Enabling Auditing of Smart Contracts Through Process Mining

Flavio Corradini, Fausto Marcantoni, Andrea Morichetta, Andrea Polini,
Barbara Re[(✉)], and Massimiliano Sampaolo

University of Camerino, Camerino, Italy
{flavio.corradini,fausto.marcantoni,andrea.morichetta,
andrea.polini,barbara.re,massimiliano.sampaolo}@unicam.it

Abstract. The auditing sector is acquiring a strong interest in the diffusion of blockchain technologies. Such technologies guarantee the persistence, and authenticity of transactions related to the execution of a contract, and then enable auditing activities. In particular, they make possible to check if observed sequences of transactions are in line with the possibly expected ones. In other words, auditing blockchain transactions allow users to check if the smart contract fits the expectation of the designers, that for instance could check if a given activity is performed or if it satisfies a given set of properties. In such a setting we propose a methodology that exploits process mining techniques to evaluate smart contracts, and to support the work of the auditor. Models resulting from the mining can be used to diagnose if the deployed application works as expected, and possibly to continuously improve them. We illustrate the use of our approach using a small, but real, case study.

Keywords: Blockchain · Smart contract · Process mining · Audit

1 Introduction

The adoption of blockchain-related technologies is spreading over many different contexts. When adopted in a new context [26] they generally have disruptive effects on traditional business, and they introduce novel ways of interactions (e.g., payment [19], agriculture [23] and others). Such transformations involve not only companies but also public authorities, that are currently recognising the potentialities of such technologies. The success of the "blockchain" is also confirmed by the significant interest of the community towards a technology able to guarantee trust natively using a faultless and robust validation system (e.g., in the last two years there have been 3.7 million Google searches for blockchain). The other key factor in the success of such a technology is tied to smart contracts. A smart contract is very much similar to a real physical contract which however takes the form of a digital artefact, and it can be used to establish business relations. These relations are enforced automatically via transactions as soon as the terms of the contract are fulfilled, and then the transactions are

M. H. ter Beek et al. (Eds.): Gnesi Festschrift, LNCS 11865, pp. 467–480, 2019.
https://doi.org/10.1007/978-3-030-30985-5_27

stored in the blockchain. The execution of a smart contract results in a set of activities that are carried out in a particular order. The order of execution of the activities describes the business logic of the contracts, and provide evidence to the interested partners about their completion.

Even though in a contract it is generally useful to define an order for the permitted actions, the smart contract specification does not provide mechanisms to enforce an order. It is then generally useful to define new methodologies for auditing the control flow of blockchain-based applications [13], and then to check if the actual execution of the contract functions conforms to the expected ones. Process mining is certainly a possible strategy to support auditors in such checks. Indeed, previous experiences show the possible benefits of process mining [2] in relation to auditing activities [4,20]. Such experiences underline the possibility to perform a better analysis of the process flow based on historical data as well as the possibility of auditing processes on-the-fly. Up to now, blockchain has the potential to impact the audit sector making particularly significant the application of process mining as a supporting technique.

In this paper, we illustrate the methodology, and we report the results we obtained in applying process mining for auditing smart contracts. In particular, we consider the list of transactions resulting from the execution of RotoHive, that is an online fantasy sport running weekly tournaments. The application has been implemented as a smart contract on the Ethereum blockchain, that provides a set of functions that a player can invoke to play in a tournament. In running process mining we apply three different algorithms: the Heuristics Miner [24], the Inductive Miner [21,22], and the Split Miner [10]. Fitness, precision and generalisation are measured to check the quality of the mining activity. The major benefits of our methodology are as follows.

– Reduce time and cost for auditing contracts usually done manually on a set of transactions randomly selected.
– Improve the effectiveness of auditing, since by looking at all the transactions, auditors will inevitably find more exceptions requiring follow-up.
– Make it easier to investigate deviations highlighting anomalies at run-time.

The rest of the paper is organised as follows. Section 2 provides an overview of blockchain technology and process mining. Sections 3 introduces the methodology we follow in our study, while Sect. 4 presents the case study we consider as well as recommendations resulting from the conducted analysis. Section 5 presents related works available in the literature. Finally, Sect. 6 closes the paper with some remarks and opportunities for future works.

2 Background

This section presents the relevant notions related to blockchain, with a particular focus on Ethereum, and process mining.

