Avaliação de Desempenho de Blockchains Permissionadas Hyperledger Orientada ao Planejamento de Capacidade de Recursos Computacionais

Francisco A. Silva[1], **Glauber D. Gonçalves**[1], **Iure fé**[1]
Leonel Feitosa[1] **e André Soares**[1]

[1]Universidade Federal de Piauí – PI – Brasil

{leonel,faps,ggoncalves,iure.fe,andre.soares}@ufpi.edu.br

Resumo. *Hyperledger Fabric é uma plataforma para redes blockchains permissionadas que permite o armazenamento e o acesso distribuído a dados de forma segura e auditável para aplicações corporativas. Existe um crescente interesse por aplicações dessa plataforma, mas o seu uso requer a configuração de uma blockchain com diferentes etapas de processamento de requisições. As diversas configurações possíveis impactam nas qualidades não funcionais da plataforma, em especial desempenho e custo. Este artigo propõe um modelo de Rede de Petri Estocástica (SPN) para modelar o desempenho de requisições na plataforma Hyperledger Fabric com variadas parametrizações para blockchain, capacidade de computadores e taxas de requisições. Apresentamos também um estudo de utilização do modelo que serve como uma exemplificação para auxiliar os administradores de redes blockchains permissionadas a adequar suas configurações encontrando o melhor desempenho para aplicações. O modelo permitiu, por exemplo, identificar o tamanho do bloco que leva a um tempo médio de resposta excessivamente alto (variando de 1 a 25 segundos) causado por alto enfileiramento de requisições.*

Abstract. *Hyperledger Fabric is a platform for permissioned blockchain networks that enables secure and auditable distributed data storage and access for enterprise applications. There is a growing interest in applications of this platform, but its use requires the configuration of a blockchain with different stages of request processing. The various possible configurations impact the non-functional qualities of the platform, especially performance and cost. This article proposes a Stochastic Petri Net (SPN) to model the performance of requests on the Hyperledger Fabric platform with different parameterizations for blockchain, computer capacity and request rates. We also present a study of the use of the model that serves as a practical guide to help administrators of permissioned blockchain networks to adapt their configurations, finding the best performance for applications. The model allowed, for example, to identify the block size that leads to an excessively high mean response time (ranging from 1 to 25 seconds) caused by high requests queuing.*

1. Introdução

Blockchain é uma tecnologia disruptiva particularmente para o setor produtivo, i.e., indústria e serviços, pois fornece recursos para o registro público, seguro e descentralizado de dados. O principal propósito da tecnologia blockchain é permitir o registro

de transações entre duas entidades, ou seja, pessoas ou organizações, que podem não se conhecer, e assim não terem confiança mútua. Dentre as principais propriedades de blockchain para o registro de transações destacam-se imutabilidade, auditabilidade e descentralização de dados. Essas propriedades resultam da unificação de diferentes tecnologias para o desenvolvimento de sistemas distribuídos como criptografia assimétrica, protocolos de consenso e redes par-a-par.

Contudo, a tecnologia blockchain precisa avançar no aspecto de desempenho para consolidar sua aplicação ao setor produtivo, que usualmente requer transações de baixa latência, semelhantes às redes de cartões de crédito, o que não é observado atualmente nas populares redes públicas de cripto ativos Bitcoin e Ethereum [Sousa et al. 2021]. Nesse sentido, arquiteturas blockchain em redes privadas (ou permissionadas) vem sendo propostas [Androulaki and et al. 2018]. Hyperledger Fabric é uma das plataformas para blockchains permissionadas mais populares atualmente[1] com recursos para a implantação de uma rede entre organizações e desenvolvimento de aplicações no topo dessa rede. Nesse caso, os participantes da rede formam um consórcio e arcam com o custo da infraestrutura, buscando melhor desempenho em relação às redes blockchain públicas.

Um aspecto importante em redes blockchains permissionadas é o seu desempenho, em especial, atraso médio de uma requisição para registrar dados na blockchain e vazão da rede blockchain em termos de requisições por segundo. A plataforma Hyperledger Fabric em particular pode alcançar diferentes níveis de desempenho via variações na infraestrutura da rede e na estrutura da blockchain [Guggenberger et al. 2022]. A infraestrutura é formada por computadores organizados em rede par-a-par que executam etapas no protocolo de consenso distribuído da plataforma, e de acordo a etapa necessitam de diferentes capacidades computacionais. Por sua vez, a estrutura da blockchain é impactada especialmente por dois parâmetros que são o tamanho do bloco em quantidade de transações e o tempo limite para geração de novos blocos (*timeout*).

Neste trabalho propomos o uso de Rede de Petri Estocástica (SPN) para modelar o desempenho de blockchains Hyperledger Fabric com diferentes parametrizações. SPNs são conhecidas pelo alto grau de representatividade, sendo mais intuitivos que opções convencionais, como cadeias de Markov, para representar concorrência, paralelismo, e sincronização em sistemas [Silva et al. 2017, Pinheiro et al. 2019, Rodrigues et al. 2019, Ferreira et al. 2019, Silva et al. 2022]. Os trabalhos relacionados na Seção 2 propuseram modelos para analisar a disponibilidade e custos na implantação [Melo et al. 2022, Melo et al. 2021], identificar gargalos [Xu et al. 2021, Sukhwani et al. 2018, Yuan et al. 2020] e o comportamento da rede em situações de ataque [Shahriar et al. 2020]. Nenhum desses esforços, no entanto, modela a quantidade de recursos disponíveis para executar requisições, considerando recursos de enfileiramento e processamento paralelo. Adicionalmente, as modelagens propostas não detalham as etapas de endosso, ordenação e *commit*, minimizando o complexo fluxo de requisições no protocolo Hyperledger Fabric em uma única etapa.

