# ST-Drop: Uma Nova Estratégia de Gerenciamento de Buffer em Redes Oportunistas D2D

**Michael D. Silva**[1], **Ivan O. Nunes**[1], **Raquel A. F. Mini**[2], **Antonio A. F. Loureiro**[1]

[1]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais
Belo Horizonte – MG – Brasil

[2]Departamento de Ciência da Computação – Pontifícia Universidade Católica de Minas Gerais
Belo Horizonte – MG – Brasil

{micdoug, ivanolive, loureiro}@dcc.ufmg.br, raquelmini@pucminas.br

***Abstract.*** *In D2D opportunistic networks, nodes act as relays for transmitting messages to other nodes according to an opportunistic routing algorithm. In this scenario, each node uses a buffer with limited capacity to store these messages temporarily until they are propagated to a neighbor node according to an opportunistic routing protocol. However, when multiple messages are forwarded in the network, the number of incoming messages may exceed the nodes' capacity, causing a buffer overflow. Therefore, message dropping policies are very important to this problem, because when a message is dropped, there is a chance that other copies of this message still exist in the network. This work proposes a new buffer management algorithm for opportunistic routing in D2D networks named ST-Drop (Space-Time-Drop). We evaluate our solution in three different types of opportunistic routing algorithms: epidemic-based, probabilistic, and social-aware. We conduct simulations using two different publicly available data sources and consider different network traffic loads. Compared to other message drop policies, ST-Drop obtained the highest message delivery ratio in all considered scenarios and the lowest overhead when applied to the state-of-art social-aware and probabilistic routing algorithms, namely, Bubble Rap and Prophet.*

***Resumo.*** *Em redes oportunistas D2D, os nós funcionam como intermediadores na transmissão de mensagens. Nesse cenário, cada nó da rede usa um buffer de memória limitada para armazenar mensagens temporariamente até serem encaminhadas para um nó vizinho de acordo com um algoritmo de roteamento oportunístico. Quando várias mensagens são encaminhadas na rede, o número de mensagens recebidas por um nó pode exceder sua capacidade de armazenamento, causando um estouro de buffer. Assim, políticas de descarte de mensagens são muito importantes para este problema, já que quando uma mensagem é descartada por um nó, existe a possibilidade de outras cópias dessa mensagem existirem na rede. Este trabalho propõe uma nova política de descarte de mensagens chamada ST-Drop (Space-Time-Drop). A solução foi avaliada em três diferentes tipos de algoritmos de roteamento: abordagem epidêmica, probabilístico e ciente de contexto social. Foram realizadas simulações com dois traces de mobilidade disponíveis publicamente e consideradas diferentes cargas de tráfego na rede. Os resultados mostram que a ST-Drop, quando comparada com outras políticas de descarte de mensagens, obteve os melhores valores de taxa de entrega em todos os cenários com o menor nível de sobrecarga na rede quando utilizado em algoritmos cientes de contexto social e probabilísticos, respectivamente BubbleRap e Prophet.*

## 1. Introdução

Nos últimos anos, houve uma grande evolução dos dispositivos móveis. Tais dispositivos nunca estiveram tão presentes no dia a dia das pessoas, sendo utilizados para as mais diversas tarefas, nas quais destacam-se o acesso a aplicativos de redes sociais, e-mail e Web. Como consequência, o nível de tráfego imposto às redes celulares vem aumentando significativamente. Como a expansão da capacidade e da infraestrutura das redes celulares apresenta alto custo e, as vezes, é até inviabilizada por outros fatores, surge a oportunidade de pesquisar soluções alternativas para o problema. Dentre as soluções propostas, a comunicação direta entre os dispositivos, denominada D2D (*Device-to-Device*) mostra-se como uma solução altamente viável, visto que a estrutura de rede pré-existente é reaproveitada para melhorar o fluxo. Estudos recentes [Aijaz et al. 2013, Andreev et al. 2014, Antonopoulos et al. 2014, Asadi et al. 2014, Bastug et al. 2014, Doppler et al. 2009, Nunes et al. 2016b, Pyattaev et al. 2013, Yang et al. 2013] mostram que a comunicação D2D é uma boa solução para disseminação de dados e soluções para diminuir a carga nas estações rádio-base.

Na comunicação D2D, os dispositivos trocam mensagens quando eles estão suficientemente próximos uns dos outros, ou seja, quando um contato ocorre, que é definido pela mobilidade humana. Assim, as conexões são intermitentes e não se pode assumir uma rota fim-a-fim entre dois dispositivos. Esse tipo de cenário foi discutido primeiramente no contexto de redes DTN (*Delay Tolerant Networks* – Redes Tolerantes a Atrasos), no qual o nó armazena a mensagem recebida em um *buffer* até que ocorra uma oportunidade para transmiti-la. Esse paradigma é conhecido como *store-carry-and-forward* [Fall 2003]. No cenário de comunicação das redes celulares, essa estratégia pode ser utilizada para transmitir conteúdo que não tenha restrições de tempo de entrega, utilizando a comunicação dispositivo a dispositivo (D2D) até que o conteúdo chegue ao seu destino. Dessa forma, as estações base podem delegar a transmissão desse conteúdo via D2D, diminuindo o seu nível de carga. Esse tipo específico de comunicação D2D é frequentemente denominado de Comunicação D2D Multi-Hop, ou Comunicação D2D Oportunista [Li et al. 2014, Nunes et al. 2016b].

Na comunicação D2D oportunista não se pode assumir rotas fim-a-fim e, assim, algoritmos de roteamento usados em redes tradicionais não funcionam nesse cenário. Algoritmos de roteamento oportunístico podem ser classificados em duas categorias [Misra et al. 2016]: a primeira categoria é a dos algoritmos *single-copy* (cópia única), no qual somente uma cópia de cada mensagem é encaminhada na rede e, o maior desafio é decidir quais rotas devem ser exploradas. Alguns exemplos de algoritmos *single-copy* são Direct Delivery [Spyropoulos et al. 2004], First Contact [Jain et al. 2004] e SimBet [Daly and Haahr 2007]. A segunda categoria é a dos algoritmos *multi-copy* (múltiplas cópias), que encaminham simultaneamente na rede múltiplas cópias da mensagem, explorando diferentes rotas D2D. Os algoritmos de roteamento *multi-copy* podem ainda ser classificados como limitados, quando é definido um limite superior de cópias de uma mesma mensagem, e ilimitados caso contrário. Os principais exemplos de algoritmos de roteamento *multi-copy* limitados são *Spray and Wait* [Spyropoulos et al. 2005] e *Spray and Focus* [Spyropoulos et al. 2007]. Alguns exemplos de algoritmos de roteamento *multi-copy* ilimitados são *Epidemic* [Vahdat et al. 2000], *Prophet* [Lindgren et al. 2003] e *Bubble Rap* [Hui et al. 2011].