Blockchain and Ethereum. A blockchain is a distributed ledger composed by a linked list (cf. chain) of records called blocks [26]. Each block contains a limited number of transactions in its body, while the header includes, among other things, the hash of the current block and the hash of the previous block. New blocks are added to the chain at regular intervals of time by the so-called "miners". These are computational nodes related to the blockchain infrastructure that is needed to derive the hash of a block. The mining process and the use of consensus protocols permit us to verify the genuineness of the transactions included in each block. Finally, the replication of the chain in any node of the network guarantees decentralization and trustworthiness, without the need of a third party independent authority. The blockchain ideas have been initially proposed to support payment systems based on cryptocurrencies. In the last years, its adoption spread off in many different contexts, also about the inclusion of additional mechanisms, such as that of *smart contracts*. These can be considered as special programs which are executed over the blockchain infrastructure, whose nodes are now equipped, in some specific technologies such as Ethereum, with computational power. The execution of smart contracts produces transactions to be stored in the blockchain, thus ensuring trust among the parties.

Ethereum is a concrete implementation of the blockchain that includes support for the execution of smart contracts [33]. This is the technology we used in our approach. In Ethereum every node connected to the Ethereum network embeds an instance of the Ethereum Virtual Machine (EVM). The operations executed in the EVM, like storage of information or contract instructions have an associated economic cost defined in terms of *GAS*, which is the unit measuring the amount of computational effort needed for the execution of the operation. The execution cost has two main advantages: (i) it reduces the risk of malicious computational tasks, and (ii) it encourages mining activities by network participants and, hence, it permits to keep the overall system working. Indeed, miners are rewarded for each block they mine with a default amount of Ethers plus the sum of the transaction fees included in the block. Currently, the most prominent language to write smart contracts for Ethereum is *Solidity* (https://solidity.readthedocs.io/).

Process Mining. Process Mining is a discipline in between data mining and computational intelligence on the one hand, and process modeling and analysis on the other [2]. Process mining aims to extract non-trivial and useful information from event logs available in today's information systems for discovering, monitoring and improving real processes [3]. It is an evidence-based approach, and this ensures a closer correspondence between modeled and observed behavior because the evaluation and definition of the model are based on real process execution traces.

In process mining, we can distinguish different activities such as discovery and conformance. The first technique, **discovery**, produces a model from an event log without using any a priori information, and usually, the discovered model is a process model expressed in a formal notation. The second class is **conformance**; it allows users to compare a process model with an event log of the same process. This is a useful technique to check whether a process as inferred from the log corresponds to the expected model and vice versa.

The discovery activity is generally based on an algorithm able to produce a model from a log. Over the years several mining algorithms have been developed, each with its proper characteristics [9]. In this paper, we apply three of them, such as Heuristics Miner, Inductive Miner and Split Miner, and we shortly discuss the results we get.

- The **Data-aware Heuristic Miner (DHM)** is an algorithm for discovering process models where the behaviour is obscured in the event logs by noise, infrequent outliers or recording errors [24]. Data-aware Heuristic Miner uses the data attributes and dependency condition to distinguish infrequent paths from random noise by using classification techniques directly embedded in the discovery algorithm built upon the Heuristic Miner. The discovered models are, then, visualized as Causal Nets (C-Nets), a concise graphical notation with clear semantics, which includes information on split and join gateways.
- The **Inductive Miner** is an algorithm based on a divide-and-conquer approach [21,22]. Such an approach is applied to the log splitting it into sub-logs and then recursively applied to these sub logs until they contain only a single activity. In this way, the problem of discovering a process model for a log is broken down in discovering several sub-processes, one for each sub-log. The algorithm ensures to return a sound, fitting and block-structured process model in finite time.
- The **Split Miner** is an algorithm similar to the heuristic miner, however experiments showed that the algorithm is 2–6 times faster than other state-of-the-art methods [10]. The first step of the algorithm constructs the Directly-Follows Graph; then it detects self-loops and short-loops to discover concurrency relations between pairs of tasks. Whenever a likely concurrency relation between two tasks is discovered, the arcs between these two tasks are pruned from the Directly-Follows Graph resulting in a pruned Directly-Follows Graph. In the third step filtering is applied to the pruned Directly-Follows Graph to strike balanced fitness and precision, still maintaining low control-flow complexity. In the fourth step, split gateways are discovered for each task in the filtered pruned Directly-Follows Graph with more than one outgoing arc. This is followed by the discovery of join gateways that is the last step of the algorithm.

