Algoritmos de Controle de Congestionamento Justos e Eficientes para Aplicações TCP em Redes Sensíveis a Perda de Pacotes e a Atrasos

Marcos Vinícius da Costa Madeira[1] **e Diogo Menezes Ferrazani Mattos**[1]

[1] LabGen/MídiaCom – TET/IC/PPGEET/UFF
Universidade Federal Fluminense (UFF)
Niterói, RJ – Brasil

Abstract. *As the number of devices connected to the Internet increases, the response of applications to competition for network resources and, consequently, to congestion has become a critical issue that affects service quality and user experience. Next-generation networks, such as beyond 5G, 6G, and low Earth orbit (LEO) satellite networks, face additional challenges due to their high sensitivity to delays and packet losses. This paper proposes three TCP congestion control algorithms (CCA) models, named Expolinear, Fractional, and Linear, which leverage packet loss information to dynamically adjust the congestion window (CWND), improving efficiency and adaptability to varying network conditions. We have implemented the algorithms as Linux kernel modules and compared them to traditional algorithms. The proposed models demonstrated up to ten times greater transmission rate efficiency in high packet loss scenarios while maintaining fairness in bandwidth usage among competing flows.*

Resumo. *À medida que o número de dispositivos conectados à Internet aumenta, a resposta das aplicações à competição por recursos de rede e, consequentemente, ao congestionamento, tornou-se uma questão crítica que afeta a qualidade do serviço e a experiência do usuário. Redes de próxima geração, como as além do 5G, 6G e redes de satélites de órbita terrestre baixa (LEO), enfrentam desafios adicionais devido à alta sensibilidade a atrasos e perdas de pacotes. Este artigo propõe três modelos de algoritmos de controle de congestionamento para o TCP (CCA), denominados Expolinear, Fracionário e Linear, que utilizam informações sobre perdas de pacotes para ajustar dinamicamente a janela de congestionamento (CWND), melhorando a eficiência e a adaptabilidade às condições variáveis da rede. Os algoritmos propostos são implementados como módulos do núcleo do Linux e são comparados com algoritmos tradicionais. Os modelos propostos demonstraram uma eficiência até dez vezes superior na taxa de transmissão em cenários com alta perda de pacotes, mantendo a equidade no uso da largura de banda entre fluxos concorrentes.*

1. Introdução

O tráfego de dados em redes móveis cresceu 21% entre o terceiro trimestre de 2023 e o terceiro trimestre de 2024, impulsionado principalmente pelo aumento do número de *smartphones* e pelo consumo médio mais elevado de dados por usuário. Nesse cenário, ao final de 2024, tráfego de vídeo respondia por aproximadamente 74% de todo o

tráfego móvel[1], evidenciando a relevância crescente de conteúdo interativos e com grande consumo de banda. Além disso, estima-se que mais de 60% do tráfego global de Internet seja originado de dispositivos móveis, com a proporção de usuários que acessam a rede via telefone celular ultrapassando 90%[2], reforçando a importância de estratégias focadas em dispositivos móveis, bem como a necessidade de infraestrutura e otimizações específicas para lidar com o alto volume de dados em vídeo e outras mídias interativas. Paralelamente, a variação do tempo de ida e volta (*Round Trip Time* - RTT) em diferentes tecnologias de rede reflete a diversidade de características físicas e arquiteturais entre essas infraestruturas. Enquanto redes terrestres baseadas em fibras ópticas apresentam RTTs influenciados pela velocidade de propagação da luz no material e por percursos muitas vezes maiores que a distância geográfica, redes de satélites de órbita terrestre baixa (*Low Earth Orbit* - LEO) podem oferecer RTTs menores em longas distâncias, pois o sinal viaja no vácuo [Conde et al., 2024]. Ainda assim, fatores como atrasos de processamento e enfileiramento impactam ambos os cenários, especialmente em redes congestionadas ou tecnologicamente menos avançadas.

Os computadores, *smartphones*, sensores IoT (Internet das Coisas) e demais dispositivos que compartilham informações entre si são ubíquos no mundo contemporâneo e tendem a elevar sua presença no futuro próximo. Assim, o tráfego nas redes de computadores fica cada vez maior e, também, aumenta a probabilidade de ocorrência de competição por recursos de rede e congestionamento, o que tem como corolário a perda de pacotes e a redução na qualidade do serviço experienciada pelos usuários. Uma forma de acomodar tal tráfego adicional é aumentar as especificações físicas das redes, como, por exemplo, os *buffers* e a largura de banda. Outra forma é empregar algoritmos de controle de congestionamento (*Congestion Control Algorithms* - CCAs) mais eficientes.

Os CCAs podem ser implementados nos dispositivos finais, tais como computadores, servidores e *smartphones*, em dispositivos no núcleo da rede, como *switches* e roteadores, ou em uma combinação de ambos. Os sistemas propostos e analisados neste artigo se enquadram no primeiro caso, mais especificamente, são CCAs do TCP (*Transmission Control Protocol*). Tais algoritmos de controle de congestionamento ajustam dinamicamente a taxa de transmissão dos dados para evitar congestionamento na rede, incrementando e decrementando a janela de congestionamento (*Congestion Window* - CWND). Além disso, a aplicação de um CCA eficiente pode auxiliar na resolução do desafio oriundo da necessidade de recuperação rápida quando ocorrem perdas de pacotes nas redes de nova geração, tais como redes móveis 5G e 6G, redes em malha sem fio e satelitais de baixa órbita, e nas redes de alta velocidade. Dessa forma, é possível proporcionar suporte a aplicações sensíveis a atrasos e com alta demanda de largura de banda, como videoconferências e jogos *online*.

Este artigo propõe três novas variantes para o controle de congestionamento do TCP, denominadas: Linear, Fracionário e Expolinear. Tais CCAs utilizam informações sobre perdas de pacotes para tomar decisões e distinguem-se, entre si e dos algoritmos disponíveis, na forma como reduzem e incrementam a CWND, cada um utilizando uma função diferente. O Expolinear e o Linear usam o tempo de ida e volta (RTT) nas funções

[1]Disponível em `https://www.ericsson.com/en/reports-and-papers/mobility-report/dataforecasts/mobile-traffic-update`.

