ESCALONAMENTO DE TAREFAS COMUNICANTES CONSIDERANDO O CONTROLE E A PREVENÇÃO DE CONGESTIONAMENTOS EM REDES

Enzo Belló Boscatto, Anderson Henrique da Silva Marcondes, Guilherme Piêgas Koslovski

INTRODUÇÃO

Em sistemas de processamento distribuído modernos, como o Apache Spark, o desempenho das aplicações depende fortemente do equilíbrio entre a capacidade de processamento local e a eficiência da rede do datacenter. Durante operações de redistribuição de dados (*shuffle*), a saturação dos canais de comunicação introduz atrasos significativos que vão além da perda pontual de pacotes. Quando ocorre congestionamento severo na infraestrutura, os executores podem ser impedidos de avançar na computação por indisponibilidade momentânea de dados, caracterizando o fenômeno de *starvation* induzido por I/O. Nesse estado, a CPU permanece ociosa aguardando a chegada de fluxos de rede, resultando em subutilização de hardware e aumento no tempo total de execução. O objetivo deste trabalho consistiu em investigar e caracterizar o *starvation* em aplicações distribuídas, validar seu comportamento em ambientes virtuais e físicos e implementar mecanismos de monitoramento granular e marcação de pacotes que permitam atuar no controle de congestionamento orientado pela demanda da aplicação.

DESENVOLVIMENTO

Inicialmente, foi estruturado um cluster Apache Spark composto por 5 instâncias (1 master e 4 workers) sobre máquinas virtuais com Ubuntu Server, conectadas através de uma máquina virtual roteadora dedicada. Foi utilizada uma aplicação customizada com geração intensiva de pacotes pequenos para induzir o gargalo de rede durante o shuffle, comparando-a com aplicações padrão como o GroupByTest sob cenários com e sem tráfego de fundo, gerado via ferramenta iPerf. O nó roteador foi configurado com disciplinas de enfileiramento RED/HTB e protocolo DCTCP com suporte a ECN. Na camada de rede, foram coletadas métricas de backlog de fila (bytes e pacotes a cada 0,1 s) e, na camada de aplicação, foi acoplado um listener customizado ao Spark para extrair métricas de tarefas, como tempo de execução (Task Time), tempo efetivo de uso de CPU (CPU Time) e Flow Completion Time (FCT).

Em continuidade aos resultados obtidos na primeira etapa, a infraestrutura foi replicada nos servidores do Laboratório de Processamento Paralelo e Distribuído (LabP2D) da UDESC/CCT, preservando a lógica da aplicação e do cluster. Para permitir a intervenção direta no transporte, foi desenvolvido um script monitor para acompanhar a execução das tarefas e capturar o estado dos buffers de recepção. O monitor analisa as conexões ativas e identifica um possível evento de starvation quando três condições simultâneas são atendidas: (i) o volume acumulado excede o mínimo de 15 MB; (ii) a taxa de transmissão na janela recente cai abaixo do limiar de 5 MB; e (iii) o uso de CPU sofre uma queda para 30% da média recente. O monitor então realiza a marcação dos pacotes (SO_MARK) nos fluxos prováveis de sofrerem com o fenômeno de starvation. Essa marcação aciona o módulo de controle de rede, que ajusta dinamicamente a janela de congestionamento (cwnd) do TCP com base na métrica Starvation Ratio (SR), priorizando os fluxos críticos e evitando a ociosidade do processador.

RESULTADOS

Os testes no ambiente virtualizado validaram a hipótese de starvation induzido pela infraestrutura. Sob tráfego concorrente, o backlog das filas no roteador apresentou picos superiores a 700 kB, elevando drasticamente o Flow Completion Time (FCT) das operações de leitura e escrita conforme ilustrado na Figura 1. O tempo médio de execução da aplicação customizada saltou de 295,4s no

cenário isolado para 622,6s sob contenção de rede, um aumento superior a 110%. Já o tempo de utilização de processamento (CPU Time) permaneceu inalterado entre os cenários enquanto a duração da comunicação aumentou em ordens de magnitude, comprovando que a degradação decorreu da ociosidade da CPU à espera de pacotes de dados.

