BT-MANET: A Novel BitTorrent-Like Algorithm for Video On-Demand Streaming over MANETs

C. Rodrigues, and V. Rocha

***Abstract*—This paper proposes a novel BitTorrent-like algorithm for video-on-demand streaming over mobile ad hoc networks: the BT-MANET algorithm. Its conceptual innovations mainly lie on (i) a flexible data-transmission scheme between direct neighbors and on (ii) a sliding window to prioritize data request, settling a compromise between data diversity and playing continuity. Through a number of simulations and assessing four different competitive metrics, we are able to validate our proposal and confirm its attractive performance for on-demand streaming. For instance, it is shown that BT-MANET may support twice the bit rate of a nominal theoretical scheme, besides being reasonably competitive when compared to an alternative wired implementation. Within this context, the key contribution of this paper is to provide valuable insights for real protocol designs targeted at multimedia on-demand streaming over mobile ad hoc networks. General conclusions and avenues for further research are included at the end of this paper.**

***Index Terms*—Streaming, MANET, BitTorrent, multimedia, video on demand.**

I. Introdução

STREAMING é a principal técnica para transmissão de conteúdo multimídia em redes de comunicação. Também é ponto pacífico que as redes sem fio estão sendo cada vez mais utilizadas. Para exemplificar, o *streaming* de vídeo sob demanda e o *download* de arquivos vão juntos corresponder a mais de 81% de todo o tráfego IP da Internet em 2021 [1]. Além disso, naquele mesmo ano, o tráfego gerado por dispositivos sem fio corresponderá a mais de 63% de todo tráfego IP mundial [1].

As redes móveis *ad hoc*, conhecidas também pelo acrônimo MANETs (*Mobile Ad Hoc Networks*), têm papel de destaque no atual e no futuro cenário anunciado. Essas redes possuem arquitetura *peer-to-peer* (P2P) e seus nós (ou *peers*) se comunicam sem depender de qualquer infraestrutura fixa, podendo atuar como servidores, clientes e roteadores. Comunicações em ambientes inóspitos, operações de busca e resgate, conferências, aulas e, ainda, atividades de entretenimento são algumas das inúmeras aplicações para as redes MANETs [2][3].

Dentro deste contexto, este artigo tem como problema de pesquisa a seguinte questão pertencente a literatura recente (e.g., [4]–[7]): como implementar *streaming* de vídeo sob demanda em MANETs? Para contribuir com a solução dessa questão, propõe-se aqui um novo algoritmo P2P, denominado BT-MANET. A proposta baseia-se no conhecido e eficiente BitTorrent [8], que é um protocolo P2P para replicação de arquivos na Internet.

As inovações conceituais do BT-MANET se ancoram nos dois pontos fulcrais a seguir. Primeiro, emprega-se um esquema flexível de transmissão de dados entre nós móveis que se comunicam diretamente (i.e., nós vizinhos), o qual criteriosamente admite o compartilhamento de dados via *unicast* e *broadcast*. Segundo, implementa-se uma janela deslizante para priorização dos dados requisitados pelos nós vizinhos, buscando estabelecer um compromisso entre a diversidade de dados no sistema e a continuidade da reprodução no cliente.

A avaliação do algoritmo BT-MANET é realizada por simulações de cenários de distribuição de um arquivo de vídeo via *streaming* sob demanda. São definidas quatro diferentes métricas de performance, as quais permitem fazer inferências sobre a qualidade de serviço (QoS) do sistema, bem como sobre a escalabilidade da solução em face do crescimento da rede em termos do número de nós móveis.

O restante deste artigo é organizado como segue. A Seção II revisa sucintamente o protocolo BitTorrent e a conceituação de MANET. A Seção III discute trabalhos relacionados. A Seção IV apresenta o novo algoritmo. A Seção V traz experimentos, resultados e análises. Por fim, conclusões gerais e trabalhos futuros constituem a Seção VI.

II. Fundamentos

A. Protocolo de Replicação BitTorrent

O arquivo a ser replicado é dividido em pedaços. Cada pedaço é então dividido em blocos. Os pedaços têm 256,0 KB de tamanho, enquanto os blocos têm 16,0 KB. Apesar de os blocos serem a unidade de dados na rede física, a análise da replicação considera somente os pedaços transferidos [8]. Por simplicidade, os termos *peer* e nó são doravante utilizados indistintamente ao longo do restante deste texto. A operação do BitTorrent é então brevemente descrita a seguir, considerando a visão de um *peer p* recém-chegado ao sistema.

C. K. S. Rodrigues, Universidade Federal do ABC, Santo André, São Paulo, Brasil, carlo.kleber@ufabc.edu.br.

V. E. M. Rocha, Universidade Federal do ABC, Santo André, São Paulo, Brasil, vladimir.rocha@ufabc.edu.br.

O *peer p* deve obter, a partir de um servidor *web*, um arquivo de metadados, relativo ao arquivo de interesse que se deseja baixar. O arquivo de metadados inclui a localização do *tracker*, que é a entidade coordenadora da comunicação entre os *peers* que vão participar da replicação do arquivo de interesse. Esse conjunto de *peers* constitui o *swarm* [9][8].

O *tracker*, após contatado, envia ao *peer p* uma lista *M*, que contém os *peers* que já estão no *swarm*. Essa lista possui tipicamente 50 *peers*. O *peer p* tenta então aleatoriamente estabelecer conexões TCP com os *peers* dessa lista *M*. Os *peers* com os quais ocorrem conexões bem-sucedidas constituem o conjunto de vizinhos. Os vizinhos são então os *peers* para os quais e a partir dos quais o *peer p* pode enviar e receber pedaços do arquivo de interesse [9][8].