Em redes D2D oportunistas, os nós precisam cooperar entre si atuando como intermediadores na transmissão de mensagens de outros nós. Para que essa cooperação seja possível, cada nó da rede deve compartilhar seus próprios recursos, alocando memória para armazenar mensagens em um *buffer*, por exemplo. Esse *buffer* é utilizado para armazenar temporariamente as mensagens sendo encaminhadas e sua capacidade é limitada. Por isso, o nível de tráfego na rede, associado às decisões tomadas pelo algoritmo de

roteamento utilizado, podem levar a um estouro de *buffer* do dispositivo. No caso dos algoritmos de roteamento *multi-copy*, políticas de descarte de mensagens representam uma solução viável para esse problema, pois quando uma mensagem é descartada, existe a possibilidade de outras cópias da mensagem existirem na rede [Akestoridis et al. 2014]. No caso de algoritmos de roteamento *single-copy*, esse problema geralmente é amenizado através do uso de outros mecanismos de controle de congestionamento.

O problema de gerenciamento do *buffer* é ainda mais crítico nos algoritmos de roteamento *multi-copy*, pois além de permitirem a existência de múltiplas cópias de uma mensagem, normalmente não existe um dispositivo central com conhecimento global da rede. Sendo assim, os nós não sabem quando uma mensagem é entregue ao destino. Nos algoritmos *multi-copy*, mesmo quando uma mensagem é entregue no destino, suas cópias permanecem na rede até que seu TTL seja excedido ou outro mecanismo da rede as descarte [Bindra and Sangal 2012]. Um mecanismo eficiente de gerenciamento de *buffer* é um requisito fundamental para esse cenário.

Neste trabalho é proposta uma política de gerenciamento de *buffer* para algoritmos de roteamento *multi-copy* em redes oportunistas D2D, denominada *Space-Time Drop* (ST-Drop). Essa política utiliza informações locais para mensurar a cobertura de tempo e espaço de uma mensagem na rede. Resumidamente, a cobertura de espaço de uma mensagem indica o número de dispositivos que carregam essa mensagem e a cobertura de tempo indica o período de tempo que uma mensagem é mantida por um dispositivo, como discutido na Seção 3. A ideia básica é que uma mensagem com maior cobertura de tempo e espaço tem maior probabilidade de já ter sido entregue ao destino, desta forma, ela pode ser descartada primeiro. Experimentos realizados com um *trace* de mobilidade real, contendo 115 dispositivos, e um *trace* sintético, contendo 500 dispositivos, mostram que a política ST-Drop, associada a algoritmos de roteamento oportunista, permite alcançar uma boa taxa de entrega de mensagens com baixo nível de sobrecarga na rede.

Este trabalho está organizado da seguinte forma: a seção 2 discute trabalhos de pesquisa relacionados. A seção 3 apresenta a política de descarte de mensagens ST-Drop, com uma descrição detalhada de seu algoritmo. A seção 4 descreve a metodologia experimental utilizada para validar a ST-Drop, incluindo a descrição dos algoritmos de roteamento oportunistas utilizados, os *traces* de mobilidade e todos os parâmetros de simulação. A seção 5 apresenta os resultados de simulação e os discute. Por fim, a seção 6 apresenta uma discussão final onde são apontadas direções futuras do trabalho.

## 2. Trabalhos Relacionados

Na comunicação D2D, os dispositivos devem alocar uma porção de memória para ser utilizada como *buffer* de armazenamento temporário das mensagens recebidas por eles. A ideia básica, para evitar situações de sobrecarga no *buffer*, mantendo a eficiência do algoritmo de roteamento oportunista utilizado, consiste na utilização de mecanismos de controle de congestionamento. Em [Silva et al. 2015], é feita uma classificação dos mecanismos de controle de congestionamento, fazendo um estudo extenso sobre políticas de descarte de mensagens utilizadas em redes oportunistas. Nesta seção, são discutidas as principais políticas de descarte de mensagens propostas na literatura.

Em [Lindgren and Phanse 2006], é proposta a política de descarte *Evict Most Forwarded* (MOFO), que se baseia na ideia de que uma mensagem que foi encaminhada um maior número de vezes tem maior chance de já ter sido entregue, mesmo se for descartada localmente. Também é proposta a política *Evict Shortest Life Time First* (SHLI), que se baseia na ideia de que faz mais sentido alocar recursos para mensagens que tenham um maior valor de TTL, pois, intuitivamente, essas mensagens têm maior probabilidade de serem entregues. Já a política *Drop Largest* [Rashid and Ayub 2010] seleciona primeiro

as mensagens com maior tamanho de serem descartadas, pois, ao fazer isso, mais espaço é liberado no *buffer* com menor número de descartes. Outra política básica é a FIFO (*First In First Out*), no qual, como o nome sugere, mensagens que foram recebidas a mais tempo são descartadas primeiro.

Em [Rashid et al. 2013] é proposta a política *Message Drop Control Source Relay* (MDC-SR), no qual o nó deixa de receber mensagens não destinadas a ele quando seu *buffer* alcança um nível de ocupação definido por um parâmetro. Com esta abordagem, a política diminui o descarte de mensagens efetuadas na rede. Experimentos realizados mostram que a política MDC-SR otimiza o desempenho dos algoritmos de roteamento *Epidemic*, *Prophet* e *First Contact* em relação à taxa de entrega e sobrecarga na rede.

Em [Krifa et al. 2008] é proposta a política *Global Knowledge Based Drop* (GBD-Drop), no qual primeiramente é assumido um conhecimento global da rede para decidir quais mensagens devem ser descartadas, levando a um desempenho ótimo. Uma versão distribuída do algoritmo é proposta, no qual são utilizadas técnicas de aprendizagem estatística para aproximar o conhecimento global da rede. Experimentos realizados mostram que tanto a versão centralizada quanto a distribuída alcançam melhores resultados quando comparadas com políticas básicas. A política GDB-Drop pode ainda ser parametrizada para alcançar melhor tempo médio de entrega ou taxa de entrega.

Em [Rashid et al. 2011] é proposta a política de descarte *E-Drop*, no qual mensagens com tamanho maior ou igual à mensagem sendo recebida pelo dispositivo são descartadas primeiro, quando é necessário liberar espaço no *buffer*. No caso em que a mensagem sendo recebida tem tamanho maior que todas as mensagens no *buffer* do dispositivo, o descarte é avaliado seguindo a lógica da política FIFO. Essa política minimiza o número de descarte de mensagens, pois ela vai normalmente remover somente uma mensagem do *buffer* para receber uma nova mensagem. Simulações realizadas com modelos de mobilidade mostram que a política E-Drop tem melhor desempenho que a política MOFO.

