Workload prediction in BTC blockchain and application to the confirmation time estimation

Ivan Malakhov$^{[0000-0002-1176-3543]}$, Carlo Gaetan$^{[0000-0002-1361-9959]}$, Andrea Marin$^{[0000-0002-5958-1204]}$, and Sabina Rossi$^{[0000-0002-1189-4439]}$

Università Ca' Foscari Venezia, Via Torino 155, 30173 Venice, Italy
{ivan.malakhov,gaetan,marin,sabina.rossi}@unive.it

Abstract. Blockchains are distributed ledgers storing data and procedures in an immutable way. The validation of the information stored therein as well as the guarantee of its immutability can be achieved without the need of a central authority. Proof-of-work is the maximum expression of the distributed nature of such systems, and requires miners to spend a large amount of energy to secure the blockchain. The cost is mostly paid by the end-users that offer fees to support the validation of their transactions. In general, higher fees correspond to shorter validation delays. However, given the limited throughput of the system and variability of the workload, the fee one needs to offer to satisfy a certain requirement on the validation delay strongly depends on the intensity of the workload that, in turns, is subject to high variability.
In this work, we propose a time series analysis of the workload of Bitcoin blockchain and compare the accuracy of Facebook Prophet model with a ARIMA model. We take into account the periodicity of the workload and show by simulations how these predictions, accompanied with their confidence intervals, can be used to estimate the confirmation delays of the transactions given the offered fees.

Keywords: Blockchain · Confirmation time analysis · Time series analysis.

1 Introduction

Blockchains have been attracting more and more attention from the research community from the economical, security and application points of view. More recently, the quantitative analysis of blockchains has also emerged as an important research challenge.

The blockchain distributed ledger has substantially three main roles: (i) verify the information or procedures that end-users wants to store according to some rules, (ii) guarantee the immutability of the stored information and (iii) make the information or procedure publicly available. While there are several ways to achieve these goals, in this paper, we focus on the most popular one, namely the *proof-of-work* (PoW).

PoW has been introduced by the seminal paper [12] by the pseudonym Satoshi Nakamoto with the aim of creating a distributed ledger for economical transactions based on the cryptocurrency Bitcoin (BTC). Many public blockchains use PoW as consensus algorithm and, recently, it has been proposed also for permissioned blockchains [10]. Therefore, henceforth, we will focus on BTC blockchain since it is, together with Ethereum, the mostly used and known blockchains. However, the methodologies and discussions that we propose can be easily extended to other blockchain systems with similar characteristics.

In BTC, the blockchain stores transactions into blocks. Blocks have a maximum size of 1MB and are generated, on average, every 10 minutes. This means that the maximum throughput of the system is fixed by design. Transactions are proposed by the end users and are sent to the *miners* for being processed.

Miners maintain a queue called *Mempool* that contains all the pending transactions, i.e., the transactions sent by end users but that have not been added to a block yet. When a transaction is included in a block, we say that it is *confirmed*, i.e., it is permanently stored in the system. The transaction residence time in the Mempool is called *confirmation delay*. This delay is crucial in determining the Quality-of-Service (QoS) of applications based on blockchains.

In order to understand the quantitative dynamics of the transaction conformations, we need to review the procedure implemented by the miners to secure the blockchain system. Each miner selects from the Mempool a set of transactions to fill a block, then he/she checks their integrity (e.g., when there is a transfer of cryptocurrency it verifies that there is not double spending) and finally it works on a computational problem that requires a large amount of energy in order to be solved. This latter step is the PoW. The miner that firstly announces the solution of its computational problem is entitled to add his/her new block to the blockchain after the other peers have verified the correctness of the solution.

In order to cover the energy and hardware costs, miners receive a certain amount of cryptocurrency when they succeed in a block consolidation: some is freshly created by the system and given to the miners and then they receive the fees promised by the owners of the transactions added to new block. These fees are offered by the end users on a voluntary base, i.e., they can even offer 0 BTC. However, the miners tend to select from the Mempool those transactions that offer the highest fee.

From the end-user's point of view, an interesting trade off arises: on the one hand, he/she wishes to offer the lowest possible fee to reduce the running costs of his/her activities, on the other he/she may have some requirements on the QoS, e.g., the need to confirm the transaction within a certain amount of time. For example, the transaction may be associated with a trading speculation and hence must be confirmed in a few minutes, or may be a bid for a certain auction with a deadline.

Blockchain systems can be studied as distributed systems by means of formal methods in the style of [5, 6]. Queueing theory allows us to study the relation between the holding time in the Mempool and the arrival intensity of the transactions. Clearly, when the holding times increase, the transaction fees also increase. However, while for high fees consolidated in few blocks it is safe to assume that the arrival intensity is time homogeneous, for transactions offering low fees this is unrealistic.

In this paper, we study the problem of predicting the traffic intensity in BTC blockchain with the aim of parameterising a simulation model that studies the expected confirmation time of transactions. After collecting data about the transaction arrival process at our BTC node, we use these traces to train two possible prediction models: one based on Facebook Prophet model [13], and the other is the well-known Autoregressive Integrate Moving Average (ARIMA) model. Both models provide confidence intervals in the prediction of the arrival process and allow us to consider pessimistic-, average- and optimistic-case scenarios. After comparing the two predictive models, we study by simulation the transaction confirmation time as a function of the offered fees and compare the results obtained with the real trace as input with those obtained by using the predicted trace as input.

The paper is structured as follows. In Section 2, we discuss the related work done in similar fields. Section 3 describes the motivation of this paper and gives a brief description of the applied prediction models. In Section 4, we examine the ARIMA and Prophet forecasts accuracy after certain hours from the transaction arrival and the accuracy of the predictions on the expected confirmation time using Monte Carlo simulations. Finally, Section 5 concludes the paper and provides an insight for future work.

2 Related work

Statistical analysis on blockchain and in particular BTC system have been widely investigated in the recent years. However, most of the research efforts have been devoted to the prediction of the conversion rate to USD or other currencies (see, e.g., [11, 4]).

In our case, we are interested in studying the cost of transaction fees. Most of the previous works assume a time-homogeneous arrival process, as in [7, 8, 1] which can be reasonable for expensive transactions that are confirmed within one hour from their request. However, when the delay is longer, the fluctuations of the arrival process cause the model with the homogeneity assumption to generate inaccurate predictions.

In addition, [9] provides a similar contribution by demonstrating the stationary analysis of the queueing model and the definition of the customer priority classes. However, the authors focus on a game theoretical framework where they attempt to find correlations between the fee fluctuations and the miners' economical incentive.

Another work [14] analyses the transaction fees in the blockchain networks. However, their research is related to the Ethereum blockchain and particularly the smart contract transactions.

