Performance Modeling and Evaluation of Hyperledger Fabric: An Analysis Based on Transaction Flow and Endorsement Policies

Carlos Melo, Glauber Gonçalves, Francisco A. Silva, and André Soares
Universidade Federal do Piauí (UFPI), Picos - PI, Brazil
{casm, ggoncalves, faps, andre.soares}@ufpi.edu.br

Abstract—Blockchain is a paradigm derived from distributed systems, protocols, and security concepts. However, can blockchain applications provide services in industrial environments, especially concerning performance issues? In blockchains, long response times can impair both user and service experience, and intensive resource use may increase the costs of service provision. The proposed paper tries to answer this question by evaluating the performance of one of the most popular permissioned blockchain platforms, the Hyperledger Fabric (HLF). We provide a framework for performance evaluation based on modeling and experimentation. The results indicate that block size and arrival rate can compromise throughput (by -70%), latency (by +1,500%), and environment utilization (by +28%) and that multiple gateways can reduce latency (by -75%), and throughput (by -60%).

Index Terms—Performance, Modeling, Hyperledger Fabric

I. INTRODUCTION

Hyperledger Fabric (HLF) is an open-source permissioned blockchain platform that has been evaluated in various industry and corporate scenarios [1]. Unlike public blockchains, such as Bitcoin and Ethereum, based on anonymous participation, HLF focuses on the network among entities that share common goals but may lack full trust in each other due to business competition. Thus, this blockchain platform requires the identification of every participant in the network organized into private communication channels.

HLF offers identifiable nodes, varying security levels for transaction execution with endorsement policies, and an architecture that executes-orders-validates transactions with modular ordering for both Crash Fault Tolerance (CFT) and Byzantine Fault Tolerance (BFT) models [1]. The HLF perform better than Bitcoin and Ethereum's order-execute-model while bolstering security [2].

Typically, transactions carried out on permissioned networks are not linked to consumption of *gas* or monetary cost. There is no charge for a transaction to be executed, which facilitates its use in private environments [3], requiring higher throughput and lower latency at the lowest possible financial cost. However, quantifying the resources necessary for these applications' provision and full utilization has become an important task related to the performance evaluation process.

Performance in the context of blockchain is a key factor used to demonstrate the viability of adopting this technology. It has become a relevant research topic in recent years, with studies focused on performance modeling and measurement [2], [4]–[11]. However, due to the volatility of this research field, most works need to consider the new versions and readjustments introduced in the context of these platforms and how these changes impact the service provisioning performance.

Also, some of these works, like [12], do not provide a model validation, while others, such as [2], do not provide generalizable performance results, attaining it to their experimental environment. This paper uses measurement and modeling to compose a generalizable performance evaluation *framework* within the scope of HLF 2.5 and beyond.

This work offers two main contributions: (i) a generalized performance model representing the HLF transaction flow and (ii) a performance evaluation through case studies to validate and demonstrate the feasibility of the proposed model. The results allow HLF service providers to adjust parameters according to their infrastructure, focusing on *throughput*, latency, and resource utilization. Furthermore, we evaluate the impact of endorsement policy and distribute endorsement on these metrics, enabling organizations that do not share mutual trust to enjoy the benefits of blockchains.

The next sections are organized as follows: Section II presents related works in modeling and performance evaluation of blockchains. Section IV introduces the base model and discusses its applications. Section III provides an overview of the transaction flow within the Hyperledger Fabric platform. Section V describes the experimental study and the validation process of the proposed model for the HLF platform. Section VI presents a set of case studies and derivations of the proposed model, as well as the main results obtained in each evaluation scenario. Finally, Section VII summarizes our final considerations, limitations, and future works.

II. RELATED WORKS

The Hyperledger Fabric platform has been extensively discussed in previous research, focusing on performance metrics. This paper seeks to update the state of the art and fill gaps left by existing studies.