O modelo SPN proposto na Seção 4 oferece recursos úteis para a configuração e planejamento de blockchains permissionadas baseadas na plataforma Hyperledger Fabric, o que pode contribuir de forma relevante para o estabelecimento dessa tecnologia

[1] https://www.ibm.com/topics/hyperledger

no setor produtivo. Nesse sentido, avaliamos a partir de três casos de usos na Seção 5 os compromissos entre configurações da blockchain (tamanho e *timeout* de blocos) e capacidade computacional de componentes arquiteturais que impactam no tempo de requisição e vazão da rede. O primeiro caso mostra que a plataforma Hyperledger Fabric requer um ajuste cuidadoso da capacidade dos computadores que realizam a última etapa da requisição (*commit*), sujeito a tempos médios de respostas longos ou gasto desnecessário de recursos. Por sua vez, configurações no tamanho do bloco e *timeout* mostradas no segundo caso impactam decisivamente na etapa intermediária da requisição (ordenação), e.g., observamos variações no tempo médio de resposta entre 1 e 25 segundos ao aumentar o tamanho do bloco em apenas uma unidade. Partindo dessa observação, o modelo proposto permitiu identificar no terceiro caso momentos em que computadores da rede saturam nas etapas iniciais da requisição (endosso e ordenação) dado a interação entre o tamanho de bloco e o *timeout*, i.e., alternância entre blocos de tamanho máximo ou incompleto, o que também pode explicar a latência de requisições em Hyperledger Fabric.

2. Trabalhos Relacionados

Nesta seção apresentamos um estudo do estado da arte do tema. Diversos trabalhos nos últimos anos buscaram avaliar métricas de desempenho, disponibilidade e o comportamento das infraestruturas de implantação de sistemas Blockchain por meio da utilização de modelos. Em [Melo et al. 2022, Melo et al. 2021] os autores utilizaram modelos para analisar as métricas de disponibilidade de sistemas Blockchain. Foram elaborados modelos para serem usados como ferramentas de planejamento de provisionamento. Para isso, foram utilizadas Cadeias de Markov de Tempo Continuo (CTMC), modelos de Diagramas de Bloco de Confiabilidade (RBD) e Redes de Petri Estocásticas – SPN para modelar sistemas na plataforma de blockchain. Também foram realizadas comparações de custos na implantação da infraestrutura em ambientes privados e públicos. Os autores concluíram que os modelos propostos podem auxiliar no planejamento de aplicações de blockchain, permitindo identificar gargalos na infraestrutura e na comparação dos custos na implantação.

Outros trabalhos foram desenvolvidos para auxiliar focadamente no planejamento das características de desempenho dos sistemas blockchain no framework Fabric. Os modelos criados em [Xu et al. 2021, Sukhwani et al. 2018, Yuan et al. 2020] observaram a vazão e latência baseadas na variação da taxa de chegada, configuração do tamanho do bloco, e intervalo de bloco. [Xu et al. 2021] utilizou um modelo analítico baseado em equações e comparado com simulações, já em [Yuan et al. 2020] os autores utilizaram Redes de Petri Estocásticas Generalizadas (GSPN). [Sukhwani et al. 2018] utilizou Redes de Recompensa Estocásticas (SRN), esse modelo permitiu além das métricas de desempenho, calcular tamanho médio da fila de cada um dos nós. Os autores concluíram que o tempo para completar o endorso é afetado pelo número de nós e as políticas utilizadas. Também foram analisados os demais processos do Hyperledger Fabric identificando os gargalos e apresentando vários cenários possíveis.

Por fim, modelos também foram utilizados para estudar o comportamento de sistemas Blockchain em situações de ataque. Em [Shahriar et al. 2020] os autores usaram em Redes de Petri capazes de identificar as vulnerabilidades que podem ser exploradas e as ameaças que o sistema está exposto. Os autores em [Zhou et al. 2021] utilizaram uma rede CTMC para avaliar a disponibilidade de um sistema de bitcoin sob um ataque

Eclipse em diferentes taxas.

Nós propomos neste trabalho um modelo SPN para calcular características de desempenho da rede blockchain. Semelhante a alguns trabalhos apresentados acima, o modelo calcula as métricas: tempo médio de resposta, vazão e utilização. Como contribuições adicionais, o modelo também calcula duas métricas novas que são as taxas de acionamentos por bloco de tamanho máximo ou por *timeout*, bem como a probabilidade de descartes de requisições. Nenhum dos trabalhos modelou a quantidade de recursos disponíveis para executar as requisições. O modelo proposto neste trabalho considera tanto a capacidade de recursos de enfileiramento, quanto os recursos de processamento. Diferente dos trabalhos relacionados, também detalhamos a etapa de ordenação, dividindo-a em processamento unitário de requisições, formação de bloco e envio de bloco. Todos os trabalhos relacionados simplificam estas etapas em apenas uma ação.

3. A Plataforma Hyperledger Fabric

A plataforma para redes blockchain permissionadas Hyperledger Fabric é uma das mais populares atualmente. Ela é um projeto de código fonte aberto envolvendo mais de 35 organizações e 200 desenvolvedores.[2] Essa rede blockchain usa a estratégia executar-ordenar-validar para processar blocos de requisições. Dessa forma há a separação entre a execução e a ordenação de requisições em blocos, o que permite melhor escalabilidade e desempenho em comparação às estratégias ordenar-executar de blockchains públicas como Bitcoin e Ethereum [Androulaki and et al. 2018]. A estratégia Hyperledger Fabric pode ser organizada em três etapas essenciais para o registro de requisições em blockchain, que são endosso, ordenação e *commit*, i.e., a confirmação dos blocos de requisições.

A Figura 1 ilustra o fluxo de uma requisição partindo da aplicação cliente para a rede Hyperledger Fabric sob as três etapas mencionadas, onde cada etapa pode ser executada em vários computadores da rede. Inicialmente, a aplicação cliente envia a proposta de requisição, i.e., uma transação, para um computador participante da rede. Inicia-se então etapa de *endosso*, onde ocorre a simulação de execução da requisição na blockchain, seguido da resposta à aplicação com o endosso da requisição ou sua negativa. A aplicação aguarda endossos de outros computadores participantes da rede, conforme a quantidade configurada na rede, geralmente 50% dos participantes mais um; então envia a requisição para o computador ordenador, iniciando a etapa de *ordenação*. Nessa etapa são recolhidas requisições da rede até alcançar os valores definidos para os parâmetros tempo limite (*timeout*) ou tamanho do bloco para gerar um novo bloco. Novos blocos são encaminhados aos computadores responsáveis pela etapa de *commit*, que consiste na validação das requisições no bloco, seu encadeamento à blockchain e atualização do estado global da blockchain para consultas rápidas, e.g., variáveis das aplicações e saldos de contas.