It is worth noticing that processes resulting from the mining are different in term of representation language. All of them can be traced back, up to some transformations, to BPMN [27] that is the target language we use in this paper being well-know and understandable to auditors.

To measure the quality of a discovered model in comparison to the event log that generated it several quality parameters have been defined [30]. Among the other we refer to:

- **Fitness**: permits to measure the extent to which the discovered model can accurately reproduce the cases recorded in the log;
- **Precision**: permits to measure how much additional behaviour is included in the model i.e. a poor precision means that a model admits much additional behaviour with respect to that reported in the log;

- **Generalization**: permits to measure how much the model just reproduce the behaviour reported in the log i.e. a low level of generalization means that the model cannot handle much more behavior with respect to the one reported in the log, maybe because not yet observed.

Overall the purpose of mining is to discover a model representative of the behaviour expressed by the event log and "to guess" additional behaviour. To generate a process model in line with reality, the algorithms should maintain a proper balance between overfitting and underfitting. The former property means that the generated model is too specific and only admits behaviour similar to that observed, while the latter property, however, presents a model too general which also accepts behaviours that are probably unrelated to the observed one.

3 Enabling the Auditing of Blockchain Contract

In this paper, we envisage a scenario based on process mining techniques to support auditing of processes "enacted" using smart contracts. In Fig. 1 we illustrate how the methodology we propose fits in the life-cycle of business transactions established through a smart contract. In particular, given a set of requirements on the transactions, a developer will define a smart contract (expressed in Solidity in our case), that will be successively deployed and executed over a blockchain infrastructure (EVM in our case). The execution of the contract will lead to a set of related transactions stored in the blockchain. At that point, it is important to check that, among other checks, the sequence of actions and interactions put in place by the contract participants are in line with what was expressed in the requirements. To enable such auditing activity we conceived and implemented the ABC (Auditing Blockchain Contracts) methodology that we will detail in the following. The methodology consists of four phases executed one after the other iteratively as represented in Fig. 2.

Smart Contract Transactions Retrieval. The first phase of the methodology consists in the selection of a smart contract to be audited from the blockchain. The proposed approach has some interest in case the contract embeds a complex behaviour in terms of ordering of the contract foreseen operations. In general, not all contracts implement complex behaviours, since they contain single functionality usually not correlated each others. In this work, we are interested in challenging contracts with a complex logic since we believe that auditing can give greater benefit in case of complex behaviour. In this work, we select two requirements to consider the contract auditable: (i) the number of recorded transactions should be higher of a given threshold calculated on the dimension of the contract (i.e., in our case we set such threshold to 100), and (ii) it should contain at least one user any links to multiple interactions on different methods of the contract to observe meaningful emergent behaviours. Indeed, if this were not the case, we would not have a concrete order on the operations of the contract. From a technical point of view, the described operations are pretty

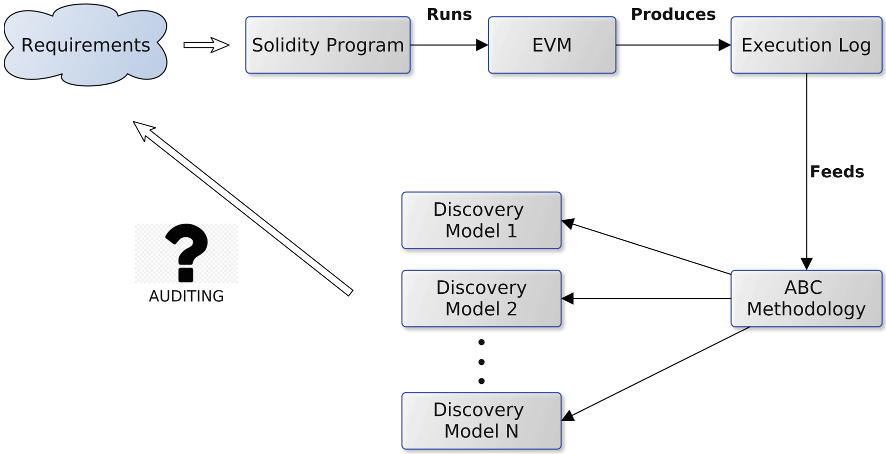

Fig. 1. ABC methodology context.

straightforward. We developed a simple application in C# integrating the Etherscan API, that permits to scan the blockchain looking for contracts according to pre-selected requirements, and it allows users to get the list of transactions in JSON format.