Em [Li et al. 2009] é proposta a política *N-Drop*, no qual as mensagens que foram transmitidas um número de vezes maior que uma constante pré-determinada são descartadas primeiro. Simulações realizadas mostram que a política tem desempenho melhor que a política *FIFO* quando utilizada em conjunto com o algoritmo de roteamento *Epidemic*. Já em [Ayub and Rashid 2010] é proposta a política de descarte *T-Drop*, no qual mensagens com tamanho dentro de um intervalo $T$ predefinido são descartadas primeiro. Simulações mostram que a *T-Drop* tem desempenho melhor que a política FIFO quando utilizada em conjunto com os algoritmos de roteamento *Epidemic* e *Prophet*.

Em [Naves et al. 2012] são propostas duas novas políticas de gerenciamento de *buffer*. A primeira é a *Least Recently Forwarded* (LRF), no qual mensagens que foram encaminhadas mais recentemente são descartadas primeiro. A segunda é a *Less Probable Sprayed* (LPS), na qual as mensagens que têm menor probabilidade de ser entregues são descartadas primeiro. Experimentos realizados com *traces* de mobilidade reais, aplicando os algoritmos de roteamento *Epidemic* e *Prophet*, mostram que os algoritmos têm desempenho superior às políticas FIFO e MOFO em relação a taxa de entrega e sobrecarga na rede.

As políticas discutidas nesta seção, em sua grande maioria, se baseiam somente em uma única métrica local, como por exemplo, tempo no qual a mensagem é carregada, número de transmissões, TTL e tamanho da mensagem. A utilização dessas métricas consiste em uma boa estratégia, pois elas são independentes do algoritmo de roteamento utilizado. Porém, a sua utilização de maneira isolada pode não ser suficiente para decidir de forma eficiente quando descartar uma mensagem ou não. A política ST-Drop, proposta

neste trabalho, combina algumas dessas métricas para criar um novo mecanismo de decisão de descarte. Na próxima seção o funcionamento da ST-Drop é descrito com mais detalhes.

## 3. O Algoritmo ST-Drop

A formulação da política ST-Drop parte do princípio que uma mensagem com uma maior cobertura de tempo e espaço na rede tem maior probabilidade de já ter sido entregue ao destino, sendo assim, ela pode ser descartada primeiro. Baseado nessa ideia, foi definido um coeficiente de espaço $S_c$ que mede a cobertura de espaço de uma mensagem na rede, e o coeficiente de tempo $T_c$ que mede a cobertura de tempo da mensagem na rede. A partir disso, é possível calcular a cobertura de tempo-espaço $ST_c$ usando a seguinte função:

$$ST_c = S_c \cdot T_c. \tag{1}$$

Essa primeira formulação não define de que forma os coeficientes $S_c$ e $T_c$ são calculados. Algoritmos de gerenciamento de *buffer* em redes oportunistas D2D devem preferencialmente ser implementados de forma distribuída, ou seja, devem se restringir a usar somente informações locais. Além disso, a política ST-Drop foi projetada para funcionar de forma independente do algoritmo de roteamento utilizado. Assim, foi definido que a política ST-Drop utilizaria a combinação de métricas locais, utilizadas por outros algoritmos discutidos anteriormente, para calcular os coeficientes $S_c$ e $T_c$ propostos.

O primeiro problema consiste em calcular a cobertura de espaço. Em redes celulares, os contatos são gerados a partir da mobilidade humana. Intuitivamente, pode-se dizer que uma mensagem carregada por uma única pessoa, terá menos oportunidades de ser transmitida, quando comparada com uma mensagem carregada por duas ou mais pessoas, pois ela estará restrita à mobilidade de uma única pessoa. Portanto, pode-se dizer que o número de dispositivos que carregam uma mensagem é um indicador sensato da cobertura de espaço de uma mensagem, pois a cobertura de espaço de uma mensagem é um resultado direto da combinação da mobilidade dos dispositivos que a carregam. Desta forma, a política ST-Drop utiliza a mesma métrica utilizada pela política MOFO para calcular a cobertura de espaço, ou seja, o número de vezes que uma mensagem foi transmitida por um nó. O número de transmissões de uma mensagem é um indicador local do número de dispositivos que carregam uma mensagem.

O segundo problema consiste em calcular a cobertura de tempo. A política FIFO explora o tempo em que as mensagens são carregadas por um nó para decidir qual mensagem descartar primeiro. Ao usar somente essa métrica, uma mensagem criada recentemente pode ser descartada ao invés de uma mensagem com TTL quase expirando. Já a política SHLI utiliza o TTL das mensagens para decidir qual deve ser descartada primeiro. Ao usar somente essa métrica, uma mensagem com valor de TTL relativamente grande pode ser mantida indefinidamente no *buffer* de um nó que tem pouca possibilidade de entregá-la ao destino. Outro problema relacionado com essa abordagem é que em cenários reais as mensagens criadas podem ter valores de TTL iniciais diferentes, o que pode levar a uma mensagem com TTL inicial relativamente baixo a ser penalizada. Com o objetivo de amenizar esses problemas, a política ST-Drop combina as ideias das políticas FIFO e SHLI, normalizando o tempo que a mensagem está sendo carregada $CT$ (*Carrying Time*) pelo seu TTL. Assim, o valor de $T_c$ é definido pela função:

$$T_c = \frac{CT}{TTL}. \tag{2}$$

Com esta formulação é possível calcular a cobertura de tempo e espaço de uma mensagem usando somente informações locais. O Algoritmo 1 apresenta a política ST-Drop, que só é acionada quando não há espaço suficiente no *buffer* para receber uma

nova mensagem. O algoritmo ordena as mensagens em ordem decrescente pelos seus valores de $ST_c$, criando assim uma fila de descarte. As mensagens com $ST_c$ igual a 0 são retiradas desta fila, o que implica em estas mensagens nunca serem descartadas. O algoritmo considera que descartar uma mensagem com cobertura de tempo e espaço igual a 0 não é justo, pois a mensagem não foi retransmitida para nenhum outro nó neste caso. Após essas etapas, a primeira mensagem da fila é descartada até que seja liberado espaço suficiente para receber a nova mensagem, ou a fila se torne vazia. Se após os descartes foi possível liberar espaço suficiente, o algoritmo retorna uma confirmação para receber a nova mensagem. Caso contrário, o dispositivo deve rejeitar a mensagem que acabou de ser recebida.