To the best of our knowledge, this is the first study that aims at predicting the intensity of the transaction arrival process by using time series analysis and predicting on the confirmation time based on the offered fee.

3 Background and motivation

This section describes the goal of the paper and provides background information about the models used for the prediction of the arrival rate of transactions.

3.1 The problem of predicting the minimum fee for QoS

PoW is a method for both reaching consensus among the miners and guaranteeing the immutability of the blockchain contents. More precisely, it quantifies the expected energy cost required to modify a confirmed transaction. The more computational power (usually called hashpower) the miners invest the more secure the distributed ledger is. For this reason it becomes crucial to incentivize more miners to join the network with some rewards.

In the BTC blockchain, miners are rewarded in two ways: i) for each confirmed block, the miner who created it receives a certain amount of cryptocurrency and ii) for each transaction included in the block, the same miner receives the fee offered by the user who created that transaction.

In BTC the cryptocurrency is the Bitcoin but since its value is high (at the moment, $1\text{BTC} \simeq 35,000$ USD), fees are usually expressed in Satoshi (sat) where $1\text{BTC}{=}10^8\text{sat}$.

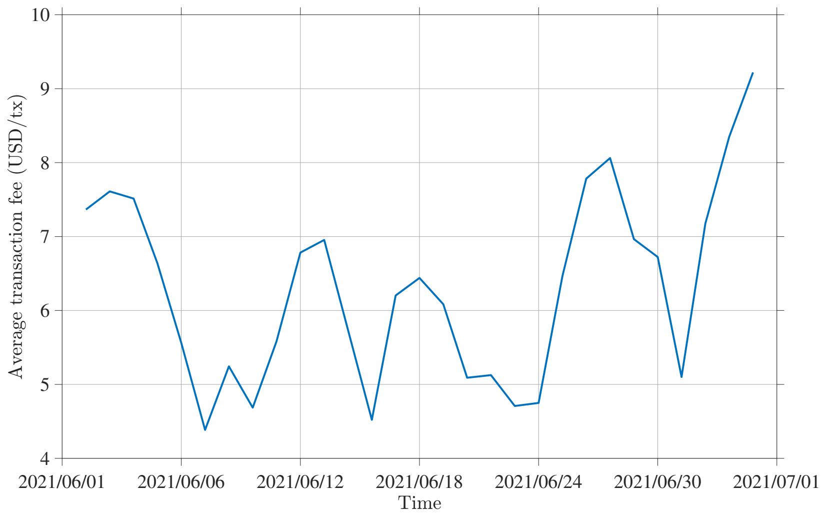
While the former reward is going to be dismissed in the next years, the latter plays a crucial role in understanding the QoS of applications that use BTC blockchain. Indeed, miners aim at maximising their profit and thus choose to include in the block the transactions with the highest fee.

Transaction fees are known to be subject to high fluctuations as shown by Figure 1a. We may notice that the average fee for a transaction can vary from around 4.5 to 9 USD in a month. How to decide which fee to offer to have an expected confirmation delay?

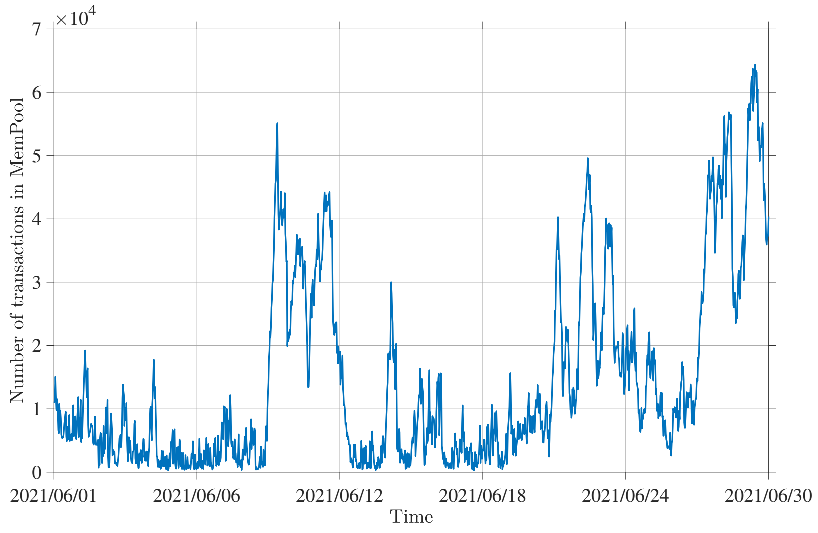
It is important to understand that the answer to this question depends on several state variables of the blockchain. First, we should consider the Mempool occupancy (usually called improperly Mempool size), i.e., the backlog of the transactions that are waiting to be confirmed. Figure 1b shows the trace of the Mempool occupancy in the month of June 2021. The are several bursts that clearly affect the decision on the fee to be offered.

However, the most important factor is the transaction arrival process. Recall that all the transactions arriving after a tagged transaction t offering a fee per byte f will overtake t if they offer more than f. Fee per byte is commonly used to compare the cost of transactions because these may have different sizes. Since block sizes are fixed (i.e., a block can contain approximately $2,300$ transactions at most) and the inter-generation time of blocks is on average 10 minutes, this implies that the competition among the transactions gets tougher when the traffic is higher. Figure 2b shows the distribution of the fee per byte offered under heavy-load conditions as measured by our monitor.

Summarising, the confirmation time of a transaction t arriving at time τ depends on the following aspects:

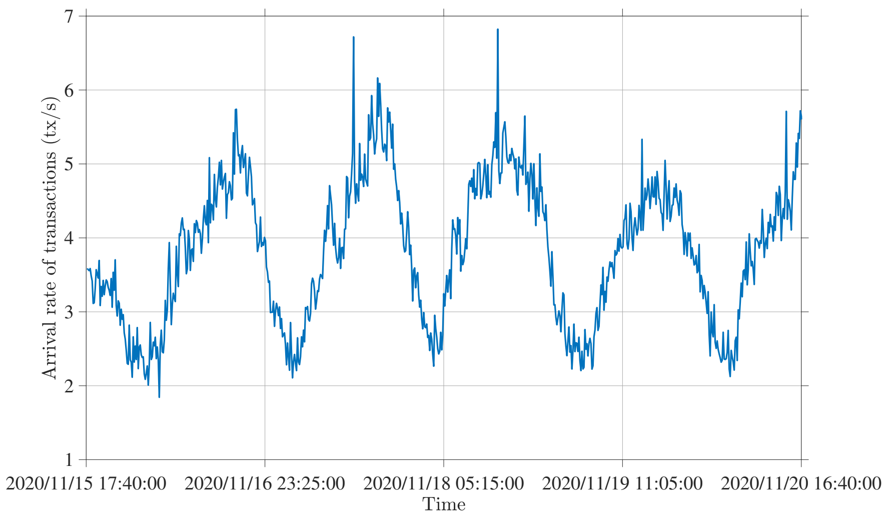


(a) Transaction fee in USD in the Bitcoin network. The data are retrieved from `http://www.blockchain.com`

