Melhorando a escalabilidade do mecanismo de consenso Committeeless Proof-of-Stake

Inácio Defilippi Gonçalves, Marco Amaral Henriques

Faculdade de Engenharia Elétrica e de Computação (FEEC)
Universidade Estadual de Campinas (Unicamp)
Campinas – SP – Brasil

i288637@dac.unicamp.br, maah@unicamp.br

Abstract. *The Committeeless Proof-of-Stake (CPoS) consensus mechanism eliminates the validation committees found in traditional Proof-of-Stake systems by relying on local lotteries and network monitoring. However, two issues hinder its adoption at larger scales: (i) the computation of the probability distribution function and (ii) the restriction of control parameters to integer values, which limits fine-grained consensus tuning. This work modifies and evaluates both issues, enabling operation in larger deployments and more precise calibration of the node selection rate per round. Experiments on a distributed cluster suggest that both approaches reduce congestion and improve block throughput.*

Resumo. *O mecanismo de consenso Proof-of-Stake Sem Comitê (CPoS - Committeeless Proof-of-Stake) elimina os comitês de validação do sistema Proof-of-Stake ao basear-se em sorteios locais e no monitoramento da rede. No entanto, ele tem dois problemas de escalabilidade: (i) o cálculo da função de distribuição de probabilidade e (ii) a restrição dos parâmetros de controle a valores inteiros, o que limita o ajuste fino do consenso. Este trabalho trata os dois problemas, permitindo a operação em escalas maiores e uma calibração mais precisa do número de nós produtores de blocos por rodada. Experimentos em um cluster distribuído sugerem que ambas as abordagens reduzem o congestionamento e melhoram a a taxa de vazão de blocos.*

1. Introdução

A tecnologia blockchain busca descentralizar a confiança digital por meio de mecanismos de consenso distribuído [Nakamoto 2009, Bashir 2020]. Para mitigar o alto consumo energético e limitações de escalabilidade do *Proof-of-Work* (PoW) [Antonopoulos 2014], a indústria tem transicionado para protocolos *Proof-of-Stake* (PoS) [King and Nadal 2012]. O PoS oferece maior eficiência energética e capacidade de vazão, o que motivou sua adoção por grandes redes públicas, como a Ethereum [Antonopoulos and Wood 2018, Buterin 2014]. Nesses sistemas, a elegibilidade para a criação de blocos associa-se diretamente ao capital (*stake*) alocado pelos nós.

Contudo, as implementações tradicionais de PoS frequentemente dependem da eleição de comitês de validação para atestar o estado da rede. Ainda que esse comitê

O presente trabalho foi realizado com apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq (Brasil), por meio de bolsa de Iniciação Científica (PIBIC). Nele foram utilizadas ferramentas de IA para revisão de texto e depuração de códigos.

não seja conhecido a priori para evitar seu comprometimento, seus membros tornam-se potenciais alvos centralizados, suscetíveis a ataques e corrupção. Visando mitigar essa vulnerabilidade, o protocolo *Committeeless Proof-of-Stake* (CPoS) propõe uma arquitetura que evita tais comitês. O mecanismo opera por meio de sorteios locais privados, com cada nó buscando o consenso de forma independente e probabilística após avaliar as mensagens de seus pares [Martins 2021].

A expectativa teórica do número de nós sorteados a cada rodada para propor um bloco é definida pelo parâmetro τ. Por tratar-se de uma expectativa média probabilística, esse parâmetro poderia assumir qualquer valor real. Contudo, a implementação original restringe-o aos números inteiros, o que limita o controle granular da rede. Configurações fixas em $\tau = 1$ podem induzir baixa produção de blocos, afetando a vazão operacional de transações. Por outro lado, valores de τ muito elevados podem produzir um número de nós sorteados muito maior que o necessário, dificultando o consenso e aumentando o tráfego de rede.

Para viabilizar a adoção do protocolo em redes de grande porte, este trabalho também aborda o problema de estouro numérico (*overflow*) no cálculo da função de distribuição acumulada (CDF). A refatoração matemática do sorteio para o espaço logarítmico permitiu uma computação estável da probabilidade, independentemente do tamanho da rede. Neste contexto, este artigo relata resultados preliminares da pesquisa em andamento focada na escalabilidade geral do mecanismo CPoS. O código base do protocolo encontra-se aberto e disponível publicamente. As otimizações matemáticas específicas detalhadas neste trabalho encontram-se disponíveis no repositório do projeto[1].

2. O gargalo do cálculo da distribuição binomial

A arquitetura do protocolo CPoS baseia-se na avaliação local de uma variável aleatória com distribuição $Binomial(total\ stake, p)$ [Grimmett and Stirzaker 2020]. Neste modelo, a probabilidade p é definida pela razão entre a expectativa global de nós sorteados τ e o *total stake* $= N$ (um *stake* para cada um dos N nós): $p = \tau/N$. O protocolo resolve as funções de sorteio e de limites de ramificação (*fork threshold*) por meio do cálculo da função de distribuição acumulada (CDF, Eq. 1).

$$F(k; N, p) = \text{Prob}(X \leq k) = \sum_{i=0}^{k} \binom{N}{i} \cdot p^i \cdot (1-p)^{N-i}. \tag{1}$$

Esse método gera um crescimento combinatório insustentável, resultando em estouro numérico (*OverflowError*) no padrão de ponto flutuante IEEE 754 quando $N > 34$ (precisão simples), $N > 170$ (precisão dupla) e $N > 1754$ (precisão quádrupla).

Para erradicar esse problema, este trabalho otimizou os métodos de cálculo. Uma primeira estratégia substituiu o somatório discreto pela função beta incompleta regularizada $I_{1-p}(N-k, k+1)$ [Press et al. 2007], via pacote `scipy.stats` da linguagem Python. Analiticamente, aplica-se a identidade beta-binomial [Grimmett and Stirzaker 2020], com a equivalência: $I_{1-p}(N-k, k+1) = \sum_{i=0}^{k} \binom{N}{i} p^i (1-p)^{N-i}$. A vantagem computacional é a avaliação inteiramente no espaço logarítmico via função gama. Ao evitar o cômputo de números combinatórios intermediários, a refatoração suprimiu os estouros

[1] https://github.com/regras/cpos_v2

na memória, permitindo o cálculo estável para N na ordem de milhões. Uma segunda estratégia buscou atacar outro problema: o algoritmo original realiza uma busca linear exaustiva de complexidade $O(N)$. A estrutura foi otimizada com a adoção da função quantil (*percent point function* - PPF) da referida distribuição. Por usar algoritmo iterativo de busca binária sobre a própria CDF no espaço logarítmico, reduziu-se a complexidade analítica do sorteio para $O(\log(N))$.

3. O hiato do domínio discreto

Além dos gargalos computacionais, a restrição original do parâmetro τ ao domínio dos números inteiros cria um dilema de calibração difícil de equalizar. Quando a rede opera com $\tau = 1$, a probabilidade de sorteio frequentemente resulta em rodadas sem nenhum nó sorteado. Isso causa escassez de propostas e gera ociosidade sistêmica, subutilizando a infraestrutura disponível. Por outro lado, o incremento para valores inteiros superiores eleva a variância do processo de sorteio. O problema central não reside na coexistência de nós sorteados simultâneos, mas sim na ocorrência de picos anômalos com dezenas de nós sorteados em uma mesma rodada. Sob a presença de cargas pesadas, tentar propagar blocos cheios de transações a partir de todos esses nós simultaneamente pode ultrapassar a largura de banda disponível. Esse fluxo maciço e redundante gera atrasos de propagação que desencadeiam ramificações, comprometendo a convergência da cadeia.

A solução desenvolvida consistiu em alterar boa parte dos algoritmos para o domínio contínuo dos números reais, permitindo explorar valores intermediários de τ com precisão de ponto flutuante. A flexibilização viabilizada por essa variação contínua tem potencial de mitigar simultaneamente a ociosidade e a sobrecarga.

4. Metodologia de avaliação e infraestrutura

Para avaliar o impacto do parâmetro τ contínuo na estabilidade da rede, desenvolveu-se uma infraestrutura de testes em ambiente distribuído. A rede foi orquestrada via *Docker Swarm* sobre um *cluster* de 22 máquinas físicas, sendo 19 com 20 vCPUs/32GB RAM e 3 com 8 vCPUs/8GB RAM, totalizando até 66 contêineres simultâneos, três por máquina. A camada de aplicação de cada nó executou o protocolo CPoS sob a linguagem Python (imagem `mwalbeck/python-poetry:1.5-3.11`), gerenciando seu banco de dados *MariaDB* local. Diferentemente de uma implantação confinada a um único hospedeiro, essa topologia expõe o protocolo a latências de rede reais entre máquinas fisicamente distintas, condição necessária para avaliar a degradação sob concorrência em cenários próximos de produção.

A execução dos testes foi automatizada por um *script shell* orquestrador, variando o parâmetro τ a cada 50 rodadas ininterruptas. Para assegurar a independência das iterações, o orquestrador sanitizou a infraestrutura entre execuções, recriando os contêineres e, com eles, os bancos de dados locais de cada nó. Para emular o tráfego de uma rede em produção, os testes injetaram 250 KiB de dados arbitrários em cada bloco proposto, simulando o conjunto de transações. Essa carga força a camada P2P a lidar com transferências intensas frente a múltiplos sorteios, realçando atrasos de propagação gerados perante falhas de calibração paramétrica. Apesar de $\approx$250 KiB ser um valor aproximado, ele representa um valor razoável com base nas estatísticas da rede Ethereum e não tem relação com a escolha do valor ideal de τ.

O desempenho foi extraído dos históricos de operação da rede. O indicador principal avaliado foi a vazão (*throughput*) da rede, medida como o comprimento da cadeia local de cada nó dividido pelo número de rodadas. Avaliou-se tambéma taxa de blocos efetivamente confirmados por rodada e a quantidade de ramificações estruturais (*forks* por rodada). Realizou-se também uma avaliação computacional isolada para validar a refatoração matemática do cálculo da distribuição binomial. O experimento mediu o tempo médio de processamento e a ocorrência de *OverflowError* para diferentes valores de *total stake* ($N \in [10, 10^6]$), considerando uma unidade de *stake* por nó.

5. Resultados e discussão

Tabela 1. Tempo de execução (ms) do cálculo da distribuição de probabilidade

Total Stake (N)	**Implementação Legada**	**Implementação Otimizada**
10	0.0031	0.0649
100	0.1560	0.0641
1.000	25.9338	0.0650
10.000	*OverflowError*	0.0653
100.000	*OverflowError*	0.0661
1.000.000	*OverflowError*	0.0680

Os resultados evidenciaram o gargalo estrutural da implementação legada. A rotina original apresenta interrupção por *OverflowError* a partir de $N = 10.000$, confirmando as limitações no processamento dos coeficientes binomiais. Ademais, o método legado exibe aumento expressivo (166 vezes) no tempo de execução ao aumentar N em 10 vezes, de 100 (0,156 ms) para 1.000 (25,93 ms), sugerindo impacto considerável ao manipular inteiros grandes. Em contrapartida, a versão otimizada baseada na função beta incompleta e busca binária (PPF) demonstrou estabilidade nas escalas avaliadas. Na faixa testada, o tempo da nova implementação permaneceu praticamente constante na marca de 0.065 ms, com baixo desvio padrão. Este comportamento mostra um custo muito baixo, dominado pelo *overhead* fixo da chamada da função em Python. Essa tendência sinaliza a viabilidade computacional do CPoS para cenários de maior escala, aproximando-se do comportamento esperado de $O(\log(total\ stake))$.

Para avaliar os efeitos da transição para o domínio contínuo sob uma infraestrutura de rede real, o cluster distribuído foi submetido a cenários de comunicação intensa, com τ variando de 1.0 a 7.0 e três repetições por configuração. A Figura 1 mostra que a vazão cresce de 0.48 ± 0.12 blocos por rodada em $\tau = 1.0$ (regime de ociosidade) para um patamar entre 0.95 e 1.0 blocos por rodada entre $\tau \approx 3.0$ e $\tau = 5.5$, próximo do máximo teórico de 1 bloco/rodada. A partir de $\tau = 6.0$, a vazão declina de forma perceptível, chegando a 0.72 ± 0.11 em $\tau = 7.0$, acompanhada por crescimento na taxa de forks (de 0.21 a 0.31) e por desvios padrão elevados nos pontos de transição ($\tau = 6.0$ e 6.5), um sinal de que o início da degradação tem caráter tanto gradual quanto probabilístico, com rodadas específicas de alta concorrência ainda pesando bastante no resultado de cada execução individual. Um teste complementar, desenhado para maximizar o número de repetições ($n = 20$) dentro do tempo disponível sem alterar o funcionamento da rede, mostra que a fração de execuções instáveis permanece elevada em toda a faixa $\tau \in [5.0, 6.5]$ (20%, 30%, 30% e 50%, respectivamente), consistente com o caráter probabilístico do fenômeno.

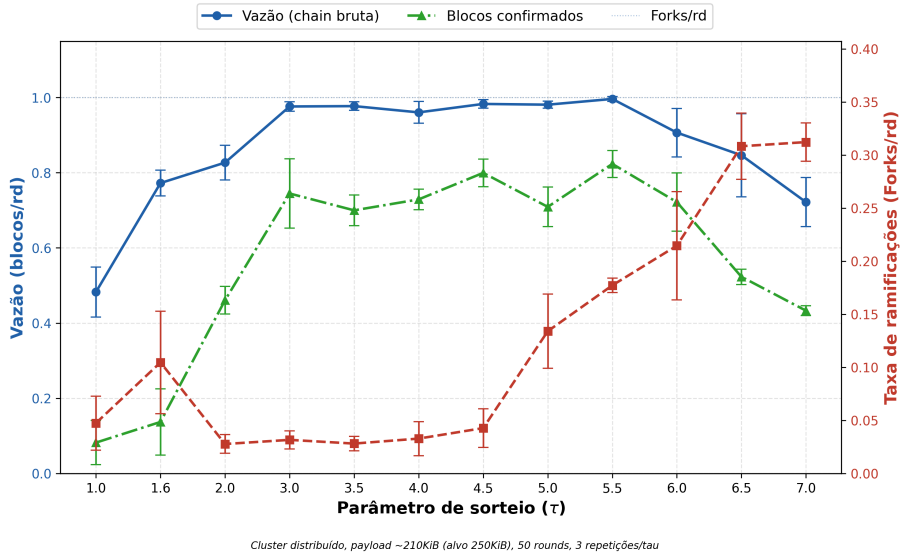

Figura 1. Vazão (*throughput*), blocos confirmados e taxa de ramificações (*forks*) por rodada, variando τ de 1.0 a 7.0 sob carga de $\approx$250 KiB (média $\pm$ erro padrão, 3 repetições).

6. Conclusões e trabalhos futuros

A viabilidade do protocolo CPoS em cenários de tráfego denso requer precisão matemática e flexibilidade paramétrica. Os ensaios isolados sugerem que as otimizações algorítmicas integradas à avaliação de sorteio ajudam a mitigar as falhas de *OverflowError* e favorecem o processamento da cadeia probabilística. A transposição para o domínio dos números reais atenua o dilema de calibração associado aos parâmetros inteiros. Os resultados de rede sugerem que a vazão se mantém elevada, próxima do máximo teórico, ao longo de boa parte da faixa avaliada, declinando de forma perceptível apenas nos valores mais altos de τ; nesses pontos de transição, a alta variabilidade entre execuções indica uma influência probabilística, ligada a rodadas isoladas de alta concorrência.

Como trabalho futuro, pretende-se eliminar o limite do número de rodadas usados nos testes, representando uma rede em produção ininterrupta, e avaliar o desempenho sob condições de carga mais pesada, com blocos de até 1 MiB. Por fim, sugere-se a avaliação de um mecanismo de propagação segmentada de blocos em duas etapas na camada P2P. Nessa abordagem, apenas o cabeçalho do bloco seria difundido inicialmente, postergando o envio do corpo para o momento de confirmação do nó vencedor. A associação dessa otimização com a calibração do sorteio fracionário possui o potencial de reduzir a redundância de tráfego e melhorar a escalabilidade do protocolo.

Referências

Antonopoulos, A. M. (2014). *Mastering Bitcoin: unlocking digital cryptocurrencies*. O'Reilly Media.

Antonopoulos, A. M. and Wood, G. (2018). *Mastering Ethereum: building smart contracts and dapps*. O'Reilly Media.

Bashir, I. (2020). *Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained*. Packt Publishing, 3rd edition.

Buterin, V. (2014). A next-generation smart contract and decentralized application platform. `https://ethereum.org/en/whitepaper/`.

Grimmett, G. and Stirzaker, D. (2020). *Probability and random processes*. Oxford university press.

King, S. and Nadal, S. (2012). Ppcoin: Peer-to-peer crypto-currency with proof-of-stake.

Martins, D. F. G. (2021). Um novo mecanismo de consenso probabilístico para blockchains públicas. Master's thesis, Universidade Estadual de Campinas.

Nakamoto, S. (2009). Bitcoin: A peer-to-peer electronic cash system.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press.