4. Modelo SPN

Esta seção apresenta um modelo estocástico para representar e computar características de desempenho de uma rede blockchain permissionada Hyperledger Fabric, conforme a arquitetura apresentada na seção anterior. Nesse propósito, exploramos especificamente

[2]Mais informações podem ser encontradas em *https://hyperledger-fabric.readthedocs.io*.

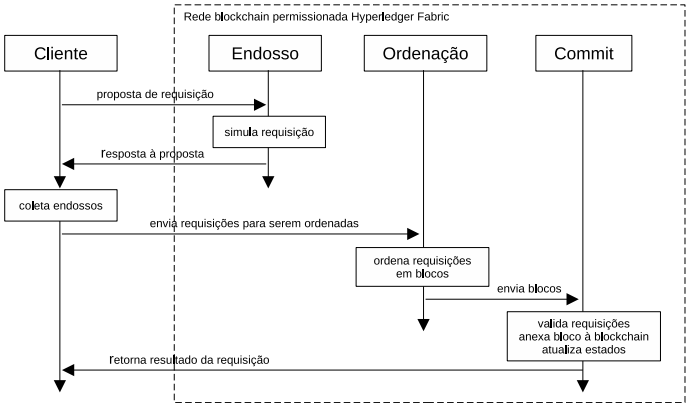

Figura 1. Fluxo de uma requisição na plataforma Hyperledger Fabric.

as Redes de Petri Estocásticas (SPN). Uma rede de Petri é um grafo bipartido direcionado que pode ser usado na modelagem e descrição de sistemas; contém componentes em sua estrutura, como conjuntos de locais e transições; e é estocástica quando cada transição está associada a um atraso de disparo aleatório que segue um processo estocástico. SPNs são conhecidos pelo alto grau de representatividade, sendo mais intuitivos que opções convencionais, como cadeias de Markov, para representar concorrência, paralelismo, e sincronização, em sistemas variados [Silva et al. 2022]. O objetivo do modelo proposto é auxiliar administradores de sistemas baseados nessa arquitetura na tarefa complexa de ajustar vários parâmetros adequadamente para alcançar níveis de desempenho desejáveis. Logo, o modelo deve ser útil para checar o efeito de mudanças no sistema, antes mesmo que elas sejam implementadas.

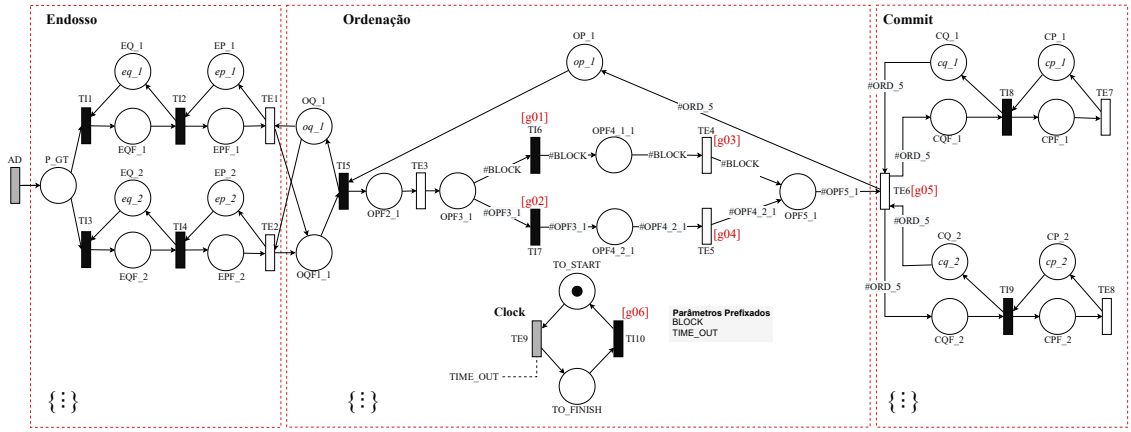

Figura 2. Modelo SPN para o processamento de transações em redes blockchain permissionadas Hyperledger Fabric.

A Figura 2 representa o modelo SPN proposto. Uma aplicação gera requisições (i.e., transações) e as transmite para a rede blockchain que é composta por três etapas macro *endosso, ordenação* e *commit*, mostradas na figura da esquerda para a direita. Vale ressaltar que essas três etapas podem ser realizadas em N computadores. Por questão de espaço, simplificamos a descrição do modelo SPN a seguir utilizando 2 computadores para o endosso e *commit* e um computador para ordenação. A aplicação é representada por uma transição de cor cinza de distribuição determinística, ou seja, a cada intervalo de

tempo (arrival delay - AD) é gerada uma nova requisição.

O *endosso* é a primeira etapa da requisição e inicia quando o token chega no local P_GT. Um dentre dois computadores (acionados através de T11 ou T13) é escolhido para processar a requisição. Cada computador é representado por dois "triângulos"com comportamentos semelhantes. O primeiro triângulo ilustra uma fila de entrada, e o segundo triângulo uma fila de processamento. A fila de entrada não possui um tempo associado. Portanto, as transições TI2 e TI4 são imediatas. Os tempos de endosso são atribuídos às transições temporizadas TE1 e TE2. Os locais nos topos dos triângulos representam a capacidade de fila ou processamento de cada computador. Por exemplo, o computador identificado com o final 1 da etapa de endosso possui um número de nós computacionais (núcleos de processamento ou contêineres) igual a EP_1. O enfileiramento ocorre quando a capacidade nos triângulos é exaurida. Nesse exemplo, caso o número de tokens de capacidade de processamento em EP_1 seja igual a zero, então haverá um enfileiramento no local EQ_1.