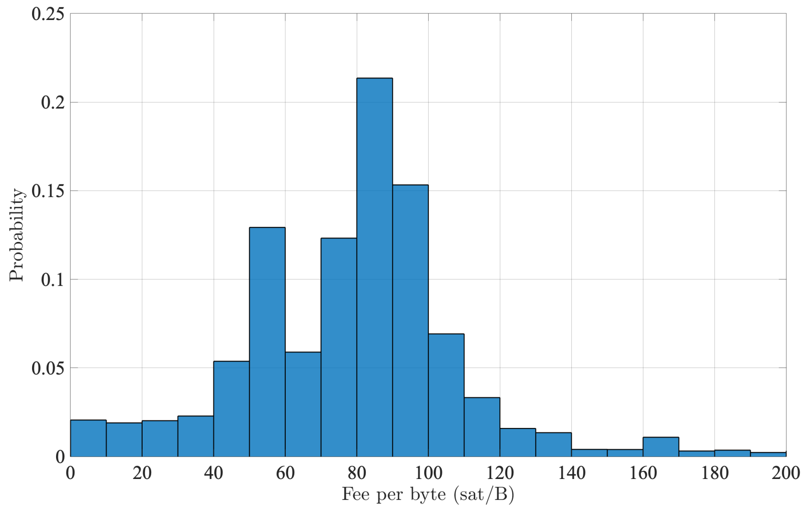


(b) Memory Pool size in transactions in the Bitcoin network. The data are retrieved from `http://www.blockchain.com`

Fig. 1: Blockchain network indicators.



(a) Arrival rate of transactions as a function of time with step of 10 minutes. The data are retrieved from the installed node.



(b) Empirical probability density function of fee per byte in heavy workload conditions. The data are retrieved from the installed node.

Fig. 2: Continue. Blockchain network indicators.

- The arrival rate of the transactions after τ and before the confirmation of t, limited to those whose fees are higher than the fee offered by t;
- The state of the Mempool at τ;
- The distribution of fees offered by the other users.

In general, we measure the confirmation delay in number of blocks rather than in seconds. However, it is well-known that the time between consecutive block consolidations is approximately exponential with mean 600s [3].

Figure 2a shows the intensity of the arrival process in a period of time. This is subject to high variability and exhibits a clear seasonality. Therefore, in any procedure aimed at predicting the expected confirmation time of transactions, we must implicitly or explicitly deal with the prediction of the arrival intensity at the moment in which the transactions is sent to the ledger.

The goal of this paper is that of evaluating the quality of the predictions of the arrival process given by two popular approaches to time series analysis: the ARIMA and Facebook Prophet models. Moreover, we will use these predictions to assess, by simulation, the quality of the estimations of the expected confirmation time of transactions.

3.2 Background on the ARIMA model

ARIMA(p,d,q) model [2] one of the most widely used models for statistical forecasting a time series of observations X_t. The ARIMA equation is a linear (i.e., regression-type) equation in which the predictors consist of lags of the dependent variable and/or lags of the forecast errors. The general model can be written as

$$(1-\phi_1 L-\cdots-\phi_p L^p)(1-L)^d X_t = c + (1+\theta_1 L+\cdots+\theta_q L^q)\varepsilon_t$$

where L is the lag-operator, i.e. $L^k a_t = a_{t-k}$ and ε_t is a white noise. The value p refers to the "AutoRegressive" component and represents the number of lagged observations included in the model. The "Integrated part" of the ARIMA model indicates that the data values have been replaced with the difference between their current and previous values, i.e. $(1-L)x_t = x_t - x_{t-1}$. The value d is a number of times that the raw observations are differenced. In general, differencing refers to the transformation applied to non-stationary time series in order to make them stationary by attempting to remove the deterministic components such as trends or periodicities. The value q, stands for the size of the "Moving Average" window for the forecast errors. Automatic identification of the orders p, d, q and statistical estimation of the parameters $\phi_1, \dots, \phi_p, \theta_1, \dots, \theta_q$ can be done easily (see [2]).

The data are collected every 10 minute and our time series exhibit seasonality with frequency of $144 = 24 \times 6$ which is exactly 24 hours in 10-minutes terms. In our experiment, using the Akaike Information Criterion, we identify a special instance of the ARIMA model, namely a multiplicative seasonal model [2]:

$$(1-\phi_1 L-\phi_2 L^p)(1-L)(1-L^{144})X_t = (1+\theta_1 L+\theta_2 L^2)\varepsilon_t.$$

3.3 Background on the Facebook Prophet model

The Prophet model [13] is a modular regression model with interpretable parameters that can be adjusted in order to optimize the prediction response.

The authors use a decomposable time series model with three key components, namely trend, seasonality, and holidays. The model may be represented as follows:

$$X_t = g_t + s_t + h_t + \varepsilon_t \,.$$

where g_t refers to the trend function that simulates non-periodic changes in the value of the time series, s_t describes periodic changes of the series, that is any seasonality effects, and h_t stands for the effects of holidays which occur on rather irregular pattern over one or more days. The error term ε_t is still white noise and represents any idiosyncratic changes of the model.

What is more, one of the features of g_t can be changepoint prior scale. The changepoints allow to incorporate trend changes in the growth models and stand for the points in time at which the trend is supposed to change its vector. It can be set manually otherwise it will be done automatically. This feature modulates the flexibility of the automatic changepoint selection. Larger values will allow many changepoints and small ones - few.

The authors frame the forecasting problem as a curve-fitting exercise, which differs from the models that account for the temporal dependence structure in the data. Although they miss some inferential benefits of using a generative model, e.g., the ARIMA model, their approach provides several practical advantages such as the fast fitting, ability to use irregular time data, flexible tuning of the trend, and seasonality behaviour.

4 Evaluation of the accuracy in performance predictions

This section consists of two parts. First, we study the accuracy of the ARIMA and Prophet predictions on the time series of the transaction arrivals in the BTC blockchain. This allows us to obtain a punctual value of the prediction after τ hours from the last considered arrival of transaction and its confidence interval. Thus, for each epoch, we have a predicted expected value, a lower bound that represents the optimistic scenario and an upper bound leading to the pessimistic scenario.

The second contribution of the section is the estimation of the accuracy of the predictions on the expected confirmation time by means of Monte Carlo simulations of the confirmation process. The simulation uses as input three values of the confidence interval (lower, upper and central) to obtain an optimistic, pessimistic and expected estimation of the confirmation time.