Com as máquinas físicas utilizando o monitor e a marcação de pacotes para controlar o starvation, no Spark com rede congestionada a abordagem reduziu o tempo total de execução em até 50 segundos (Figura 2) quando comparada aos protocolos TCP Reno e Cubic. Também, é possível notar que os resultados apresentam menores variações nos valores de tempo de execução e de rede, indicando maior estabilidade quando realizada a marcação dos pacotes.

CONSIDERAÇÕES FINAIS

O trabalho desenvolvido ao longo deste último ano de iniciação científica permitiu entender a fundo como os gargalos de rede afetam o processamento distribuído na prática. Foi possível constatar que os mecanismos tradicionais de controle de congestionamento são cegos para o estado da aplicação: eles apenas reduzem a velocidade da transmissão sem saber se a CPU está ficando desocupada à espera de pacotes. Os testes com Apache Spark mostraram que é possível recuperar a vazão nos momentos críticos da execução, reduzir o tempo total e garantir mais estabilidade nas tarefas.

Palavras-chave: processamento distribuído; apache spark; congestionamento de rede; starvation ratio; computação em nuvem.

ANEXOS

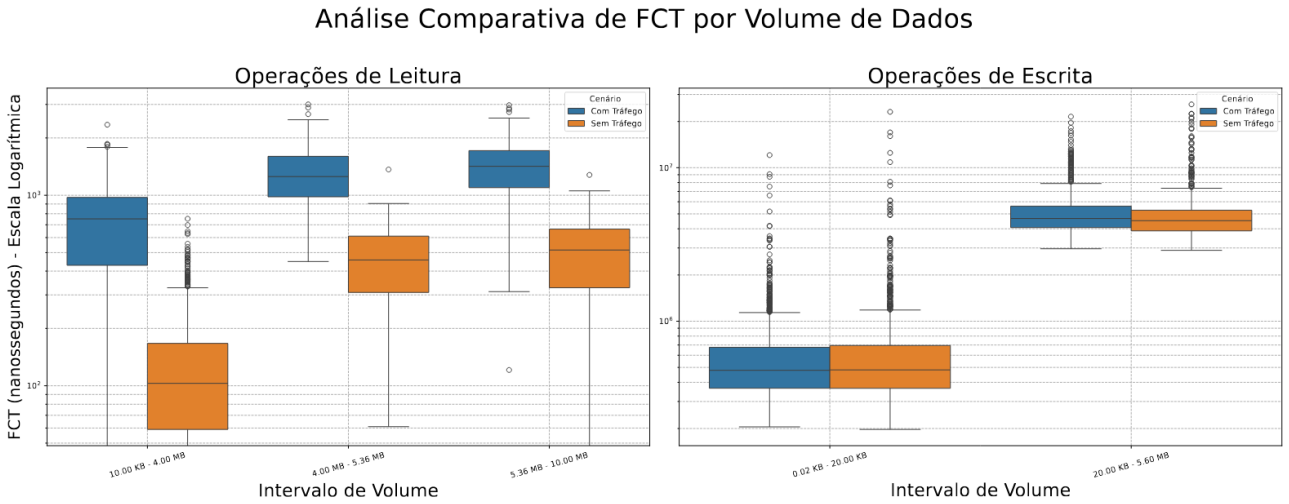


Figura 1. *Comparativo boxplot de FCT (ns) por intervalo de volume de dados (MB) das tarefas durante a execução da aplicação customizada.*