Na etapa seguinte, a *ordenação*, há a indicação de um único computador para processamento. A fila de entrada segue o mesmo padrão da etapa anterior e com uma capacidade igual a oq_1. No processamento da ordenação ocorre o maior risco de gargalos, pois há uma série de regras descritas a seguir para os tokens prosseguirem para a etapa final. O processamento da ordenação conta com a capacidade op_1. O token efetivamente entra na etapa de ordenação quando está no local OPF2_1. A transição TE3 indica um pré-processamento individual da requisição para torná-la apta a formar um bloco. No local OPF3_1 ocorre o acúmulo de tokens e formação de um bloco que pode ser de dois tipos (completo ou parcial). Diferentes tamanhos de bloco podem ser configurados através da variável #BLOCK. Caso o número de tokens em OPF3_1 seja igual a #BLOCK, esse tokens formam um bloco completo e seguem caminho pela transição superior TI6. Essa transição é acionada apenas se #OPF3_1 = #BLOCK. O componente Clock do modelo serve de gatilho para indicar que expirou o tempo limite (*timeout*) para formação do bloco completo. Assim, o bloco parcial de tamanho #OPF3_1 segue caminho pela transição TI7, que é acionada apenas se #TO_FINISH = 0. Os blocos (parciais ou completos) que foram formados são enfim processados nas transições TE4 e TE5. Quando o bloco chega em OPF5_1, há a reinicialização do Clock (token muda de TO_FINISH para TO_START).

A etapa de *commit* é iniciada quando o bloco formado alcança a transição TE6, sendo realizada em todos os computadores alocados para essa etapa. O *commit* é finalmente executado pelas transições TE7 e TE8. A Tabela 1 apresenta os principais elementos do modelo. A Tabela 2 mostra as condições de guarda para o funcionamento apropriado do modelo.

4.1. Métricas

O tempo médio de resposta (*MRT*) pode ser obtido a partir da Lei de Little [Jain 1990], que relaciona o número médio de requisições em progresso em um sistema ($RequestsInProgress$), a taxa de chegada ($Arrival\_Rate$) e MRT. Como mostrado anteriormente, uma requisição se subdivide em um conjunto de requisições, e é diretamente impactada pela taxa de chegada. A taxa de chegada é o inverso do tempo de chegada. Considerando a transição para tempo entre chegadas do modelo, temos

Tabela 1. Descrição dos elementos principais do modelo, incluindo transições e marcaçoes.

Tipo	Elemento	Descrição
Lugares	P_GT	Espera por novas requisições (conjuntos de trabalhos)
Transições Determinísticas	AD	Tempo de chegada entre requisições
	TE9	Tempo atribuído ao TIME_OUT
Transições Temporizadas	TE1, TE2	Tempo de processamento da fase de Endosso
	TE3	Tempo de processamento para a montagem do bloco
	TE4, TE5	Tempo de processamento do bloco
	TE6	Tempo de entrada para o Commit
	TE7, TE8	Tempo de processamento para a realização do Commit
Transições Imediatas	TI1, TI3	Entrada para a fila do Edosso
	TI2, TI4	Entrada para o processamento no Edosso
	TI5	Entrada para a fila da Ordenação
	TI6	Entrada para realizar a ordenação após atingir o tamanho do bloco
	TI7	Entrada para realizar a ordenação após atingir o clock do TIME_OUT
	TI8, TI9	Entrada para o processamento do Commit
	TI10	Restart do TIME_OUT
Marcações dos Lugares	eq1_1, eq1_2	Capacidade associada a fila do Endosso
	ep1_1, ep1_2	Capacidade associada ao processamento do Endosso
	oq_1	Capacidade associada a fila do Ordenador
	op_1	Capacidade associada ao processamento do Ordenador
	cq1_1, cq1_2	Capacidade associada a fila do Commit
	cp1_1, cp1_2	Capacidade associada ao processamento do Commit

Tabela 2. Expressões de guarda cujos índices estão destacados em vermelho no modelo.

Transição	Índice	Expressão
TI6	g01	(#OPF3_1>0)
TI7	g02	(#TO_FINISH=1)AND(#OPF3_1>0)
TE4	g03	(#OPF4_1_1>0)
TE5	g04	(#OPF4_1_2>0)
TE6	g05	(#OPF5_1>0)
TI10	g06	(#OPF5_1>0)

que $Arrival\_Rate = \frac{1}{Arrival\_Delay}$. Vale ressaltar que a Lei de Little requer um sistema estável, ou seja, que possua uma taxa de requisições menor que a taxa de processamento dos servidores. Portanto, a equação correspondente à Lei de Little para MRT utilizada no nosso modelo é expressa na Equação 1.

$$\textbf{MRT} = \frac{RequestsInProgress}{Arrival\_Rate} \tag{1}$$

A Equação 2 define $RequestsInProgress$ especificamente para o modelo apresentado. Para calcular o número de requisições em progresso no sistema, precisa-se somar a quantidade de tokens em cada um dos locais que representam uma requisição em andamento. Na equação 2, $Esp(Local)$ representa a esperança estatística de existir tokens em "Local", onde $Esp(Local) = (\sum_{i=1}^{n} P(m(Local) = i) \times i)$. Em outras palavras, $Esp(Lugar)$ indica quantos tokens ocupam aquele Local.

$$
\begin{aligned}
\textbf{RequestsInProgress} &= RequestsInProgress\_End + \\
&RequestsInProgress\_Ord + RequestsInProgress\_Comm \\
\textbf{RequestsInProgress\_End} &= Esp(P\_GT) + Esp(EQF\_1) + Esp(EPF\_1) + \\
&Esp(EQF\_2) + Esp(EPF\_2) \\
\textbf{RequestsInProgress\_Ord} &= Esp(OQF1\_1) + Esp(OPF2\_1) + Esp(OPF3\_1) + \\
&Esp(OPF4\_1\_1) + Esp(OPF4\_2\_1) + Esp(OPF5\_1) \\
\textbf{RequestsInProgress\_Comm} &= Esp(CFQ\_1) + Esp(CFP\_1) + Esp(CFQ\_2) + \\
&Esp(CFP\_2)
\end{aligned}
\tag{2}
$$

A Equação 3 define a probabilidade de haver perdas de requisições (DP_PROB). Para calcular o descarte é necessário não restar mais nenhuma capacidade de enfileiramento na entrada do sistema. $P(Local = n)$ calcula a probabilidade de existirem n tokens em "Local". O símbolo $\wedge$ indica um AND lógico.