**Algorithm 1** Algoritmo ST-Drop

**Input:** msgRecebida
**Input:** buffer
**Output:** receber

**if** msgRecebida.tamanho $>$ buffer.capacidade **then**
  **return** false
**end if**
*Ordena as mensagens em ordem decrescente por $ST_c$ :*
fila_descarte $\leftarrow$ ordenar(buffer)
*Remove as mensagens com $ST_c = 0$*
fila_descarte.filtrar()
*Descarta mensagens até que seja liberado espaço suficiente*
tamanhoRecebido $\leftarrow$ msgRecebida.tamanho
disponivel $\leftarrow$ buffer.espacoDisponivel
**while** disponivel $<$ tamanhoRecebido e não fila_descarte.vazia **do**
  msg $\leftarrow$ fila_descarte.removerPrimeira()
  buffer.descartar(msg)
  disponivel $\leftarrow$ ++msg.tamanho
**end while**
**if** disponivel $<$ tamanhoRecebido **then**
  **return** false
**else**
  **return** true
**end if**

## 4. Metodologia de Simulação

Esta seção descreve as simulações realizadas para avaliar a política ST-Drop. Primeiramente, são descritas as métricas de avaliação de comunicação oportunista D2D que foram utilizadas. Depois são apresentados os algoritmos de roteamento aplicados nas simulações e são descritos os dois *traces* de mobilidade utilizados para simular a movimentação dos nós e, consequentemente, a geração de contatos. Por fim, são expostos e discutidos os parâmetros e configurações utilizados.

### 4.1. Métricas

Para comparar o desempenho da política de descarte ST-Drop com as outras políticas, foram utilizadas as seguintes métricas:

- **Taxa de Entrega**: avalia o percentual de mensagens entregues ao destino ao longo do tempo;

- **Número de Transmissões**: avalia o número de retransmissões por mensagem criada na rede, ou seja, o número de transmissões D2D que cada algoritmo executou ao longo do tempo, normalizado pelo número de mensagens criadas.

Políticas de descarte de mensagens para redes D2D devem idealmente alcançar um bom custo benefício considerando essas métricas, ou seja, devem alcançar um alto nível de taxa de entrega com baixo número de transmissões. As mensagens entregues ao destino com sucesso representam aquelas mensagens que as estações base não precisarão transmitir, economizando, assim, recursos computacionais do dispositivo (e.g., processamento e energia) e do canal de comunicação. Um grande número de transmissões de mensagens pode impactar negativamente a experiência dos usuários, por exemplo, aumentando o gasto de energia dos dispositivos.

### 4.2. Algoritmos de Roteamento Oportunista

Para avaliar o desempenho da política ST-Drop, foram utilizados algoritmos de roteamento que se baseiam em três abordagens distintas, permitindo, assim, avaliar o comportamento da solução proposta quando integrada a diferentes algoritmos de roteamento. O primeiro algoritmo avaliado foi o *Epidemic* [Vahdat et al. 2000], também conhecido como *Flooding*. Este algoritmo é bastante simples, visto que não se baseia em funções de utilidade ou métricas similares. Nesse algoritmo os nós da rede a cada encontro trocam todas mensagens não comuns entre eles. Quando se considera um cenário onde os dispositivos da rede têm *buffers* com capacidade ilimitada, o algoritmo *Epidemic* alcança o valor máximo possível de taxa de entrega, porém também alcança um número muito alto de transmissões. Porém, quando se considera um cenário mais realista, no qual os dispositivos têm *buffers* com capacidade de armazenamento limitada, esse algoritmo por si só pode não alcançar o melhor percentual de taxa de entrega possível.

O segundo algoritmo de roteamento considerado nas simulações é o *Prophet* [Lindgren et al. 2003]. Esse algoritmo utiliza o histórico de contatos dos nós para calcular a probabilidade de um determinado elemento entregar uma mensagem. Quando dois nós entram em contato, eles trocam somente as mensagens as quais o outro nó tem maior probabilidade de entrega. A ideia básica do *Prophet* consiste em considerar que um par de nós que se encontrou recentemente tem grande probabilidade de se encontrar novamente.

O terceiro algoritmo de roteamento utilizado é o Bubble Rap [Hui et al. 2011]. Este algoritmo se baseia nos conceitos sociais de comunidade e popularidade. Nesse algoritmo, cada nó é associado a pelo menos uma comunidade social, que, por sua vez, é definida como um conjunto de nós densamente conectados (um grupo de nós que tem mais contatos entre si do que com outros). A popularidade de um nó é calculada a partir do número de contatos distintos que um determinado nó teve ao longo do tempo. A ideia básica do Bubble Rap é encaminhar as mensagens para nós que tenham um maior nível de popularidade global até que a mensagem seja alcançada por um membro da comunidade destino (uma das comunidades associadas ao nó destino). A partir deste momento, a mensagem é então encaminhada somente entre membros da comunidade destino com base no nível de popularidade local (considerando somente os membros da comunidade), até que a mensagem chegue ao destino. As comunidades e valores de centralidade, necessários para utilizar o Bubble Rap, foram obtidos a partir dos algoritmos *Distributed K-Clique Community Detection* [Hui et al. 2007] e *C-Window* [Hui et al. 2011], da forma que foram definidos na formalização original do algoritmo de roteamento.

Como foi discutido, foram utilizados algoritmos de roteamento bem distintos, baseados em estratégias de epidemia (inundação), funções de probabilidade e ciente de contexto social. Assim, pôde-se avaliar o impacto de diferentes tipos de roteamento no desempenho da política ST-Drop.

**Tabela 1. Níveis de Tráfego por Cenário**

| Cenário | | Mensagens/Dia | Total |
|---|---|---|---|
| NCCU | 50 | 58 | 522 |
| | 100 | 115 | 1035 |
| SWIM | 50 | 250 | 1250 |
| | 100 | 500 | 2500 |

### 4.3. *Traces* de Mobilidade

Foram utilizados dois *traces* de mobilidade para simular a movimentação dos nós da rede. O primeiro é o *trace* NCCU [Tsai and Chan 2015], que utiliza dados reais coletados a partir de um experimento realizado com 115 estudantes em um campus universitário durante 15 dias. O segundo é um *trace* sintético gerado a partir do modelo de mobilidade *Small World in Motion* (SWIM) [Kosta et al. 2014]. Esse *trace* se baseia em um modelo de mobilidade tido com estado da arte, que captura a existência de comunidades sociais que influenciam a mobilidade humana. O *trace* utilizado simula a movimentação de 500 pessoas por um período de 11 dias. A utilização dos dois *traces* permite avaliar a ST-Drop sob duas escalas de rede diferentes.

### 4.4. Execução

Para simular a utilização dos algoritmos de roteamento com diferentes políticas de descarte de mensagens, foi utilizado o simulador de redes oportunistas *The One* [Keränen et al. 2009], um simulador de eventos discreto que não oferece suporte para utilização de algoritmos de gerenciamento de *buffer* personalizados, nem para o algoritmo de roteamento Bubble Rap. Assim, foi necessário implementar esses algoritmos[1] para realizar as simulações.