Transactions Clustering. The second phase of the methodology performs clustering activities on the retrieved blockchain transactions. The main challenge at this point is connected to the selection of the clustering criteria. Smart contracts do not integrate the notion of traces; each transaction represents something performed without any links with other transactions. To implement a significant correlation and to generate a set of traces we need to cluster transactions according to some logic.

In our approach, we solve the problem of creating traces grouping together transactions coming from the same sender. This means that a new trace is generated for each user. This trace contains the list of transactions exchanged between the user, and the contract ordered according to their timestamps. The main drawback of this clustering methodology refers to the possibility of correlating sequential operations just because they are executed one after the other, even if in reality they do not have any causal dependency.

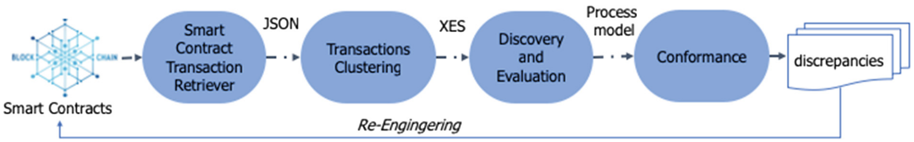

Fig. 2. ABC methodology.

From a technical perspective, in the clustering step we take the JSON file produced in the previous phase, and we generate an event log, which is then stored in a file in XES format [28].

Discovery and Evaluation. The third phase of the methodology performs a process mining discovery activity. In this work we have used three different discovery algorithms: the Heuristics Miner [24], the Inductive Miner [21,22], and the Split Miner [10]. We consider the Inductive Miner and the Split Miner thanks to its performance characteristics [9], and we also include the Heuristic Miner because it generally performs better with respect to quality criteria [12]. The used algorithms generate three different models that are compared using quality measures like fitness, precision and generalization [1]. Running three algorithms the auditor has the possibility to consider a wider spectrum of possible behaviours. Indeed the three resulting models collectively represent different and possible working scenarios. From a technical perspective, we take in input the log in the XES format and using the Apromore process mining tool[1] we discover the behavioural model emerging from the recorded transactions using Split Miner, while we use ProM[2] in the case of Heuristic and Inducting. Finally, ProM was used to compute quality measure.

Conformance. The last phase of the methodology analyses the models generated by the discovery phase, to find discrepancies concerning what is expected by the specified requirements. This is the most important phase of the auditing activity; furthermore, this analysis phase will lead to a successive contract re-engineering in case of unsatisfactory results. In the presented approach this activity does not include automatic support, yet. Nevertheless, it is clear that model checking techniques [15,16], to check interesting temporal properties, seem to be a perfect fit for such an activity. Clearly, in such a case it will be necessary to equip the auditor with user-friendly tools to define relevant properties out of the requirements list.

4 Process Mining in Blockchain: The RotoHive Case

In this section, we show the methodology in practice considering a real case study such as RotoHive[3]. More details on data used in the experiment as well as resulting model are available on-line[4].

4.1 RotoHive Overview

RotoHive is a fantasy sport running weekly tournaments. Every Tuesday a new tournament starts, and users are asked to rank National Football League (NFL)

[1] http://apromore.unicam.it.

[2] http://www.promtools.org.

[3] https://www.rotohive.com.

[4] http://pros.unicam.it/blockchainauditing/.

players by role based on projected performance for the week. RotoHive user submissions are then rated against real player performances. At the end of Monday night football matches, top performing RotoHive users are paid according to the rank of the selected players. This process repeats on Tuesday morning when the next weekly tournament begins. Roto can then be staked to user submissions to win a portion of a separate weekly Ethereum prize pool.

4.2 ABC Methodology in Practice

Considering **the smart contract transactions retrieve** activity, the RotoHive application was selected since it contains more than 3000 transactions[5] distributed over 4 months (from August to December 2018), and it includes several users. This characteristics make it a quite challenging scenario for experimenting with the proposed approach.

The **transactions were clustered** and formatted in a XES file considering the users interacting with the contract. Each trace is identified by a tag containing the address of the user, and a list of events performed by the user on the contract. Each event contains the name of the method called if it is completed and the corresponding timestamp. Listing 1.1 shows an excerpt of the XES file representing a trace performed by a user for the RotoHive *stake* method resulting in a transaction.