$$
\textbf{DP\_PROB} = P((EQ\_1 = 0) \wedge (EQ\_2 = 0)) \tag{3}
$$

O nível de utilização de recursos é calculado para cada triângulo (fila ou processamento). Nós focamos na utilização da parte de processamento por questão de exemplificação da simulação apresentada a seguir. Assim, a utilização é a divisão do número esperado de tokens em um local (por onde passam os tokens executados) pela respectiva capacidade total. Por exemplo, a utilização média do processamento da etapa de endosso para o computador com identificador 1 é dada pela Equação 4. A utilização da etapa como um todo é dada pela média das utilizações dos computadores executados naquela etapa.

$$
\textbf{U\_END\_1} = \frac{Esp(EP\_1)}{ep\_1} \tag{4}
$$

A vazão do sistema é representada pela taxa de saída de elementos que entraram no sistema. Observa-se o ponto de saída do sistema, que no nosso caso é o *commit*, e calcula-se a taxa de saída em tal etapa. Note que a etapa de *commit* é feita paralelamente por N computadores. Assim, a vazão é dada pela média das vazões das máquinas que participam na etapa. Observando um "triângulo", a vazão é dada pela divisão do número esperado de tokens em execução dividido pelo tempo de serviço da transição temporizada subsequente. Assim, a vazão para o computador com identificação 1 no modelo apresentado é dada pela Equação 5. O tempo alocado a uma transição é dado por t(Transição).

$$
\textbf{TP\_1} = \frac{Esp(CPF\_1)}{t(TE7)} \tag{5}
$$

A bifurcação de caminhos a partir do local `OPF3_1` foi feita para obter a taxa de acionamento por bloco completo ou *timeout* alcançado. Tais taxas seguem o mesmo padrão exemplificado anteriormente para a vazão. Por fim apresentamos duas métricas inerentes ao contexto de blockchain: BLOCK_CALL_RATE e TIME_OUT_CALL_RATE. Tais métricas são dadas respectivamente pelas equações 6 e 7. A partir delas pode-se analisar os compromissos entre o *timeout* e o tamanho do bloco e calibrá-los adequadamente para diferentes cenários.

$$\textbf{BLOCK\_CALL\_RATE} = \frac{Esp(OPF\_1\_1)}{t(TE4)} \tag{6}$$

$$\textbf{TIME\_OUT\_CALL\_RATE} = \frac{Esp(OPF\_2\_1)}{t(TE5)} \tag{7}$$

5. Resultados

Nesta seção apresentamos três estudos de caso. Cada estudo de caso possui um objetivo bem específico, focando em um ou dois parâmetros em particular. Os estudos de caso são úteis tanto para obter novas descobertas sobre o funcionamento do modelo e consequentemente do sistema real, como também ilustrar como o modelo pode ser explorado. A representação do modelo e a computação dos resultados da análise numérica foram obtidos com a ferramenta Mercury [Maciel et al. 2017]. Nos estudos de caso utilizamos a mesma quantidade de computadores representados no modelo apresentado na Seção 4. Para os parâmetros do modelo, utilizamos inicialmente os valores mostrados na Tabela 3 para capacidade de fila (eq, oq e cq), capacidades de processamento para marcações de endosso (ep), ordenação (op) e commit (cp) e tempos de serviço nas transições (TE). Tais parâmetros foram baseados em testes reais com a plataforma Hyperledger e consideramos as capacidades de processamento como número de computadores, usualmente contêineres na plataforma, e capacidade de filas como posições de memória totais para a referida etapa. Vale ressaltar que esses parâmetros podem ser diferentes para cada realidade a depender do poder computacional que o analista está avaliando.

Tabela 3. Parâmetros iniciais de configuração do modelo usados nos estudos de caso.

Tipo	Parâmetros	Valor
Capacidade	ep_1, ep_2, op_1, cp_1, cp_2	6
	eq_1, eq_2, oq_1, cq_1, cq_2	100
Tempo	TE1, TE2, TE3	5 ms
	TE6	10 ms
	TE7, TE8	80 ms
	TE4, TE5	2 ms

5.1. Estudo de Caso 01 - Variação da Capacidade de Commit

Neste primeiro estudo de caso variamos a capacidade de processamento dos computadores da etapa de *commit*. Testamos três possibilidades de quantidade de contêineres disponíveis em cada um dos dois computadores. As marcações no modelo responsáveis pelo processamento no *commit* são respectivamente `cp_1` e `cp_2`. Os valores para tais marcações foram variados da seguinte forma: `cp_1` = `cp_2` = $[2, 4, 6]$. Para obter uma visão macro, variamos também a taxa de chegada com valor mínimo de 0.0025 req/ms, e valor máximo de 0.3 req/ms e incremento de 0.01565 req/ms. Para fins de análise utilizamos uma requisição por bloco, então configuramos um *timeout* alto (`TIME_OUT`=10000ms) e um tamanho de bloco baixo (`BLOCK`=1). Com tais valores, todo o fluxo do modelo passa pela formação de bloco completo.

A Figura 3 apresenta os resultados do estudo de caso 01, onde oito gráficos mostram o grande impacto da variação dos parâmetros em questão. Os três gráficos de

