. Entrega de mensagens de variados algoritmos

6. Conclusões e trabalhos futuros

Neste trabalho foi proposto o algoritmo distribuído *Multi-Priority Alternative Journey Routing Protocol* (MP-AJRP) que foi desenvolvido com o objetivo de maximizar a entrega de mensagens aos destinos considerando as restrições de largura de banda dos enlaces e as limitações de capacidade de armazenamento dos nós, utilizando um mecanismo de seleção de jornadas alternativas menos congestionadas. Além disso, diferentes políticas de priorização de envio das mensagens do *buffer* foram implementadas e avaliadas. Este algoritmo foi avaliado utilizando *traces* gerados pela rede veicular *DieselNet* e verificou-se que, embora não realize nenhum tipo de replicação de mensagens, seus resultados superaram os de alguns algoritmos de roteamento em DTNs que consideram tal funcionalidade. Vale ressaltar ainda que políticas menos onerosas de replicação de mensagens podem ser incluídas ao MP-AJRP a fim de otimizar os resultados obtidos pelo algoritmo. Tais modificações podem reduzir a diferença de desempenho ou até superar os resultados dos algoritmos que foram aqui comparados com o MP-AJRP.

Como continuação deste trabalho, pretende-se desenvolver algoritmos de encaminhamento que adotam outras métricas de roteamento como, por exemplo, a de entrega das mensagens considerando seus *deadlines*. Além disso, pretende-se desenvolver mecanismos para melhorar o desempenho do algoritmo de encaminhamento usando, para isso, replicação de mensagens de aplicação e incorporação de mais informações relevantes nas mensagens de controle. Outros pontos importantes a serem pesquisados são a avaliação do desempenho do algoritmo proposto neste trabalho em outras topologias e também a comparação deste com outros algoritmos existentes na literatura.

Referências

Argolo, A., Dolejsi, L. D. A., E.Uchoa, e Subramanian, A. (2009). Roteamento em redes tolerantes a atrasos e desconexões com restrições de buffer e largura de banda. In *XLI Simpósio Brasileiro de Pesquisa Operacional*.

Balasubramanian, A., Levine, B., e Venkataramani, A. (2007). Dtn routing as a resource allocation problem. In *ACM SIGCOMM*, pages 373–384.

Bui-Xuan, B., Ferreira, A., e Jarry, A. (2003). Evolving graphs and least cost journeys in dynamic networks. In *Modeling and Optimization in Mobile, Ad-Hoc, and Wireless Networks*, pages 141–150.

Burgess, J., Gallagher, B., Jensen, D., e Levine, B. N. (2006). Maxprop: Routing for vehicle-based disruption-tolerant networks. In *IEEE Infocom*, pages 1–11.

Cerf, V. (2007). Rfc 4838: Dtn architecture. Technical report, IETF.

Chen, C. (2005). Advanced routing protocol for satellite and space networks. Master's thesis, Georgia Institute of Technology.

Jain, S., Fall, K., e Patra, S. (2004). Routing in a delay tolerant network. In *ACM SIGCOMM*, pages 145–158.

Lindgren, A., Doria, A., e Schelén, O. (2004). Probabilistic routing in intermittently connected networks. In *SAPIR Workshop*, pages 239–254.

Merugu, S., Ammar, M., e Zegura, E. (2004). Routing in space and time in networks with predictable mobility. Technical report, Georgia Institute of Technology.

Oliveira, C. T. e Duarte, O. C. M. B. (2007). Uma análise da probabilidade de entrega de mensagens em redes tolerantes a atrasos e desconexões. In *XXV Simpósio Brasileiro de Redes de Computadores*, pages 293–305.

Oliveira, E. C. R. e Albuquerque, C. V. N. (2009). Nectar: A dtn routing protocol based on neighborhood contact history. In *24th ACM Symposium on Applied Computing*, pages 359–365.

Peleg, D. (2000). *Distributed Computing: a locality-sensitive approach (Monographs on Discrete Mathematics and Applications)*. Society for Industrial Mathematics.

Ramanathan, R., Basu, P., e Krishnan, R. (2007). Towards a formalism for routing in challenged networks. In *ACM MobiCom workshop on Challenged Networks*, pages 3–10.

Santos, A., Rocha, G., Drummond, L., e Gorza, M. (2008). Algoritmos distribuídos para roteamento em redes tolerantes a atrasos e desconexões. In *Workshop em Sistemas Computacionais de Alto Desempenho*, pages 35–42.

Spyropoulos, T., Psounis, K., e Raghavendra, C. S. (2005). Spray and wait: An efficient routing scheme for intermittently connected mobile networks. In *ACM WDTN*, pages 252–259.

Vahdat, A. e Becker, D. (2000). Epidemic routing for partially connected ad hoc networks. Technical report, Duke University.