Há dois tipos de *peers* no *swarm*: *leechers* e *seeds*. Um *leecher* é um *peer* que está baixando pedaços de um arquivo, mas que também permite que outros *peers* baixem pedaços dele. Um *seed* é um *peer* que já tem todos os pedaços do arquivo desejado, mas que permanece no sistema para permitir que outros *peers* baixem pedaços dele. O intercâmbio de pedaços entre os *peers* é controlado pelas políticas de seleção de *peers* e de seleção de pedaços, como explicado a seguir.

A política de seleção de *peers* (ou algoritmo de *choke*), permite que cada *peer p* do *swarm* escolha quais *peers*, dentre os seus vizinhos, podem receber os pedaços que ele possui. Os três vizinhos que fornecem pedaços para *p* com as mais altas taxas de *upload* são os escolhidos [9][8].

Os *peers* escolhidos são colocados no estado de *unchoked*, e os demais são colocados no estado de *choked*. Um *slot* de *upload* de dados do *peer p* é alocado para cada um dos escolhidos. Essa seleção é repetida tipicamente a cada 10 segundos, constituindo o *regular unchoking* [9][8].

Há ainda o *optimistic unchoking*. Tipicamente a cada 30 segundos, cada *peer p* do *swarm* seleciona aleatoriamente outro *peer* de seu conjunto de vizinhos para receber os pedaços que ele possui. Nesse caso, um *slot* de *upload* do *peer p* é então alocado para esse vizinho [9][8].

A política de seleção de pedaços, por sua vez, é usada para que os *peers* decidam sobre que pedaços solicitar ao passar para o estado de *unchoked*. Cada *peer p* do *swarm* possui um conjunto de pedaços mais raros, i.e., pedaços menos replicados no conjunto de vizinhos. Após passar para o estado de *unchoked*, o *peer p* então solicita o próximo pedaço considerando seu conjunto de pedaços mais raros e os pedaços disponíveis no vizinho que o passou para o estado *unchoked*. Após receber o pedaço, o *peer p* então avisa a todos os seus vizinhos sobre o pedaço recebido [9][8].

Embora reconheça-se a sua eficiência para a replicação de arquivos na Internet, o BitTorrent não pode ser diretamente utilizado para *streaming*, pois os pedaços são transmitidos fora de ordem e sem preocupação com tempo de entrega [10]. É verdade que já há propostas na literatura visando a sua adaptação para *streaming* em redes cabeadas (e.g., [11] e [12]); no entanto, o seu emprego em MANETs ainda exige uma maior discussão, constituindo-se em uma lacuna da literatura (e.g., [2] [13]–[17]) que é abordada neste trabalho.

B. Definição de MANET

Como mencionado, as MANETs são redes de arquitetura P2P que operam sem depender de qualquer infraestrutura fixa. Seus nós são livres para se mover aleatoriamente e se organizarem arbitrariamente. A topologia da rede pode, assim, mudar rapidamente e de forma imprevisível [13][14].

Cada nó móvel pode se comunicar diretamente com outro nó móvel que esteja dentro de seu alcance, i.e., usando um enlace de comunicação de salto único (do inglês, *one-hop wireless link*). Para se comunicar com um nó fora de seu alcance direto, um nó utiliza nós intermediários para retransmitir suas mensagens, i.e., usando um enlace de comunicação de múltiplos saltos (do inglês, *multiple-hop wireless link*) [14][18]. Portanto, os nós atuam como clientes, servidores e roteadores em simultâneo.

A evolução das tecnologias móveis e a redução de custos por parte dos fabricantes possibilitaram uma vertiginosa popularização do uso de *smartphones*, PDAs, *laptops* e semelhantes. Essa popularização trouxe consigo o desenvolvimento de uma diversidade de aplicações para MANETs, incluindo desde comunicações militares até atividades em educação e entretenimento [13][14][18].

Essa mesma evolução tecnológica propiciou o surgimento de variações da MANET original, que podem ser admitidas como subcategorias ou como novos tipos independentes. Dentre as mais comuns, citam-se as três seguintes: 1) *Vehicular Ad Hoc Network* (VANET), cuja característica principal é ter o nó móvel sendo um veículo, e.g., um caminhão [19]; 2) *Flying Ad Hoc Network* (FANET), cuja característica principal é ter o nó móvel sendo um *Veículo Aéreo Não Tripulado* (VANT), também conhecido como *drone* [20]; e 3) *Wireless Mesh Network* (WMN), cuja característica principal é ter uma rede hierarquizada, constituída por uma infraestrutura fixa e parcialmente cabeada para suportar a comunicação, denominada de *mesh backbone*, e também por uma parte que abarca os clientes da rede, denominados de *mesh clients*, contemplando distintos tipos de dispositivos sem fio [21][22]. Para fins de análise desses diversos tipos de redes, destaca-se a importância do uso de modelos de mobilidade para realização de experimentos realísticos [23].

Para finalizar esta seção, é importante ressaltar que o algoritmo BT-MANET visa exclusivamente a redes MANETs, considerando sua definição original, e a aplicações cuja área geográfica onde os nós se movimentam é plana, sem obstáculos e sem cobertura da Internet. Esse cenário pode representar, e.g., parte de um *campus* universitário, onde os alunos desejam assistir a uma aula ao ar livre, ou mesmo um saguão de aeroporto, em que os clientes assistem a um filme enquanto esperam o momento do embarque.

III. TRABALHOS RELACIONADOS

Esta seção discorre sucintamente sobre alguns dos mais importantes trabalhos da literatura sobre o desenvolvimento de protocolos para a transmissão de arquivos multimídia em MANETs. Opta-se por uma citação preferencialmente cronológica, em que se realça a concepção das propostas.