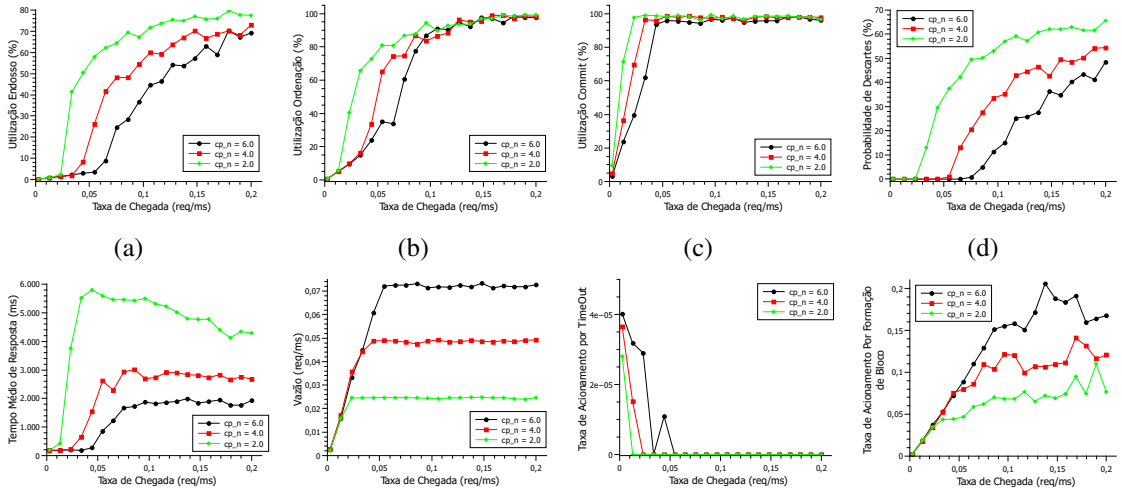

Figura 3. Estudo de Caso 01 - Variação da Capacidade do Commit - (a) Utilização - Endosso (b) Utilização - Ordenação (c) Utilização - Commit (d) Probab. de Descartes (e) MRT (f) Vazão (g) Taxa de Acionamento por Timeout (h) Taxa de Acionamento por Bloco

utilização (Figura 3a, 3b e 3c) possuem comportamentos semelhantes de crescimento em função do aumento da taxa de chegada. No entanto, a utilização do *commit* atinge o pico mais rapidamente em função da variação da capacidade computacional. Outra observação relevante é que a mudança da capacidade não impacta na utilização após determinadas taxas de chegada, por exemplo, 0.05 req/ms na etapa de *commit*. Tal observação é útil para orçamentos de equipamentos mais adequados à uma demanda específica. A probabilidade de descartes (Figura 3d) é calculada em função da utilização do ponto de entrada do sistema, no caso o endosso. Assim, observa-se que os gráficos Figura 3a e Figura 3d são semelhantes. Caso o componente de endosso alcance alto nível de utilização, todos os componentes internos do sistema também estarão sobrecarregados e assim haveremos alta probabilidade de descartes. Para o MRT (Figura 3e), comparando `cp_n` = 4 e `cp_n` = 6, temos que até a taxa de chegada igual 0.05 req/ms há pouca diferença (abaixo de 1000 ms). O MRT de `cp_n` = 2 é bem acima dos demais e há uma queda nesta configuração pois o número de requisições no sistema é tão alto que a taxa de chegada passa a ter um impacto maior sobre o MRT. A vazão (Figura 3f) cresce em um ritmo constante até estagnar. A estagnação ocorre em função da utilização do commit (Figura 3c). No momento que a utilização do commit atinge 100% de utilização não tem como aumentar mais. A taxa de acionamento por timeout (Figura 3g) possui valores muito abaixo de zero pois configuramos um timeout alto (10000ms) para que o fluxo de dados não seguisse por este caminho, mas sim pelo caminho da formação de bloco completo. A taxa de acionamento por bloco (Figura 3h) seria proporcional à taxa de chegada se não existisse a restrição de recursos principalmente na etapa de *commit*. A taxa de acionamento por bloco com `cp_n`=2 está abaixo das demais pois menos requisições passam.

Como descobertas principais desse estudo de caso podemos destacar que a etapa de *commit* tem um papel fundamental e deve ter sua capacidade cuidadosamente configurada. Caso haja uma alta restrição de sua capacidade, rapidamente o desempenho do sistema cairá refletindo em todas as métricas. Por outro lado, apesar de ter um aumento

proporcional nas três capacidades analisadas, temos que o MRT com cp_n=4 e cp_n=6 são muito semelhantes e portanto o analista pode optar por usar uma capacidade cp_n=4 e economizar recursos.

5.2. Estudo de Caso 02 - Impactos Individuais do Tamanho de Bloco e Timeout

Neste segundo estudo de caso procuramos observar o comportamento das métricas segundo duas óticas. Primeiro variamos o tamanho do bloco fixando um *timeout* alto (TIME_OUT=10000ms). Segundo, variamos o *timeout* fixando um tamanho de bloco alto (BLOCK=10). O objetivo desse estudo de caso é analisar de forma individual o impacto de cada um dos dois parâmetros. Assim, fixamos a taxa de chegada em 0.1 req/ms e usamos demais parâmetros seguindo a Tabela 3. A Figura 4 apresenta os resultados.

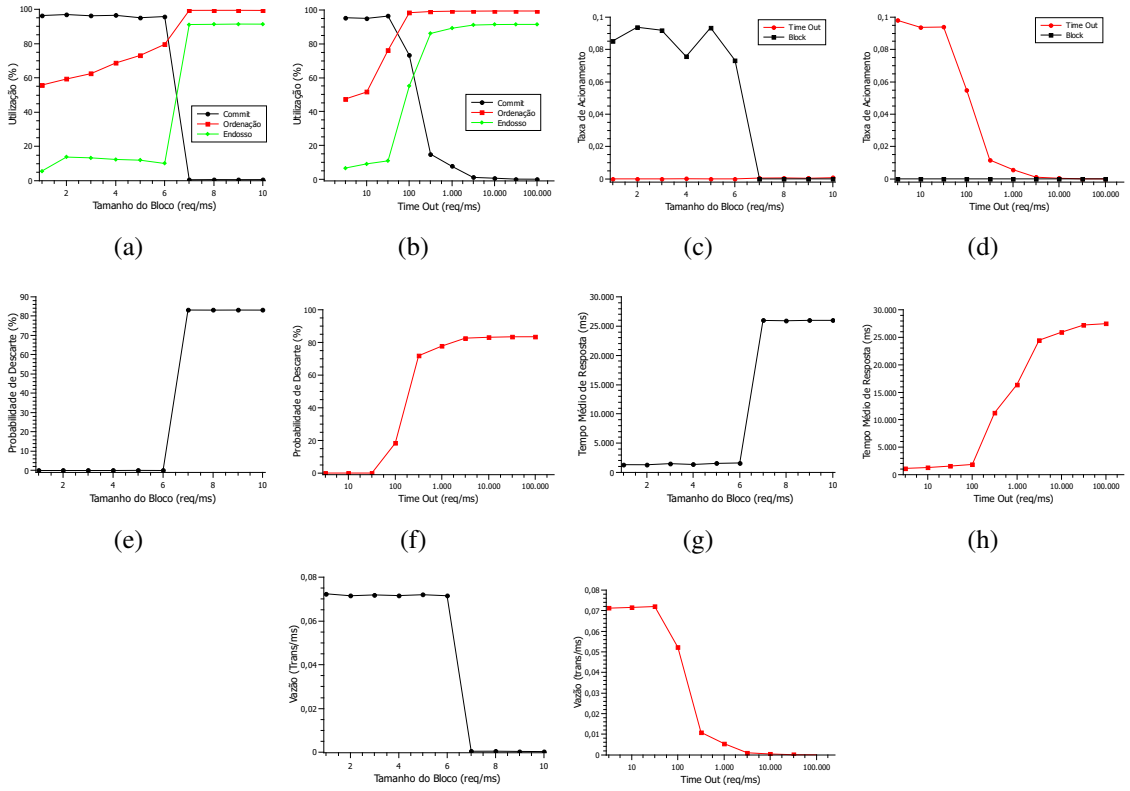

Figura 4. Resultados do Estudo de Caso 02 Variando Tamanho do Bloco e Time-Out - (a) Utilização - Variando Bloco (b) Utilização - Variando TimeOut (c) Taxa de Acionamento por Bloco (d) Taxa de Acionamento por TimeOut (e) Probab. de Descartes - Variando Bloco (f) Probab. de Descartes - Variando TimeOut (g) MRT - Variando Bloco (h) MRT - Variando TimeOut (i) Vazão - Variando Bloco (j) Vazão - Variando TimeOut

Para a utilização (Figura 4a e 4b), pode-se observar que no endosso e ordenação há um aumento da utilização proporcional ao aumento do bloco e *timeout*. No entanto, a variação de *timeout* leva ao pico de utilização para endosso e ordenação mais rapidamente do que a variação de bloco. Isso indica que para tais parâmetros, o timeout possui um impacto maior sobre utilização do que tamanho de bloco. A utilização do commit cai com o aumento do gargalo pois menos requisições passam a chegar nesta etapa. Novamente,

a queda de utilização do *commit* por *timeout* acontece antes do que com bloco pois a variação de *timeout* faz exaurir os recursos mais previamente. Por sua vez, a queda da utilização do *commit* na variação de bloco é mais brusca (de 100 a 0%) do que na variação de *timeout*. Pode-se observar, portanto, que a mudança de uma unidade no tamanho do bloco pode ter grande impacto sobre tal métrica. Para a taxa de acionamento (Figura 4c e 4d), como há uma observação focada em um parâmetro por vez, um dos parâmetros possui taxa de acionamento sempre nula. Para ambos os casos, quando se aumenta o gargalo, a taxa de acionamento é reduzida. Na variação de bloco, a taxa de acionamento se anula com bloco de tamanho 7. Na variação de *timeout*, a taxa de acionamento vai a zero com *timeout* igual a 5000ms. Vale observar que tamanho de bloco igual a 7 e *timeout* igual 5000ms são pontos de inflexão em todos os gráficos, ou seja, o respectivo valor passa a ser nulo ou atinge um pico nestes pontos. A probabilidade de descartes (PD) e o tempo médio de resposta (MRT) aumentam à metida que o gargalo aumenta pois o tempo de espera em fila aumenta. PD (Figura 4e e 4f) chegou a 80% em ambos casos. O MRT (Figura 4g e 4h) variou entre 1000ms e 25000ms. A vazão (TP) (4i e 4j), semelhante à utilização do commit, cai com o aumento do gargalo. O valor inicial de TP é 0.07req/ms. Este valor está abaixo da taxa de chegada (0.1 req/ms) pois o sistema possui alguma sobregarga em todas as suas subpartes. Para constatar isso, basta observar a utilização do commit em 100% nos pontos iniciais (4a e 4b).

Como principais observações para este segundo estudo podemos destacar que ambos parâmetros (tamanho de bloco e *timeout*) possuem um alto impacto sobre o comportamento do sistema de modo individual. O aumento de tais parâmetros causa gargalos significativos na etapa de ordenação. Esse gargalo vai aumentando até chegar a um ponto máximo que chamamos de ponto de inflexão, que para este estudo foi causado no bloco de tamanho 7 e *timeout* em 5000ms. A alteração do tamanho do bloco em apenas uma unidade pode fazer alterar em grande medida ao MRT. Alterando o tamanho do bloco de 6 para 7, fez alterar o MRT de 1 para 25 segundos.

5.3. Estudo de Caso 03 - Interação entre Bloco e TimeOut

No estudo de caso anterior nós observamos o comportamento da variação dos dois parâmetros de tamanho de bloco e *timeout* sem que tais parâmetros interferissem um no outro. Dessa forma foi possível perceber como cada um dos parâmetros interferiu nas métricas estudadas. Quando variamos anteriormente o tamanho do bloco, fixamos o *timeout* em 10000ms, assim a taxa de acionamento por *timeout* permaneceu nula.

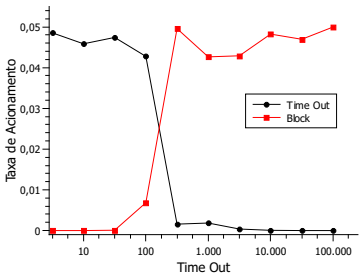

Figura 5. Interação entre Bloco e TimeOut

Agora, no terceiro estudo de caso variamos o *timeout* com a mesma faixa de valores do estudo anterior fixando o tamanho de bloco com valor mais baixo (`BLOCK` = 6). O objetivo desta vez foi obter um resultado de interação entre as duas taxas de acionamento. A Figura 5 apresenta tal resultado. O gráfico apresenta um cruzamento das duas linhas. À medida que se aumenta o timeout, a taxa de acionamento por timeout cai, pois, este caminho se torna mais restritivo. Como vai demorar mais para acionar por timeout, isso dá mais chance de o bloco completo ser formado. Pesquisadores de re-

des blockchain relatam que é algo complexo calibrar estes dois parâmetros justamente por esta possível interação [Thakkar et al. 2018, Sukhwani et al. 2018]. O presente modelo contribui, portanto, com esta previsão de comportamento do sistema.