Para avaliar a política ST-Drop, foi necessário primeiramente definir a capacidade do *buffer* dos nós da rede e um modelo de tráfego de mensagens. Como discutido em [Grasic and Lindgren 2012], não existe um padrão bem definido em relação à definição de modelos de tráfego e mensagens para testar soluções neste cenário. Portanto, foram escolhidos valores que permitissem a correta avaliação dos resultados. A capacidade do *buffer* dos nós foi definida como 1 GB. As mensagens têm sete dias de TTL inicial e são geradas de forma aleatória com distribuição uniforme ao longo do tempo de simulação. O tamanho das mensagens é definido dentro do intervalo [50 MB, 100 MB] por uma distribuição de probabilidade uniforme. A origem e o destino de cada mensagem também são selecionados a partir de uma distribuição de probabilidade uniforme sobre os nós da rede.

Foram considerados três níveis de tráfego de mensagens relativos ao número de dispositivos de cada cenário. O primeiro gera a cada dia um número de mensagens igual a 50% do número de nós, o segundo gera 75% e o último gera 100%. Os resultados do cenário com 75% de nível de tráfego não são expostos, devido à limitação de espaço, porém isso não impacta na análise apresentada. A Tabela 1 apresenta o número de mensagens geradas nos níveis de tráfego de 50% e 100%.

Cada combinação de cenário e nível de tráfego foi simulada 10 vezes com diferentes sementes de geração de origem, destino e tamanho de mensagens. A taxa de entrega e o número de transmissões de cada cenário foram calculados e são apresentados utilizando um intervalo de confiança de 95%

[1] O código fonte das alterações está disponível em `https://github.com/micdoug/the-one/tree/58e207ac44d5012fac5d103da781d30266164216`

## 5. Resultados

Os resultados de simulação dos cenários NCCU100 e NCCU50 estão representados nas Figuras 1 e 2, respectivamente. Os resultados dos cenários SWIM100 e SWIM50 estão representados nas Figuras 3 e 4, respectivamente. Os valores referentes à taxa de entrega são apresentados em função do tempo de vida das mensagens (até 7 dias nos experimentos realizados) e não do tempo total de simulação. Desta forma é possível ter uma visão mais clara do impacto das políticas na taxa de entrega de cada mensagem. Já os valores referentes ao número de transmissões efetuadas são apresentados considerando todo o tempo de simulação, já que a transmissão de cada mensagem impacta o *overhead* da rede como um todo.

Ao analisar os resultados percebe-se que a evolução da taxa de entrega e do número de transmissões exibe o mesmo comportamento mesmo em diferentes níveis de tráfego. Isso mostra que as políticas mantém um comportamento estável sob diferentes níveis de carga. Pode-se notar também que o trace SWIM exibe um ambiente mais desafiador em relação à entrega de mensagens quando comparado com o trace NCCU. Isso ocorre devido à características referentes ao próprio padrão de contatos exibido pelo trace. Desta forma, os valores de taxa de entrega para todas as políticas no trace SWIM são bem próximos, e, tecnicamente equivalentes quando considera-se o intervalo de confiança.

Os resultados de taxa de entrega no trace NCCU mostram que a política SHLI tem vantagem expressiva nos primeiros três dias do tempo de vida das mensagems, considerando-se os três algoritmos de roteamento utilizados. Isso se deve ao fato desta política eliminar mensagens com valores de TTL altos. Desta forma, a maioria das mensagens entregues se concentram neste espaço de valores de TTL baixos. Isso é evidenciado pelo fato da política SHLI não apresentar um aumento significativo em sua taxa de entrega após o período inicial de três dias. As políticas E-Drop, FIFO e MOFO têm desempenho abaixo das políticas SHLI e ST-Drop para praticamente qualquer valor de TTL considerado, nos três algoritmos de roteamento considerados. A política ST-Drop exibe um crescimento progressivo de taxa de entrega, alcançando o melhor desempenho ao final da simulação para os três algoritmos de roteamento testados. As outras políticas de descarte, porém, alcançam seu valor máximo de taxa de entrega rapidamente, exibindo um comportamento de estabilização rápida.

O crescimento progressivo da taxa de entrega ao utilizar a ST-Drop é consequência do fato da política não descartar mensagens com coeficiente de tempo-espaço igual a 0. Na formulação do algoritmo, definiu-se que o coeficiente de espaço é medido pelo número de vezes que uma mensagem foi retransmitida. Sendo assim, quando uma mensagem ainda não foi retransmitida por um nó, ela não será descartada. Desta forma, garante-se que pelo menos uma cópia da mensagem será mantida na rede, e, as cópias mantidas tendem a se concentrar em nós situados no final do caminho traçado pela mensagem. Tem-se assim uma aproximação do mecanismo de encaminhamento tradicional em redes *"forward-and-drop"*, no qual, ao encaminhar uma mensagem, o nó a exclui de sua memória. Esse efeito torna-se mais forte à medida que o nível de sobrecarga na rede aumenta. Como as mensagens são encaminhadas por caminhos cada vez maiores e não são descartadas completamente da rede, as mensagens continuam sendo entregues mesmo quando considera-se valores de TTL mais altos. As outras políticas de descarte tendem a realizar um descarte precoce das mensagens quando a rede exibe níveis de sobrecarga altos, o que explica a estabilização rápida.

Os resultados de número de transmissões mostram que a política ST-Drop tem o melhor desempenho quando utilizada com os algoritmos de roteamento BubbleRap e Prophet em ambos traces. Em contrapartida, quando utilizada com o algoritmo Epidemic ela exibe o maior número de transmissões entre as políticas avaliadas. Isso também é ex-

plicado pelo efeito *"forward-and-drop"*. O algoritmo Epidemic não define uma restrição em relação aos caminhos que uma mensagem pode percorrer na rede. Como a ST-Drop exerce o efeito de empurrar a mensagem para outros nós, as mensagens tendem a percorrer caminhos cada vez maiores, mesmo quando já foram entregues. Este efeito pode ser amenizado com a utilização de um mecanismo de eliminação de mensagens já entregues, como por exemplo o VACCINE [Haas and Small 2006]. Os algoritmos de roteamento BubbleRap e Prophet criam naturalmente uma fronteira no caminho das mensagens definida pelos seus critérios de encaminhamento, desta forma a política ST-Drop funciona muito bem nestes dois cenários, apresentando valores de números de transmissões bem mais baixos, quando comparada com as outras políticas.