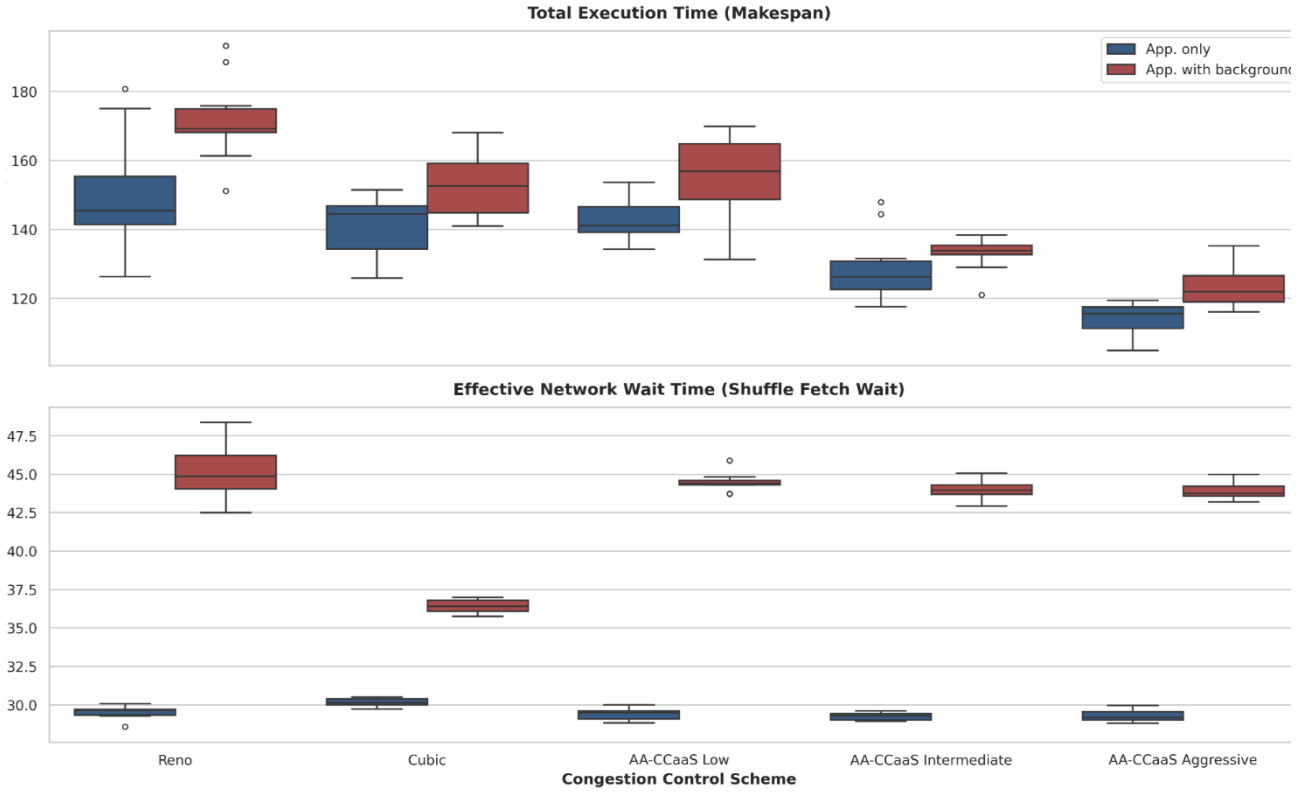


Figura 2. *Gráficos boxplot de média do Tempo Total de Execução (s) e Tempo de Espera Efetivo (s) da aplicação.*

REFERÊNCIAS BIBLIOGRÁFICAS

ALIZADEH, M. et al. **Data center TCP (DCTCP)**. In: ACM SIGCOMM CONFERENCE, 2010. Proceedings [...]. New York: ACM, 2010. p. 63-74.

DUKKIPATI, N.; MCKEOWN, N. **Why Flow-Completion Time is the right metric for congestion control**. ACM SIGCOMM Computer Communication Review, New York, v. 36, n. 1, p. 59-62, 2006.

SALLOUM, S. et al. **Big data analytics on Apache Spark**. International Journal of Data Science and Analytics, v. 1, n. 3, p. 145-164, 2016.

MARCONDES, A. et al. **Starvation ratio: letting applications drive datacenter congestion control**. In: SEMINÁRIO INTEGRADO DE SOFTWARE E HARDWARE (SEMISH), 53., 2026, Gramado/RS. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2026. p. 674-685. ISSN 2595-6205.