Em [24], os autores propõem uma extensão do protocolo BitTorrent para emprego em MANETs. Essa extensão consiste na adição de uma interface entre o núcleo do BitTorrent, na camada de aplicação, e a camada de transporte, de modo a permitir o envio e o recebimento de

dados via *broadcast*, além do tradicional envio via *unicast*. Embora os resultados obtidos sejam satisfatórios para o tempo de *download* do arquivo, não são realizados experimentos para *streaming* sob demanda.

Em [25], os autores prestam uma significativa contribuição para a pesquisa de *streaming* em MANETs. É apresentado um amplo *survey* de 44 trabalhos relacionados. São discutidas principalmente as concepções dos protocolos, estratégias de avaliação de performance, métricas utilizadas, e limitações dos resultados. Em que pese o fato de não serem apresentadas novas propostas, o trabalho serve como valiosa orientação para o desenvolvimento de projetos.

Em [26], apresenta-se um protocolo de replicação distribuído que considera a capacidade individual de armazenamento do nó móvel, além da periodicidade de encontro entre os nós detentores dos arquivos e os demais nós da rede. Embora extensivas simulações comprovem a satisfatória performance da proposta, os resultados obtidos apontam para otimizações com foco apenas no tempo de *download* do arquivo, não sendo realizados experimentos para *streaming* sob demanda.

Em [27], é proposto um esquema de escalonamento de pedaços de vídeo, denominado de DSSSA (*Delay-Sensitive Segment Scheduling Algorithm*). Esse esquema usa estimativas de atraso fim a fim coletadas por um protocolo de roteamento proposto em [28], seguindo assim um paradigma de desenvolvimento *cross-layer*. A validação se dá por simulações comparativas com esquemas tradicionais, em que são observadas otimizações na taxa de admissão e na taxa de recepção de dados com sucesso. Essa proposta, junto com aquela apresentada em [29], constituem as duas únicas referências, dentre aquelas consultadas, a considerar explicitamente limites de tempo para entrega do arquivo, que é fundamental para realizar *streaming*.

Por fim, em [29], são propostos dois algoritmos com base no protocolo BitTorrent para MANETs. Sua filosofia baseia-se no uso de novos critérios para a seleção de peers e, como no algoritmo aqui proposto (i.e., BT-MANET), segue o paradigma de desenvolvimento *layered architecture* e emprega uma janela deslizante para seleção de pedaços. No entanto, diferentemente do algoritmo BT-MANET, o tráfego de dados é exclusivamente via *unicast*. A avaliação é feita por meio de simulações e os resultados obtidos permitem concluir que os algoritmos são eficientes.

Ante o exposto, o algoritmo BT-MANET combina peculiaridades de trabalhos anteriores em uma única solução, em que se destacam: baseia-se no protocolo BitTorrent; aplica-se à *streaming* sob demanda, sendo inteiramente reativo; usa janela deslizante para seleção de pedaços; admite tráfego *unicast* e *broadcast*; e explora apenas a camada de aplicação (i.e., segue o paradigma de desenvolvimento *layered architecture*). O novo algoritmo é, assim, independente de protocolos de roteamento [16][30], de simples implementação, e de convivência harmoniosa com outras aplicações já instaladas [7].

IV. Novo Algoritmo: BT-MANET

Esta seção apresenta o novo algoritmo BT-MANET. A apresentação é feita por meio da explicação de suas políticas de seleção de *peers* e de seleção de pedaços. Subtende-se conhecido o arcabouço conceitual apresentado na Seção II.

Para a política de seleção de *peers*, tem-se o entendimento, explicado a seguir, admitindo a visão de um *peer p* que recebe a lista *M* a partir do *tracker*. Os vizinhos do *peer p* são escolhidos aleatoriamente a partir da lista *M*, informada pelo *tracker* do sistema. Os *peers* escolhidos são colocados no estado *choked* e classificados em *interessados* e *não interessados*: se um vizinho não tem todos os pedaços que o *peer p* possui, este vizinho é classificado como *interessado* no *peer p*; caso contrário, é classificado em não interessado no *peer p*. A banda de *upload* do *peer* é dividida em 4 *slots* de dados: 3 *slots* para *regular unchoking*; 1 *slot* para *optimistic unchoking*.

A transmissão dos pedaços ocorre apenas entre vizinhos e se dá via *unicast* ou *broadcast*. Não existe, assim, transmissões indiretas, i.e., passando por nós intermediários. Por fim, o Quadro 1 traz uma descrição dos processos de *regular unchoking*, que ocorre a cada 10 s, e *optimistic unchoking*, que ocorre a cada 30 s, respectivamente, dessa política.

QUADRO I
Processos De Regular Unchoking e Optimistic Unchoking

Processo: *Regular unchoking*
Início
Passo 1: Se o *peer* for um *leecher*
Então os *peers interessados* são ordenados em função de suas taxas de *upload* de dados. Os três *peers* de maiores taxas são escolhidos e aloca-se 1 *slot* a cada um deles. Em caso de empate, a escolha é aleatória;

Passo 2: Se o *peer p* for um *seed*
Então os *peers interessados* são ordenados em função da taxa de *download* de dados a partir do *peer p*. Aos três peers de maiores taxas, alocam-se os *slots* de *upload* do *peer p*, sendo 1 *slot* para cada *peer*. Em caso de empate, a escolha é aleatória. Se é a primeira vez que o pedaço é solicitado, então a transmissão é em *brodacast*; caso contrário, em *unicast*.
Fim

Processo: *Optimistic unchoking*
Início
Conforme proposta original do protocolo BitTorrent: 1 *slot* de dados é alocado a um *peer interessado* escolhido aleatoriamente.
Fim.