It is worthy of notice that, while the expected confirmation delay is monotonic increasing with respect to the arrival rate, the relation between waiting time and intensity of the arrival process is not linear and hence the intervals obtained in the confirmation delay predictions are not symmetric with respect to the prediction obtained using the expected arrival rate.

4.1 Comparison of time series prediction models

This section describes the accuracy of the estimates as well as their insights obtained by the aforementioned prediction models.

In order to collect the time series, we have installed a BTC mining node and logged the transactions announced at its Mempool. We have collected the data for five days and obtained our dataset that was coherent with the information available on specialised websites but with higher granularity (see Fig. 2a). Additionally, we analysed the distribution of transaction fees of the Bitcoin clients in heavy load conditions (see Fig. 2b).

In order to train the models, we divided our dataset in two parts with the same size: the first one has been used to train the models, while the second part has been used to assess the accuracy of the prediction.

For both the models, we use prediction intervals with a coverage of 95%.

Fig. 3 and 4 show predictions of the transaction arrival intensity provided by the Prophet and ARIMA models, respectively. What is more, Fig. 3a and 3b illustrate the prediction deviation due to the choice of different changepoint prior scale values, namely, 0.06 and 0.07 accordingly. Thus, the outcome of the Prophet model at the parameter 0.07 gives the best prediction, according to our experiment. In our assessment, we will use the best results.

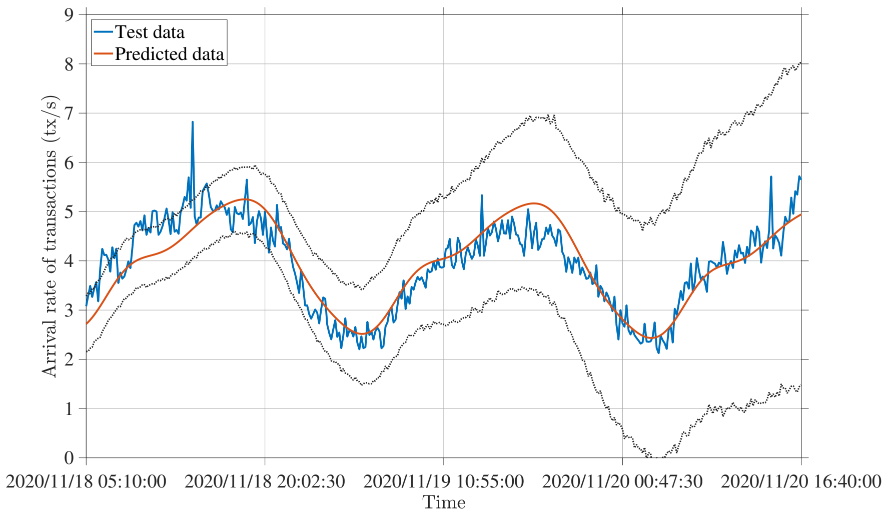
For both the plots, we used the first 2.5 days of data to train the model, and then we predicted the future arrivals. We show the test data of our dataset (blue line), the prediction of the model (red line) and the confidence intervals (grey lines). As expected, as the prediction time is moved far in the future, the confidence interval becomes wider. However, for practical applications, predictions are useful when performed within approximately 10 or 12 hours, otherwise it is very likely that the transaction is delay tolerant.

Even before formally testing the accuracy of the predictions with an error measure, we may notice that Prophet seems to give a better accuracy in this context.

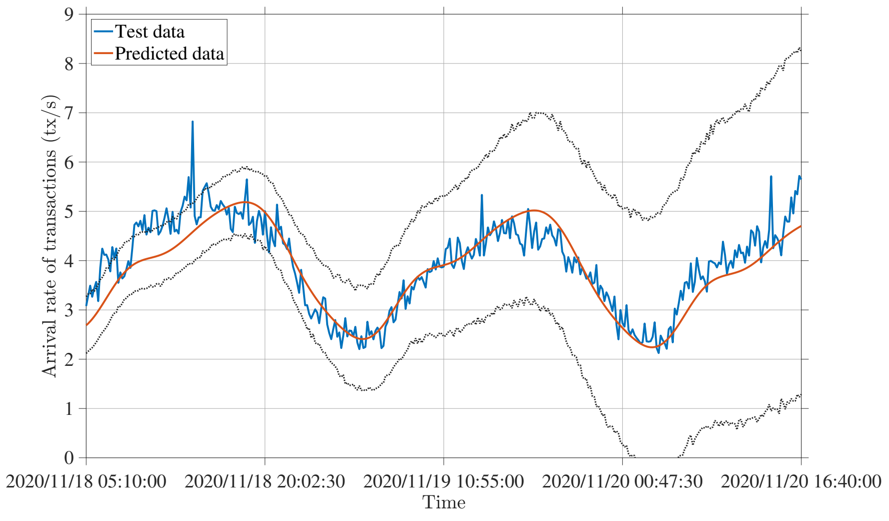
Now, we consider the predictions of the Prophet and ARIMA model at fixed time intervals. More precisely, given an interval τ, at each time t we use all the data up to t to train the model, and forecast the value of the time series at time $t + \tau$.

Fig. 6 shows the comparison of the predictions obtained with the Prophet and ARIMA models for different values of τ. We can see that, although the ARIMA predictions tend to be more noisy, both the models show rather good predictions of the test data.

More precisely, in Table 1 we compute the absolute errors of the Prophet and ARIMA predictions. According to our experiments, Prophet outperforms ARIMA, especially for short term predictions. Henceforth, we will carry out our experiments by using the Prophet model.



(a) The comparison at changepoint prior scale of 0.06.



(b) The comparison at changepoint prior scale of 0.07.

Fig. 3: Comparison of the actual arrival rate of transactions and the predicted response based on the Prophet model with different changepoint prior scale.