In [7], we introduced models to assess the use of computational resources in the context of HLF. Using Continuous Time Markov Chains (CTMCs) and Stochastic Petri Nets (SPNs), we demonstrated the effectiveness of these formalisms in modeling and evaluating HLF-based applications, particularly in detecting infrastructure bottlenecks. However, the model proposed in this paper extends the scope of the

previous work, assessing metrics such as throughput and transaction latency not addressed by previously proposed models.

In other studies like [8]–[10], the authors modeled general system performance metrics from the perspective of HLF. Jiang et al. [9], for example, used a hierarchical modeling approach for HLF 1.4, analyzing indicators such as throughput, latency, and system utilization. Whereas Wu et al. [8] developed a queue theory-based model focused on the flow of a transaction within the scope of HLF 2.0. Ke and Park [10] proposed queue models considering different service rates.

Finally, in other works like [6], [11], the authors evaluated the influence of arrival rates and block sizes on HLF throughput and latency. Still, as we will see in the present work, these are not the only factors significantly impacting these metrics. Furthermore, in Yuan et al. [11], the authors modeled the platform using Generalized Stochastic Petri Nets (GSPN). In contrast, in Sukhwani et al. [6], the authors used Stochastic Reward Networks (SRN); both models are isomorphic to the SPNs presented in this work, that is, they have the same symbolic power; however, they do not contemplate the transaction flow for more recent versions of HLF.

This work extends those studies, especially directly extends our previous work [12], by presenting an SPN model for the Hyperledger Fabric platform, focusing on the HLF 2.5+ version addressing the concept of the gateway and the possibility of it being a single point of failure and a pre-platform bottleneck.

III. ARCHITECTURE

Figure 1 demonstrates the transaction flow in the network through a process subdivided into four stages: *proposal*, *endorsement*, *ordering*, and *commit*. The figure also shows network components, such as the client and gateway, where transactions are submitted to the blockchain, and peers that must maintain consistent replicas of the blockchain.
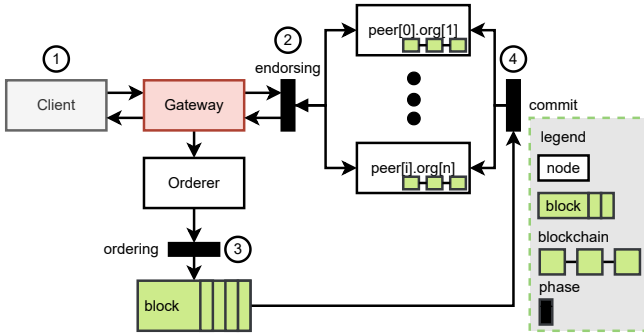

Fig. 1: Hyperledger Fabric - Transaction Flow

1) From version 2.5 of HLF, the flow begins with the client application sending a transaction proposal to a *gateway*, which acts as an intermediary between the client and the peers; 2) Endorsing peers will simulate the transactions and determine if they meet the pre-established requirements in the application's smart contract, which is a self-executing contract with business rules directly written into code; 3)

The transaction is then forwarded to the ordering peers (*orderer*), who will insert it into a block together with other transactions; 4) The generated block is subsequently persisted on the blockchain through the *commit* process.

A transaction in HLF consists of read and write operations that change the values of data objects in the blockchain. Transactions can range from creating a new data object to transferring the account owner of objects. In turn, the endorsing phase checks whether transaction operations lead to a consistent state of the blockchain by just simulating a given transaction. For example, it is common for the simulation to verify whether the sender indeed owns the goods and that the recipient exists in a transaction transferring goods between accounts. It is also common for endorsing peers to belong to different geographically distributed organizations (*org*) that do not have mutual trust in each other.

The endorsed transaction returns to the gateway, and it will be forwarded to the orderers responsible for managing new blocks. Blocks have a predefined size and a maximum wait time to be formed (*batch timeout*). Once transactions are ordered into a block, this block will be sent to the committers nodes, which will process and insert it into their replicates of the blockchain to ensure integrity, consistency, and immutability of data objects. Note that endorsing and commit phases can execute in the same network node, as shown in Figure 1, or be separated into distinct nodes depending on the blockchain governance model.