Para a política de seleção de pedaços, tem-se o seguinte entendimento. Admita que o arquivo a ser transmitido possui *t* pedaços: 1, 2, …, *t*. Seja então *d* o pedaço que corresponde ao ponto de reprodução corrente. Ainda, seja *W* uma janela deslizante, e seja *w* o seu tamanho em número de pedaços. A janela *W* compreende então os seguintes pedaços: $[d; d + w]$. A janela *W* é dinamicamente atualizada, conforme os pedaços são reproduzidos. Por exemplo, seja j_{play} o número de pedaços reproduzidos. O primeiro pedaço de *W* é então atualizado para $(d + j_{play})$, e o último pedaço para $(d + w + j_{play})$.

Agora, seja *V* um *buffer* contido em *W*, e seja *v* o seu tamanho em número de pedaços. O primeiro pedaço de *V* é

sempre coincidente com o primeiro pedaço de *W*. Contando a partir do primeiro pedaço de *W*, se *v* pedaços consecutivos já tiverem sido recuperados, mas ainda não tiverem sido reproduzidos, então solicitar o pedaço mais raro dentre aqueles contidos em *W* que ainda não tenham sido recuperados; caso contrário, solicitar o próximo pedaço consecutivo não recuperado até que *v* pedaços consecutivos tenham sido recuperados, mas ainda não tenham sido reproduzidos.

Note que, diferentemente do BitTorrent original, o pedaço a ser solicitado não é o mais raro. A alternância de método para escolher o próximo pedaço visa a estabelecer um compromisso entre a diversidade de pedaços no sistema, estimulada pela política do mais raro, e a continuidade na reprodução do arquivo, estimulada pela política gulosa, tal como deve ser em um sistema de *streaming* sob demanda.

V. Análise de Performance

A. Modelagem, Cenários e Métricas

Os modelos são implementados no ambiente de simulação *PeerSim* [31]. O sistema computacional consiste em um PC Intel i7, 2,6 GHz de *clock*, 24,0 GBytes de RAM, e sistema operacional GNU/Linux. Os resultados de simulação têm intervalos de confiança de 95% que estão dentro do limite de 5% dos valores médios reportados, tendo sido realizadas 30 execuções (rodadas).

Os cenários simulados possuem *n* nós que se movimentam aleatoriamente sobre uma área plana *A*, sem obstáculos, de 100 m x 100 m. A velocidade de cada nó é escolhida entre 0 e 2 m/s, e a sua direção entre 0 e 360 graus. O *throughput* de cada nó é de 4 Mbps por *slot* (*upload* e *download*). Dentre esses *n* nós, há um *tracker* e um *seed* que operam ininterruptamente e são alcançáveis por todos os nós.

Na camada de rede, admite-se o protocolo de roteamento baseado no AODV (*Ad hoc On-Demand Distance Vector*) [32] do simulador WiFi Direct [28]. Para as camadas MAC e física, são considerados, respectivamente, os protocolos CSMA/CA e 802.11 [34], com percentual de perda de quadros configurado em 1,0%.

O arquivo a ser distribuído por *streaming* sob demanda corresponde a um vídeo com codificação de 300,0 kbps. Esse vídeo é dividido em 390 pedaços de 256,0 kbytes, e cada pedaço é dividido em blocos de 16,0 kbytes. O arquivo tem, portanto, tamanho total aproximado de 100,0 Mbytes. Apesar de os blocos serem a unidade de dados na rede física, a análise considera apenas os pedaços transmitidos.

São avaliadas quatro métricas de performance, assim definidas: (i) *download operacional*, corresponde à taxa média para receber o vídeo; (ii) *tempo de download*, corresponde ao tempo médio para receber o vídeo; (iii) *tempo de descontinuidade*, corresponde ao tempo médio de interrupção durante a reprodução do vídeo; (iv) *descontinuidades*, corresponde ao número médio de pedaços indisponíveis durante a reprodução do vídeo.

B. Organização dos Experimentos

São realizados cinco experimentos. Os quatro primeiros admitem três variantes da solução algorítmica proposta na Seção IV, considerando $n = 100$, e ainda uma versão denominada de *cabeada*. Essas variantes e a versão *cabeada* são implementadas para permitir uma maior facilidade de análise dos resultados e entendimento das conclusões.

As três variantes diferenciam-se no Passo 2 do processo de *regular choking* (vide Quadro 1, Seção IV), como segue. A primeira variante, denominada *manet*, não emprega *broadcast*; a segunda, denominada *bc*1, realiza *broadcast* de um pedaço apenas quando ocorre sua primeira solicitação; finalmente, a terceira, denominada *bc*2, realiza *broadcast* de um pedaço quando ocorrem a primeira e, também, a segunda solicitação do mesmo.

A versão *cabeada*, por sua vez, difere das três variantes por aplicar-se à uma rede com fio e a nós sem mobilidade. Esta versão serve como um referencial teórico ideal para comparação de performance, explicitando o desempenho ótimo que pode ser eventualmente alcançado. Esta versão emprega o mesmo algoritmo da variante *manet*, mas admitindo o protocolo de roteamento OSPF (*Open Shortest Path First*).

Para esses quatro primeiros experimentos, tem-se que o tamanho do *buffer V* varia desde 10 pedaços (cerca de 2% do arquivo) até 90 pedaços (cerca de 20% do arquivo), e a janela *W* tem tamanho fixo de 390 pedaços, correspondendo ao total do arquivo.

Por fim, o quinto experimento avalia a escalabilidade da solução BT-Manet ante o aumento do valor de *n*, bem como realiza uma análise comparativa com outro trabalho anterior da literatura. Para tanto, levam-se em conta as quatro métricas definidas, a variante identificada como de melhor performance nos quatro experimentos anteriores. Os valores dos parâmetros de configuração desse experimento são explicitados mais adiante.

C. Resultados e Análises

Na Fig. 1, pode-se observar o experimento para *download operacional*, que corresponde à taxa média para receber o vídeo. Como esperado, a versão *cabeado* é a de melhor performance, alcançando até 1.000,0 kbps. As variantes *bc*1 e *bc*2 têm performances semelhantes, alcançando até 600,0 kbps.