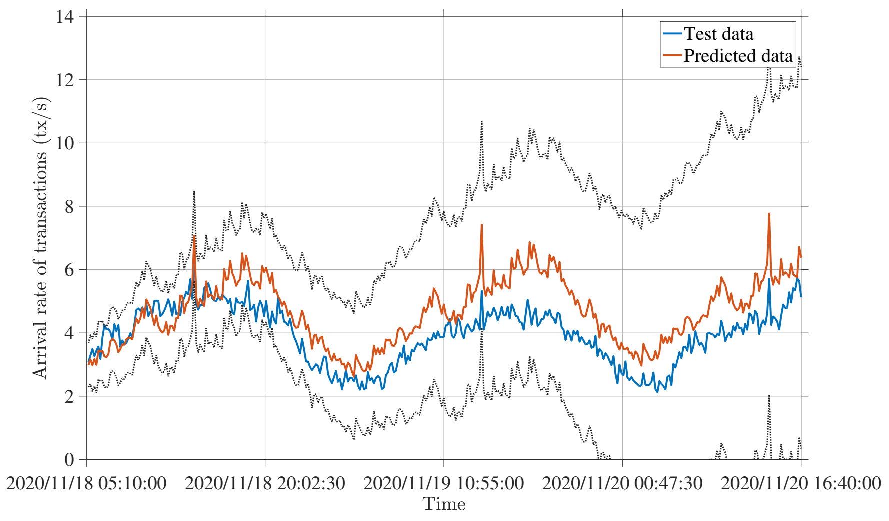
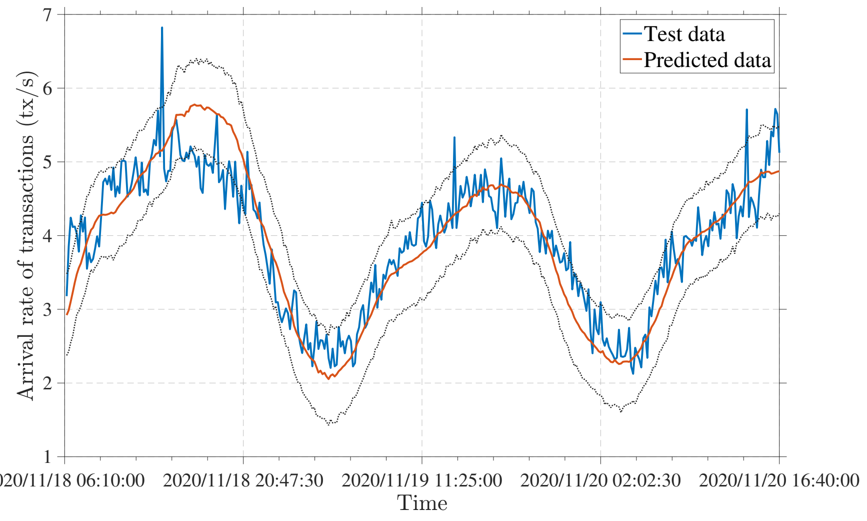


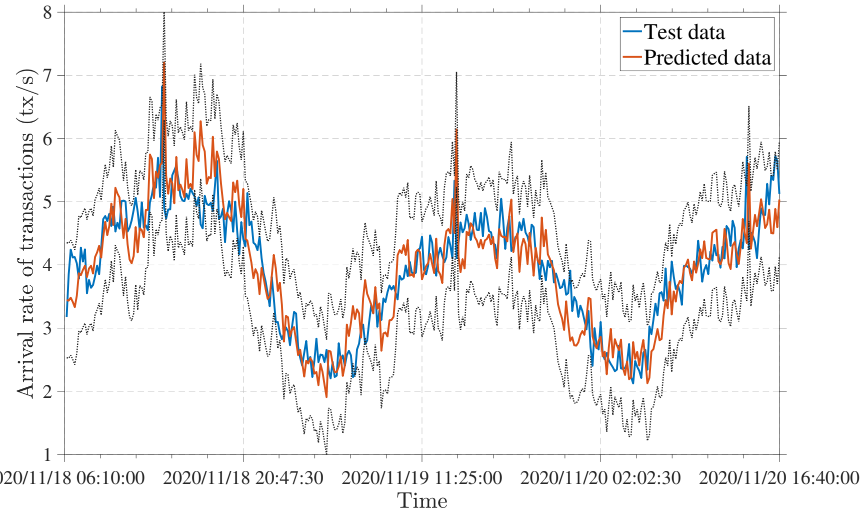
Fig. 4: Comparison of the actual arrival rate of transactions and the predicted response based on the ARIMA model.

Table 1: Mean Absolute Errors of Prophet and ARIMA models with different size of the prediction horizon.

Prediction horizon τ in hours	Prophet Error	ARIMA Error
1	0.3120	0.3668
2	0.3366	0.3896
4	0.3966	0.4219
12	0.6333	0.6416

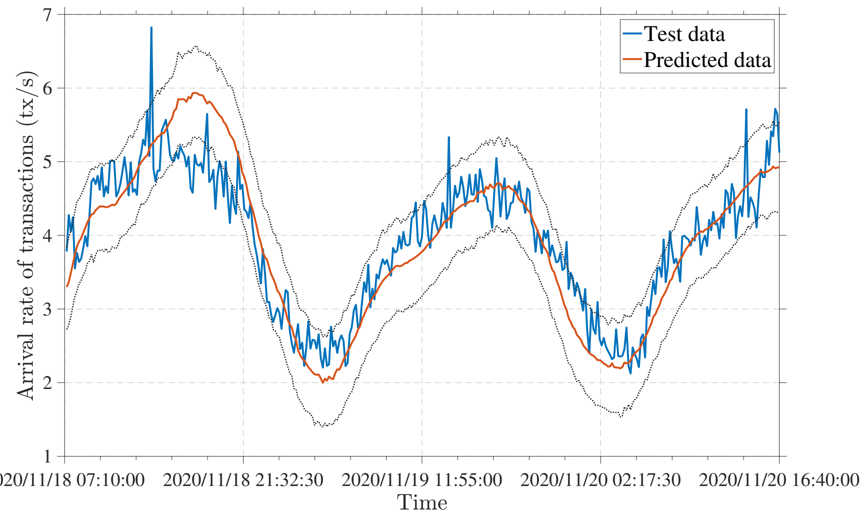


(a) Comparison of the actual arrival rate of transactions and the predicted response for $\tau = 1$ hour ahead based on the Prophet with changepoint prior scale of 0.07.

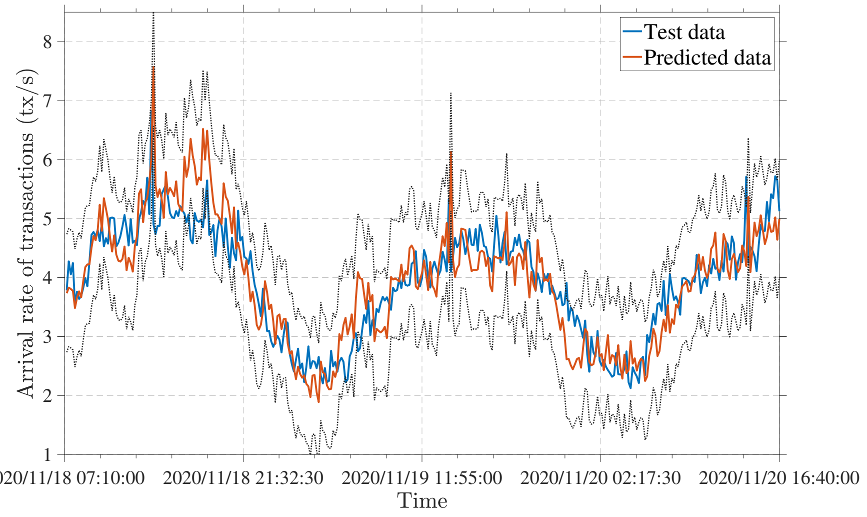


(b) Comparison of the actual arrival rate of transactions and the predicted response for $\tau = 1$ hour ahead based on the ARIMA model.