IV. PROPOSED MODEL

Figure 2 presents the proposed model based on the high-level transaction flow of the HLF platform defined in Figure 1. All transitions in the model follow single-server semantics and exponential distribution with distinct times.

Transaction flow in the model begins with triggering the **Arrival** transition, depicted in gray at the top left of the model. The first highlighted area composed by a rectangle formed by dashed green lines emphasizes the activities occurring within the gateway.

Transactions arriving through Arrival are directed to the **G1** location, symbolizing the *gateway* in the HLF environment, with G1 connected to the **GQ** (*gateway queue*) location through the **TPT** (*transaction posted*) transition. Transactions exceeding the *gateway* capacity are discarded. Transactions entering the gateway proceed to the intermediate state **IS1** via the timed **TPT** transition and are directed towards the endorsing peers through the **BF1** transition.

The endorsing peers and their respective capacities are represented by the locations **E1**, **E2**, **E1Q**, and **E2Q**. A garbage collector mechanism, **TGC**, is present in all endorsing peers, reducing the probability of transaction *starvation* within the system and the waste of computational resources generated by software aging.

After endorsement, the transaction reaches the **ST** (*Stamped Transaction*) state through the average endorsement time represented by the **E1T** and **E2T** transitions. A transaction in the modeled environment receives endorsement from two peers, meaning the **ST** location will have up to
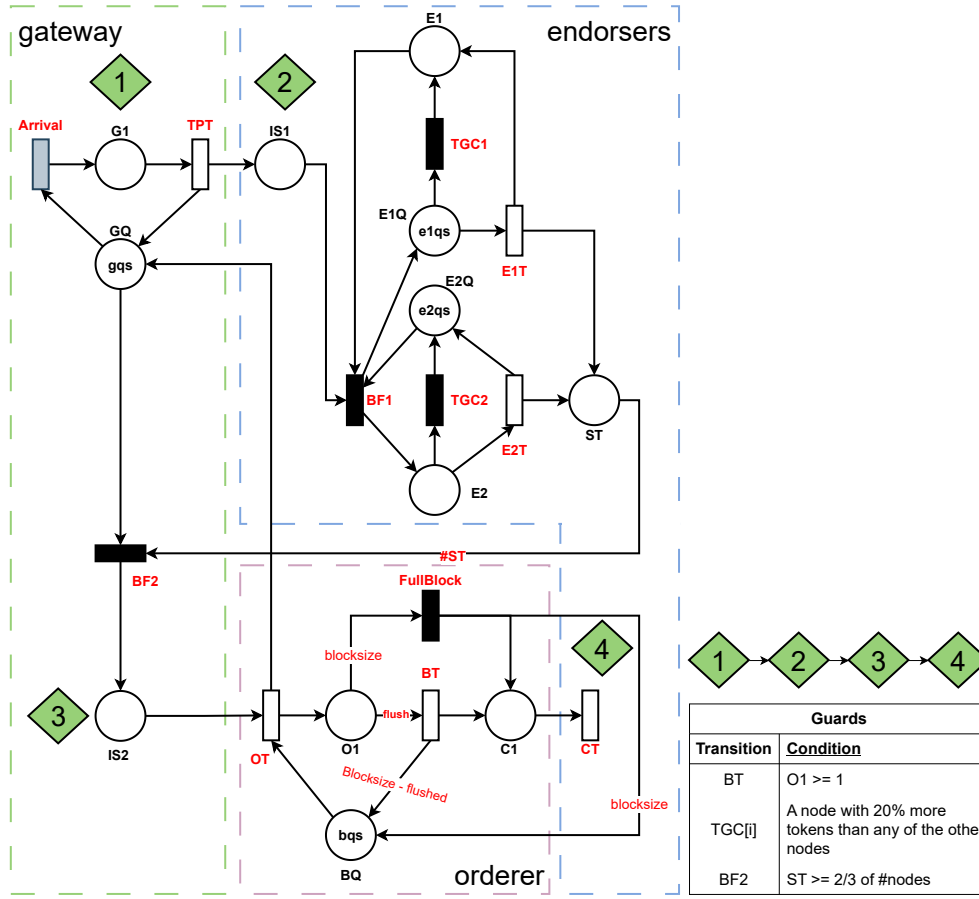
Fig. 2: Proposed Model - Hyperledger Fabric