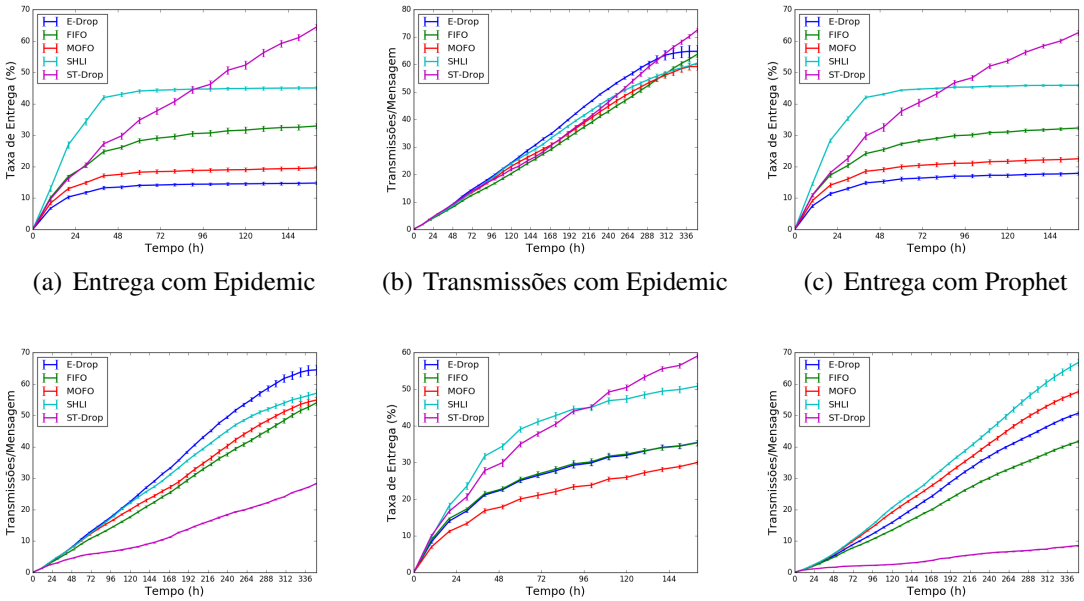
No geral, a política ST-Drop obteve os melhores resultados de taxa de entrega para todos os três algoritmos de roteamento. Em relação ao número de transmissões, a política ST-Drop obteve os melhores resultados em todos os cenários com os algoritmos de roteamento Prophet e Bubble Rap. A política SHLI obteve melhores resultados para valores de tempo de entrega menores, pois nessa política, as mensagens com maior tempo de entrega são descartadas primeiro, levando a um comportamento esperado. Porém quando se considera todo o intervalo de tempo de entrega avaliado, a política ST-Drop obteve melhores resultados que a SHLI.

Vale a pena destacar que, no algoritmo Prophet, a política ST-Drop obteve um número de transmissões muito menor quando comparada às outras políticas. Em relação ao algoritmo de roteamento Bubble Rap, os resultados foram ainda melhores, pois a política ST-Drop alcançou um número de transmissões até três vezes menor que a segunda melhor política. Essa última observação é extremamente importante, pois, atualmente os algoritmos de roteamento oportunista baseados em contexto social são considerados como o estado da arte. A afirmação pode ser confirmada quando comparamos os valores de taxa de entrega e número de transmissões com os três algoritmos de roteamento: Epidemic, Prophet e Bubble Rap. Com ambos *traces* de mobilidade, o Bubble Rap (estratégia baseada no contexto social) obteve os melhores resultados de taxa de entrega com menor número de transmissões, especialmente quando utilizado em conjunto com a ST-Drop.

## 6. Conclusão

Este trabalho propôs uma nova política de gerenciamento de *buffer* em redes oportunistas D2D chamada Space-Time-Drop (ST-Drop). O princípio dessa política é que uma mensagem com maior cobertura de tempo e de espaço na rede tem maior probabilidade de já ter sido entregue, portanto pode ser descartada primeiro. A cobertura de tempo e espaço é calculada a partir da combinação de ideias de políticas mais simples, usando somente informações locais. Assim, a solução pode ser facilmente implantada em ambientes distribuídos. A política ST-Drop foi avaliada a partir de simulação com os algoritmos de roteamento Epidemic, Prophet e Bubble Rap, no qual foram utilizados três níveis diferentes de tráfego. Os resultados mostraram que a política ST-Drop obteve os melhores valores de taxa de entrega com os três algoritmos de roteamento e número de transmissões extremamente mais baixo com os algoritmos de roteamento Prophet e Bubble Rap. Considerando todas as combinações de algoritmos de roteamento oportunista e políticas de gerenciamento de buffer avaliadas, a combinação do Bubble Rap com a política ST-Drop obteve, em todos os experimentos, o melhor custo benefício, ou seja, a melhor razão entre taxa de entrega e número de transmissões. Esse resultado mostra que a política ST-Drop contribui significativamente para melhorar a relação custo-benefício de redes oportunistas D2D.

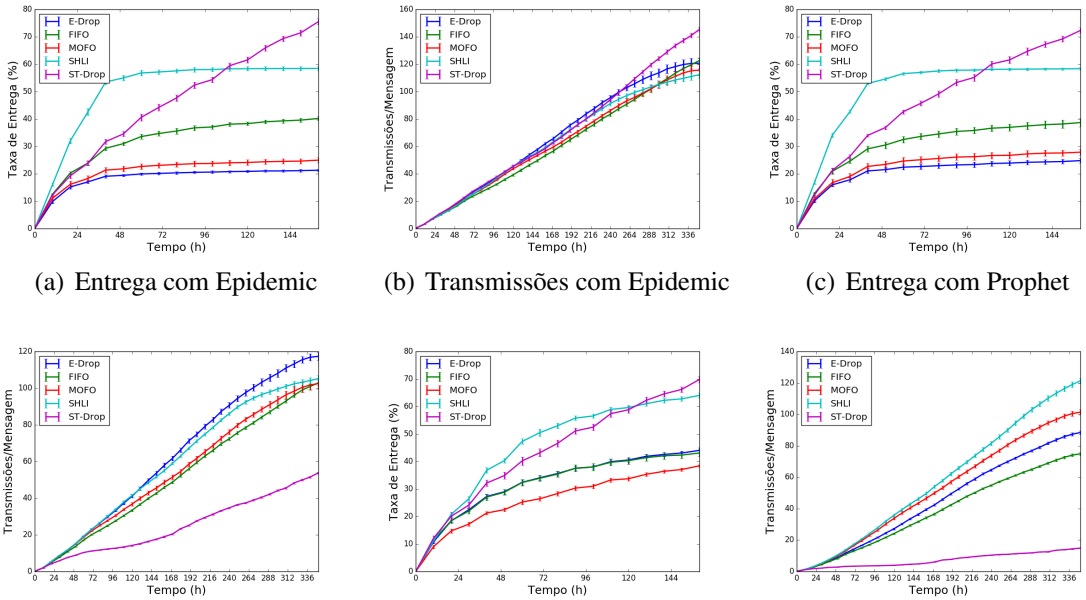
Como trabalho futuro, a política ST-Drop deve ser aplicada a outros algoritmos de roteamento oportunista D2D. Como a política ST-Drop obteve bons resultados com um algoritmo baseado em contexto social, seria interessante aplicá-la a outros algoritmos de