Fig. 5: Comparison of the Prophet and ARIMA prediction models at prediction horizon $\tau = 1$ hour and confidence interval of 0.95.

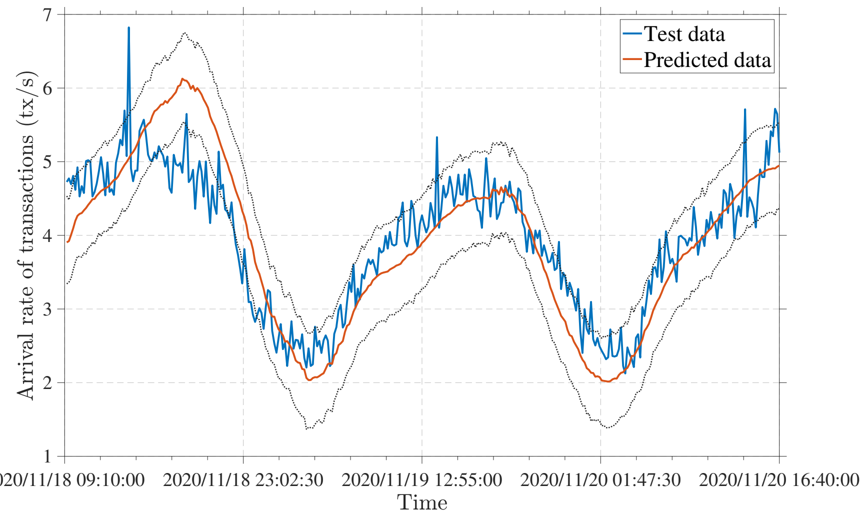


(a) Comparison of the actual arrival rate of transactions and the predicted response for $\tau = 2$ hours ahead based on the Prophet with changepoint prior scale of 0.07.

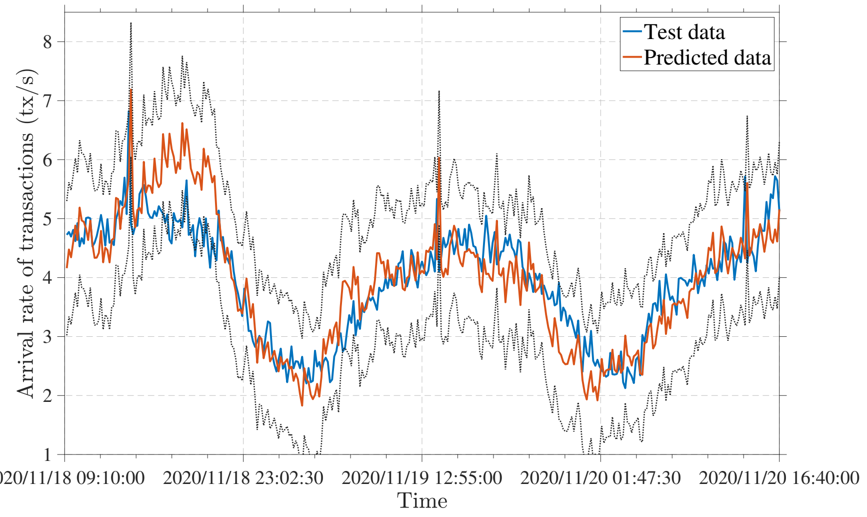


(b) Comparison of the actual arrival rate of transactions and the predicted response for $\tau = 2$ hours ahead based on the ARIMA model.

Fig. 6: Comparison of the Prophet and ARIMA prediction models with prediction horizon $\tau = 2$ hours and confidence interval of 0.95.

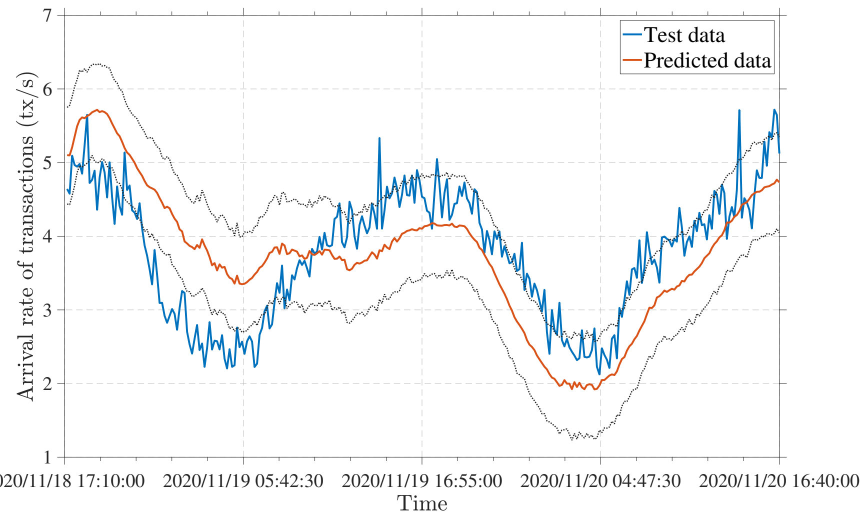


(a) Comparison of the actual arrival rate of transactions and the predicted response for $\tau = 4$ hours ahead based on the Prophet with changepoint prior scale of 0.07.

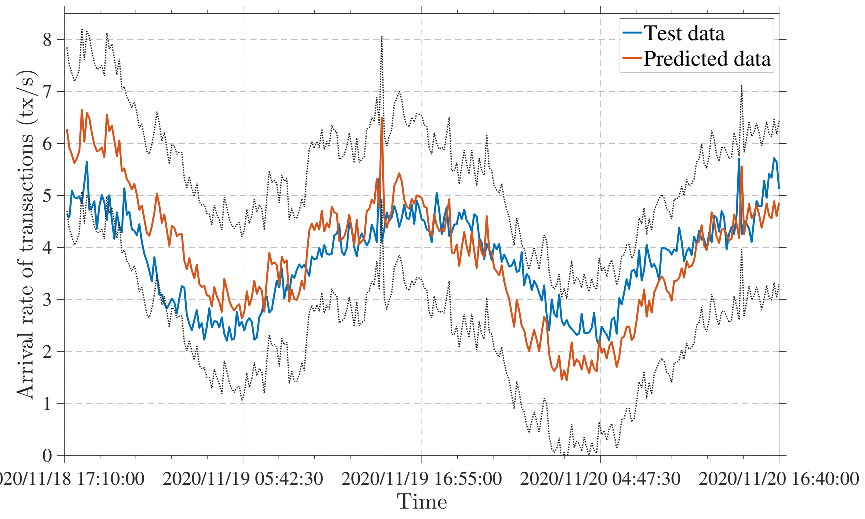


(b) Comparison of the actual arrival rate of transactions and the predicted response for $\tau = 4$ hours ahead based on the ARIMA model.