two markings representing a single transaction. The attested transaction waits for the gateway's availability to receive it (**BF2**) and then moves to the intermediate state **IS2** via the **BF2** branching transition, indicating that the transaction is between the gateway and the orderer.

The **OT** (*ordering time*) transition represents the time to order transactions into a new block through the ordering peer (**O1**). The **BQ** (*block queue*) location indicates the block size or the maximum number of transactions required to complete it [12].

In HLF, a block can be partially or filled. Upon reaching the full capacity of a block, transactions in **OQ** move to the **C1** location through the **FullBlock** transition. The alternative flow used for a partially filled block goes through the **BT** (*Batch Timeout*) location, which assists in preventing *starvation* by establishing the maximum time a transaction will be in the ordering phase.

The **flush** variable on the arc connecting the **O1** location to the **BT** transition ensures the transfer of transactions to **BQ** without exceeding the block size (*blocksize*). Blocks enter the persistence phase through the **C1** location via the **CT** transition. New blocks are *committed* to the other peers. For better accuracy, guard expressions in **BF2**, **BT**, and **TGCs**, which are presented in Figure 2, are responsible for restricting the firing of these transitions, ensuring the model's efficiency and representativeness when compared to the real environment.

V. EXPERIMENTAL METHODOLOGY AND VALIDATION

The experiments to validate the base model were conducted using the basic network of Hyperledger Fabric. The basic network, maintained by the HLF developers, enables user creation, verification, and transfer of assets. The focus of the experiments was exclusively on asset creation transactions due to their higher resource demand than querying operations and transferring goods between accounts.

The setup of the experimental environment included a full node with two endorsing peers and one orderer deployed in Docker containers. The test machine had four physical cores, 8 GB of RAM, and 80 GB of storage. The software consisted of Ubuntu 22.04, Hyperledger Fabric 2.5, and Docker 24. The orderer, also a Docker container, organized transactions into blocks and ensured their distribution and persistence in the network. A command-line interface (CLI) in a separate container facilitated the interaction between the client application and the blockchain network through the container containing the gateway.

In the validation process of the proposed model, the client application, developed in TypeScript, sent an average of 10 transactions per second (tps) to the platform's gateway. The client sent the transactions to the system following

an exponential distribution, the same adopted for the timed transitions of the proposed model.

The goal of the experiments was to validate the model, not to optimize resource usage through high demand that would result in an overload on the peers. However, we will see in our case studies that many components were on the verge of resource exhaustion, even with a low arrival rate. Thus, the transaction arrival rate was adjusted to ensure the execution of all phases in the model, including the times required for forming partial and complete blocks.

Measurements were carried out in the environment, and the obtained data were used to evaluate the proposed model. The experimental evaluation involved submitting 400 transactions to the real system. The average times obtained for each stage were then used to validate the proposed model through 400 simulations. The Mercury tool [13] was used for the simulations, which focused on latency as the key metric. The measured results indicated an average latency of 1131 ms with a standard deviation of 408 ms. An average latency of 1110 ms with a standard deviation of 18 ms for the simulated results.

A two-sample t-test was conducted, resulting in a p-value of 0.3. With a significance level α of 0.05, the test indicated no statistically significant difference between the results obtained through model evaluation and experiments. Therefore, with 95% confidence, there is no evidence to reject the hypothesis that the proposed base model accurately represents the system.