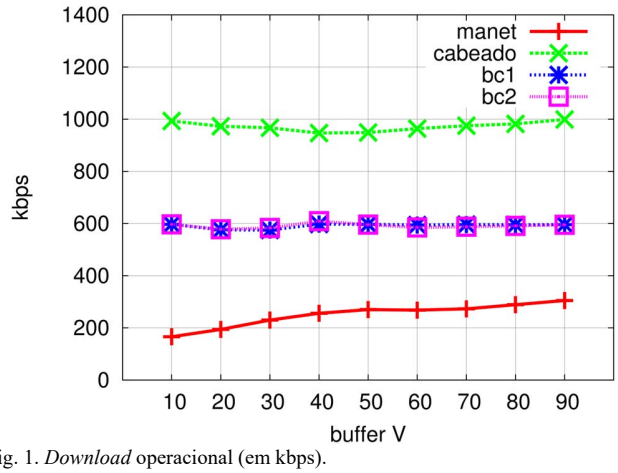

Fig. 1. *Download* operacional (em kbps).

A *manet* é a de pior performance, limitada superiormente a 240,0 kbps. É também possível constatar que as variantes *bc*1 e *bc*2 podem comportar arquivos codificados em até 600,0

kbps, independentemente do tamanho do *buffer V*. Esse valor máximo se torna especialmente interessante no caso de sistemas de vídeo adaptativo ou no caso de desejar-se uma codificação de maior taxa para obter maior qualidade de visualização. Em síntese, embora as variantes fiquem 40% abaixo do patamar teórico ideal, que corresponde àquele da versão *cabeado*, o resultado absoluto é bem atrativo, pois corresponde a duas vezes o valor de codificação do arquivo, i.e., 300,0 kbps.

Na Fig. 2, pode-se observar o experimento do *tempo de download*. A variante *manet* é a de pior performance e, mesmo em sua melhor condição (i.e., v = 90), precisa de quase o dobro de tempo se comparada com as variantes *bc*1 e *bc*2. A variante *cabeado*, por sua vez e como esperado, é a de melhor performance. Como exemplo, para v = 20, os *tempos de download* são respectivamente 820,0 segundos (*cabeado*), 1.385,0 segundos (*bc*1 e *bc*2), e 4.106,0 segundos (*manet*). Esses resultados confirmam as observações alcançadas na discussão do *download operacional*, no sentido de que as variantes *bc*1 e *bc*2 têm ambas performances satisfatórias, ficando em patamar aceitável de desvantagem com relação à referência ideal (i.e., versão *cabeado*).

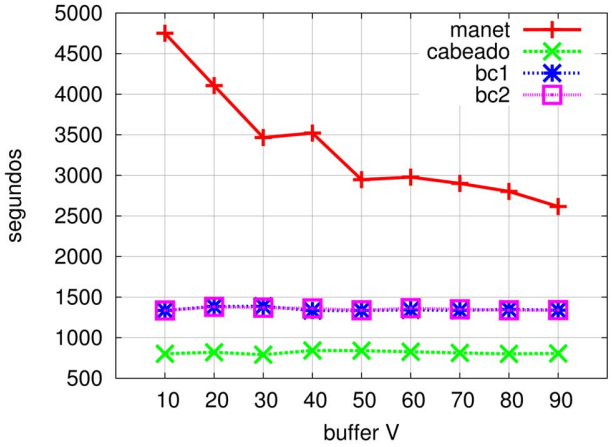

Fig. 2. Tempo de *download* (em segundos).

Na Fig. 3, pode-se observar o *tempo de descontinuidade*, que, conforme definido, corresponde ao tempo médio de interrupção durante a reprodução do vídeo. A variante *manet* possui o maior *tempo de descontinuidade*, mas, se assemelha às outras variantes para $v \geq 50$. Para $v \geq 30$, as variantes *bc*1 e *bc*2 e a versão *cabeado* não apresentam diferenças entre si e possuem performances satisfatórias. Interessantemente, no intervalo de $10 \leq v < 30$, as variantes *bc*1 e *bc*2 são inclusive melhores que a versão *cabeado*. Isto deve-se ao fato de que o *broadcast* permite que um nó, que ainda não fez a requisição pelo pedaço, o receba mesmo não tendo passado pelo *unchocking*, diminuindo assim o seu tempo de espera.

Na Fig. 4, tem-se a análise da métrica *descontinuidades*. A variante *manet* é a de pior performance e, mesmo em sua melhor condição (i.e., v = 90), apresenta quase o dobro de pedaços indisponíveis se comparada com as variantes *bc*1 e *bc*2. As variantes *bc*1, *bc*2 e *cabeado* não apresentam uma diferença significativa entre si, começando com aproximadamente 1,0 pedaço indisponível, para $v \leq 30$, e finalizando com menos de 1 pedaço, para $v > 30$. Esses resultados são bastante positivos e confirmam as observações alcançadas nas discussões anteriores sobre a satisfatória performance das variantes *bc*1 e *bc*2, pois as interrupções medidas nesse experimento não são significativas.

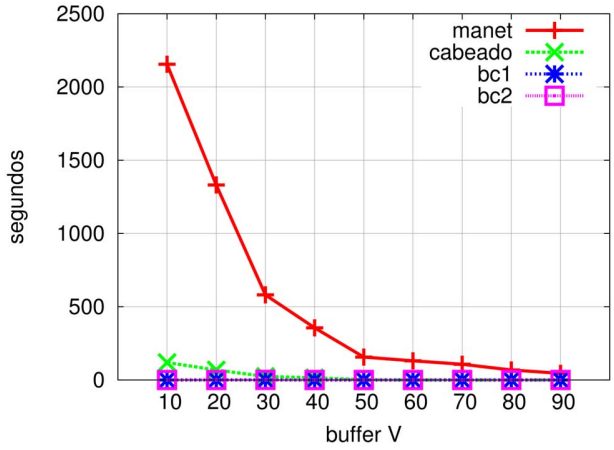

Fig. 3. Tempo de descontinuidade na reprodução (em segundos).