Fig. 7: Comparison of the Prophet and ARIMA prediction models with prediction horizon $\tau = 4$ hours and confidence interval of 0.95.



(a) Comparison of the actual arrival rate of transactions and predicted response for $\tau = 12$ hours ahead based on the Prophet prediction approach by Facebook with changepoint prior scale of 0.07.



(b) Comparison of the actual arrival rate of transactions and predicted response for $\tau = 12$ hours ahead based on the ARIMA model.

Fig. 8: Comparison of the Prophet and ARIMA prediction models with prediction horizon $\tau = 12$ hours and confidence interval of 0.95.

4.2 Simulations

In this section, we are interested in determining the accuracy of the estimation of the expected confirmation time using the Prophet prediction model to determine the arrival intensity of the transactions.

We resort to Monte Carlo simulations whose structure can be summarised as follows:

- We consider a fixed sequence of transaction arrivals. This can be trace-driven by our dataset or obtained by the models (optimistic-, average- or pessimistic-case scenarios).
- The generation of the blocks occurs at random time intervals, exponentially distributed with average 10 minutes. This follows from the memoryless characteristic of the mining process and from the invariant properties of the BTC blockchain.
- At a block generation instant, the most valuable transactions of the Mempool are confirmed and removed from the queue. We assume that the block contains $2,300$ transactions. Transaction fees are chosen probabilistically using the distribution of Figure 2b.
- Initially, the Mempool is populated with a fixed amount of transactions. These transactions offer a fee per byte according to the distribution of Figure 2b. Notice that, although this is an approximation since the cheapest transactions tend to remain in the Mempool, the comparison remains fair since the initial Mempool population is the same for all the scenarios.

More precisely, we number the transactions from $-M$ to ∞, where M is the initial Mempool size, transaction 0 is the tagged transaction whose confirmation time is measured, and transaction denoted by $i > 0$ are those arriving after the tagged one.

Transaction t_i is denoted by a pair (τ_i, f_i), where τ_i is the arrival time and f_i the offered fee. For $i \leq 0$, $t_i = 0$. f_i is sampled from the distribution of Figure 2b independently of τ_i. τ_i, for $i > 0$ are obtained from the real traces or from the predictions of Prophet. Notice that, in practice, the fees may be dependent from the system state (Mempool size, intensity of the arrival process) but in this context we use the simplifying assumption of independence since we mainly focus on the accuracy of the predictive power of the Prophet model.

Let $\mathcal{T}$ be the set of transactions.

Let $X_1, X_2, \ldots$ be the sequence of block consolidation times, and assume $X_0 = 0$. Then, $X_{i+1} - X_i$, $i \geq 0$, are i.i.d. exponential random variables with mean 10 minutes.

The state of the simulation model is described by a collection of transactions in the Mempool, denoted by $\mathcal{M}_i$, where the subscript i expresses that the state is associated with the instant immediately after the consolidation of block i.

The set of transactions arriving during the consolidation of the $(i+1)$-th block, but after the consolidation of the i-th, can be denoted by:

$$\mathcal{A}_i = \{t_i \in \mathcal{T} : \tau_i > X_i \wedge \tau_i \leq X_{i+1}\}$$

Now, let $\mathcal{F}(\mathcal{M})$ the set of at most $2,300$ transactions with the highest fee present in $\mathcal{M}$.

Thus we have the following recursive relation:

- $\mathcal{M}_0 = \{t_i \in \mathcal{T} : \tau_i \leq 0\}$
- $\mathcal{M}_{i+1} = \mathcal{M}_i \cup \mathcal{A}_i \setminus \mathcal{F}(\mathcal{M}_i \cup \mathcal{A}_i)$

Thus, the confirmation time T_c for the tagged transaction is given by:

$$T_c = \min\{i : t_0 \notin M_i\} \,.$$

The Monte Carlo simulation experiment consists of $10,000$ samples of T_c for a fixed fee f_0. Then, the expected confirmation time is obtained by averaging the sample values. The experiments have been repeated 30 times and the estimates have been used to determine the confidence interval for the expected confirmation delay. To avoid confusion, we omit the confidence interval from the plot. For a confidence of 95% we have a maximum relative error of 7%.

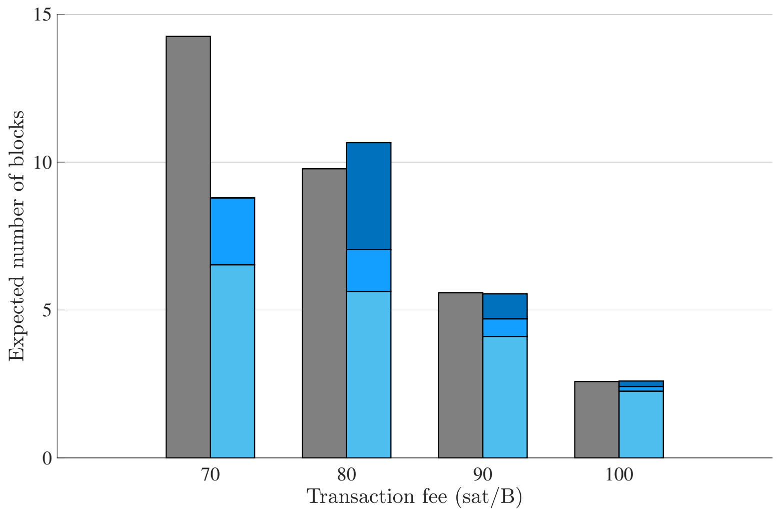
For each scenario that we consider, the tagged transaction offers a certain fee per byte that controls the confirmation time: the higher the fee, the quicker the process.

According to the trace of arrival that we use, we obtain 4 estimates: the first using the real data, the second using the average prediction of Prophet, the third and fourth using the trace given by the lower and upper bounds of the confidence intervals determined by the Prophet. These two latter scenarios can be interpreted as pessimistic and optimistic cases in terms of confirmation delay.