VI. CASE STUDIES

The reference data for the model are the average times for each factor obtained experimentally through measurement. Table I summarizes these values, which will serve as input parameters in the model.

TABLE I: Input Values for the Model and Their Variations

Factor	Baseline	Range {min,max}
Arrival Rate (λ)	10/s	{5, 18}
Block Timeout (BT)	2000 ms	{1000, 3000}
Endorsement Time (E1T and E2T)	160 ms	{80, 240}
Ordering Time (OT)	15ms	{7.5, 22.5}
Commit Time (CT)	1150 ms	{575, 1725}
Transaction Posting Time (TPT)	10 ms	{5, 15}
Capacity Sizes (E1Q, E2Q, GQ)	10	{5, 15}
Block Size (BQ)	10	{5, 50}

A. Latency, Throughput, and Utilization

The first analyzed metric is latency, derived from Little's Law [14], which describes the relationship between the average number of pending transactions in a system and its respective arrival rate. We calculate the latency for the proposed model according to Equation (1), where $P(m(\text{Place}) = i)$ is the probability of i markings in the place.

$$E(\text{Place}) = \sum_{i=1}^{n} P(m(\text{Place}) = i) \times i, \qquad (1)$$

Figure 3a shows how the arrival rate influences latency. It is observed that an increase in the arrival rate results in

an increase in latency, from 1110 ms for a rate of 10 tps to 20000 ms for a rate of 100 tps.

Figure 3b demonstrates the effect of queue parameters on latency. Block size (BQ) stands out, as larger blocks may delay filling and processing new blocks.

Figure 3c highlights the influence of timing parameters on latency. The endorsement time (ET) has the most significant impact. Other times, even with variations above or below, have a lesser impact on this metric.

The throughput metric measures the number of transactions carried out over a period of time [2]. This paper focuses on the number of transactions persisted on the blockchain per second. Throughput can be calculated using Equation 2, with **E(Place)** representing transactions in **O1** and **C1**, and *t(Transition)* indicating the time associated with the **CT** transition (commit time).

$$\textbf{Throughput} = \frac{\text{E(Place)}}{\text{t(Transition)}} \qquad (2)$$

The arrival rate constantly impacts the system's throughput. Meanwhile, Figure 4a shows that the negative impact of block size (BQ) on throughput is greater than that of other queues in the system, and this is intensified by the maximum time until a block is completed (BT), as can be seen in Figure 4b, the greater the value of BT, the lower the system's throughput.

Smaller blocks result in higher throughput, a finding previously confirmed in [12] and now supported by a refined model and a new version of Fabric that includes a gateway and a set of intermediate states that increase precision and improve the accuracy of the obtained results. Transactions processed quickly flow more efficiently between endorsement, ordering, and commit.

The utilization metric quantifies the expected number of markings in a specific place **E(Place)** [15], divided by the total capacity of another location directly related to the first, for example, the gateway (G1) and its corresponding queue (GQ). The combination of locations in modeling effectively characterizes a resource queue, as shown in Equation 3.

$$\textbf{Utilization} = \frac{\text{E(Place)}}{\text{Resource Capacity}} \qquad (3)$$

We focus on utilizing the gateway that receives requests from clients and endorsers. As the gateway is a single point of failure, since there is only one gateway in the assessed environment, it is natural to be overloaded. Figure 5a shows that the system becomes overloaded with arrival rates below 5 transactions per second, and upon reaching a rate of 20 tps, the gateway utilization reaches 95%.

Figure 5c presents the factors related to time and their impact on the utilization. The BT has a significant effect; the shorter the time for block formation, the lower the system utilization, which indicates that ordering impacts the gateway more than the arrival of new transactions from the client.