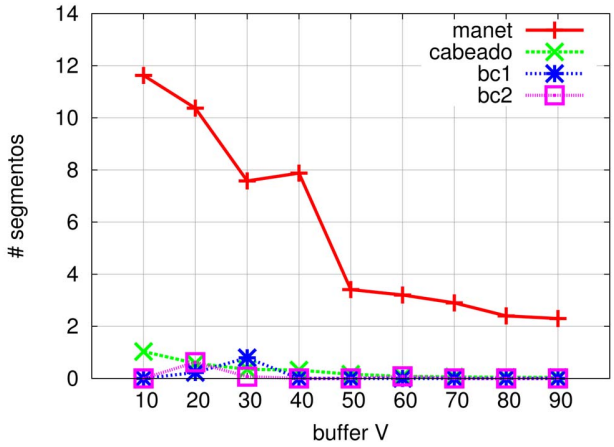

Fig. 4. Descontinuidade na reprodução (em pedaços ou segmentos).

A partir dos resultados e discussões tidas, é possível concluir que o algoritmo BT-MANET é capaz de prover uma capacidade de recuperação do arquivo que corresponde a duas vezes a taxa de codificação do arquivo (e.g., vide Fig. 1), e as descontinuidades medidas não são significativas (e.g., vide Fig. 3), estando inclusive dentro do mesmo patamar daquelas de redes cabeadas. Mais pontualmente, também pode ser concluído que o tamanho do *buffer* v pode ser estimado em apenas 2% do tamanho total do arquivo, pois a performance já é satisfatória (e.g., vide Fig. 3). Veja que, quanto menor puder ser esse tamanho do *buffer*, menor será a necessidade de uma recuperação de pedaços gulosa, favorecendo o emprego mais frequente da política do pedaço mais raro, garantindo então a maior diversidade de pedaços e, notadamente, a eficiência global sistêmica. Também é possível concluir pontualmente que o número de envios via *broadcast* de cada pedaço pode ser unitário (i.e., apenas uma vez), tendo em vista que as variantes *bc*1 e *bc*2 apresentam performances semelhantes. Isso simplifica o processo de transmissão de dados no sistema. Assim, por sua maior simplicidade, *bc*1 termina então sendo uma melhor opção que *bc*2.

Nas Figs. 5 e 6, tem-se o último experimento realizado, o qual trata sobre a *escalabilidade* da solução algorítmica aqui proposta e também realiza uma análise comparativa com uma proposta anterior da literatura. Explica-se que a *escalabilidade* diz respeito à avaliação do desempenho da proposta em face do aumento do número de nós na rede. Quanto menor impacto houver no desempenho, mais escalável é a proposta.

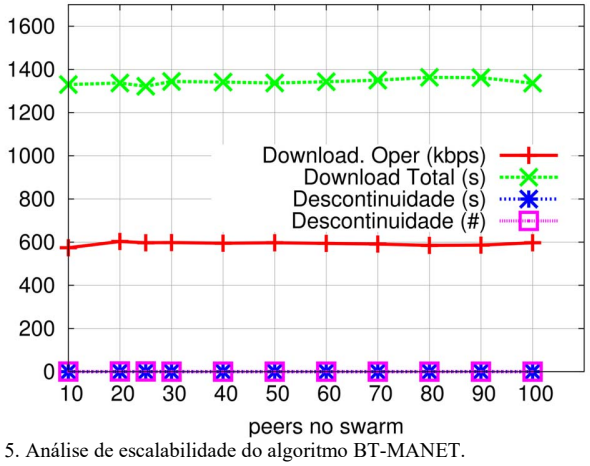

Fig. 5. Análise de escalabilidade do algoritmo BT-MANET.

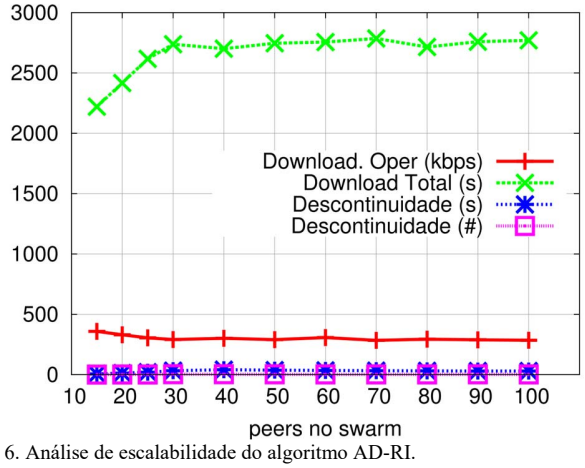

Fig. 6. Análise de escalabilidade do algoritmo AD-RI.

Para este experimento, o tamanho do *buffer V* é fixado em 10 pedaços, a *janela W* segue fixada em 390, apenas a variante *bc*1 é considerada, e a proposta anterior da literatura denominada de AD-RI (Algoritmo de Distância Mínima com Reciprocidade Indireta) [29]. Essa proposta foi escolhida para comparação por ser recente, de satisfatória performance, explorar apenas a camada de aplicação (i.e., paradigma *layered architecture*), e ter o mesmo propósito de aplicação do algoritmo BT-MANET.

Explica-se que, sob AD-RI, um *peer* seleciona os *peers* para os quais fará o *upload* de pedaços considerando tanto a distância geográfica (i.e., menor distância implica maior probabilidade de ser escolhido), quanto a reciprocidade indireta (i.e., maior compartilhamento de pedaços com outros *peers* implica maior probabilidade de ser escolhido).