Para este último estudo algumas conclusões interessantes podem ser feitas, como por exemplo: (i) o modelo permite determinar qual o ponto onde as duas linhas se cruzam, ou seja, o ponto onde mais blocos completos passam a ser formados do que blocos parciais; (ii) a taxa de acionamento por bloco atinge uma estabilidade em determinado ponto pois o sistema atingiu alto grau de enfileiramento. (iii) para um `TIME_OUT` = 10ms, apenas blocos parciais serão formados; (iv) para um `TIME_OUT` = 10000ms, apenas blocos completos (de tamanho 6) serão formados. A alternância entre bloco de tamanho máximo ou incompleto pode a aumentar a variabilidade do tempo de requisição dificultando a sua previsão. Cabem aos administradores da rede blockchain permissionada configurar o tamanho de bloco e *timeout* de interesse para a taxa de requisição observada na rede.

6. Conclusão

Neste trabalho foi proposto um modelo SPN para analisar o desempenho de uma blockchain permissionada na plataforma Hyperledger Fabric. O modelo calcula as métricas tempo médio de resposta, vazão e utilização, bem como a probabilidade de descartes de requisições, em função de variações em diferentes parâmetros da blockchain como tamanho e *timeout* de bloco, além de variações na taxa de chegada de requisições. Adicionalmente, o modelo proposto considera variações na capacidade de recursos de enfileiramento e recursos de processamento. Três casos de usos foram utilizados para analisar os compromissos entre configurações da blockchain (tamanho e *timeout* de blocos) e capacidade computacional de componentes arquiteturais que impactam no atraso e vazão da rede. Esses estudos de caso proveem uma exemplificação sobre a análise de desempenho da plataforma Hyperledger Fabric. Trabalhos futuros incluem estender o modelo para possibilitar atribuir pesos para as decisões, por exemplo, para qual conjunto de máquinas participantes da rede blockchain atribuir uma determinada etapa do protocolo com maior probabilidade e explorarmos tal funcionalidade como estudo de caso.

Referências

Androulaki, E. and et al. (2018). Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. In *Proc. of the EuroSys Conference*.

Ferreira, L., da Silva Rocha, E., Monteiro, K. H. C., Santos, G. L., Silva, F. A., Kelner, J., Sadok, D., Bastos Filho, C. J., Rosati, P., Lynn, T., et al. (2019). Optimizing resource availability in composable data center infrastructures. In *2019 9th Latin-American Symposium on Dependable Computing (LADC)*, pages 1–10. IEEE.

Guggenberger, T., Sedlmeir, J., Fridgen, G., and Luckow, A. (2022). An in-depth investigation of the performance characteristics of hyperledger fabric. *Computers & Industrial Engineering*, 173:108716.

Jain, R. (1990). *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. John Wiley & Sons.

Maciel, P., Matos, R., Silva, B., Figueiredo, J., Oliveira, D., Fé, I., Maciel, R., and Dantas, J. (2017). Mercury: Performance and dependability evaluation of systems with

exponential, expolynomial, and general distributions. In *Proc. of PRDC*, pages 50–57. IEEE.

Melo, C., Araujo, J., Dantas, J., Pereira, P., and Maciel, P. (2022). A model-based approach for planning blockchain service provisioning. *Computing*, 104(2):315–337.

Melo, C., Dantas, J., Pereira, P., and Maciel, P. (2021). Distributed application provisioning over ethereum-based private and permissioned blockchain: availability modeling, capacity, and costs planning. *The Journal of Supercomputing*, 77(9):9615–9641.

Pinheiro, T., Silva, F. A., Fé, I., Oliveira, D., and Maciel, P. (2019). Performance and resource consumption analysis of elastic systems on public clouds. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 2115–2120. IEEE.

Rodrigues, L., Endo, P. T., and Silva, F. A. (2019). Stochastic model for evaluating smart hospitals performance. In *2019 IEEE Latin-American Conference on Communications (LATINCOM)*, pages 1–6. IEEE.

Shahriar, M. A., Bappy, F. H., Hossain, A. F., Saikat, D. D., Ferdous, M. S., Chowdhury, M. J. M., and Bhuiyan, M. Z. A. (2020). Modelling attacks in blockchain systems using petri nets. In *Proc. of TrustCom*, pages 1069–1078. IEEE.

Silva, F. A., Brito, C., Araújo, G., Fé, I., Tyan, M., Lee, J.-W., Nguyen, T. A., and Maciel, P. R. M. (2022). Model-driven impact quantification of energy resource redundancy and server rejuvenation on the dependability of medical sensor networks in smart hospitals. *Sensors*, 22(4):1595.

Silva, F. A., Kosta, S., Rodrigues, M., Oliveira, D., Maciel, T., Mei, A., and Maciel, P. (2017). Mobile cloud performance evaluation using stochastic models. *IEEE Transactions on Mobile Computing*, 17(5):1134–1147.

Sousa, J. E. d. A., Oliveira, V., Valadares, J., Dias Goncalves, G., Moraes Villela, S., Soares Bernardino, H., and Borges Vieira, A. (2021). An analysis of the fees and pending time correlation in ethereum. *International Journal of Network Management*.

Sukhwani, H., Wang, N., Trivedi, K. S., and Rindos, A. (2018). Performance modeling of hyperledger fabric (permissioned blockchain network). In *Proc. of NCA*, pages 1–8. IEEE.

Thakkar, P., Nathan, S., and Viswanathan, B. (2018). Performance benchmarking and optimizing hyperledger fabric blockchain platform. In *Proc. of MASCOTS*, pages 264–276. IEEE.

Xu, X., Sun, G., Luo, L., Cao, H., Yu, H., and Vasilakos, A. V. (2021). Latency performance modeling and analysis for hyperledger fabric blockchain network. *Information Processing & Management*, 58(1):102436.

Yuan, P., Zheng, K., Xiong, X., Zhang, K., and Lei, L. (2020). Performance modeling and analysis of a hyperledger-based system using gspn. *Computer Communications*, 153:117–124.

Zhou, C., Xing, L., and Liu, Q. (2021). Dependability analysis of bitcoin subject to eclipse attacks. *International Journal of Mathematical, Engineering and Management Sciences*, 6(2):469.