Furthermore, Figure 5b confirms the impact of ordering on the gateway by showing that very large blocks (BQ) reduce the system utilization. Therefore, more efficient orderers are
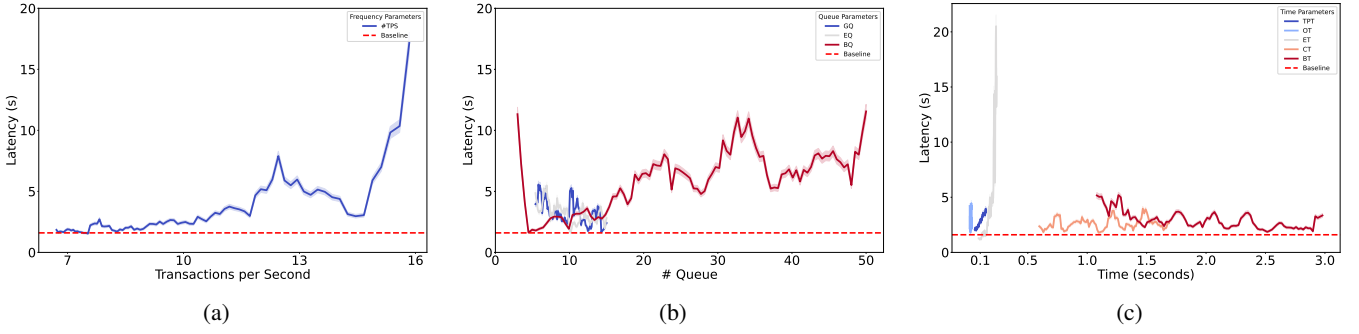
(a) (b) (c)
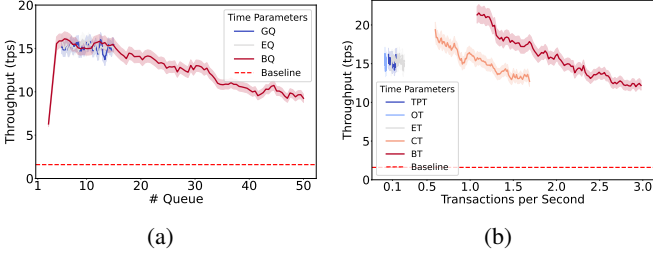
Fig. 3: Impact of Parameters on Latency

(a) (b)

Fig. 4: Impact of parameters on throughput

a better investment than a larger number of endorsers from the perspective of the evaluated scenario.

B. Endorsement Policy

The endorsement of a transaction is based on a set of policies described in the source code of that blockchain or in its smart contract. The policies specify how many and which nodes or peers need to agree or attest to the integrity of that transaction through their signature. The three main endorsement policies are **AND**, **OR**, and **K-out-of-N** (KooN).

Assuming we have three endorsing peers. If an AND type endorsement policy is adopted, then all three peers need to endorse the transaction; if it's OR, at least one of the three must endorse it; for KooN, we determine how many peers need to endorse it, for example, *2-out-of-3*, indicates that two out of the three need to do so. Once the transaction is endorsed, it returns to the gateway to be redirected to the orderer.

Figure 6a shows that adopting a K-out-of-N type endorsement policy provides greater stability to the system in terms of latency and offers extra margin in the availability of the environment since some endorsing nodes may fail, but the service will continue to operate.

Figure 6b shows that adopting endorsement policies has little to no impact on the overall system throughput for the evaluated scenarios. However, it is important to note that the main effect of adopting endorsement policies is on the security of the environment and the relationship between the organizations that comprise it.

C. Distributed Endorsement and Multiple Gateways

We expanded the scope of the proposed model by introducing a new gateway aimed at reducing overload and mitigating the negative impacts on our single point of failure at the system's entrance. Additionally, we considered three geographically distributed endorsers in a **2-out-of-3** policy by adding latency time to the endorsement process.

This case study aims to assess how the metrics of interest behave with changes in the components that most impacted each of them. For instance, latency, endorsement time, and arrival rate had the greatest impact, while block size was critical for throughput and utilization. We used Table I as the basis for this case study.