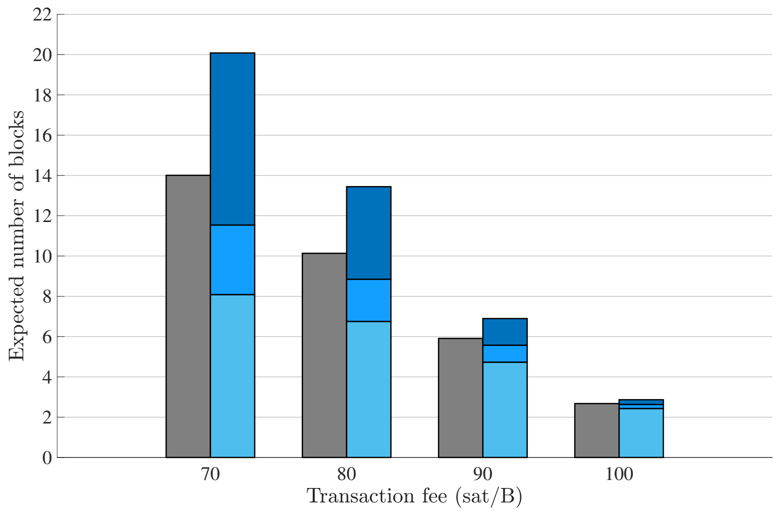
Fig. 9a and 9b show the expected number of blocks required for the transaction confirmation for different offered transaction fees. The grey bars refer to the expected number of blocks obtained from the real data while the stack of the second bar in light, normal and dark blue represent the optimistic-, average- and pessimistic- case scenarios, respectively, derived from the predicted data.

In the scenario of Figure 9a, the data were derived from the time series shown in Figure 3b, and the arrival time of the tagged transaction is 2020/11/18 05:10. Thus, the first half of the dataset was used to train the model. While in the second scenario (Figure 9b), the arrival time for the tagged transaction is 2020/11/19 08:30:00, and hence the training data include all the series up to that epoch.

The inspection of Figure 9a shows that the pessimistic case scenario for 70 sat/B is absent: this happens because the transaction is dropped before its confirmation (usually after 72 hours of residence in the Mempool). A second observation is that, especially in heavy-load (70 and 80 sat/B), the distance between the optimistic prediction and the average is smaller than that from the pessimistic and the average. This is due to the non-linearity of expected response time of a queueing system with respect to the arrival intensity. Finally, for this scenario, we notice that while the prediction obtained with the dataset



(a) The simulation results at 50% of the training data.



(b) The simulation results at 70% of the training data.

Fig. 9: Simulation results based on the actual data (grey bar) compared to the results of the Prophet predicted response (blue bars) with the optimistic, average and pessimistic cases and the initial Mempool occupancy of $10,000$ transactions and different amount of the training data.

is always within the optimistic and pessimistic cases, it seems to be closer to the latter. Indeed, Fig. 3b shows that predicted values for the first period of time are rather underestimated by the model. To confirm this explanation, we can look at the beginning of the next prediction interval (2020/11/19 08:30:00) when the prediction accuracy is higher. In this case, there is a good matching between the predicted average confirmation time and that obtained by using the real dataset (see Figure 9b).

5 Conclusion

In this paper, we have applied two different time series forecast models, namely the Prophet by Facebook and ARIMA, in order to predict the arrival rate of the transactions at the Mempool of the Bitcoin network. According to our experiments, the Prophet model provides more accurate predictions in terms of the absolute errors.

Moreover, we have investigated if these predictions can be used to parameterise a model aimed at estimating the expected confirmation time of a transaction given its offered fee. We have shown two scenarios and in both cases we obtained valuable predictions that can be used to study the trade off between the blockchain running costs and the quality of service.

Although our study has been carried out for the BTC blockchain, it can be extended to any similar system where transactions are chosen from the Mempool according to an auction (e.g., Ethereum blockchain).

Future works have several directions. First, it would be important to compare the approach proposed here with other forecasting models, e.g., based on machine learning. Second, an analytical model of the queueing processes associated with the transactions should be studied to avoid the computationally expensive Monte Carlo simulations required to obtain the prediction on the expected confirmation time.

References

1. Balsamo, S., Marin, A., Mitrani, I., Rebagliati, N.: Prediction of the consolidation delay in blockchain-based applications. In: Proc. of Int. Conf. on Performance Engineering (ICPE). pp. 81–92 (2021)
2. Box, G.E., Jenkins, G.M., Reinsel, G.C., Ljung, G.M.: Time Series Analysis: Forecasting and Control. John Wiley & Sons (2015)
3. Decker, C., Wattenhofer, R.: Information propagation in the bitcoin network. In: IEEE P2P 2013 Proceedings. pp. 1–10. IEEE (2013)
4. Faghih Mohammadi Jalali, M., Heidari, H.: Predicting changes in Bitcoin price using grey system theory. Financial Innovation **13**(6) (2020)
5. Fourneau, J., Marin, A., Balsamo, S.: Modeling energy packets networks in the presence of failures. In: Proc. of 24th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS. pp. 144–153. IEEE Computer Society (2016)

6. Gallina, L., Hamadou, S., Marin, A., Rossi, S.: A probabilistic energy-aware model for mobile ad-hoc networks. In: Proc. of Analytical and Stochastic Modeling Techniques and Applications - 18th International Conference, ASMTA. Lecture Notes in Computer Science, vol. 6751, pp. 316–330. Springer (2011)
7. Kasahara, S., Kawahara, J.: Effect of Bitcoin fee on transaction-confirmation process. J. of Industrial & Management Optimization **15**(1), 365–386 (2019)
8. Kawase, Y., Kasahara, S.: Priority queueing analysis of transaction-confirmation time for bitcoin. J. of Industrial & Management Optimization **16**(3), 1077–1098 (2020)
9. Li, J., Yuan, Y., Wang, F.Y.: Analyzing Bitcoin transaction fees using a queueing game model. Electronic Commerce Research pp. 1–21 (2020)
10. Malakhov, I., Marin, A., Rossi, S., Smuseva, D.: Fair work distribution on permissioned blockchains: a mobile window based approach. In: Proc. of IEEE International Conference on Blockchain. pp. 436–441. IEEE (2020)
11. Mudassir, M., Bennbaia, S., Unal, D., Hammoudeh, M.: Time-series forecasting of bitcoin prices using high-dimensional features: a machine learning approach. Neural Computing and Applications (2020)
12. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2009), http://www.bitcoin.org/bitcoin.pdf
13. Taylor, S.J., Letham, B.: Forecasting at scale. The American Statistician **72**(1), 37–45 (2018)
14. Zarir, A.A., Oliva, G.A., Jiang, Z.M., Hassan, A.E.: Developing cost-effective blockchain-powered applications: A case study of the gas usage of smart contract transactions in the ethereum blockchain platform. ACM Transactions on Software Engineering and Methodology (TOSEM) **30**(3), 1–38 (2021)