Listing 1.1. Log Excerpt.

```
<trace>
   <string key= ``concept:name” value=``0xd12c89fe9dccb84dd8fc2ba426dffe94169”/>
      <event>
      <string key= ``concept:name” value=``stake”/>
      <string key= ``lifecycle:transition” value=`` complete”/>
      <date key= ``time:timestamp” value=``2018-10-12T04:39:00.000+02:00”/>
      <string key= ``event” value= ``stake”/>
      </event>
     .
     .
     .
      <event> ... </event>
</trace>
```

The **process discovery** resulted in three models generated applying the Split Miner, Inductive Miner and the Heuristic Miner. The processes are depicted in Figs. 3, 4 and 5 respectively.

The three discovered models contain the same number of tasks, with two principal dominant behaviours, one representing the users playing the game and the other covering the behaviour of the administrator. The path representing the users contains just one task *stake* closed in a loop, indicating that a player can perform multiple stakes in each tournament. The path representing the administrator is composed of two initial tasks *constructor* (i.e., 0x60806040) and *settokencontract* indicating the first initialization of the game followed by the tasks representing the tournaments. In the tournament we have *createtournament* for the creation of a new tournament followed by the operation performed once the tournament is completed *releaseroto*, *rewardroto*, *destroyroto*, and *closetournament.*

[5] https://etherscan.io/address/0x0d19d264207a3afad4094f26b693ff5590361b0d.

Analysing the models we can state that the behaviour of each player is rather simple, and all the models reproduce a similar structure. Different is the case for the administrator part where the three models differ significantly for the tasks executed at the end of each tournament: *destroyroto, releaseroto, rewardroto*, and *closetournament*. The Split Miner, in Fig. 3, admits a first occurrence of the *rewardroto* task, and then the other tasks. In particular, *destroyroto* when executed occurs always after *releaseroto*. The Inductive Miner, in Fig. 4, admits to create the tournament and then two paths are possible. It can complete or execute *rewardroto* followed by two paths in parallel. The first includes the possibility to eventually execute several times *releaseroto*, while the other can execute *destroyroto* follow by *closetournament* tasks enclosed in a loop structure. This two paths are successively synchronised, and then the process ends. The Heuristic Miner, in Fig. 5, instead admits *createturnament* that is always followed by *rewardroto*. Than the three tasks *releaseroto*, *destroyroto* and *closeturnament* can be execute in sequence. Eventually *releaseroto* and *destroyroto* can be skipped.

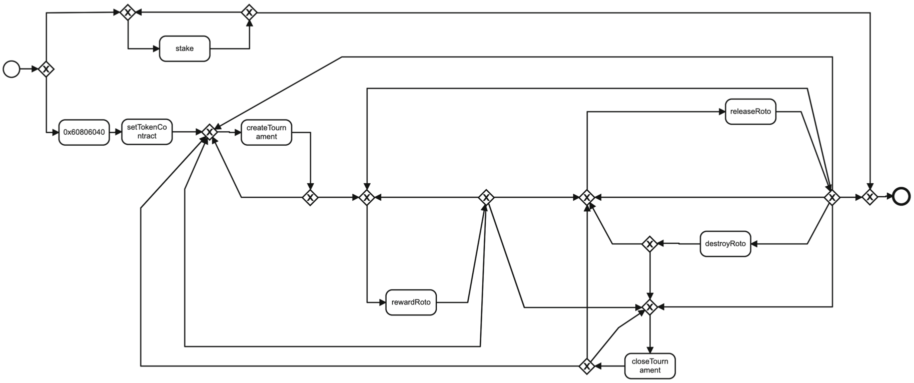

Fig. 3. RotoHive Split Miner.

To evaluate the quality of the mining algorithms applied to the RotoHive case study we considered fitness, precision, and generalisation in Table 1. Generally, we can observe that both fitness and generalization values are quite good for all 3 algorithms, while precision is more variable, and in general observed values are lower. Notably, having models with a value of fitness equal to one guarantees that all the traces in the log can be reproduced by it.

All three models represent quite well the application domain, so it is challenging to choose the best process mining algorithm to be used for audit applied to the blockchain domain. At this point, the evaluation is up to the auditor, who must consider all the models and their quality. If the auditors are interested in a model reflecting better the whole log our best solution is the Split Miner or Inductive Miner with the highest value of fitness, but with the drawback of low precision. If the auditors are more interested in highest value of precision they

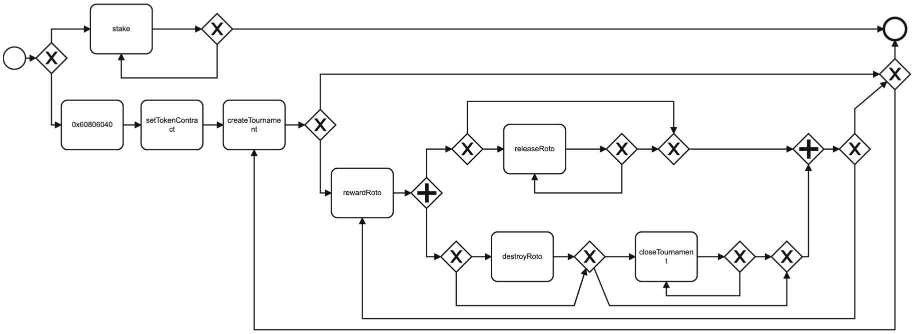

Fig. 4. RotoHive Inductive Miner.

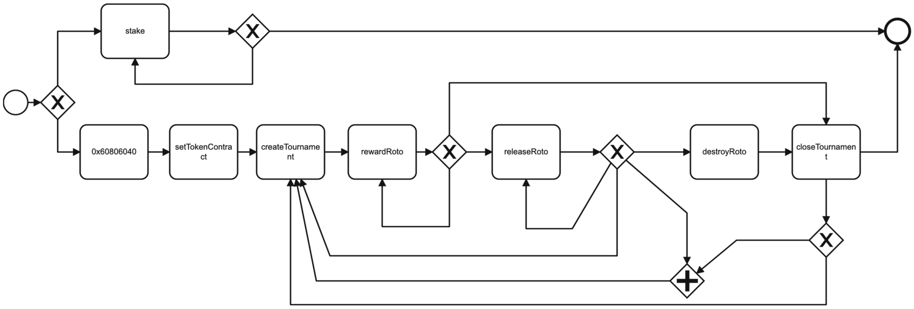

Fig. 5. RotoHive Heuristic Miner.

should use the Heuristic Miner loosing a bit the quality of the other parameters that are slightly below the others but not significantly.

4.3 Discussion

The used approach has lead to good results with sound models discovered, and pretty good quality parameters measured. The usage of more than one mining algorithm seems somehow desirable for auditing purpose. Indeed, the objective of the auditors is to identify any potential risk, and to make a careful assessment on what happened, but also on what could potentially happen. In this sense, we are working to make the presentation of potential risks easier.

Potentially the methodology used could also be useful to understand how a user interacts with a system, and to compare different behaviours with the expected one defined by requirements. Going even further in this direction we could understand the characteristics of certain users by analysing how they interacted on different systems. The designer after an accurate evaluation of the divergences can also decide to review the contract to force or avoid specific behaviour.

Table 1. RotoHive quality measures.

Algorithm	Fitness	Precision	Generalization
Split Miner	1	0.20486	0.99897
Inductive Miner	1	0.20389	0.99881
Heuristic Miner	0.92307692	0.5	0.99872

5 Related Work

In this section, we refer to the research available in the literature that inspired our work. We first discuss other papers proposing process mining techniques for audit, then we discuss solutions to enable secure and trustworthy auditing of logs.

Much effort has been devoted to the application of process mining techniques to auditing scenarios. Here in the following, we refer to those contributions supporting, as in the case of our approach, a semi-automatic strategy.

Dogana and Curbera [17] present a semi-automatic auditing approach in cases where there is no process execution engine. Ghose and Koliadis [18] present a broad auditing framework suitable to check the compliance status of a business process against given regulations. Zerbino et al. [34] propose a novel methodology for auditing information systems; they also discuss an application on the information exchange among port stakeholders. The authors provide operational guidance bridging the gaps of the current approaches for off-line information system auditing. Similarly to our approach the proposed methodology promotes the process reengineering, and for revising the boundaries in the process flow of the port community system. Accorsi and Stocker [5] use conformance checking for security auditing. They also discuss a case study employing a bank scenario and a real-life loan application process. Conformance checking is also introduced by Ramezani et al. [29]. In this paper the check considers the control flow and the normative requirements. Mayers et al. [25] use process mining and conformance checking analysis techniques to identify anomalous behaviour and cyber-attacks using industrial control systems data logs. Moreover, Arya et al. [7] use event logs collected in real time to run conformance on the operational process. The obtained results are also compared with simulated event logs to perform more accurate conformance checking. Different from our work none of the considered papers take into account blockchain transactions as a log for auditing those applications based on blockchain.

Finally, we considered solutions to enable secure and trustworthy auditing of logs. Among the others, we refer to Ahmad et al. [8], and Sutton and Samavi [31,32] discussing the possibility of the blockchain to enable privacy auditing. In particular, Ahmad et al. [8] present a scalable and tamper-proof system. Sutton and Samavi [32] provide a mechanism for log integrity and authenticity verification, by means of compliance checking queries. These papers underline the importance of performing the auditing in blockchain-related scenarios even though they do not propose any possible solutions for such an activity.

6 Conclusions and Future Work

The increasing adoption of blockchain technology disrupts traditional businesses, and it introduces a novel way to sign and run contracts. The combined use of blockchain technologies and process mining presents novel challenges and opportunities for auditing activities that can rely on trustworthy logs.

In this paper, we present the results we obtained in applying process mining for auditing Ethereum applications. In particular, we consider RotoHive's generated transactions. This is an on-line fantasy sport that runs weekly tournaments. The auditing activity has been performed using the discovered models and considering fitness, precision and generalization.

In the future, we plan to continue our programme to support auditors of blockchain-based applications effectively. Therefore we aim at enlarging the study running a more extensive validation, and considering a broad set of different blockchain-based applications that can be optimized via cost/reward method [6]. We also intend to deepen our research on the possible selection of one or more mining algorithm, and their suitability and checking its performance and effectiveness. Moreover, we would evolve the methodology with a prototype suitable to run auditing activity in a user-friendly manner. Finally, we would like to explore other analysis techniques for auditing, i.e. monitoring [11] and conformance [14].

Acknowledgement. It is really our pleasure to take part in Stefania's Festschrift. The cooperation with her and her group is somehow recent, nevertheless it has been very profitable, and inspiring both from the professional and human profile. In particular the cooperation has strengthened in relation to the Learn PAd European research project where both UNICAM and ISTI–CNR were partners. The work we present here can be somehow considered a germination of the research carried on together within Learn PAd. We thank Stefania for her friendship, and wish her all the best for the future.

References

1. Van der Aalst, W., Adriansyah, A., van Dongen, B.: Replaying history on process models for conformance checking and performance analysis. Wiley Interdisc. Rev. Data Min. Knowl. Disc. **2**(2), 182–192 (2012)
2. van der Aalst, W.M.P.: Process Mining - Data Science in Action, 2nd edn. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
3. van der Aalst, W.M.P., et al.: Process mining manifesto. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM 2011. LNBIP, vol. 99, pp. 169–194. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28108-2_19
4. van der Aalst, W.M.P., van Hee, K.M., van der Werf, J.M.E.M., Verdonk, M.: Auditing 2.0: using process mining to support tomorrow's auditor. IEEE Comput. **43**(3), 90–93 (2010)

5. Accorsi, R., Stocker, T.: On the exploitation of process mining for security audits: the conformance checking case. In: Symposium on Applied Computing, pp. 1709–1716. ACM (2012)
6. Aceto, L., Larsen, K.G., Morichetta, A., Tiezzi, F.: A cost/reward method for optimal infinite scheduling in mobile cloud computing. In: Braga, C., Ölveczky, P.C. (eds.) FACS 2015. LNCS, vol. 9539, pp. 66–85. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-28934-2_4
7. Adriansyah, A., van Dongen, B.F., van der Aalst, W.M.P.: Towards robust conformance checking. In: zur Muehlen, M., Su, J. (eds.) BPM 2010. LNBIP, vol. 66, pp. 122–133. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20511-8_11
8. Ahmad, A., Saad, M., Bassiouni, M., Mohaisen, A.: Towards blockchain-driven, secure and transparent audit logs. In: 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, pp. 443–448. ACM (2018)
9. Augusto, A., et al.: Automated discovery of process models from event logs: review and benchmark. IEEE Trans. Knowl. Data Eng. **31**, 686–705(2018)
10. Augusto, A., