CENTRO UNIVERSITÁRIO FEI

DANILO BIZARRIA DE OLIVEIRA

GERALDO LUCAS FERNANDES DO AMARAL

MARKEL PEDROSA DUARTE DE MACEDO

MATHEUS FERREIRA MESQUITA

# AVALIAÇÃO DE DESEMPENHO DE TRANSAÇÕES EM PROTOCOLOS *BLOCKCHAIN*: um estudo de caso com Ethereum

São Bernardo do Campo

2022

DANILO BIZARRIA DE OLIVEIRA
GERALDO LUCAS FERNANDES DO AMARAL
MARKEL PEDROSA DUARTE DE MACEDO
MATHEUS FERREIRA MESQUITA

# AVALIAÇÃO DE DESEMPENHO DE TRANSAÇÕES EM PROTOCOLOS *BLOCKCHAIN*: um estudo de caso com Ethereum

Trabalho de Conclusão de Curso apresentado ao Centro Universitário FEI, como parte dos requisitos necessários para obtenção do título de Bacharel em Ciência da Computação. Orientado pelo Prof. Dr. Calebe de Paula Bianchini.

São Bernardo do Campo

2022

Avaliação de Desempenho de Transações em Protocolos Blockchain : um estudo de caso com Ethereum / Danilo Bizarria de Oliveira...[et al.].
São Bernardo do Campo, 2022.
36 f. : il.

Trabalho de Conclusão de Curso - Centro Universitário FEI.
Orientador: Prof. Dr. Calebe de Paula Bianchini.

1. blockchain. 2. processamento de transações. 3. sistemas distribuídos. I. Bizarria de Oliveira, Danilo. II. Fernandes do Amaral, Geraldo Lucas. III. Duarte de Macêdo, Markel Pedrosa. IV. Ferreira Mesquita, Matheus. V. de Paula Bianchini, Calebe, orient. VI. Título.

Elaborada pelo sistema de geração automática de ficha catalográfica da FEI com os dados fornecidos pelo(a) autor(a).

DANILO BIZARRIA DE OLIVEIRA
GERALDO LUCAS FERNANDES DO AMARAL
MARKEL PEDROSA DUARTE DE MACEDO
MATHEUS FERREIRA MESQUITA

**AVALIAÇÃO DE DESEMPENHO DE TRANSAÇÕES EM PROTOCOLOS *BLOCKCHAIN*:** um estudo caso com Ethereum

Trabalho de Conclusão de Curso apresentado ao Centro Universitário FEI para obtenção do título de Bacharel em Ciência da Computação. Orientado pelo Prof. Dr. Calebe de Paula Bianchini.

Comissão julgadora

Prof. Dr. Calebe de Paula Bianchini
Orientador

Prof. Dr. Ricardo Destro
Examinador

Prof. Dr. Fernando Pujaico Rivera
Examinador

São Bernardo do Campo
14/06/2022

**RESUMO**

Sistemas de processamento de transações são parte essencial de negócios *online*, especialmente os relativos a transações financeiras, que exigem tanto confiança quanto desempenho para entregar o serviço esperado. Uma ferramenta de *blockchain* é capaz de oferecer os objetivos mencionados, dado que sua arquitetura distribuída favorece a escalabilidade em relação a sistemas centralizados, enquanto que seu protocolo de consenso formaliza a validação de transações. Ainda que tais qualidades sejam comuns às diferentes redes de *blockchain*, suas medidas oscilam entre implementações, e têm sua variação ampliada entre a crescente quantidade de versões do protocolo disponíveis, o que dificulta a caracterização de qual alternativa é mais apropriada para um dado cenário de execução. A abordagem proposta neste trabalho é da avaliação do potencial de escalabilidade da rede Ethereum, e seu comportamento em cenários progressivamente mais distribuídos. Por meio da execução e medição de seu desempenho em 5 diferentes escalas (de 1 a 16 nós, em progressão geométrica de razão 2), foram coletadas métricas de *throughput* e latência, de forma a ilustrar a escalabilidade da rede, com destaque para um aumento observado de cerca de 10× o número de transações por segundo, quando comparados um nó isolado e uma rede de 16 nós, esta que também proporcionou uma queda de mais da metade da latência com relação ao cenário centralizado.

Palavras-chave: *blockchain*, processamento de transações, sistemas distribuídos

**ABSTRACT**

Transaction processing systems are an essential part of online businesses, especially those related to financial transactions, which demand both trust and performance to deliver the expected service. A blockchain tool is capable of providing the mentioned objectives, given that its distributed architecture favors scalability in comparison to centralized systems, while its consensus protocol formalizes transaction validation. Even though such qualities are common to different blockchain networks, their metrics oscillate between implementations, and have their variation amplified among the growing number of versions of the protocol available, what makes it difficult to characterize which alternative is the most appropriate to a given execution scenario. The proposed approach of this work is the evaluation of the scalability potential of the Ethereum network, and its behavior in progressively more distributed scenarios. Through the execution and measurement of its performance in 5 different scales (from 1 to 16 nodes, in a geometric progression of common ratio 2), throughput and latency metrics were collected to illustrate the network's scalability, with the highlight of an observed increase of about 10× the number of transactions per second, when comparing an isolated node to a 16-node network, the latter also provided a decrease of more than half the latency of the centralized scenario.

Keywords: *blockchain*, transaction processing, distributed systems

## LISTA DE ILUSTRAÇÕES


Figura 1 – Funcionamento de uma *blockchain* . . . . . 14
Figura 2 – Uma árvore de blocos, com a definição da maior cadeia feita pela regra de cadeia mais longa e pelo protocolo GHOST . . . . . 15
Figura 3 – Diagrama de Metodologia . . . . . 21
Figura 4 – Funcionamento do agente de testes (Caliper) . . . . . 22
Figura 5 – Diagrama da Proposta Experimental . . . . . 25
Figura 6 – Diagrama do ambiente configurado para uma rede de 4 nós . . . . . 27
Figura 7 – *Throughput* (tps): Caliper centralizado vs. Caliper distribuído . . . . . 29
Figura 8 – Latência (s): Caliper centralizado vs. Caliper distribuído . . . . . 29
Figura 9 – *Throughput* (tps) × Número de nós . . . . . 31
Figura 10 – Latência (s) × Número de nós . . . . . 31

## LISTA DE TABELAS

Tabela 1 – Máquinas do tipo A . . . . . 20
Tabela 2 – Máquinas do tipo B . . . . . 20
Tabela 3 – *Throughput* (tps) × Número de *workers* do Caliper . . . . . 28
Tabela 4 – Latência (s) × Número de *workers* do Caliper . . . . . 30
Tabela 5 – *Throughput* (tps) × Número de nós . . . . . 32
Tabela 6 – Latência (s) × Número de nós . . . . . 33

**SUMÁRIO**


**1 INTRODUÇÃO** . . . . . 8
1.1 OBJETIVO . . . . . 8
1.2 MATERIAL COMPLEMENTAR . . . . . 8
**2 TRABALHOS RELACIONADOS** . . . . . 9
**3 CONCEITOS FUNDAMENTAIS** . . . . . 13
3.1 *BLOCKCHAIN* . . . . . 13
3.2 BITCOIN . . . . . 14
3.3 GHOST . . . . . 15
3.4 ALGORAND . . . . . 16
3.5 PROTOCOLO DE CONSENSO . . . . . 17
**3.5.1 *Proof-of-Work*** . . . . . 17
**3.5.2 *Proof-of-Stake*** . . . . . 18
3.6 MÉTRICAS DE DESEMPENHO . . . . . 19
**4 METODOLOGIA** . . . . . 20
4.1 PREPARAÇÃO DO AMBIENTE . . . . . 20
4.2 DEFINIÇÃO DA CARGA DE TESTES . . . . . 21
4.3 INSTANCIAÇÃO DO AGENTE DE TESTES (CALIPER) . . . . . 22
4.4 EXECUÇÃO DOS TESTES . . . . . 23
4.5 COLETA DE DADOS . . . . . 24
4.6 ANÁLISE DE DADOS . . . . . 24
**5 PROPOSTA EXPERIMENTAL** . . . . . 25
**6 RESULTADOS E DISCUSSÃO** . . . . . 28
6.1 SESSÕES DE TESTES . . . . . 28
**6.1.1 Teste Centralizado** . . . . . 28
**6.1.2 Testes Distribuídos** . . . . . 30
**7 CONCLUSÃO** . . . . . 34
**REFERÊNCIAS** . . . . . 35

# 1 INTRODUÇÃO

Sistemas de processamento de transações são parte essencial da arquitetura de muitos negócios e serviços *online*, especialmente os relativos a transações financeiras, que exigem tanto confiança quanto desempenho para entregar o serviço esperado. Na maioria dessas arquiteturas, existe uma entidade centralizadora, a qual se encarrega de validar as transações inerentes do sistema.

Dado sua arquitetura distribuída, uma ferramenta de *blockchain* (AL-BASSAM et al., 2018; GILAD et al., 2017; LI et al., 2020; NAKAMOTO, 2008) favorece a escalabilidade em relação a sistemas centralizados, enquanto seu protocolo de consenso formaliza a validação de transações. Isso possibilita haver confiança no estado do sistema, sem que se faça necessário haver confiança mútua entre os nós da rede, ou confiança em uma entidade centralizadora, confia-se apenas na criptografia e em seu protocolo de consenso. Dessa forma, usuários de uma rede *blockchain* são capazes de realizar transações acordadas entre si, validadas através de assinaturas digitais, e persistidas globalmente através de um consenso estabelecido entre seus participantes que, uma vez concretizado, não pode ser trivialmente revertido.

Tendo em mente a diferença fundamental entre o modelo transacional de sistemas distribuídos, onde existe uma entidade centralizadora, e o modelo *blockchain*, que é descentralizado e valida transações através de consenso entre os nós, a proposta deste trabalho é medir e analisar o potencial de escalabilidade de uma rede *blockchain* levando em consideração duas métricas fundamentais de desempenho: *throughput* e latência.

## 1.1 OBJETIVO

Este trabalho apresenta uma avaliação de desempenho de transações em *blockchain*, realizando estudos de caso com o protocolo Ethereum.

## 1.2 MATERIAL COMPLEMENTAR

Os arquivos para a instanciação da rede Ethereum, assim como os arquivos de configuração dos testes, e o código-fonte dos contratos executados, estão disponíveis em `https://github.com/dbodanilo/caliper-benchmarks/tree/v0.3.2-ethereum`.

# 2 TRABALHOS RELACIONADOS

Neste capítulo são apresentadas, em ordem cronológica, as publicações que serviram como base para a elaboração, definição do escopo e execução deste trabalho.

No trabalho de Nakamoto (2008), é proposta uma solução ao problema de gasto-duplo em redes de pagamento *peer-to-peer*, sem a necessidade de uma autoridade central. Para obter tal garantia, a rede controla a ordenação das transações ao conectar linearmente seus *hashes* em uma cadeia contínua, que é confirmada por *Proof-of-Work*. Dessa forma, o autor calcula que a probabilidade de um ataque bem sucedido cai exponencialmente com o aumento da vantagem da cadeia honesta, medida pela diferença entre o seu comprimento e o comprimento de uma cadeia maliciosa.

No artigo de Sompolinsky e Zohar (2015) é investigada a implicação de ter um valor de *throughput* muito alto para transações na segurança da Bitcoin contra ataques de gasto-duplo. Os autores abordam essa preocupação de segurança pela regra GHOST, uma modificação na maneira que o nó de Bitcoin constrói e reorganiza as cadeias de blocos. Os resultados apresentados mostram que com um *throughput* alto, invasores substancialmente mais fracos são capazes de reverter pagamentos que fizeram, mesmo depois de serem considerados aceitos pelos destinatários.

No artigo de Eyal et al. (2016) foi apresentado Bitcoin-NG, um protocolo de *blockchain* desenvolvido com foco em escalabilidade. Os autores implementaram o protocolo e executaram um experimento em larga escala com 15% do tamanho do sistema operacional de Bitcoin, utilizando os mesmos clientes dos protocolos. Os experimentos realizados demonstraram que Bitcoin-NG escala de forma otimizada, com a largura de banda limitada pela capacidade individual dos nós e a latência limitada apenas pela propagação do tempo da rede.

Em Gervais et al. (2016) foi apresentada uma estrutura quantitativa para analisar as implicações de segurança e desempenho de vários consensos e parâmetros de rede de *blockchains* de *Proof of Work (PoW)*. Nessa estrutura, são utilizadas estratégias adversárias ideais para gasto-duplo e mineração egoísta, levando em consideração as restrições do mundo real, como propagação de rede, diferentes tamanhos de bloco, intervalos de geração de bloco, mecanismo de propagação de informações e o impacto de ataques de eclipse. A estrutura apresentada permite capturar implantações baseadas em PoW, além de variantes de *blockchain* de PoW que são instanciadas com parâmetros diferentes e comparar as compensações entre seu desempenho e provisões de segurança.

No artigo de Francesco, Malavolta e Lago (2017) é utilizada a metodologia de estudo de mapeamento sistemático para identificar, classificar e avaliar o atual estado da arte nas arquiteturas de microsserviços. São avaliadas três perspectivas: tendencias, focos de pesquisa e potencial de adoção pela indústria. Foi desenvolvido um *framework* de classificação composto de três facetas, uma para cada uma destas três perspectivas. Neste estudo, foram selecionadas 71 estudos de 300 *papers* potencialmente relevantes, foi definido um *framework* de classificação e aplicado à estes estudos. Após, os dados foram sintetizados para produzir uma visão clara do estado da arte em arquitetura de microsserviços e feito considerações à respeito de como os resultados do estudo em questão podem ser transferidos e adotados pela indústria. O resultado deste estudo é um *framework* reutilizável para classificar, comparar e avaliar soluções de arquitetura, métodos e técnicas específicas para microsserviços.

No artigo de Gilad et al. (2017) foi apresentada Algorand, uma nova cripto-moeda que confirma transações com a latência do pedido enquanto escala muitos usuários. Os autores implementaram e avaliaram o desempenho de 1.000 máquinas virtuais EC2, simulando até 500.000 usuários. Os resultados mostram que Algorand confirma transações em menos de um minuto, 125× o *throughput* do Bitcoin e, segundo os autores, quase não gera penalidade para a escalabilidade de mais usuários.

Em Al-Bassam et al. (2018), é detalhado o desenho de Chainspace: um registro distribuído que suporta contratos inteligentes definidos pelos usuários, e executa transações fornecidas pelos usuários em seus objetos. Uma implementação do sistema é avaliada quanto a escalabilidade e funcionalidades, e tem seu desempenho medido. Visando escalabilidade, Chainspace distribui seu estado e a execução de transações em *shards*, e, para consistência, implementa o *Sharded Byzantine Atomic Commit* (S-BAC), um protocolo distribuído que combina *commits* atômicos e *Byzantine Fault Tolerance* (BFT). Com uma implantação em contêineres Amazon EC2, é avaliado que o *throughput* de transações no sistema escala linearmente com o número de *shards*, em até 22 transações por segundo para cada *shard* adicionado, suportando até 350 transações por segundo com 15 *shards*.

No trabalho de Roos et al. (2018), é desenhado e avaliado o SpeedyMurmurs, um algoritmo de roteamento para redes descentralizadas de *Path-Based Transactions* (PBT). A descoberta de caminhos é feita por um método de roteamento baseado em *embedding*, e se utiliza de estabilização por demanda ao recalcular as rotas da sua rede de PBT. A simulação de sua execução foi baseada em dados reais da rede de crédito Ripple, e indicou que o SpeedyMurmurs reduz a sobrecarga de estabilização por até duas ordens de magnitude, e a sobrecarga de roteamento de

uma transação por mais de um fator de dois. Além disso, é mantida ao menos a mesma taxa de sucesso que um roteamento baseado em marcos, enquanto completa mais rapidamente as transações.

No artigo de Thakkar, Nathan e Viswanathan (2018) foi realizado um estudo empírico abrangente para caracterizar o desempenho da Hyperledger Fabric – uma plataforma para o desenvolvimento de redes *blockchain* privadas, introduzida por Androulaki et al. (2018) – e identificar potenciais gargalos de desempenho, para ganhar melhor entendimento do sistema. Foi introduzido e estudado diversas otimizações simples, como *caching* agressivo para verificação no componente de criptografia (melhoria de 3× de performance) e a paralelização do val de política de verificação (melhoria de 7×). Além disso os autores aprimoraram e mediram o efeito de uma otimização de leitura/gravação em massa existente para CouchDB durante o estado de validação e fase de *commit*. Combinando as três otimizações foi obtido uma melhora média de *throughput* de 16× (i.e., de 140 a 2250 transações por segundo).

No artigo de Belhadi et al. (2020), são explorados cinco problemas de mineração e é proposta uma nova estrutura distribuída chamada de *Decomposition Transaction for Distributed Pattern Mining* (DT-DPM, ou "Transação de Decomposição para Mineração de Padrões Distribuídos"). Essa estrutura visa atuar nas limitações dos problemas existentes de mineração de padrões, reduzindo o espaço de pesquisa de enumeração. Os resultados obtidos revelaram que o uso do DT-DPM melhorou a escalabilidade de algoritmos de mineração aplicados em bancos de dados além de demonstrar que o DT-DPM supera a linha de base entre os algoritmos paralelos para mineração de padrões em grandes bancos de dados.

O artigo de Li et al. (2020) apresenta Conflux, um sistema descentralizado de *blockchain* que processa blocos de forma otimista e concorrente, sem descartar qualquer bloco como uma bifurcação. Para isso, as relações entre os blocos são representadas por um grafo acíclico e, partindo de uma ordem total dos blocos, é gerada deterministicamente uma ordem total das transações, que são então registradas na *blockchain*. O sistema foi avaliado em *clusters* Amazon EC2 com até 12 mil nós completos. Nessa configuração, o Conflux alcança um *throughput* de 9,6Mbps, equivalente a 3.480 transações por segundo.

No artigo de Podgorelec, Turkanović e Karakatič (2020), é abordado um método baseado em *machine learning* que busca introduzir a assinatura automatizada de transações de *blockchain*, incluindo também uma identificação personalizada de transações anômalas. Para validação desse método proposto, foi realizado experimento e analise de dados da rede pública Ethereum. Os

resultados obtidos mostram uma visão promissora sobre o método proposto, contribuindo para um uso mais amigável de aplicativos que são baseados em *blockchain*.

No artigo de Srivastava et al. (2021) é proposto o algoritmo "*pre-large weighted high-utility pattern*" (PWHUP) para integrar *high-utility patterns* (HUPs) de diferentes fontes de dados detectadas. O algoritmo PWHUP proposto considera o tamanho da fonte de dados para descobrir mais HUPs para integração. Os experimentos mostram que a abordagem projetada, segundo os autores, tem bom desempenho para integração de conhecimento e supera as soluções de não integração existentes em precisão, *recall* e tempo de execução.

# 3 CONCEITOS FUNDAMENTAIS

Este capítulo apresenta os conceitos fundamentais para a execução da metodologia proposta. Para isso, suas seções incluem os algoritmos, protocolos, e métricas utilizados no desenvolvimento do trabalho.

## 3.1 *BLOCKCHAIN*

*Blockchain* é uma tecnologia que possibilita a manutenção de um registro distribuído sobre o envio e recebimento de recursos digitais entre pares. Sua aplicação ocorre na implementação de cripto-moedas como Bitcoin (NAKAMOTO, 2008), Algo (GILAD et al., 2017) e Ether (BUTERIN, 2014), por exemplo.

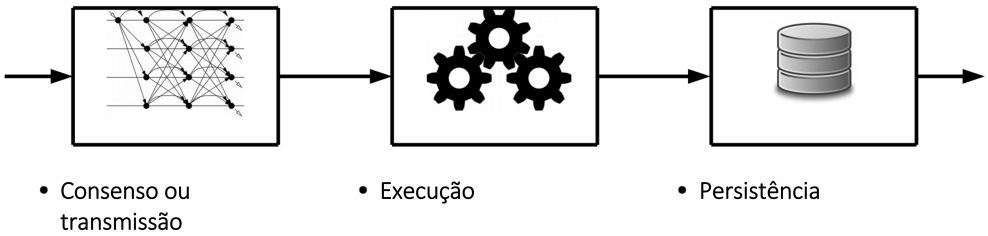
Ela é utilizada no controle de ativos virtuais. E, como analisado em Gervais et al. (2016) e Sompolinsky e Zohar (2015), se desafia a proporcionar segurança a uma base de dados num ambiente descentralizado, sem depender de confiança mútua entre seus nós, nem da presença de uma autoridade central para ditar o estado global do sistema. Trata-se de uma tecnologia de registro distribuído.

A Figura 1 ilustra o funcionamento de uma rede *blockchain* por meio de três etapas: consenso ou transmissão, execução e persistência. Num primeiro momento, as transações são recebidas dos clientes e compartilhadas com outros nós da rede (transmissão), ocorre sua agregação pelo minerador a um bloco, que, por sua vez, também é disseminado pela rede, a fim de ser aceito pela maioria dos outros nós mineradores; tal aceitação se dá por seu uso como base para o próximo bloco em mineração (consenso); as transações do bloco confirmado por consenso são validadas (execução), e o bloco é adicionado à corrente com o histórico de transações desta rede (persistência).

Seu funcionamento é, de modo geral, fundamentado na descrição de Nakamoto (2008), e tem o objetivo de administrar a posse de um ativo digital, assegurando a transferência de propriedade de forma única e acordada entre os envolvidos. Tais garantias são obtidas, primeiramente, pela assinatura digital de cada transação por ambas as partes, seguida do agrupamento de transações em um bloco, que, quando finalizado, é inserido ao estado do sistema pela referência, no bloco atual, à identificação do bloco anterior na cadeia global, e identificado, também de forma única, por meio de um *hash* do seu conteúdo.

A *Blockchain* pode ser divida em três tipos, com relação à sua implantação e permissão de acesso, sendo eles: Privada, na qual o ambiente é controlado e possui um número reduzido

Figura 1 – Funcionamento de uma *blockchain*




Fonte: Autores, adaptado de Androulaki et al. (2018)

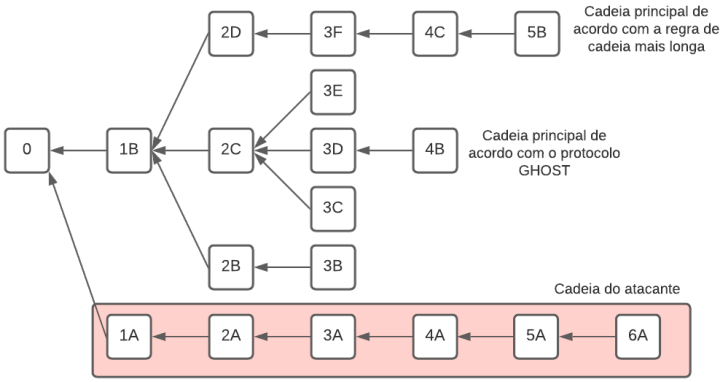
de nós – possuindo assim um maior nível de centralização e controle sobre os participantes –, pública, que representa uma rede aberta descentralizada com um maior número de nós e, por fim, de Consórcio, representando uma mistura entre as redes públicas e privadas, apresentando uma rede pública de usuários, porém mantendo um nível de centralização.

## 3.2 BITCOIN

Bitcoin – a primeira criptomoeda de que se tem conhecimento – cujo protocolo foi definido por Nakamoto (2008), é uma versão *peer-to-peer* de dinheiro eletrônico que permite que pagamentos entre duas partes sejam enviados e recebidos sem que haja o intermédio de uma instituição financeira, ao mesmo tempo que previne a possibilidade de gasto-duplo. Segundo Nakamoto (2008), uma moeda eletrônica pode ser definida como uma cadeia de assinaturas digitais. Um dono transfere uma determinada quantia para outro através de um processo que envolve assinar digitalmente, com sua chave privada, um *hash* da transação anterior e da chave pública do próximo dono. A cadeia de propriedade pode ser verificada utilizando-se as chave pública do dono anterior em cada transação.

O problema, porém, é assegurar que não haja gasto-duplo. A solução proposta pelo *white paper* é utilizar um *timestamp server* que faz um *hash* de um bloco de transações, adiciona um *timestamp* à este bloco e publica este *hash* na rede. Cada *timestamp* inclui o *timestamp* do bloco anterior no *hash*, criando assim, uma cadeia de blocos de transações, ou uma *blockchain*. O *hash* precisa preencher requisitos específicos para ser considerado válido. Esses requisitos servem para garantir o *Proof-of-Work*. O *Proof-of-Work*, serve para assegurar que um bloco não possa ser modificado uma vez que ele é incluído no *blockchain*, porque, para modificar um bloco, todos

Figura 2 – Uma árvore de blocos, com a definição da maior cadeia feita pela regra de cadeia mais longa e pelo protocolo GHOST




Fonte: Autores, adaptado de Sompolinsky e Zohar (2015)

os blocos que vieram depois dele e o referenciam também teriam que ser modificados, algo que seria computacionalmente custoso.

## 3.3 GHOST

GHOST é uma sigla para: *Greedy Heaviest-Observed Sub-Tree* (Sub-árvore observada mais pesada e gananciosa), introduzido em Sompolinsky e Zohar (2015), este protocolo é uma nova politica para definir a cadeia principal em uma árvore de blocos, com o objetivo de aumentar a segurança da cadeia contra ataques de gasto-duplo em blocos de maior tamanho, substituindo a regra de cadeia mais longa.

A Figura 2 mostra um cenário no qual uma árvore de blocos é criada por uma rede. Neste exemplo, o atacante cria uma cadeia de 6 blocos (representados pelos blocos de 1A a 6A), que não apresenta tantas bifurcações, dado que é criada de forma artificial, com a intenção de reescrever a cadeia já aceita pelos nós honestos. Esta cadeia substituiria a cadeia principal, se considerada a regra de cadeia mais longa, porém, aplicando o GHOST, a cadeia do atacante, numa sub-árvore menos "pesada", deixa de ser considerada como cadeia principal.

Ainda no artigo Sompolinsky e Zohar (2015) são apresentadas as seguintes propriedades básicas do protocolo:

Dada uma rede de *blockchain*, denota-se por $\lambda_h$ a taxa de criação de blocos dos nós honestos (os quais não tentam manipular a rede em benefício próprio). Para cada bloco $B$, $time(B)$ denota seu momento absoluto de criação (contado a partir do momento de criação do

primeiro bloco da *blockchain* – seu bloco gênesis) Por fim, $\psi_B$ é definido como o momento mais anterior no qual o bloco $B$ foi abandonado por todos os nós, ou aceito por todos os nós.

i Proposição (A convergência da História): $Pr(\psi_B < \infty) = 1$. Ou seja todos os blocos ocasionalmente estarão totalmente abandonados, ou serão totalmente adotados.

ii Proposição (Resiliência a ataques dos 50 %): Assumindo que a taxa de criação de blocos do atacante seja $q \cdot \lambda_h$, e $0 \leq q < 1$. A probabilidade que um bloco $B$ estará fora da cadeia principal em algum momento após o $time(B) + \tau$, dado que estava na cadeia principal no $time(B) + \tau$, tende a zero à medida que $\tau$ tende a $\infty$

## 3.4 ALGORAND

De acordo com o artigo de Gilad et al. (2017), Algorand é um ativo virtual projetado para confirmar transações com latência na ordem de um minuto. Em seu núcleo, é utilizado um protocolo de acordo bizantino (*Byzantine Agreement*) chamado de "$BA\star$", que pode ser dimensionado para muitos usuários, permitindo Algorand chegar a um consenso sobre um novo bloco com baixa latência e sem possibilidade de *forks*. $BA\star$ usa funções aleatórias verificáveis (VRFs) para selecionar usuários aleatoriamente em um ambiente privado e de forma não interativa.

Em seu núcleo, Algorand enfrenta três desafios:

i Ataques Sybil: um adversário cria diversos pseudônimos para gerar influência no protocolo de acordo bizantino

ii Dimensionamento para milhões de usuários: grande superioridade à escala em que o estado da arte de protocolos bizantinos operam.

iii Ataques de negação de serviço: Algorand deve continuar operando mesmo que um adversário desconecte alguns dos usuários.

A fim de prevenir ataques Sybil, Algorand determina um peso para cada usuário. $BA\star$ é projetado para garantir consenso, desde que uma fração ponderada (constante maior de 2/3) dos usuários são honestos. Para determinar o peso dos usuários, Algorand se baseia no dinheiro disponível em suas contas. Assim, contanto que mais de 2/3 do dinheiro é propriedade de usuários honestos, Algorand pode evitar bifurcações e gastos em dobro.

$BA\star$ alcança escalabilidade escolhendo um comitê – pequeno conjunto de representantes selecionados aleatoriamente do conjunto total de usuários – para executar cada etapa de seu

protocolo. Todos os outros usuários observam mensagens do protocolo, o que lhes permite aprender o bloco acordado. Para garantir que uma fração suficiente dos membros do comitê são honestos, $BA\star$ escolhe membros do comitê aleatoriamente entre todos os usuários com base em seus respectivos pesos.

Para prevenir um adversário de visar os membros do comitê, $BA\star$ seleciona os membros de maneira privada e não interativa. Isso significa que cada usuário no sistema pode determinar de forma independente se eles são escolhidos para estar no comitê, calculando uma função de sua chave privada e informações públicas do *blockchain*. Em caso de indicação da função que o usuário foi escolhido, é retornada uma *string* que prova, para os outros usuários, que os membros do comitê podem incluir suas mensagens na rede. Desde a adesão, a seleção não é interativa, um adversário não sabe qual usuário a ser direcionado até que comece a participar do $BA\star$.

Nos casos que um adversário pode ter como alvo um membro do comitê, uma vez que esse membro envie uma mensagem em $BA\star$, a $BA\star$ mitiga esse ataque, exigindo membros do comitê para falar apenas uma vez. Assim, uma vez que uma mensagem é enviada por um membro do comitê para um adversário (expondo sua identidade), o membro se torna irrelevante para $BA\star$. $BA\star$ alcança essa propriedade evitando qualquer estado privado (exceto para a chave privada do usuário), tornando capaz a igualdade de participação dos usuários e eleição de um novo comitê de membros para cada etapa do protocolo do acordo bizantino.

## 3.5 PROTOCOLO DE CONSENSO

Segundo Xiao et al. (2020), um protocolo de consenso, numa rede de *blockchain*, é a tecnologia que garante o acordo entre todos os nós sobre o registro unificado de transações, sem a ajuda de uma autoridade central.

Seu desafio é garantir o favorecimento da atuação de nós honestos, em detrimento da atuação de possíveis nós maliciosos. Para isso, pode ser descrito por dois tipos principais: algoritmos de consenso por *Proof-of-Work*, e por *Proof-of-Stake*.

### 3.5.1 *Proof-of-Work*

Com base no artigo de Gervais et al. (2016) o mecanismo de consenso de prova de trabalho (*Proof-of-Work*, ou PoW) é o mais amplo implementado mecanismo de consenso em *blockchains* existentes. PoW foi introduzido pelo Bitcoin assumindo que cada par vota com seu “poder computacional”, solucionando instâncias de prova de trabalho e construindo blocos

apropriados. Bitcoin, por exemplo, utiliza um PoW baseado em *hash*, que envolve encontrar um valor *nonce* (bytes sem conteúdo semântico sobre o bloco), de modo que o *hash* deste com parâmetros adicionais do bloco deve ser menor do que o valor alvo atual.

Ao encontrar o *nonce*, o minerador cria e encaminha o bloco na camada de rede para seus pares. Para realizar a validação, outros pares na rede podem verificar o PoW calculando o *hash* do bloco e verificar se satisfaz a condição de ser menor que o valor alvo atual estabelecido no *blockchain*. No caso do Bitcoin, o valor alvo diminui a cada 2016 blocos, fazendo com que o nível de dificuldade para minerar um bloco aumente. Este *feature* previne que um nó possa alterar um bloco já minerado, uma vez que ele teria que modificar todos os blocos posteriores, algo que seria computacionalmente inviável devido ao aumento no grau de dificuldade à medida que mais blocos são inseridos. Assim, o PoW serve para prevenir fraudes no *blockchain*.

Ainda de acordo com Gervais et al. (2016), o funcionamento da segurança do PoW é baseado no principio de que nenhuma entidade deve centralizar mais que a metade do poder de processamento, dado que tal entidade pode controlar efetivamente o sistema sustentando a cadeia mais longa. Os ataques conhecidos às *blockchains* baseados em PoW podem ser divididos em duas etapas. Primeiro, um adversário pode dobrar os gastos de um mesmo ativo para emitir duas (ou mais) transações, gastando assim mais moedas do que de fato possui. Segundo, os mineradores podem realizar ataques de mineração egoísta, esses são baseados no aumento da participação de mineração relativa no *blockchain*, causando a retenção seletiva dos blocos minerados e publicando-os gradualmente.

### 3.5.2 *Proof-of-Stake*

Em Xiao et al. (2020), *Proof-of-Stake* é definida como uma alternativa de maior eficiência energética ao *Proof-of-Work*. Um *stake* refere-se às moedas ou *tokens* da rede que são investidos por seu dono no processo de consenso da *blockchain*. Enquanto um minerador num cenário de PoW tem sua chance de propor um novo bloco à rede proporcional ao seu poder computacional, a chance de um minerador por PoS é proporcional a seu valor de *stake*.

Com isso, esse desenho de consenso move o custo de mineração de fora do sistema, quando se minera com poder computacional e eletricidade, para dentro do sistema, ao promover a mineração pelo comprometimento de um ativo digital. Ainda assim, *Proof-of-Stake* se mostra resiliente a ataques “Sybil”, nos quais um nó administrado por uma mesma entidade se apresenta à rede como diversos nós, porque o *stake* de um nó está vinculado a sua identidade na rede.

## 3.6 MÉTRICAS DE DESEMPENHO

Métricas de performance são dados numéricos, geralmente com relação ao tempo de execução, mensurados sobre um sistema, e são utilizadas na comparação entre diferentes arquiteturas, que tenham um objetivo comum.

São exemplos de métricas para comparação de desempenho:

***Throughput*** O *throughput* é a quantidade de dados de entrada que são completamente processados num determinado intervalo de tempo ( Roos et al. (2018)). *Throughput* refere-se a quantidade de dados que podem ser transferidos da origem para o destino dentro de um determinado intervalo específico de tempo. No contexto do nosso experimento, ele é a taxa de transações são propagadas em toda rede durante o intervalo de 1 segundo .

Desta forma, o *throughput* demonstra a avaliação da eficiência de um teste e quais outras métricas de desempenho podem estar afetando o funcionamento da rede, como uma latência alta, ou uma taxa de sucesso baixa. No contexto do nosso experimento, dentro da carga de teste que esta sendo executada se a taxa de transação estiver próxima da quantidade de transações por segundo, significa que o comportamento da rede no teste realizado teve alta eficiência, caso contrário o comportamento da rede não foi muito eficiente.

**Latência** A latência é o tempo decorrido entre o envio de uma mensagem e o seu recebimento pelo destinatário - Roos et al. (2018). No contexto do nosso experimento, a latência compreende o tempo decorrido desde que uma transação foi submetida até o momento em que o resultado dela está disponível para toda a rede. Uma latência alta significa que para a realização da transação o tempo decorrido para o seu recebimento foi grande, diminuindo assim a taxa média de dados de sucesso de da transação. Portanto, uma baixa latência significa um comportamento eficiente da rede.

# 4 METODOLOGIA

A metodologia deste trabalho se vê representada na Figura 3, e está organizada em 6 estágios principais, descritos neste capítulo. O estágio 1, preparação do ambiente, compreende a instanciação de recursos computacionais para a execução do protocolo uma mesma arquitetura. A seguir, o estágio 2 é o de definição da carga de testes, e objetiva fornecer uma carga comum de dados, que será transformada para o respectivo formato de entrada, antes de ser passada no protocolo. Com a carga de testes preparada, o estágio 3 é a instanciação do agente de testes, onde o qual será responsável por toda orquestração do teste que será realizado. No estágio 4, o teste é realizado, com o qual serão gerados dados indicadores de performance para avaliação na etapa seguinte. No estágio 5, de coleta de dados, as métricas originadas da execução serão agrupadas e ordenadas para a análise posterior. No estágio 6, análise de dados, serão comparadas entre os sistemas as informações colhidas após sua execução, para avaliação.

## 4.1 PREPARAÇÃO DO AMBIENTE

Tem por objetivo normalizar as condições de execução da rede. Assim a configuração das máquinas foi definida da forma descrita nas Tabelas 1 (tipo A) e 2 (tipo B):

Tabela 1 – Máquinas do tipo A

| Componente | Especificação |
|---|---|
| Processador | Intel® Core™ i5-3340S CPU @ 2.80 GHz |
| RAM instalada | 8 GB |
| Tipo de sistema | Windows 10 de 64 bits |

Fonte: Autores

Tabela 2 – Máquinas do tipo B

| Componente | Especificação |
|---|---|
| Processador | Intel® Core™ i5-6500T CPU @ 2.50 GHz |
| RAM instalada | 8 GB |
| Tipo de sistema | Windows 11 de 64 bits |

Fonte: Autores

Figura 3 – Diagrama de Metodologia

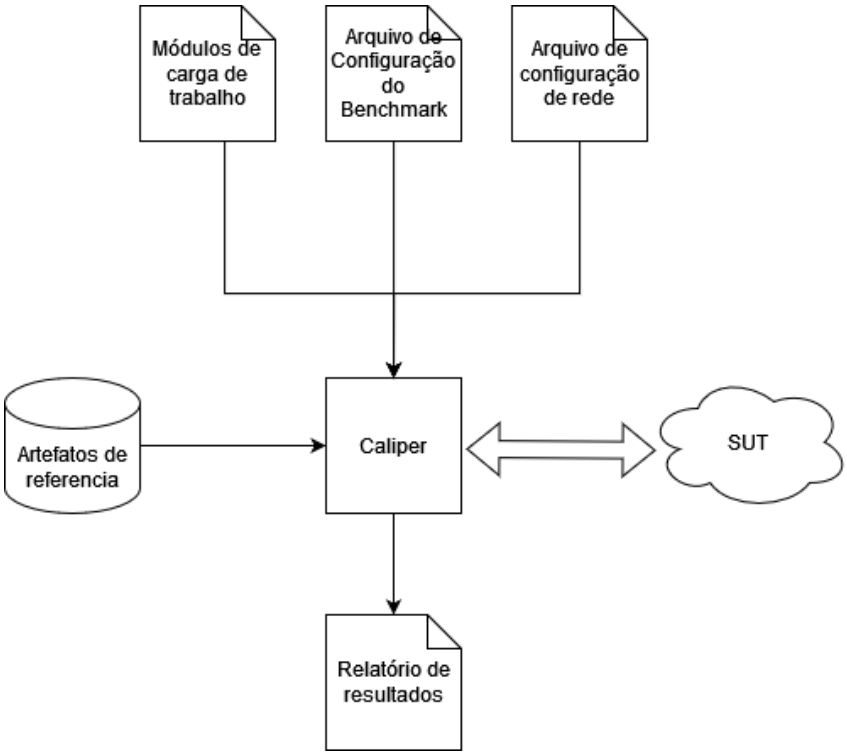
Fonte: Autores

Assim, nos testes com redes de até 8 nós Ethereum, apenas máquinas do tipo A foram utilizadas. Enquanto que a rede com 16 nós foi instanciada entre 8 estações do tipo A e 8 estações do tipo B.

## 4.2 DEFINIÇÃO DA CARGA DE TESTES

Deve-se ter em conta as diferenças de configuração dos tipos de máquinas apresentados nas tabelas 1 e 2. Nesta etapa, foi definida a configuração do teste realizado, contendo a duração do teste, quantidade de transações por segundo e a quantidade de máquinas. A definição da quantidade de transações por segundo teve relação com a quantidade máquinas que foram utilizadas em cada teste, de forma que o aumento da quantidade de transações por segundo é reflexo do aumento da quantidade de máquinas definida.

Figura 4 – Funcionamento do agente de testes (Caliper)




Fonte: Autores, adaptado de Caliper (2020)

A carga de testes estabelece uma taxa máxima de envio de transações por segundo. Essa taxa tem por objetivo enviar uma quantidade de transações por segundo superior à capacidade de processamento da rede, sem prejudicar o funcionamento desta.

## 4.3 INSTANCIAÇÃO DO AGENTE DE TESTES (CALIPER)

Nesta etapa, foi realizada a instanciação do agente da ferramenta Caliper, para a execução dos testes. No teste centralizado, foi instanciado apenas um agente (*master*), nos testes distribuídos, são instanciados agentes *workers* em cada uma das máquinas com um nó da rede sob teste.

O agente de testes é um sistema cliente do protocolo *blockchain* que gera uma carga de trabalho e mede o desempenho de seu processamento pela rede, chamada de sistema sob teste (*System Under Test* (SUT), monitorando continuamente suas respostas. Na figura 4 pode ser observada a estrutura do seu funcionamento:

**Arquivo de configuração de *benchmark*:** nesse arquivo contém a descrição de como o *benchmark* deve ser executado. Ele pode ser considerado como orquestrador do fluxo, pois nele temos informações sobre quantas rodadas de teste o Caliper deve

executar, a que taxa as transações devem ser enviadas e qual o módulo irá gerar seu conteúdo. Estão incluídas também as configurações sobre o monitoramento do SUT.

**Arquivo de configuração de rede:** é específico do SUT. O arquivo descreve a localização dos nós (endereços de *endpoint*), quais clientes estão presentes na rede e quais contratos inteligentes o Caliper deve implantar ou interagir.

**Módulos de carga de trabalho:** Quando o Caliper agenda transações para uma determinada rodada, é tarefa do módulo de carga de trabalho da rodada gerar o conteúdo das transações e submetê-lo. Cada rodada pode conter um módulo de carga de trabalho associado diferente, portanto, deve ser fácil separar sua implementação de carga de trabalho com base em fases/comportamentos.

**Artefatos de referência:** Pode haver artefatos adicionais necessários para executar um

*benchmark* que pode variar entre diferentes *benchmarks* e execuções. Eles geralmente incluem materiais criptografados necessários para interagir com o SUT, código-fonte de contrato inteligente, arquivos de configuração de tempo de execução ou pacotes para módulos de carga de trabalho.

Além de contribuir para diferenciação dos aspectos que serão comparáveis para aqueles que não serão, o agente viabilizará a consolidação de informações que auxiliarão na coleta e análise dos dados, levantando dados suficientes para a avaliação de desempenho do protocolo de acordo com o teste realizado.

## 4.4 EXECUÇÃO DOS TESTES

Nesta etapa, com o intuito de gerar informações para análise posterior, é realizado o teste de carga com o protocolo. Será feita a inicialização da rede do protocolo a ser testado, a inicialização do agente de testes, onde será definida a taxa na qual as transações serão inseridas, e a definição da quantidade de maquinas que serão utilizadas no teste. Os testes foram segregados nos seguintes tipos:

**Centralizado:** Testa de maneira centralizada o agente de testes instanciando apenas o nó *Master* do agente, mas sem realizar o processo *peer-to-peer*.

**Distribuídos:** Testa o agente de testes de maneira *peer-to-peer*, fazendo comunicação com um nó *Master*, pelos *Workers* do agente instanciados em cada um dos nós da rede.

Além disso, o processo de teste realizado pelo agente de testes (Caliper) foi:

***Fixed-feedback-rate*:** Envia transações de entradas em um intervalo fixo, além de ter uma quantidade fixa de transações por segundo. Quando as transações inacabadas excedem os tempos das transações definidas para cada cliente, o agente irá parar de enviar transações de entrada temporariamente e fará uma pausa por um período que pode ser definido.

Por fim, para a execução dos testes, serão utilizados três tipos de contratos, sendo eles:

***Open*:** Contrato responsável por simular uma abertura de conta no protocolo Ethereum.

***Query*:** Contrato responsável por listar as contas que estão disponíveis na rede.

***Transfer*:** Contrato responsável por simular a transferência de um valor fixo de uma conta para outra dentro do protocolo Ethereum.

## 4.5 COLETA DE DADOS

É feita a coleta dos dados transacionais que serão utilizados. As bases de dados aqui consideradas serão avaliadas a fim de gerar informações comparativas suficientes de cada um dos protocolos, buscando avaliar o comportamento de cada um deles, além de contribuir para um maior detalhamento para o que processo avaliativo ocorra de maneira coesa. Por fim, a partir da execução das mesmas transações pelos três sistemas, mede-se: o *throughput* em transações executadas por segundo, e sua respectiva latência.

## 4.6 ANÁLISE DE DADOS

Após sua coleta, os dados quantitativos serão comparados para cada teste realizado, tendo em conta a escala de cada tipo de métrica, bem como sua hierarquia. Como exemplo dessas diferenças, nota-se que a latência caracteriza resultados melhores em valores mais baixos, enquanto o *throughput* é classificado no sentido contrário, com resultados melhores vistos em valores mais altos. Ao mesmo tempo, por completude, tais respostas concretas serão comentadas e justificadas com características da arquitetura e da execução de cada solução.

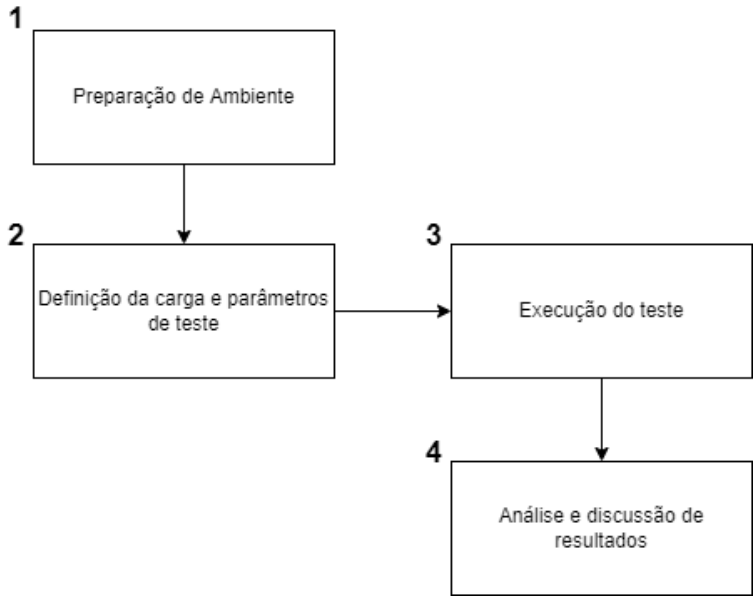
# 5 PROPOSTA EXPERIMENTAL

Este experimento destina-se a realizar uma análise de desempenho e escalabilidade de uma rede de *blockchain* que utiliza o protocolo Ethereum. Para esse objetivo, foi utilizada a ferramenta Hyperledger Caliper para gerar e distribuir uma carga de transações entre os vários nós de uma rede *blockchain* Na etapa de preparação do ambiente, as máquinas utilizadas foram configuradas com as ferramentas selecionadas para este experimento: Caliper, Docker, Mosquitto, além dos nós Ethereum. Estes nós são conectados entre si, formando, assim, uma rede *blockchain*. Na figura 5 é apresentado um diagrama das etapas que foram realizadas.

O Mosquitto - ferramenta de mensageiria (*message broker*) - é utilizado para mediar a comunicação entre os nós (*master* e *workers*) do Caliper, agente de testes do experimento. O Caliper é utilizado para atuar sobre os nós do Ethereum, sendo o responsável pelo envio das transações, além da coleta das métricas que compõem o relatório de desempenho. Ambas estas ferramentas são executadas através do Docker (ferramenta de conteinerização).

Na etapa de definição dos parâmetros de teste, definimos os parâmetros que regem a atuação do Caliper. Entre estes parâmetros, os que estão relacionados à medição do desempenho da rede são: a quantidade de *workers* do Caliper, a quantidade de transações por segundo (TPS), e a quantidade de transações pendentes. Este último parâmetro serve para definir o número

Figura 5 – Diagrama da Proposta Experimental




Fonte: Autores

máximo de transações que o Caliper deve permitir estar sem confirmação pela rede, e retomar sua taxa de envio apenas quando houver sua confirmação. Isso garante que o Caliper não envie transações muito além do limite de processamento da rede, o que poderia ocasionar a interrupção do serviço e, consequentemente, dos testes.

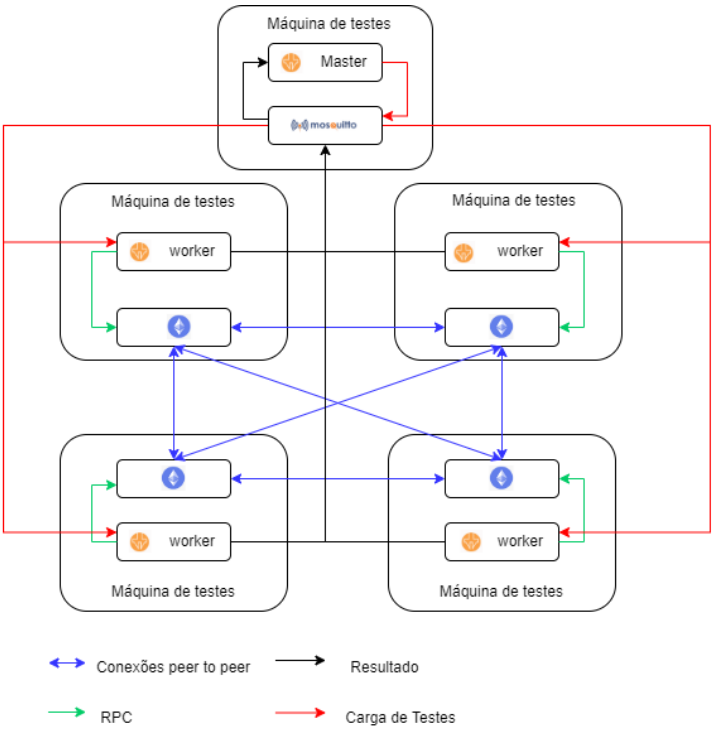
Na etapa de Sessões de Teste, a rede de *blockchain* e os nós do Caliper já estão instanciados; o nó *master* do Caliper envia a carga de transações para um tópico do Mosquitto. Os nós *workers* do Caliper consomem o conteúdo dessa fila: a carga de transações a serem transmitidas para os seus respectivos nós da Ethereum, que, por sua vez, as inserem no bloco em mineração e as propagam na rede.

Foram realizadas cinco sessões de teste, onde se escalou a rede Ethereum quanto a sua quantidade de nós, em progressão geométrica de razão 2, Portanto, a primeira sessão se dá com 1 nó; a segunda, com 2; com 4 na terceira, e assim por diante, até a última sessão, com 16 nós. Em cada teste temos apenas um nó Ethereum por máquina, um *worker* do Caliper associado a este nó rodando em cada máquina, e uma outra máquina dedicada a rodar a instância do Mosquitto e o nó *master* do Caliper. Durante o processo descrito nesta etapa, o Caliper coleta valores das métricas de *throughput*, latência e taxa de sucesso.

Na última etapa deste experimento - Análise e Discussão de Resultados - as métricas obtidas pelo Caliper foram a base para de uma análise desempenho e potencial de escalabilidade da rede Ethereum de *blockchain*.

A Figura 6 representa um ambiente com 5 máquinas, onde 4 máquinas são pertencentes à rede Ethereum e a 5ª máquina representa o nó *master* do agente de testes.

Figura 6 – Diagrama do ambiente configurado para uma rede de 4 nós




Fonte: Autores

# 6 RESULTADOS E DISCUSSÃO

Este capítulo apresenta os resultados obtidos dos experimentos detalhados na proposta experimental, envolvendo redes distribuídas, *blockchain* e contratos inteligentes.

## 6.1 SESSÕES DE TESTES

Neste experimento, foram feitas quatro execuções por teste descrito. A fim de gerar métricas para comparação de duas configurações do agente de testes, apenas na escala de quatro máquinas foram realizadas uma sessão de testes de maneira centralizada e outra de maneira distribuída. A seguir, são apresentados de forma agregada os resultados da realização dos testes.

### 6.1.1 Teste Centralizado

Neste experimento, para verificação do comportamento do agente de testes sobre a rede, foram realizados dois testes sobre uma rede de quatro nós Ethereum: o primeiro com um agente de testes atuando de maneira centralizada, o segundo com o agente de testes distribuído como os nós da rede sob teste. Como critérios avaliativos, os gráficos nas Figuras 7 e 8 foram consolidados com os resultados dos testes.

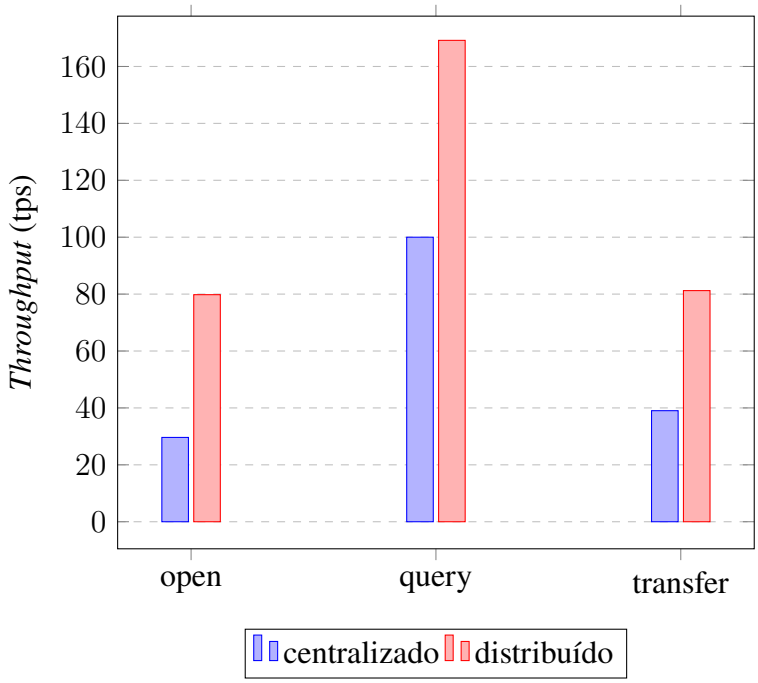
Com este teste, é possível verificar nas Tabelas 3 e 4 que o agente de testes atuando de maneira distribuída em conjunto com os nós da rede, contribui para o aumento da taxa média de transações processadas (*throughput*). Por outro lado, de maneira centralizada, o agente de testes foi diagnosticado como um gargalo para a taxa média de transações sucedidas, uma vez que a rede apresentava capacidade de processamento suficiente para uma maior quantidade de transações efetuadas com sucesso.

Tabela 3 – *Throughput* (tps) × Número de *workers* do Caliper

| *workers* | open | | query | | transfer | |
|---|---|---|---|---|---|---|
| | média | desvio padrão | média | desvio padrão | média | desvio padrão |
| 1 | 29.63 | 2.25 | 100.00 | 0.00 | 39.03 | 4.34 |
| 4 | 79.80 | 1.50 | 169.18 | 59.25 | 81.23 | 27.73 |

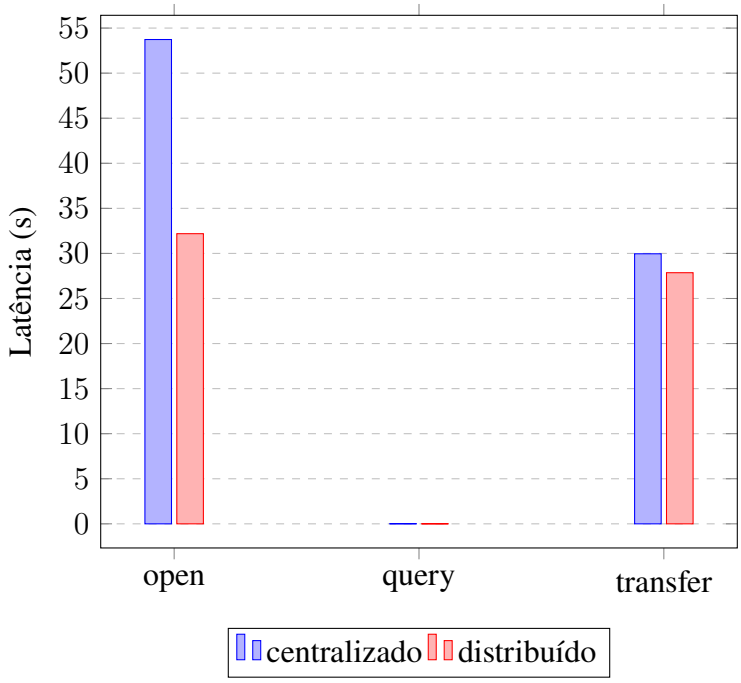
Fonte: Autores

Figura 7 – *Throughput* (tps): Caliper centralizado vs. Caliper distribuído




Fonte: Autores

Figura 8 – Latência (s): Caliper centralizado vs. Caliper distribuído




Fonte: Autores

Tabela 4 – Latência (s) × Número de *workers* do Caliper

| *workers* | open | | query | | transfer | |
|---|---|---|---|---|---|---|
| | média | desvio padrão | média | desvio padrão | média | desvio padrão |
| 1 | 53.73 | 40.65 | 0.02 | 0.02 | 29.96 | 20.37 |
| 4 | 32.19 | 1.35 | 0.01 | 0.01 | 27.86 | 4.94 |

Fonte: Autores

### 6.1.2 Testes Distribuídos

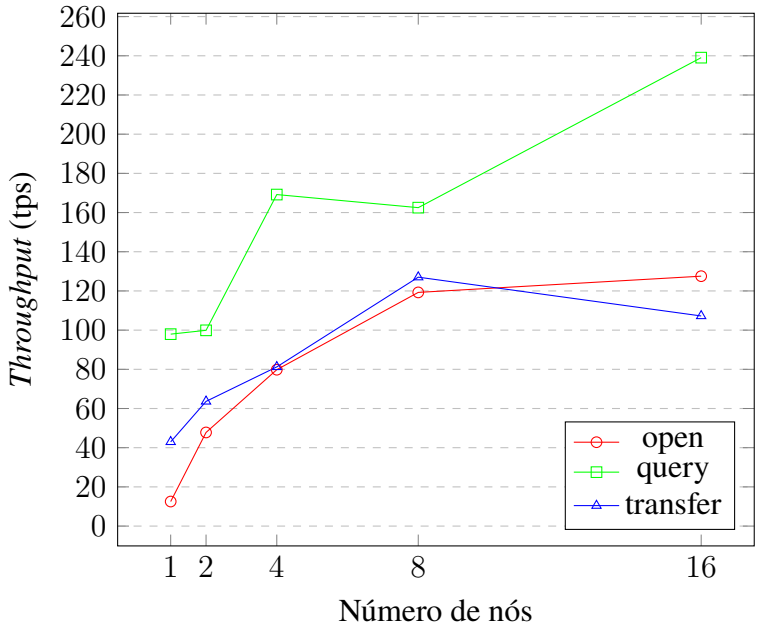
Nesse experimento, para avaliação do comportamento da rede, foram realizados testes de escalabilidade tanto com a rede, como com o agente de testes. Em suma, foi feita a progressão geométrica de razão 2 para demonstrar a escalabilidade da rede.

Os resultados relacionados à taxa média de transações processadas com sucesso podem ser observados na Figura 9. E observa-se que, nos testes com o contrato *open*, conforme a progressão da quantidade de maquinas foi sendo realizada, a taxa média de transações processadas com sucesso foi crescente. Nos testes de *query*, o comportamento descrito também pode ser observado, porém o teste realizado com 8 máquinas teve uma pequena queda na taxa média de transações processadas com sucesso. Após queda, o comportamento crescente se manteve. Já nos testes de *transfer* foi observado a tendência crescente da taxa média de transações processadas com sucesso conforme progressão da quantidade de máquinas, mantendo assim os padrões de escalabilidade da rede.

Como ilustrado na Figura 10, observa-se nos testes com o contrato *open* que, conforme a progressão da quantidade de maquinas foi sendo realizada, a latência teve comportamento decrescente. Nos testes de *query*, pode ser observado que a latência se manteve estável, uma vez que esse processo não exige tanto processamento da rede. Já nos testes de *transfer*, puderam ser observados dois comportamentos: o primeiro, um comportamento crescente que pode ser observado na progressão de um até oito nós, e o segundo, já na escala de dezesseis máquinas, no qual a latência foi reduzida de maneira significativa.

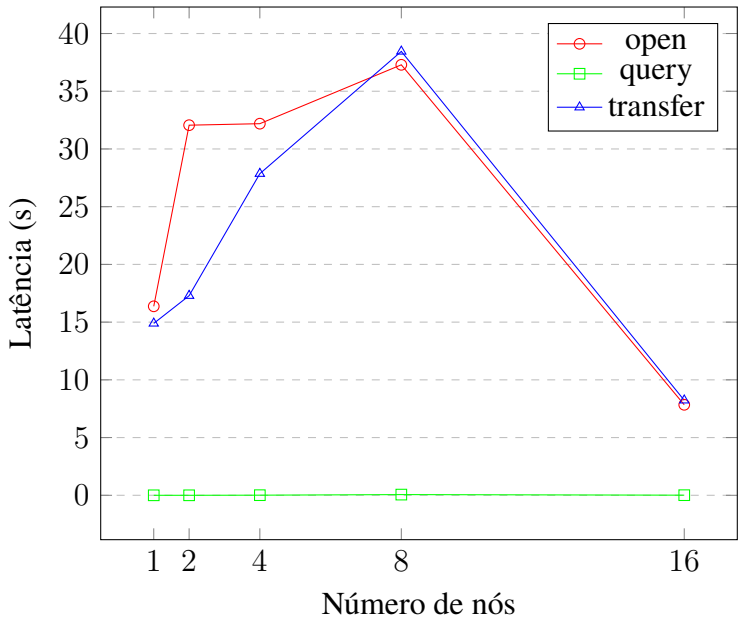
Além dos dados de desempenho apresentados anteriormente, é possível verificar informações relacionadas à variação de comportamento da rede nos testes aplicados. As Tabelas 5 e 6 representam a média o desvio padrão entre as quatro execuções de cada teste, cada um definido pelo número de nós na rede.

Figura 9 – *Throughput* (tps) × Número de nós




Fonte: Autores

Figura 10 – Latência (s) × Número de nós




Fonte: Autores

Tabela 5 – *Throughput* (tps) × Número de nós

| nós | open | | query | | transfer | |
|---|---|---|---|---|---|---|
| | média | desvio padrão | média | desvio padrão | média | desvio padrão |
| 1 | 12.53 | 5.16 | 97.95 | 1.03 | 42.98 | 2.10 |
| 2 | 47.78 | 3.90 | 99.90 | 0.12 | 63.63 | 1.89 |
| 4 | 79.80 | 1.50 | 169.18 | 59.25 | 81.23 | 27.73 |
| 8 | 119.25 | 6.12 | 162.50 | 84.00 | 127.03 | 6.40 |
| 16 | 127.53 | 66.90 | 239.05 | 0.31 | 107.23 | 45.40 |

Fonte: Autores

Analisando o conjunto de informações apresentadas na Tabela 5, é possível tirar algumas conclusões. Nos testes com o contrato *open*, por exemplo, a média do *throughput* seguiu a escala da rede, e manteve seu desvio padrão estável nas escalas de 1 a 8 nós, tendo variação significativa apenas com 16 nós. Uma possível causa desse aumento é a sobrecarga das máquinas do tipo A ao executarem simultaneamente os nós da rede sob teste e os *workers* do agente de testes, o que pode ter impactado no processamento das transações.

Nos testes com o contrato *query*, a média do *throughput* seguiu continuamente a escala da rede, porém, com 4 e 8 nós observou-se uma variação significativa em seu desvio padrão. Uma possível causa desse aumento é a sobrecarga das máquinas do tipo A como já explicado anteriormente.

Por fim, nos testes com o contrato *transfer* a média seguiu a escala da rede de 1 a 16 nós, tendo uma pequena variação apenas com 16 nós, seu desvio padrão manteve-se estável nos testes de 1, 2 e 8 nós, mas mostrou variação nos testes realizados com 4 e 16 maquinas. Uma possível causa desse aumento é além da sobrecarga das maquinas do tipo A, os testes com o contrato *transfer* foram sempre os últimos a serem realizados em cada rodada, pode-se observar que as variações de desempenho ocorridas em qualquer um dos testes anteriores refletiram em uma variação do *throughput* do teste aumentando assim seu desvio padrão.

Analisando o conjunto de informações apresentadas na Tabela 6 é possível tirar algumas conclusões. Nos testes com o contrato *open* e com o contrato *transfer* é possível notar um comportamento bem parecido, pois, a média da latência se manteve estável nas escalas de 1 a 8 nós, tendo variação significativa apenas com 16 nós, já seu desvio padrão teve variação apenas nos testes de 1 e 8 nós. Uma possível justificativa para o crescimento é a sobrecarga das máquinas do tipo A, o executarem simultaneamente os nós da rede sob teste e os *workers* do agente de testes, o que pode ter impactado na quantidade de tempo para a requisição ser concluída. No

Tabela 6 – Latência (s) × Número de nós

| nós | open | | query | | transfer | |
|---|---|---|---|---|---|---|
| | média | desvio padrão | média | desvio padrão | média | desvio padrão |
| 1 | 16.37 | 10.33 | 0.00 | 0.00 | 14.89 | 5.57 |
| 2 | 32.06 | 1.97 | 0.00 | 0.00 | 17.29 | 1.89 |
| 4 | 32.19 | 1.35 | 0.01 | 0.01 | 27.86 | 4.94 |
| 8 | 37.29 | 12.19 | 0.06 | 0.08 | 38.45 | 13.00 |
| 16 | 7.84 | 2.17 | 0.01 | 0.00 | 8.23 | 1.29 |

Fonte: Autores

caso da significativa diminuição da média latência, como o teste de 16 nós foi realizado com 8 máquinas do tipo A e 8 máquinas do tipo B, mesmo com a sobrecarga das máquinas do tipo A, a compensação da rede pode ser devida às máquinas do tipo B, diminuindo a sobrecarga individualmente em cada nó e, consequentemente, processando as requisições de uma maneira mais efetiva.

Nos testes com o contrato *query* observou-se um comportamento constante tanto no desvio padrão como na média em todos os testes. Uma possível justificativa é a simplicidade do processo realizado no contrato *query*, consumindo pouco da rede e portanto mantendo-se estável em todo processo.

# 7 CONCLUSÃO

Pela estrutura descentralizada de redes *blockchain*, pode-se inferir, intuitivamente, que a escalabilidade é um atributo inerente a elas, considerando que o aumento no número de nós na rede facilita a propagação de transações. Através deste experimento, observou-se uma melhora de desempenho com a escalada da rede a mais nós, refletida no aumento de *throughput* coletado nos testes utilizando 1, 2, 4 e 8 máquinas. No teste com 16 máquinas, apesar de ter sido observada uma queda no *throughput*, em relação aos testes anteriores, esta queda é acompanhada por um queda mais acentuada na latência. Isto mostra que, apesar de haver uma perda no *throughput*, a queda mais acentuada da latência indica um aumento no desempenho geral da rede.

Para elaborar melhor este ponto, pode-se pensar no cenário observado como um *trade-off*. Analisando as transações de transferência, nos testes com 1, 2, 4 e 8 máquinas, houve um aumento gradual no *throughput*, partindo de uma média de 42.92 transações por segundo, nos testes com 1 nó, para 127.03 transações por segundo, nos testes com 8 nós, quase o triplo do *throughput* centralizado. Nestes mesmos testes, a latência média das transações de transferência tem um aumento de, aproximadamente, 2.7 vezes. Agora, observando o que acontece quando se aumenta o número de nós de 8 para 16, nota-se uma queda acentuada na latência média, que variou de 38.45 para 8.23 segundos, enquanto a queda no *throughput* foi menos expressiva, variando de 127.03 para 107.23. Isso mostra que os ganhos de desempenho com a queda de latência foram mais expressivos do que a perda de *throughput*, o que resulta em um *trade-off* positivo para o desempenho da rede.

Nas transações de abertura de conta (*open*), esta tendência é mais evidente. Nela, o *throughput* variou de 12.53 para 127.53 nos testes de 1 à 16 máquinas, sempre com ganho de *throughput* a medida que se aumentou o número de máquinas. Já a latência variou de 16.37 até 37.29 segundos nos testes de 1 à 8 máquinas, caracterizando uma perda inicial de desempenho, porém, com 16 nós, observa-se uma queda para 7.84 segundos, quase um quinto do seu valor mais alto, o que representa um ganho de desempenho com relação a todas as escalas anteriores, desde apenas 1 nó até 8 nós em funcionamento.

Observa-se, então, uma relação diretamente proporcional entre desempenho e escalabilidade na rede *blockchain* testada, o que dá indícios, assim, de uma tendência positiva entre desempenho e escala para aplicações distribuídas implementadas neste contexto.

# REFERÊNCIAS

AL-BASSAM, Mustafa et al. Chainspace: A Sharded Smart Contracts Platform. In: PROCEEDINGS 2018 Network and Distributed System Security Symposium. San Diego, CA, USA: Internet Society, fev. 2018. ISBN 1-891562-49-5. DOI: 10.14722/ndss.2018.23241. Disponível em: https://doi.org/10.14722/ndss.2018.23241.

ANDROULAKI, Elli et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In: PROCEEDINGS of the Thirteenth EuroSys Conference. Porto, Portugal: Association for Computing Machinery, 2018. (EuroSys '18). ISBN 9781450355841. DOI: 10.1145/3190508.3190538. Disponível em: https://doi.org/10.1145/3190508.3190538.

BELHADI, Asma et al. A general-purpose distributed pattern mining system. **Applied Intelligence**, v. 50, n. 9, p. 2647–2662, 2020. ISSN 1573-7497. DOI: 10.1007/s10489-020-01664-w. Disponível em: https://doi.org/10.1007/s10489-020-01664-w.

BUTERIN, Vitalik. **Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform.** [S.l.]: Ethereum Organizaiton, 2014. Disponível em: https://ethereum.org/669c9e2e2027310b6b3cdce6e1c52962/Ethereum_Whitepaper_-_Buterin_2014.pdf.

CALIPER, Project. **Architecture**. [S.l.]: Hyperledger Foundation, jun. 2020. Disponível em: https://hyperledger.github.io/caliper/v0.3.2/architecture/.

EYAL, Ittay et al. Bitcoin-NG: A Scalable Blockchain Protocol. In: 13TH USENIX Symposium on Networked Systems Design and Implementation (NSDI 16). Santa Clara, CA: USENIX Association, mar. 2016. P. 45–59. ISBN 978-1-931971-29-4. Disponível em: https://www.usenix.org/conference/nsdi16/technical-sessions/presentation/eyal.

FRANCESCO, Paolo Di; MALAVOLTA, Ivano; LAGO, Patricia. Research on Architecting Microservices: Trends, Focus, and Potential for Industrial Adoption. In: 2017 IEEE International Conference on Software Architecture (ICSA). Gothenburg, Sweden: IEEE, 2017. P. 21–30. ISBN 978-1-5090-5729-0. DOI: 10.1109/ICSA.2017.24. Disponível em: https://doi.org/10.1109/ICSA.2017.24.

GERVAIS, Arthur et al. On the Security and Performance of Proof of Work Blockchains. In: PROCEEDINGS of the 2016 ACM SIGSAC Conference on Computer and Communications Security. Vienna, Austria: Association for Computing Machinery, 2016. (CCS '16), p. 3–16. ISBN 9781450341394. DOI: 10.1145/2976749.2978341. Disponível em: https://doi.org/10.1145/2976749.2978341.

GILAD, Yossi et al. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. In: PROCEEDINGS of the 26th Symposium on Operating Systems Principles. Shanghai, China: Association for Computing Machinery, 2017. (SOSP '17), p. 51–68. ISBN 9781450350853. DOI: 10.1145/3132747.3132757. Disponível em: https://doi.org/10.1145/3132747.3132757.

LI, Chenxing et al. A Decentralized Blockchain with High Throughput and Fast Confirmation. In: PROCEEDINGS of the 2020 USENIX Annual Technical Conference (USENIX ATC 20).

USA: USENIX Association, jul. 2020. (USENIX ATC'20), p. 515–528. ISBN 978-1-939133-14-4. DOI: 10.5555/3489146.3489181. Disponível em: https://www.usenix.org/conference/atc20/presentation/li-chenxing.

NAKAMOTO, Satoshi. **Bitcoin: A Peer-to-Peer Electronic Cash System**. [S.l.]: Bitcoin Project, 2008. Disponível em: https://bitcoin.org/bitcoin.pdf.

PODGORELEC, Blaž; TURKANOVIĆ, Muhamed; KARAKATIČ, Sašo. A Machine Learning-Based Method for Automated Blockchain Transaction Signing Including Personalized Anomaly Detection. **Sensors**, Basel, Switzerland, v. 20, n. 1, 2020. ISSN 1424-8220. DOI: 10.3390/s20010147. Disponível em: https://www.mdpi.com/1424-8220/20/1/147.

ROOS, Stefanie et al. Settling payments fast and private: Efficient decentralized routing for path-based transactions. In: PROCEEDINGS 2018 Network and Distributed System Security Symposium. San Diego, CA, USA: Internet Society, fev. 2018. ISBN 1891562495. DOI: 10.14722/ndss.2018.23252. Disponível em: https://doi.org/10.14722/ndss.2018.23252.

SOMPOLINSKY, Yonatan; ZOHAR, Aviv. Secure High-Rate Transaction Processing in Bitcoin. In: BÖHME, Rainer; OKAMOTO, Tatsuaki (Ed.). **Financial Cryptography and Data Security**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015. Springer, p. 507–527. ISBN 978-3-662-47854-7. DOI: 10.1007/978-3-662-47854-7_32.

SRIVASTAVA, Gautam et al. A Pre-Large Weighted-Fusion System of Sensed High-Utility Patterns. **IEEE Sensors Journal**, v. 21, n. 14, p. 15626–15634, 2021. DOI: doi:10.1109/JSEN.2020.2991045.

THAKKAR, Parth; NATHAN, Senthil; VISWANATHAN, Balaji. Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform. In: 2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS). Milwaukee, WI, USA: IEEE, 2018. P. 264–276. DOI: 10.1109/MASCOTS.2018.00034. Disponível em: https://doi.org/10.1109/MASCOTS.2018.00034.

XIAO, Yang et al. A Survey of Distributed Consensus Protocols for Blockchain Networks. **IEEE Communications Surveys & Tutorials**, v. 22, n. 2, p. 1432–1465, 2020. DOI: 10.1109/COMST.2020.2969706.