(a) Entrega com Epidemic (b) Transmissões com Epidemic (c) Entrega com Prophet

(d) Transmissões com Prophet (e) Entrega com BubbleRap (f) Transmissões com BubbleRap

**Figura 1. Comparação de políticas de descarte de mensagens no trace NCCU com tráfego de 100% (NCCU100)**

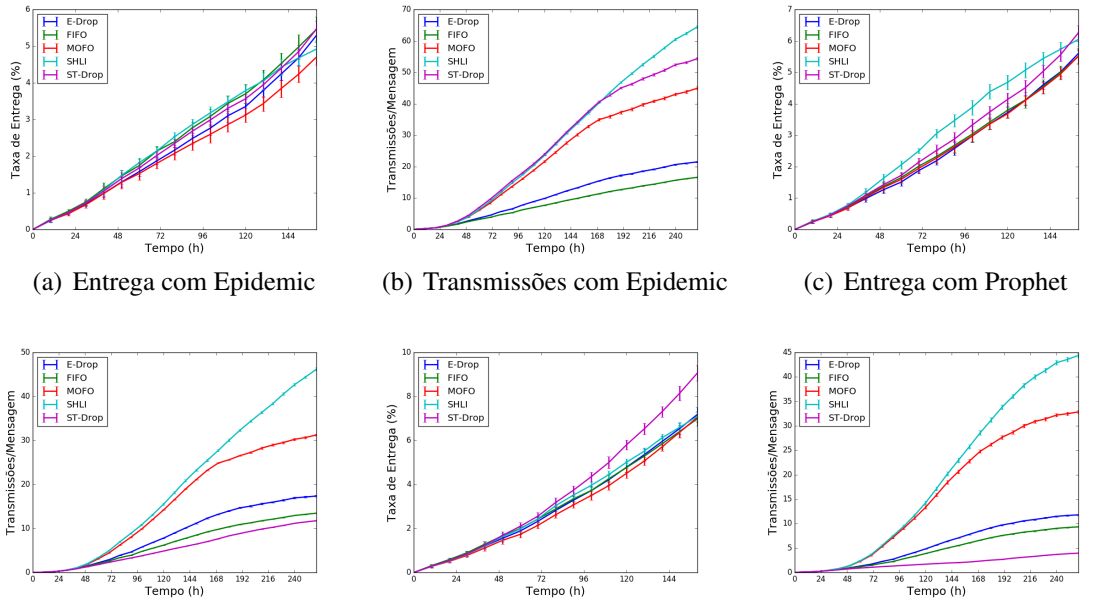



(a) Entrega com Epidemic (b) Transmissões com Epidemic (c) Entrega com Prophet

(d) Transmissões com Prophet (e) Entrega com BubbleRap (f) Transmissões com BubbleRap

**Figura 2. Comparação de políticas de descarte de mensagens no trace NCCU com tráfego de 50% (NCCU50)**

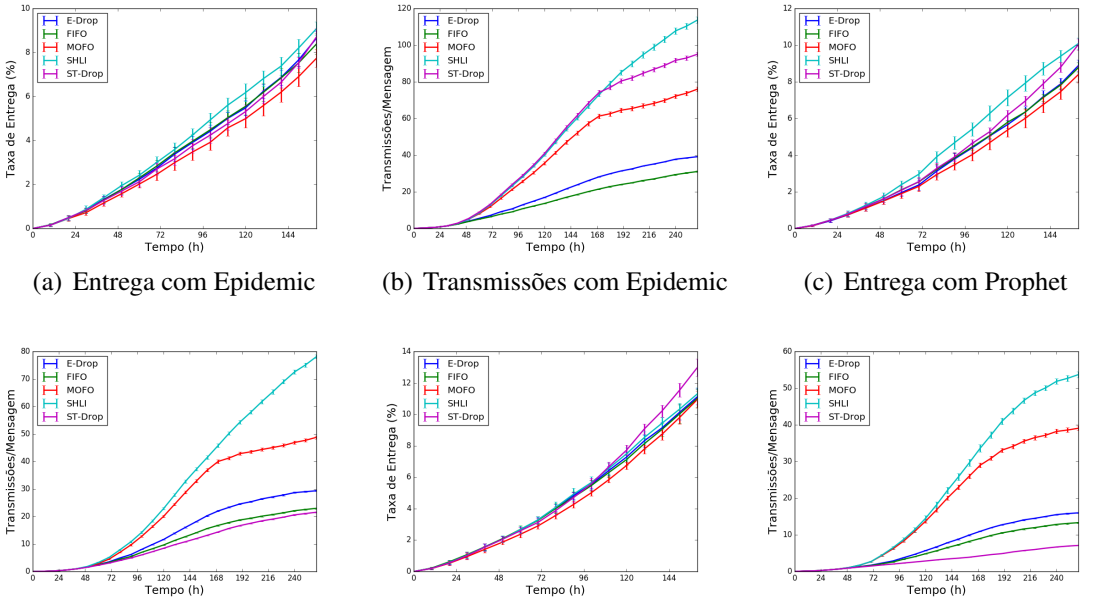
roteamento recentes que utilizam abordagem semelhante como, por exemplo, o Groups-Net [Nunes et al. 2016a, Nunes et al. 2016c]. É importante destacar que também existe a possibilidade de explorar outras métricas para calcular a cobertura de tempo e espaço das mensagens. Informações obtidas a partir da análise de contexto do dispositivo, como nível de bateria e informação espacial (GPS), podem ser utilizadas para melhorar o desempenho




(a) Entrega com Epidemic (b) Transmissões com Epidemic (c) Entrega com Prophet

(d) Transmissões com Prophet (e) Entrega com BubbleRap (f) Transmissões com BubbleRap

**Figura 3. Comparação de políticas de descarte de mensagens no trace SWIM com tráfego de 100% (SWIM100)**




(a) Entrega com Epidemic (b) Transmissões com Epidemic (c) Entrega com Prophet

(d) Transmissões com Prophet (e) Entrega com BubbleRap (f) Transmissões com BubbleRap

**Figura 4. Comparação de políticas de descarte de mensagens no trace SWIM com tráfego de 50% (SWIM50)**

da política ST-Drop.

## Referências

Aijaz, A., Aghvami, H., and Amani, M. (2013). A survey on mobile data offloading: technical and business perspectives. *IEEE Wireless Communications*, 20(2):104–112.

Akestoridis, D.-G., Papanikos, N., and Papapetrou, E. (2014). Exploiting social preferences for congestion control in opportunistic networks. In *2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 413–418. IEEE.

Andreev, S., Pyattaev, A., Johnsson, K., Galinina, O., and Koucheryavy, Y. (2014). Cellular traffic offloading onto network-assisted device-to-device connections. *IEEE Communications Magazine*, 52(4):20–31.

Antonopoulos, A., Kartsakli, E., and Verikoukis, C. (2014). Game theoretic d2d content dissemination in 4g cellular networks. *IEEE Communications Magazine*, 52(6):125–132.

Asadi, A., Wang, Q., and Mancuso, V. (2014). A survey on device-to-device communication in cellular networks. *IEEE Communications Surveys & Tutorials*, 16(4):1801–1819.

Ayub, Q. and Rashid, S. (2010). T-drop: An optimal buffer management policy to improve qos in dtn routing protocols. *Journal of Computing*, 2(10):46–50.

Bastug, E., Bennis, M., and Debbah, M. (2014). Living on the edge: The role of proactive caching in 5g wireless networks. *IEEE Communications Magazine*, 52(8):82–89.

Bindra, H. S. and Sangal, A. (2012). Need of removing delivered message replica from delay tolerant network-a problem definition. *International Journal of Computer Network and Information Security*, 4(12):59.

Daly, E. M. and Haahr, M. (2007). Social network analysis for routing in disconnected delay-tolerant manets. In *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pages 32–40. ACM.

Doppler, K., Rinne, M., Wijting, C., Ribeiro, C. B., and Hugl, K. (2009). Device-to-device communication as an underlay to lte-advanced networks. *IEEE Communications Magazine*, 47(12):42–49.

Fall, K. (2003). A delay-tolerant network architecture for challenged internets. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 27–34. ACM.

Grasic, S. and Lindgren, A. (2012). An analysis of evaluation practices for dtn routing protocols. In *Proceedings of the seventh ACM international workshop on Challenged networks*, pages 57–64. ACM.

Haas, Z. J. and Small, T. (2006). A new networking model for biological applications of ad hoc sensor networks. *IEEE/ACM Transactions on Networking (TON)*, 14(1):27–40.

Hui, P., Crowcroft, J., and Yoneki, E. (2011). Bubble rap: Social-based forwarding in delay-tolerant networks. *IEEE Transactions on Mobile Computing*, 10(11):1576–1589.

Hui, P., Yoneki, E., Chan, S. Y., and Crowcroft, J. (2007). Distributed community detection in delay tolerant networks. In *Proceedings of 2nd ACM/IEEE international workshop on Mobility in the evolving internet architecture*, page 7. ACM.

Jain, S., Fall, K., and Patra, R. (2004). *Routing in a delay tolerant network*, volume 34. ACM.

Keränen, A., Ott, J., and Kärkkäinen, T. (2009). The one simulator for dtn protocol evaluation. In *Proceedings of the 2nd international conference on simulation tools and techniques*, page 55. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

Kosta, S., Mei, A., and Stefa, J. (2014). Large-scale synthetic social mobile networks with swim. *IEEE Transactions on Mobile Computing*, 13(1):116–129.

Krifa, A., Barakat, C., and Spyropoulos, T. (2008). Optimal buffer management policies for delay tolerant networks. In *2008 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 260–268. IEEE.

Li, Y., Wu, T., Hui, P., Jin, D., and Chen, S. (2014). Social-aware d2d communications: qualitative insights and quantitative analysis. *IEEE Communications Magazine*, 52(6):150–158.

Li, Y., Zhao, L., Liu, Z., and Liu, Q. (2009). N-drop: congestion control strategy under epidemic routing in dtn. In *Proceedings of the 2009 international conference on wireless communications and mobile computing: connecting the world wirelessly*, pages 457–460. ACM.

Lindgren, A., Doria, A., and Schelén, O. (2003). Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE mobile computing and communications review*, 7(3):19–20.

Lindgren, A. and Phanse, K. S. (2006). Evaluation of queueing policies and forwarding strategies for routing in intermittently connected networks. In *2006 1st International Conference on Communication Systems Software & Middleware*, pages 1–10. IEEE.

Misra, S., Saha, B. K., and Pal, S. (2016). Opportunistic mobile networks.

Naves, J. F., Moraes, I. M., and Albuquerque, C. (2012). Lps and lrf: Efficient buffer management policies for delay and disruption tolerant networks. In *Local Computer Networks (LCN), 2012 IEEE 37th Conference on*, pages 368–375. IEEE.

Nunes, I. O., Celes, C., de Melo, P. O., and Loureiro, A. A. (2016a). Groups-net: Group meetings aware routing in multi-hop d2d networks. *arXiv preprint arXiv:1605.07692*.

Nunes, I. O., de Melo, P. O. S. V., and Loureiro, A. A. F. (2016b). Leveraging d2d multihop communication through social group meeting awareness. *IEEE Wireless Communications*, 23(4):12–19.

Nunes, I. O., de Melo, P. O. V., and Loureiro, A. A. F. (2016c). Groups-net: Roteamento ciente de encontros de grupos em redes móveis d2d. In *Proceedings of the Brazilian Symposium on Computer Networks and Distributed Systems*, Salvador, Bahia.

Pyattaev, A., Johnsson, K., Andreev, S., and Koucheryavy, Y. (2013). Proximity-based data offloading via network assisted device-to-device communications. In *Vehicular Technology Conference (VTC Spring), 2013 IEEE 77th*, pages 1–5. IEEE.

Rashid, S. and Ayub, Q. (2010). Efficient buffer management policy dla for dtn routing protocols under congestion. *international Journal of computer and Network Security*, 2(9):118–121.

Rashid, S., Ayub, Q., Zahid, M. S. M., and Abdullah, A. H. (2011). E-drop: An effective drop buffer management policy for dtn routing protocols. *Int. J. Computer Applications*, pages 118–121.

Rashid, S., Ayub, Q., Zahid, M. S. M., and Abdullah, A. H. (2013). Message drop control buffer management policy for dtn routing protocols. *Wireless personal communications*, 72(1):653–669.

Silva, A. P., Burleigh, S., Hirata, C. M., and Obraczka, K. (2015). A survey on congestion control for delay and disruption tolerant networks. *Ad Hoc Networks*, 25:480–494.

Spyropoulos, T., Psounis, K., and Raghavendra, C. S. (2004). Single-copy routing in intermittently connected mobile networks. In *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, pages 235–244. IEEE.

Spyropoulos, T., Psounis, K., and Raghavendra, C. S. (2005). Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259. ACM.

Spyropoulos, T., Psounis, K., and Raghavendra, C. S. (2007). Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility. In *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops' 07. Fifth Annual IEEE International Conference on*, pages 79–85. IEEE.

Tsai, T.-C. and Chan, H.-H. (2015). Nccu trace: social-network-aware mobility trace. *IEEE Communications Magazine*, 53(10):144–149.

Vahdat, A., Becker, D., et al. (2000). Epidemic routing for partially connected ad hoc networks.

Yang, M. J., Lim, S. Y., Park, H. J., and Park, N. H. (2013). Solving the data overload: Device-to-device bearer control architecture for cellular data offloading. *IEEE Vehicular Technology Magazine*, 8(1):31–39.