A partir dos resultados obtidos, tem-se o seguinte. Na Fig. 5 pode-se observar que o aumento de nós, variando de 10 até 100 não tem um impacto significativo sobre as métricas aqui analisadas. O mesmo pode ser observado na Fig. 6. Isso permite concluir positivamente sobre a escalabilidade das propostas BT-MANET e AD-RI. No entanto, BT-MANET supera AD-RI em quase o dobro, na métrica de *download operacional*, e diminui à metade o *tempo de download total*. Assim, pode-se então concluir que BT-MANET é superior à AD-RI. O uso de transmissão *broadcast* e a maior simplicidade das políticas de seleção de *peers* de BT-MANET, comparativamente a AD-RI, são as principais razões que justificam essa superioridade observada nos experimentos.

VI. Conclusões e Trabalhos Futuros

Este artigo apresentou um novo algoritmo para realização de *streaming* sob demanda em redes móveis *ad hoc*: algoritmo BT-MANET. Por meio de simulações e considerando quatro distintas métricas de análise, constatou-se a satisfatória eficiência e escalabilidade da proposta.

Como resultados pontuais, destacam-se: a taxa média de recuperação de dados e a continuidade de reprodução ficaram em patamares significativamente atrativos; a escalabilidade do algoritmo se mostrou satisfatória para MANETs de até 100 nós móveis; por fim, o envio de dados em *broadcast* e o emprego de uma janela deslizante para priorização dos dados a solicitar se revelaram ideias promissoras para o projeto de protocolos de *streaming* sob demanda em MANETs.

Como trabalhos futuros, apontam-se os seguintes caminhos: complementar a análise aqui realizada com modelos analíticos e medições reais; comparar o BT-MANET com outros algoritmos da literatura ante: distintas áreas de cobertura, variados modelos de mobilidade e interatividade [30][35], nós móveis heterogêneos, arquivos multimídia de múltiplas codificações [25], e métricas de desempenho mais relacionadas a qualidade da imagem, em complementação ás métricas de transmissão de vídeo sob a percepção do usuário aqui utilizadas, e.g., PSNR (*peak-signal-to-noise ratio*) e SSIM (*structural similarity index measure*); por fim, avaliar o algoritmo BT-MANET usando diferentes métodos de roteamento [16][30][2] e aplicado ao serviço de *live streaming* [36][35].

Referências

[1] CISCO, “Cisco Visual Networking Index: Forecast and Methodology, 2016–2021,” June 6, 2017. Available at: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.pdf. Accessed on: Oct. 22rd, 2018.

[2] S. Ghalib et al., “Routing protocol development for quality of service optimization of video-on-demand system over mobile ad hoc networks,” *Int. Journal of Communications Systems*, vol. 31, no. 2, Jan., 2018.

[3] R. Kumar, K. Taneja, and H. Taneja, “Performance Evaluation of MANET Using Multi-Channel MAC Framework,” *Procedia Computer Science*, vol. 133, pp. 755–762, 2018.

[4] M. Usman et al., “Performance evaluation of High Definition video streaming over Mobile Ad Hoc Networks,” *Signal Processing*, vol. 148, pp. 303–313, July, 2018.

[5] D. Chander and R. Kumar, “QoS Enabled Cross-Layer Multicast Routing over Mobile Ad Hoc Networks,” *Procedia Computer Science*, vol. 125, pp. 215–227, 2018.

[6] W. Wang et al., “On the packet delivery delay study for three-dimensional mobile ad hoc Networks,” *Ad Hoc Networks*, vol. 69, pp. 38–48, Feb., 2018.

[7] C. M. S. Ferreira et al., “An Evaluation of Cross-Platform Frameworks for Multimedia Mobile Applications Development,” *IEEE Latin America Transactions*, vol.16, no. 4, Apr. 2018.