[2]Disponível em `https://explodingtopics.com/blog/mobile-internet-traffic`.

de incremento, que são, respectivamente, exponenciais e lineares. Enquanto o valor adicionado à CWND pelo Fracionário independe do RTT, dado que esse sempre adiciona uma fração pré-estabelecida da atual CWND. Outra diferença para os demais algoritmos é a maneira como os limites e as reduções na janela de congestionamento são tratados.

Trabalhos anteriores apontam que não existe uma versão de CCA que obtenha a melhor desempenho possível em todos os casos. No entanto, a proposta apresentada neste artigo supera as limitações das abordagens existentes ao oferecer uma melhor reação às perdas de pacotes, equilibrando eficiência e equidade entre fluxos. Os algoritmos propostos foram implementados como módulos no *kernel* do Linux e foram submetidos a testes em um ambiente virtualizado. Os resultados demonstram que os CCAs propostos atingem valores dez vezes superiores aos dos modelos existentes em cenários de perda de pacotes, sem comprometer a justiça no compartilhamento de recursos.

O restante do artigo está organizado da seguinte forma. A Seção 2 elenca os trabalhos relacionados. O problema de definição do algoritmo de controle de congestionamento mais adequado a cada cenário é analisado na Seção 3. Os algoritmos de controle de congestionamento são propostos na Seção 4. A Seção 5 avalia a proposta e discute os resultados obtidos. A Seção 6 conclui o trabalho.

2. Trabalhos Relacionados

Diversas estratégias têm sido propostas na literatura para mitigar o congestionamento em redes de computadores. Algumas abordagens incluem a expansão de recursos físicos, como a substituição de cabos coaxiais por fibras ópticas, que aumentam a capacidade de transmissão. Outras estratégias se concentram no aprimoramento de técnicas de multiplexação e controle de acesso [Budhiraja et al., 2021], bem como no uso de roteamento dinâmico para reorganização das rotas em tempo real [Zhang e Yeh, 2024]. No contexto de dispositivos como roteadores e comutadores, o controle de congestionamento pode ser implementado através de controladores que utilizam informações provenientes desses dispositivos e/ou operam dentro deles [Li et al., 2024]. Essa abordagem, embora exija dispositivos programáveis capazes de fornecer dados específicos, oferece vantagens significativas, como a capacidade de ajustar dinamicamente as condições da rede em tempo real e evitar gargalos.

Abordagens específicas também foram investigadas em estudos recentes. A utilização de *buffers* de crédito tem demonstrado eficácia na gestão do congestionamento [Cho et al., 2017]. Outro estudo sugere a aplicação de controladores que operam em intervalos sub-RTT, proporcionando respostas mais rápidas a flutuações de tráfego [Arslan et al., 2023]. Adicionalmente, o uso de informações detalhadas sobre os estados de fila ajuda a prever e prevenir congestionamentos [Addanki et al., 2022]. Uma abordagem alternativa consiste na redução do volume de informações transmitidas, ajustando a taxa de compressão de acordo com as condições da rede. Essa técnica, que pode incluir a perda controlada de dados, mostra-se eficaz na redução do congestionamento, em cenários específicos, sem comprometer significativamente a experiência do usuário [Akhtar et al., 2018, Hegde et al., 2023].

Entre os algoritmos de controle de congestionamento utilizados em dispositivos finais, destacam-se algumas variantes importantes do protocolo TCP. O BBR (*Bottleneck Bandwidth and Round-trip propagation time*) é um algoritmo inovador baseado em esti-

mativas da largura de banda máxima disponível e da menor latência de propagação na rede [Cardwell et al., 2016]. Ele ajusta dinamicamente a janela de congestionamento (*Congestion Window* - CWND) para maximizar o uso da largura de banda sem sobrecarregar a rede. Diferentemente dos algoritmos tradicionais que dependem do monitoramento de perdas de pacotes, o BBR determina sua taxa de transmissão ideal com base em métricas objetivas. O BBR mantem uma taxa estável e eficiente, sendo especialmente útil em redes de alta velocidade e baixa variabilidade de congestionamento. O BBR, entretanto, enfrenta desafios em cenários em que há rápida variação da capacidade de rede.

O TCP Cubic adota uma abordagem baseada em uma função cúbica para substituir o crescimento linear tradicional da CWND [Ha et al., 2008]. Após um evento de perda, o algoritmo utiliza um crescimento côncavo para estabilizar a utilização da rede ao redor do valor máximo anterior. Em seguida, adota um crescimento convexo para explorar capacidades adicionais da rede. Essa característica garante que o TCP Cubic seja altamente escalável e eficiente, especialmente em redes de alta latência e longa distância. Além disso, o algoritmo é independente do tempo de ida e volta (RTT), proporcionando equidade entre fluxos com diferentes RTTs. Essa independência e sua capacidade de adaptação tornaram o TCP Cubic o padrão no *kernel* Linux desde 2006, consolidando sua posição como uma solução robusta e moderna.

O TCP Reno é uma das variante mais antiga que utiliza mecanismos de retransmissão rápida (*fast retransmit*) e recuperação rápida (*fast recovery*) para gerenciar congestionamentos [Padhye et al., 2000]. Quando três ACKs duplicados são recebidos, o Reno reduz a CWND pela metade e entra na fase de recuperação rápida, estabilizando a transmissão sem sobrecarregar a rede. Em casos de estouro de temporizadores do TCP (*timeouts*), a CWND é reiniciada para um segmento, e o crescimento volta à fase de início lento (*slow start*), aumentando a CWND exponencialmente até alcançar o limite seguro. Embora eficaz em cenários estáveis, o TCP Reno enfrenta limitações em redes com altas taxas de perda ou RTTs elevados, como redes móveis, em que estouros de temporizadores frequentes podem degradar significativamente o desempenho.

O TCP Vegas introduz avanços significativos no controle de congestionamento ao priorizar a prevenção de congestionamentos antes que perdas ocorram [Brakmo et al., 1994]. Ele utiliza cálculos precisos do RTT para monitorar atrasos e ajustar proativamente a CWND. O Vegas evita o crescimento exponencial descontrolado da janela ao medir a diferença entre a taxa de transmissão esperada e a observada, ajustando a CWND de forma mais granular. Além disso, o algoritmo melhora o mecanismo de retransmissão ao identificar perdas de pacotes com base no RTT em vez de depender apenas de ACKs duplicados. Essas melhorias tornam o TCP Vegas mais eficiente em condições variáveis, com maior vazão e menos retransmissões em relação ao TCP Reno.

Mais recentemente, o aprendizado de máquina tem sido integrado ao controle de congestionamento, com soluções como o TCP ex Machina [Winstein e Balakrishnan, 2013] e o Sage [Yen et al., 2023], que aprimoram algoritmos existentes. Abordagens baseadas em aprendizado por reforço, como o Gemini [Yang et al., 2023] e o TCP-Drinc [Xiao et al., 2019], permitem adaptações dinâmicas a redes modernas, aumentando a eficiência e a robustez, ao custo de maior processamento para o cálculo da janela de congestionamento e monitoramento de outras variáveis.

Embora os trabalhos citados ofereçam soluções robustas para diferentes cenários, a proposta deste estudo apresenta uma abordagem inovadora ao integrar aspectos clássicos, como as dinâmicas de crescimento e retração da janela de congestionamento, com técnicas modernas baseadas em métricas mais precisas e aprendizado de máquina. Essa combinação permite maior flexibilidade e eficiência em redes altamente variáveis, garantindo que os algoritmos possam responder rapidamente a mudanças nas condições da rede. Além disso, o estudo enfatiza a otimização da granularidade dos ajustes da CWND e a previsibilidade nas respostas, abordando desafios específicos, como a adaptação em tempo real e a redução de perdas, ainda não completamente resolvidos pela literatura existente.

3. Desafios para Controle de Congestionamento em Redes de Nova Geração

A transmissão eficiente de dados em redes de computadores exige o envio do máximo possível de informações, utilizando toda a largura de banda disponível. Contudo, os canais de comunicação geralmente são compartilhados por diversos dispositivos, não sendo dedicados a uma única conexão. Isso resulta em problemas como perda de pacotes quando vários usuários tentam consumir simultaneamente toda a capacidade do canal. Portanto, é imprescindível o desenvolvimento de mecanismos que gerenciem o uso do canal para maximizá-lo e assegurar o acesso equitativo entre os dispositivos/usuários.

Uma solução amplamente utilizada para enfrentar esses desafios é o algoritmo de controle de congestionamento (*Congestion Control Algorithm* - CCA) no protocolo TCP, que considera o princípio fim a fim da Internet e estabelece que as funções mais complexas devem ser executadas nos dispositivos localizados nas extremidades da rede [Saltzer et al., 1984]. Seguindo esse princípio, os algoritmos de controle de congestionamento do protocolo TCP são responsáveis pelo gerenciamento realizado pelas máquinas nas pontas da rede.

Essa abordagem possui vantagens significativas, como a independência do restante da rede. O funcionamento dos algoritmos não exige que dispositivos intermediários executem operações específicas de um protocolo entre o remetente e o receptor. No entanto, também apresenta desafios, como a necessidade de cooperação entre máquinas concorrendo pelo mesmo recurso, a dificuldade em interpretar corretamente as condições da rede e o tempo de resposta aos eventos de congestionamento.

Os algoritmos de controle de congestionamento devem permitir a coexistência amigável entre usuários, compartilhando igualmente a banda disponível, salvo em casos específicos de divisão ponderada. Além disso, o uso da rede deve tender ao equilíbrio, independentemente do momento em que os fluxos de dados foram iniciados. Esses requisitos são características fundamentais dos algoritmos do TCP.

Para aproveitar os recursos da rede, é essencial que o remetente interpele periodicamente o estado do canal, decidindo entre aumentar, reduzir ou manter sua taxa de transmissão. Essa tarefa é dificultada pela visão limitada que cada dispositivo possui, baseada apenas em métricas locais como o tempo de ida e volta dos pacotes (*Roud-Trip Time* - RTT), recebimento de confirmações (*Acknowledgements* - ACKs) e tamanho da janela de congestionamento (*Congestion Window* - CWND). Informações adicionais, como o número de fluxos em um enlace ou a sinalização de início de novas conexões, seriam benéficas para a tomada de decisões, mas não são informações locais às pontas da rede.

Os algoritmos baseados na perda de pacotes utilizam a técnica de aumentar a CWND quando não há perdas, ampliando o uso do canal, e de reduzi-la quando perdas são detectadas. Apesar de eficientes, esses métodos possuem limitações, como a ocupação da rede por pacotes de teste e o tempo necessário para ajustar a CWND ao valor ideal.

O tempo de resposta é um aspecto crítico, pois os algoritmos reagem às condições da rede de maneira retrospectiva. Idealmente, um algoritmo de controle de congestionamento seria capaz de prever as condições da rede e ajustar a taxa de transmissão em um intervalo de tempo τ tendendo a zero. No entanto, devido à imprevisibilidade do comportamento da rede, isso é um desafio prático. A detecção de eventos, como perdas de pacotes ou variações no RTT, e o ajuste da CWND requerem tempo, tornando necessário otimizar esses processos para minimizar atrasos.

A taxa de crescimento da CWND também impacta o desempenho. Aumentos rápidos podem acelerar a obtenção do valor ideal, mas comprometem a granularidade dos ajustes. Por outro lado, reduções excessivas da CWND podem subutilizar a rede, enquanto ajustes pequenos podem ser insuficientes para evitar congestionamento. Além disso, a diferenciação entre perdas causadas por erros de transmissão e aquelas resultantes de novos fluxos na rede representa outro desafio significativo.

A ausência de um controle centralizado que coordene os participantes é uma limitação adicional. Um sistema centralizado poderia, por exemplo, ajustar a CWND de forma otimizada para maximizar a utilização da rede, promover equidade e mitigar conflitos entre os fluxos de dados. Essas limitações destacam a complexidade do problema de controle de congestionamento e reforçam a necessidade de desenvolver algoritmos mais eficientes e adaptativos.