The endorsement time for each endorsing peer will correspond to the measured time + network latency based on the distance between the gateway and endorsers. The GCPing tool was used and started in Garanhuns, Brazil, and as a destination, the major data center of the Google Cloud Platform (GCP).

The target cities, i.e., the destinations with viable servers chosen based on the shortest distance from the starting point, are São Paulo - Brazil, Santiago - Chile, and South Carolina - USA. The respective latency for each target server was 81 ms, 136 ms, and 197 ms.

Figure 7a presents the impact of the arrival rate on the average response time in an environment with two gateways and three endorsers under the **2-out-of-3** policy. The previous behavior remains; the higher the arrival rate, the longer the response time. However, the results indicate greater stability. For the first case study, we reached a latency of 20 seconds for an arrival rate of 18 transactions per second.

Figure 7b presents the impact of block size on throughput in the same environment. We noticed the same behavior in the first case study, the larger the block size, the lower the system throughput, as large blocks will use the time to be partially filled and not their completeness.

VII. CONCLUSION

This work presented a detailed analysis of the performance of Hyperledger Fabric, employing measurement and modeling to identify systemic bottlenecks. The emphasis was on latency, throughput (TPS), and utilization metrics. The experiments focused on the endorsement, ordering, and
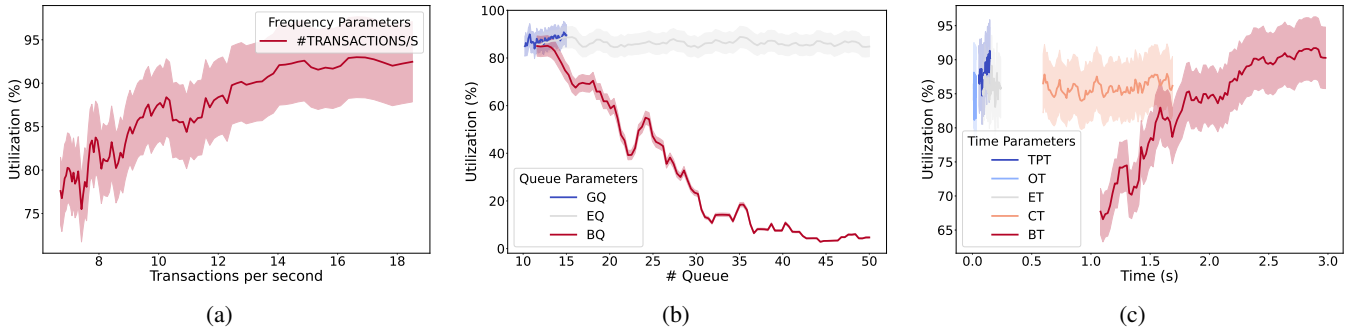
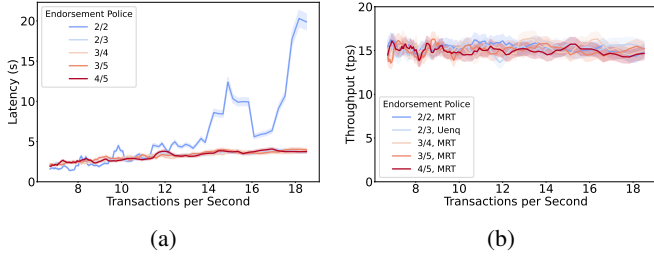Fig. 5: Impact of parameters on gateway utilization

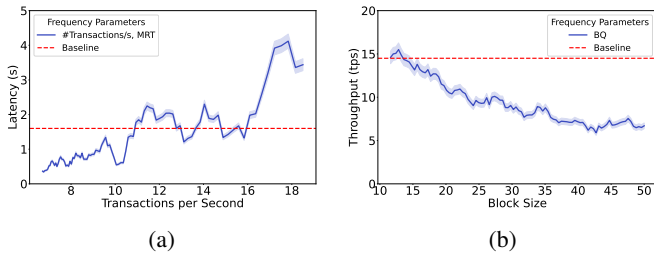Fig. 6: Latency and Throughput vs. Endorsement Policy

Fig. 7: Latency and Throughput vs. Multiple Gateways

confirmation (commit) phases of transactions in the Fabric network. We also evaluated block size and endorsement queue utilization, concluding that enhancing endorsement capacity can increase throughput and reduce latency. Finally, we examined the impact of an endorsement policy on the assessed metrics. An infrastructure with more endorsing nodes and nodes distributed across different locations under a policy where most nodes need to endorse the transaction directly affects the system's latency and throughput. Future work will explore other distributions, considering the network's expansion across multiple nodes and latency.

VIII. ACKNOWLEDGEMENT

REFERENCES

[1] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*. ACM, 2018, p. 30.

[2] P. Thakkar, S. Nathan, and B. Viswanathan, "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)*. IEEE, 2018, pp. 264–276.

[3] M. Antwi, A. Adnane, F. Ahmad, R. Hussain, M. H. ur Rehman, and C. A. Kerrache, "The case of hyperledger fabric as a blockchain solution for healthcare applications," *Blockchain: Research and Applications*, vol. 2, no. 1, p. 100012, 2021.

[4] X. Xu, G. Sun, L. Luo, H. Cao, H. Yu, and A. V. Vasilakos, "Latency performance modeling and analysis for hyperledger fabric blockchain network," *Information Processing & Management*, vol. 58, no. 1, p. 102436, 2021.

[5] T. Guggenberger, J. Sedlmeir, G. Fridgen, and A. Luckow, "An in-depth investigation of the performance characteristics of hyperledger fabric," *Computers & Industrial Engineering*, vol. 173, p. 108716, 2022.

[6] H. Sukhwani, N. Wang, K. S. Trivedi, and A. Rindos, "Performance modeling of hyperledger fabric (permissioned blockchain network)," in *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*. IEEE, 2018, pp. 1–8.

[7] C. Melo, J. Araujo, J. Dantas, P. Pereira, and P. Maciel, "A model-based approach for planning blockchain service provisioning," *Computing*, vol. 104, no. 2, pp. 315–337, 2022.

[8] O. Wu, S. Li, L. Liu, H. Zhang, X. Zhou, and Q. Lu, "Performance modeling of hyperledger fabric 2.0," in *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering 2022*, 2022, pp. 357–365.

[9] L. Jiang, X. Chang, Y. Liu, J. Mišić, and V. B. Mišić, "Performance analysis of hyperledger fabric platform: A hierarchical model approach," *Peer-to-Peer Networking and Applications*, vol. 13, pp. 1014–1025, 2020.

[10] Z. Ke and N. Park, "Performance modeling and analysis of hyperledger fabric," *Cluster Computing*, pp. 1–19, 2022.

[11] P. Yuan, K. Zheng, X. Xiong, K. Zhang, and L. Lei, "Performance modeling and analysis of a hyperledger-based system using gspn," *Computer Communications*, vol. 153, pp. 117–124, 2020.

[12] F. A. Silva, G. D. Gonçalves, L. Feitosa, A. Soares *et al.*, "Avaliaçao de desempenho de blockchains permissionadas hyperledger orientada ao planejamento de capacidade de recursos computacionais," in *Anais do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. SBC, 2023, pp. 71–84.

[13] B. Silva, R. Matos, G. Callou, J. Figueiredo, D. Oliveira, J. Ferreira, J. Dantas, A. Lobo, V. Alves, and P. Maciel, "Mercury: An integrated environment for performance and dependability evaluation of general systems," in *Proceedings of industrial track at 45th dependable systems and networks conference, DSN*, 2015, pp. 1–4.

[14] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. New York: Wiley Computer Publishing, John Wiley & Sons, Inc., May 1991.

[15] P. R. Maciel, K. S. Trivedi, R. Matias, and D. S. Kim, "Dependability modeling," in *Performance and dependability in service computing: concepts, techniques and research directions*. IGI Global, 2012, pp. 53–97.