[8] B. Cohen, "Incentives build robustness in BitTorrent," in *First Workshop on Economics of Peer-to-Peer Systems*, Berkeley, USA, 2003.
[9] A. Legout, G. Urvoy-Keller, and P. Michiardi. "Rarest first and choke algorithms are enough," in *6th ACM SIGCOM Conference on Internet Measurement*, Rio de Janeiro, RJ, Brazil, 2006.
[10] C. K. S. Rodrigues, "On the optimization of BitTorrent-like protocols for interactive on-demand streaming systems," *Int. Journal of Computer Networks and Communications (IJCNC)*,vol. 6, no. 5, pp. 39–58, 2014.
[11] L. D'Acunto et al., "BitTorrent-like P2P approaches for VoD: A comparative study," *Computer Networks*, vol. 57, no. 5, pp. 1253–1276, 2013.
[12] C. K. S. Rodrigues and M. V. M. Rocha, "A dispersão de dados como critério para a política de seleção de peers em uma rede BitTorrent para streaming sob demanda interativo," *Revista de Sistemas e Computação (RSC)*, vol. 7, no. 1, pp. 36–43, 2017.
[13] S. A. Mahmud et al., "A comparison of MANETs and WMNs: Commercial feasibility of community wireless networks and MANETs," in *First International Conference on Access Networks (AccessNet)*, Athens, Greece, Sept., 2006.
[14] J. Wang, B. Xie, and D. P. Agrawal, "Journey from mobile ad hoc networks to wireless mesh networks," *Guide to Wireless Mesh Networks, Computer Communications and Networks*, ISBN 978-184800-908-0, Springer London, 2009.
[15] Y. Jahir et al., "Routing protocols and architecture for Disaster Area Network: A survey," *Ad Hoc Networks*, vol. 82, pp. 1–14, January, 2019.
[16] J. J. Ferronato and M. A. S. Trentin, "Analysis of Routing Protocols OLSR, AODV and ZRP in Real Urban Vehicular Scenario with Density Variation," *IEEE Latin America Transactions*, vol.15, no. 9, Sept., 2017.
[17] E. P. F. da Cruz, "A Comprehensive Survey in Towards to Future FANETs," *IEEE Latin America Transactions*, vol.16, no. 3, Mar., 2018.
[18] S. Xiang and J. Yang, "Performance reliability evaluation for mobile ad hoc networks," *Reliability Engineering & System Safety*, vol. 169, pp. 32–39, Jan., 2018.
[19] S. Boussoufa-Lahlah, F. Semchedine, and L. Bouallouche-Medjkoune, "Geographic routing protocols for Vehicular Ad hoc NETworks (VANETs): A survey," *Vehicular Communications*, vol. 11, January, pp. 20–31, 2018.
[20] E. F. Cruz, "A Comprehensive Survey in Towards to Future FANETs," *IEEE Latin America Transactions*, vol. 16, no. 3, pp. 876–884, 2018.
[21] J. Wang, B. Xie, and D. P. Agrawal, "Journey from Mobile Ad Hoc Networks to Wireless Mesh Networks," In: (eds) *Guide to Wireless Mesh Networks. Computer Communications and Networks*, Springer, London, 2009.
[22] C. A. G. Silva et al., "A Study of the Mesh Topology in a ZigBee Network for Home Automation Applications," *IEEE Latin America Transactions*, vol. 15, no. 5, p. 935-942, 2017.
[23] F. Bai and A. Helmy, "A Survey of mobility models in wireless ad hoc network," In: (eds.) *Wireless Ad Hoc and Sensor Networks*, Springer, October, pp. 1–30, 2006. Available at: https://www.cise.ufl.edu/~helmy/papers/Survey-Mobility-Chapter-1.pdf Accessed on: Nov. 17th, 2018.
[24] N. Quental and P. Gonçalves, "Mobile-BitTorrent: a BitTorrent Extension for MANETs," *Revista de Exatas e Tecnológicas*, vol. 1, no. 1, 2010.
[25] S. Mantzouratos et al. "Survey of Cross-layer Proposals for Video Streaming over Mobile Ad hoc Networks (MANETs)," in *International Conference on Telecommunications and Multimedia (TEMU),* July, 2012.
[26] K. Chen and H. Shen, "Maximizing P2P file access availability in mobile ad hoc networks through replication for efficient file sharing," *IEEE Trans. Comput*., vol. 64, no. 4, pp. 1029–1042, 2015.
[27] C.-C. Hu et al., "Timely scheduling algorithm for P2P streaming over MANETs," *Computer Networks*, vol. 127, pp. 56–67, 2017.
[28] C.-C. Hu, "Delay-sensitive routing in multi-rate MANETs," *J. Parallel Distrib. Comput.*, vol. 71, no. 1, pp. 53–61, 2011.
[29] C. K. S. Rodrigues, "Efficient BitTorrent-Like Algorithms for Interactive On-Demand Multimedia Streaming over MANETs," in *24th Brazilian Symposium on Multimedia and the Web (WebMedia '18)*, Salvador, BA, pp. 29–36, Oct., 2018.
[30] S. Nazhad et al. "An efficient routing protocol for the QoS support of large-scale MANETs," *International Journal of Communication Systems*, vol. 31, no. 1, Jan., 2018.
[31] A. Montresor and M. Jelasity, "PeerSim: A scalable P2P simulator," in *9th International Conf. on Peer-to-Peer (P2P'09)*, pp. 99–100, 2009.
[32] C. Perkins, E. Belding-Royer and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing", RFC 3561, DOI 10.17487/RFC3561, 2003, Available at: https://www.rfc-editor.org/info/rfc3561, Accessed on: Oct. 22rd, 2018.
[33] B. Luciano, N. Derakhshan, and S. Guinea, "WiDiSi: A Wi-Fi direct simulator," in *IEEE Wireless Communications and Networking Conference*, pp. 1–7, 2016.
[34] B. Crow, I. Widjaja, and P. Sakai, "IEEE 802.11 Wireless Local Area Networks," *Comm. Mag*, vol.35, no. 9, pp. 116-126, 1997.
[35] D. da Silva et al. "A First Look at Mobile-Live-Users of a Large CDN," in *23rd Brazillian Symposium on Multimedia and the Web (WebMedia'17)*, Gramado, RS, Brazil, pp. 81–84, Oct. 2017.
[36] J. M. O. Neto and L. D. Nery e Silva, "A Middleware for the Development of Distributed Collaborative Video Applications," in *23rd Brazilian Symposium on Multimedia and the Web (Webmedia'17)*, Gramado, RS, Brazil, pp. 357–364, Oct., 2017.

Carlo K. S. Rodrigues é Doutor em Engenharia de Sistemas e Computação pela Universidade Federal do Rio de Janeiro (UFRJ, 2006) e Mestre em Sistemas e Computação pelo Instituto Militar de Engenharia (IME, 2000). É professor do Centro de Matemática, Computação e Cognição da Universidade Federal do ABC (UFABC), no curso de Ciência da Computação. Atua na subárea de Redes de Computadores.
http://buscatextual.cnpq.br/buscatextual/visualizacv.do?id=K4721493T1.

Vladimir E. M. Rocha é Doutor em Engenharia de Computação pela Escola Politécnica da Universidade de São Paulo (POLI-USP, 2017), e Mestre em Ciência da Computação pelo Instituto de Matemática e Estatística da mesma universidade (IME-USP, 2005). É professor do Centro de Matemática, Computação e Cognição (CMCC) da UFABC, no curso de Ciência da Computação. Atua em Sistemas Distribuídos. http://buscatextual.cnpq.br/buscatextual/visualizacv.do?id=K4753743J0.