4. Algoritmos Propostos de Controle de Congestionamento

O desenvolvimento dos diferentes algoritmos de controle de congestionamento (*Congestion Control Algorithm* - CCA) focam dispositivos que tomam decisões baseadas em informações relacionadas à perda de pacotes. Esses algoritmos implementam as etapas de partida rápida, recuperação e prevenção de congestionamento, diferenciando-se na taxa de crescimento da janela de congestionamento (*congestion window* - CWND) em cada fase. A CWND é uma variável utilizada pelos algoritmos de controle de congestionamento para definir o número máximo de segmentos que podem ser enviados sem confirmação (ACK) no protocolo TCP, representando, assim, a capacidade atual estimada da conexão. Essa abordagem permite explorar estratégias distintas para otimizar o desempenho em redes de computadores.

O artigo apresenta o desenvolvimento de três modelos teóricos de controle de congestionamento, sendo eles os modelos linear, fracionário e expolinear, cada um com características específicas para gerenciar a janela de congestionamento. O primeiro modelo, denominado linear, utiliza funções matemáticas lineares para gerenciar o incremento da CWND, diferenciando-se na partida rápida pelo uso de uma função cúbica. Durante essa fase, o crescimento da CWND é definido por

$$J_{novo} = \alpha \times t^3, \tag{1}$$

em que J_{novo} é o novo valor da CWND, α é uma constante que determina a velocidade do

aumento, e t é o tempo decorrido desde o início da sessão ou do último evento de perda de pacotes. Considerar o tempo decorrido permite minimizar o impacto de atrasos elevados no enlace, aproveitando melhor a vazão da rede.

Na fase de prevenção de congestionamento, o crescimento da janela é dado por

$$J_{novo} = b_A + \beta_A \times t, \tag{2}$$

em que b_A representa o valor inicial da CWND na fase de prevenção, e β_A é uma constante que influencia a taxa de crescimento. Essa abordagem linear garante controle gradual e previsível da CWND.

A redução da CWND ocorre em resposta às perdas de pacotes, podendo ser fixa ou adaptativa. A redução fixa diminui a janela em valores predefinidos (10%, 20% ou 50%), enquanto a redução adaptativa depende da quantidade de pacotes perdidos, variando entre 15% e 50%. A dinâmica adaptativa é dada por

$$R = 15 + \frac{pkts_{err}}{pkts_{env}} \times 35, \tag{3}$$

em que R é a percentagem de redução, $pkts_{err}$ o número de pacotes perdidos, e $pkts_{env}$ o total de pacotes enviados na janela atual. Esse mecanismo assegura uma resposta proporcional à gravidade do congestionamento.

Durante a recuperação, novas perdas podem desencadear reduções adicionais na CWND e em seu limite superior, que é redefinido para o valor atual da janela no momento da identificação da perda. O crescimento da CWND na recuperação segue

$$J_{novo} = b_r + \beta_r \times t, \tag{4}$$

em que b_r é o valor inicial da CWND na fase de recuperação e β_r é uma constante que define a velocidade de incremento. Esse modelo linear está ilustrado na Figura 1(a), evidenciando o comportamento teórico da CWND.

O segundo modelo, chamado fracionário, ajusta a CWND com base em frações de seu valor atual, diferindo do modelo linear nas fases de recuperação e prevenção de congestionamento. Na recuperação, o incremento é descrito por

$$J_{novo} = J_{atual} + \gamma_r \times \Delta_r, \tag{5}$$

em que J_{atual} é o valor atual da CWND, $\gamma_r \in \{0, 4, 5, 1\}$ é o multiplicador de Δ_r, e Δ_r é igual a 10% da diferença entre os valores da CWND antes e depois da redução causada por uma perda, pois os incrementos são múltiplos de 10. Embora Δ_r possa ser integrado a γ_r, ao manter a definição de Δ_r explícita, há maior flexibilidade para alterar o comportamento da função, permitindo ajustes finos. Enquanto isso, na prevenção, o incremento segue

$$J_{novo} = J_{atual} \times (1 + \gamma_A), \tag{6}$$

em que γ_A indica a fração a ser acrescida. A Figura 1(b) ilustra o comportamento desse modelo.

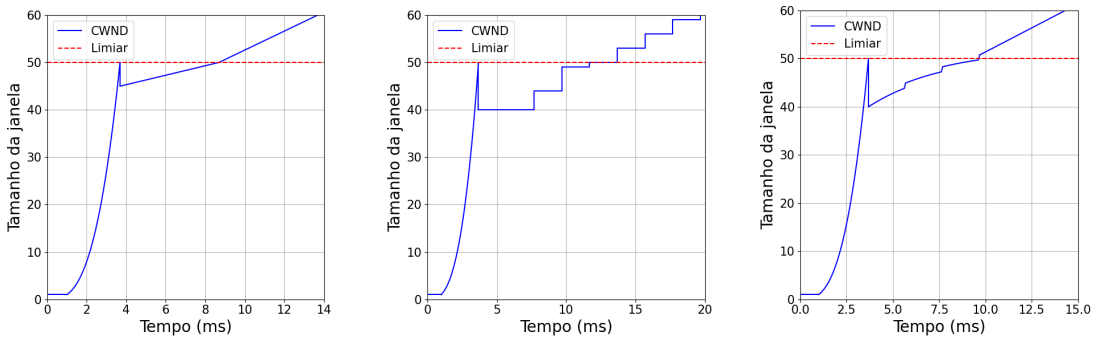
O modelo Expolinear combina funções exponenciais na recuperação e lineares na prevenção de congestionamento, mantendo as mesmas definições da partida rápida e do limite superior da recuperação. Na fase de recuperação, o ajuste segue

$$J_{novo} = b_r + \eta_r \times \Delta_r \times (1 - 2^{-\sigma t}) + 1, \tag{7}$$

em que η_r e σ são constantes que influenciam o ritmo de crescimento. Na prevenção, o incremento é descrito por

$$J_{novo} = b_A + \eta_A \times t, \tag{8}$$

em que η_A regula a velocidade de aumento. A Figura 1(c) destaca o comportamento desse modelo, evidenciando a combinação de funções que caracteriza o modelo Expolinear.



(a) Modelo linear. (b) Modelo fracionário. (c) Modelo expolinear.

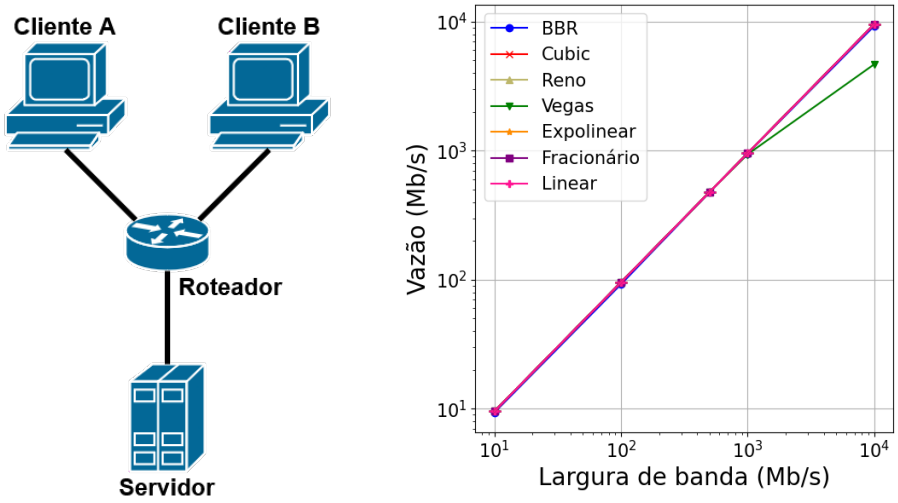
Figura 1. Representação do comportamento da CWND nos três modelos teóricos propostos. (a) Modelo linear realiza o crescimento cúbico durante a partida rápida. (b) Modelo fracionário executa ajustes granulares baseados em frações da CWND. (c) Modelo expolinear faz a combinação de funções exponenciais e lineares para adaptação dinâmica em diferentes condições de rede.

Cada modelo proposto apresenta vantagens para diferentes condições de rede. O modelo linear se destaca por sua simplicidade e previsibilidade, sendo adequado para redes com baixa variabilidade de atraso e perdas. O modelo fracionário oferece maior flexibilidade e capacidade de adaptação em cenários com flutuações moderadas de congestionamento, garantindo ajustes mais granulares. Por fim, o modelo Expolinear é ideal para redes dinâmicas e altamente voláteis, onde a combinação de crescimento exponencial e linear permite uma resposta rápida e eficiente a mudanças drásticas no estado da rede.

5. Avaliação e Resultados

Os testes foram realizados utilizando máquinas virtuais configuradas com o sistema operacional Ubuntu Server 24.04.1. Essas VMs foram interconectadas por pontes virtuais do Linux, enquanto o hipervisor KVM (Kernel-based Virtual Machine) gerenciava a virtualização. O computador hospedeiro das VMs possuía 20 GB de RAM, um

processador Intel Core i3 de 11ª geração, um SSD e o mesmo sistema operacional das VMs. O código dos algoritmos de controle de congestionamento propostos nesta pesquisa foi implementado em linguagem C e está disponível em um repositório público no GitHub[3], garantindo reprodutibilidade e acesso à implementação.



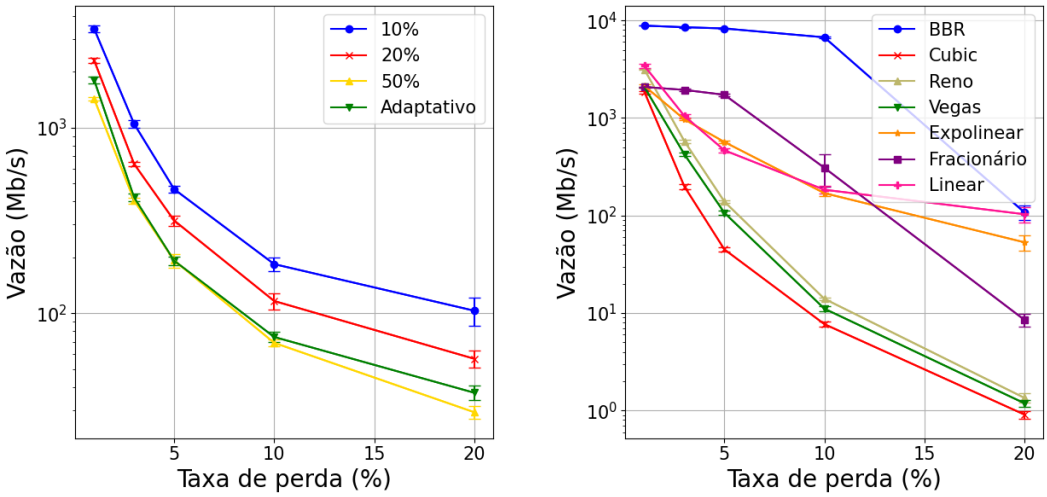
(a) Topologia da rede de testes. (b) Efeito da largura de banda nos CCAs.

Figura 2. (a) A topologia da rede de testes é composta por máquinas virtuais interconectadas via pontes virtuais no Linux, com controle de tráfego configurado no roteador. (b) O desempenho dos algoritmos de controle de congestionamento em função da largura de banda disponível. Os CCAs testados utilizam aproximadamente 96% da largura de banda disponibilizada.

O objetivo dos testes é avaliar o desempenho dos algoritmos de controle de congestionamento propostos (*Congestion Control Algorithms* - CCAs) em diferentes cenários de rede através da variação de valores de atraso, taxa de perda e largura de banda, comparando-os com os algoritmos BBR, Cubic, Reno e Vegas. Estes algoritmos foram selecionados devido às suas diferentes abordagens de controle de congestionamento: enquanto Cubic e Reno utilizam perda de pacotes como métrica, BBR se baseia na taxa de transferência e Vegas utiliza o RTT. Todos os experimentos são executados com intervalo de confiança de 99% e consistem em dez rodadas por cenário, garantindo robustez estatística e minimização de variações aleatórias nos resultados.

No experimento para avaliar o impacto da largura de banda, utilizou-se o roteador da topologia mostrada na Figura 2(a) para limitar a capacidade do enlace, configurando valores específicos de largura de banda com no subsistema de controle de tráfego do Linux (*Traffic Control* - tc) e *Network Emulator* (netem) [Hemminger et al., 2005]. Em cada teste, uma máquina cliente, configurada com um CCA específico, conecta-se ao servidor por meio da ferramenta *iperf*[4], simulando tráfego TCP por 20 segundos. Os resultados desse experimento, apresentados na Figura 2(b), indicam que, à medida que a largura de banda aumenta, todos os CCAs testados, exceto Vegas em um dos pontos, conseguem se adaptar e utilizar aproximadamente 96% da capacidade disponível. O desempenho abaixo

[3]Disponível em https://github.com/marcosvcm/CCAChikorita.
[4]Disponível em https://iperf.fr/.



(a) Efeito da perda de pacotes no modelo linear e suas versões.

(b) Efeito da perda de pacotes nos CCAs.

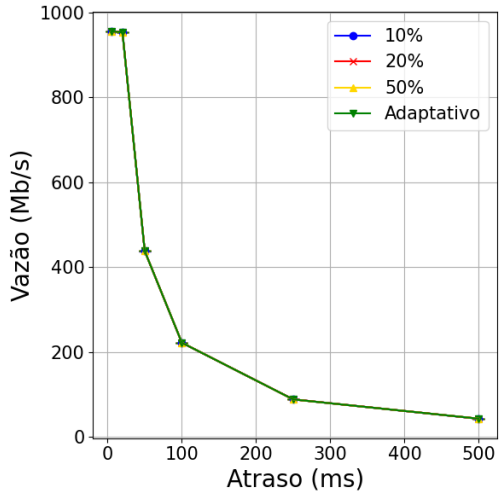
Figura 3. (a) Desempenho das versões do modelo linear sob diferentes taxas de perda de pacotes, com destaque para a redução fixa de 10% como a mais eficiente. (b) Comparação dos CCAs sob perda de pacotes, evidenciando o BBR como o mais robusto e os modelos propostos superando Cubic, Reno e Vegas em perdas acima de 3%.

da taxa máxima de 1 Gb/s pode ser atribuído ao tempo necessário para a fase de partida lenta ou a imprecisões no limite imposto pelo roteador.

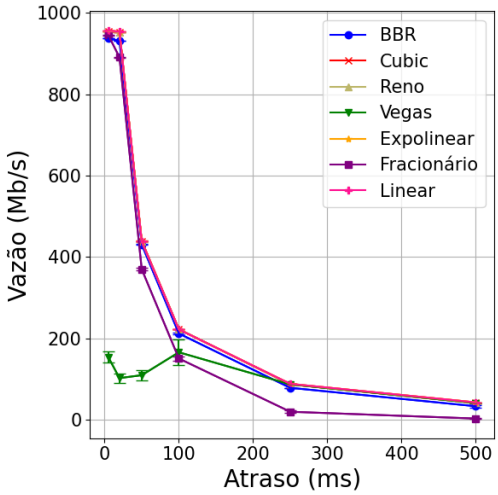
No experimento sobre perdas de pacotes, um enlace de 10 Gb/s foi configurado entre o roteador e o servidor, simulando perdas aleatórias em percentuais de 1%, 3%, 5%, 10% e 20%. Os CCAs foram avaliados individualmente com tráfego TCP gerado pelo iperf, com duração de 100 segundos por rodada. As Figuras 3(a) e 3(b) mostram, respectivamente, os resultados para diferentes versões do modelo linear proposto e para os CCAs finais. O modelo linear com redução fixa de 10% demonstrou melhor adaptação às perdas devido à menor diminuição da janela de congestionamento. O modelo adaptativo apresentou comportamento semelhante à redução fixa de 50%, sugerindo que o ajuste dinâmico aplicado frequentemente convergiu para valores próximos a essa proporção.

A análise dos CCAs nas Figuras 3(b) destaca o desempenho superior do BBR em cenários com perda de pacotes inferior a 20%, demonstrando que seu método baseado em taxa de transferência é mais robusto que os algoritmos tradicionais. Entre os CCAs propostos, o modelo fracionário superou os demais em perdas de 3%, 5% e 10%, embora os resultados para perdas de 10% mostrem sobreposição com o modelo linear. Em perdas de 20%, o modelo linear alcançou desempenho semelhante ao BBR, demonstrando competitividade em cenários de alta perda.

Para avaliar os efeitos do atraso, a largura de banda foi fixada em 1 Gb/s, e diferentes valores de atraso (5 ms a 500 ms) foram simulados entre o roteador e o servidor. Os gráficos nas Figuras 4(a) e 4(b) mostram que o atraso impacta negativamente a vazão de todos os CCAs. O modelo fracionário apresentou maior sensibilidade, pois sua adaptação da janela de congestionamento não considera o tempo de transmissão. Em contraste, os modelos linear e expolinear ajustam a janela proporcionalmente ao RTT, resultando em uma curva de desempenho mais gradual.

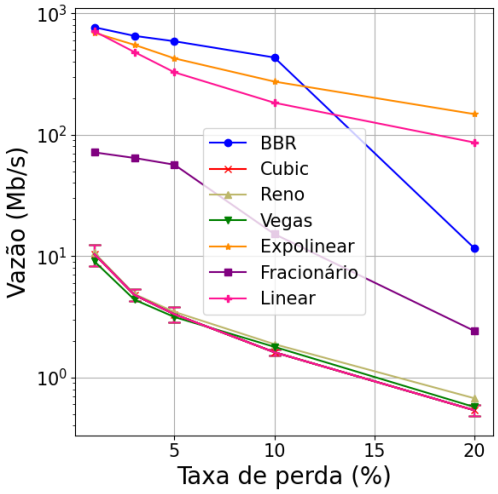


(a) Efeito do atraso no modelo linear e suas versões.

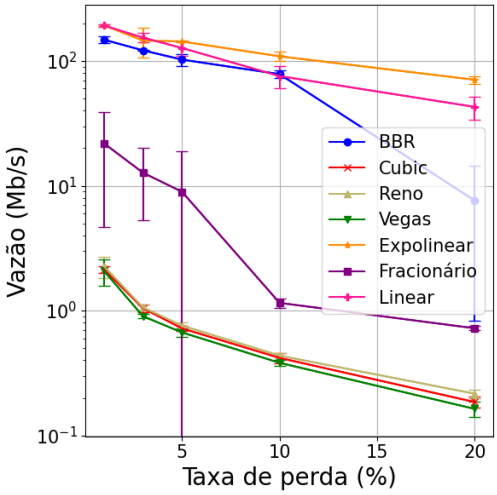


(b) Efeito do atraso nos CCAs.

Figura 4. (a) Redução exponencial da vazão das versões do modelo linear com o aumento do atraso, evidenciando sensibilidade ao RTT. (b) Impacto do atraso nos CCAs. O fracionário foi mais afetado e os linear e expolinear mostram melhor adaptação em altos RTTs devido aos ajustes proporcionais ao tempo.



(a) Efeito da perda de pacotes nos CCAs considerando um RTT de 20 ms.



(b) Efeito da perda de pacotes nos CCAs considerando um RTT de 100 ms.

Figura 5. Impacto da taxa de perda de pacotes na vazão de diferentes CCAs para atrasos de rede de (a) 20 ms e (b) 100 ms. O algoritmo Expolinear obtém melhores resultados em cenários com altas taxas de perda e atrasos elevados.

Conexões na Internet frequentemente experimentam atrasos e perdas de pacotes simultaneamente. Diante disso, o próximo experimento verifica o comportamento de diversos algoritmos de controle de congestionamento (CCAs) em cenários com taxas de perda de pacotes variáveis e atrasos fixos de 20 ms e 100 ms. Para tanto, foi configurado um enlace de 1 Gb/s entre um roteador e um servidor, simulando perdas aleatórias de pacotes em taxas de 1%, 3%, 5%, 10% e 20%. Cada CCA foi submetido a testes individuais utilizando tráfego TCP gerado pelo *iperf*, com duração de 50 segundos por rodada de teste. As Figuras 5(a) e 5(b) apresentam os resultados obtidos para os diferentes CCAs

Algoritmo	Expolinear	Fracionário	Linear
BBR	1,56 ± 0,302	2,11 ± 0,030	1,26 ± 0,166
Cubic	1,02 ± 0,001	0,96 ± 0,025	1,02 ± 0,001
Reno	1,01 ± 0,001	0,91 ± 0,024	1,01 ± 0,013
Vegas	21,02 ± 2,049	36,75 ± 13,777	20,74 ± 2,276
Expolinear	1,00 ± 0,009	0,93 ± 0,017	1,01 ± 0,000
Fracionário	1,02 ± 0,031	1,07 ± 0,013	1,13 ± 0,001
Linear	1,01 ± 0,001	0,93 ± 0,015	1,01 ± 0,001

Tabela 1. Cordialidade entre os modelos desenvolvidos e os demais CCAs. Utilizando um servidor com BBR.

Algoritmo	Expolinear	Fracionário	Linear
BBR	1,16 ± 0,013	2,13 ± 0,042	1,17 ± 0,006
Cubic	1,00 ± 0,009	0,95 ± 0,025	1,02 ± 0,011
Reno	1,01 ± 0,001	0,93 ± 0,019	1,01 ± 0,011
Vegas	23,44 ± 5,212	27,90 ± 5,365	22,97 ± 3,560
Expolinear	1,01 ± 0,000	0,92 ± 0,020	1,01 ± 0,011
Fracionário	1,13 ± 0,002	0,91 ± 0,034	1,13 ± 0,001
Linear	1,01 ± 0,001	0,91 ± 0,021	1,00 ± 0,006

Tabela 2. Cordialidade entre os modelos desenvolvidos e os demais CCAs. Utilizando um servidor com Cubic.

em cada cenário de atraso.

A Figura 5(a) demonstra a superioridade do algoritmo BBR em cenários com taxas de perda inferiores a 10%, evidenciando a robustez de seu mecanismo de controle baseado em taxa de transferência. No entanto, em condições de perda mais severas (20%), o algoritmo Expolinear se destaca, alcançando uma vazão média superior em mais de uma ordem de magnitude ao BBR. O algoritmo Linear também apresentou bom desempenho nesse cenário, consolidando a eficácia das propostas Expolinear e Linear em condições adversas. Por outro lado, o algoritmo Fracionário apresentou o pior desempenho entre os algoritmos avaliados, embora ainda superior aos CCAs tradicionais (Cubic, Reno e Vegas). A Figura 5(b) revela a superioridade dos algoritmos Expolinear e Linear em todas as taxas de perda avaliadas, com o Expolinear apresentando o melhor desempenho na maioria dos casos. O algoritmo BBR apresentou desempenho similar ao Expolinear em algumas situações, mas foi superado pelo Expolinear e Linear na maior parte dos cenários, quando as taxas de perda são superiores a 5%.

Os resultados obtidos indicam que o algoritmo Expolinear apresenta a maior vazão entre todos os algoritmos avaliados em cenários com altas taxas de perda ou atrasos elevados, como aqueles comumente encontrados em comunicações intercontinentais ou redes de acesso sem fio com alta carga e alto número usuários conectados em um mesmo ponto de acesso. Esses resultados demonstram a adequação do algoritmo Expolinear para aplicações que demandam alto desempenho em redes com condições adversas.

A justiça entre fluxos concorrentes foi analisada configurando o servidor com BBR ou Cubic e testando diferentes combinações de CCAs nos clientes A e B (Figura 2(a)). As Tabelas 1 e 2 apresentam os índices de cordialidade calculados como $x = a/b$,

em que x é o resultado exposto nas tabelas, a é a vazão registrada pelo cliente aparamentado com o CCA correspondente a coluna da tabela (alvo dos testes) e b é a vazão mensurada pelo cliente com o CCA correspondente a linha. Tem-se que quanto menor o ϵ, onde $\epsilon = |1 - x|$, mais justo, amigável, é o controlador, mais próximos um do outro são os valores de a e b. Os resultados mostram que, com exceção do Vegas, os CCAs propostos demonstram níveis satisfatórios de justiça em relação aos algoritmos existentes. A falta de justiça com o Vegas reflete a natureza do algoritmo, que ajusta sua janela de congestionamento com base no RTT, tornando-o suscetível a conflitos.

6. Conclusão

Este trabalho apresentou três novos algoritmos de controle de congestionamento para o TCP, denominados Expolinear, Fracionário e Linear, destacando suas vantagens em cenários com perda de pacotes. Os resultados experimentais demonstraram que os algoritmos propostos não apenas coexistem de forma cordial com outros CCAs, mas também proporcionam uma distribuição de banda mais equitativa e uma vazão significativamente superior, alcançando, em alguns casos, até dez vezes mais desempenho do que os algoritmos tradicionais. Além disso, os novos modelos apresentaram uma resposta ao atraso comparável aos algoritmos existentes, evidenciando sua adaptabilidade em diferentes condições de rede. Esses resultados confirmam o potencial dos CCAs propostos como alternativas promissoras para redes de nova geração, em que eficiência e robustez são essenciais. Como trabalhos futuros, pretende-se ampliar a pesquisa para cenários de rede mais complexos e realistas, como redes de centros de dados e aplicações de navegação *web*, incluindo condições adversas de latência, variação de RTT e perda de pacotes.

Agradecimentos

Este trabalho foi realizado com recursos do CNPq, FAPERJ, RNP, CAPES, CGI/FAPESP (2018/23062-5) e Prefeitura de Niterói/FEC/UFF (Edital PDPA 2020).

Referências

Addanki, V., Apostolaki, M., Ghobadi, M., Schmid, S. e Vanbever, L. (2022). Abm: active buffer management in datacenters. Em *Proceedings of the ACM SIGCOMM 2022 Conference*, SIGCOMM '22, p. 36–52, New York, NY, USA. Association for Computing Machinery.

Akhtar, Z., Nam, Y. S., Govindan, R., Rao, S., Chen, J., Katz-Bassett, E., Ribeiro, B., Zhan, J. e Zhang, H. (2018). Oboe: auto-tuning video abr algorithms to network conditions. Em *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, SIGCOMM '18, p. 44–58, New York, NY, USA. Association for Computing Machinery.

Arslan, S., Li, Y., Kumar, G. e Dukkipati, N. (2023). Bolt: Sub-RTT congestion control for Ultra-Low latency. Em *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, p. 219–236, Boston, MA. USENIX Association.

Brakmo, L. S., O'Malley, S. W. e Peterson, L. L. (1994). Tcp vegas: new techniques for congestion detection and avoidance. Em *Proceedings of the Conference on Communications Architectures, Protocols and Applications*, SIGCOMM '94, p. 24–35, New York, NY, USA. Association for Computing Machinery.

Budhiraja, I., Kumar, N., Tyagi, S., Tanwar, S., Han, Z., Piran, M. J. e Suh, D. Y. (2021). A systematic review on noma variants for 5g and beyond. *IEEE Access*, 9:85573–85644.

Cardwell, N., Cheng, Y., Gunn, C. S., Yeganeh, S. H. e Jacobson, V. (2016). Bbr: Congestion-based congestion control. *ACM Queue*, 14, September-October:20 – 53.

Cho, I., Jang, K. e Han, D. (2017). Credit-scheduled delay-bounded congestion control for datacenters. Em *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, SIGCOMM '17, p. 239–252, New York, NY, USA. Association for Computing Machinery.

Conde, J., Martínez, G., Reviriego, P. e Hernández, J. A. (2024). Round trip times (rtts): Comparing terrestrial and leo satellite networks. Em *2024 27th Conference on Innovation in Clouds, Internet and Networks (ICIN)*, p. 42–46.

Ha, S., Rhee, I. e Xu, L. (2008). Cubic: a new tcp-friendly high-speed tcp variant. *SIGOPS Oper. Syst. Rev.*, 42(5):64–74.

Hegde, P., de Veciana, G. e Mokhtari, A. (2023). Network adaptive federated learning: Congestion and lossy compression. Relatório técnico, arXiv. Disponível em https://arxiv.org/abs/2301.04430.

Hemminger, S. et al. (2005). Network emulation with netem. Em *Linux conf au*, volume 5, p. 2005.

Li, W., Wu, Z., Zhang, B., Zheng, L., Zhao, W., Zou, J. e Sun, S. (2024). Design of congestion control algorithm based on switch queue length change detection. Em *2024 IEEE 30th International Conference on Telecommunications (ICT)*, p. 01–06.

Padhye, J., Firoiu, V., Towsley, D. F. e Kurose, J. F. (2000). Modeling tcp reno performance: a simple model and its empirical validation. *IEEE/ACM Trans. Netw.*, 8(2):133–145.

Saltzer, J. H., Reed, D. P. e Clark, D. D. (1984). End-to-end arguments in system design. *ACM Trans. Comput. Syst.*, 2(4):277–288.

Winstein, K. e Balakrishnan, H. (2013). Tcp ex machina: computer-generated congestion control. *SIGCOMM Comput. Commun. Rev.*, 43(4):123–134.

Xiao, K., Mao, S. e Tugnait, J. K. (2019). Tcp-drinc: Smart congestion control based on deep reinforcement learning. *IEEE Access*, 7:11892–11904.

Yang, W., Liu, Y., Tian, C., Jiang, J. e Guo, L. (2023). Gemini: Divide-and-conquer for practical learning-based internet congestion control. Em *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications*, p. 1–10.

Yen, C.-Y., Abbasloo, S. e Chao, H. J. (2023). Computers can learn from the heuristic designs and master internet congestion control. Em *Proceedings of the ACM SIGCOMM 2023 Conference*, ACM SIGCOMM '23, p. 255–274, New York, NY, USA. Association for Computing Machinery.

Zhang, J. e Yeh, E. (2024). Congestion-aware routing and content placement in elastic cache networks. Em *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications*, p. 1471–1480.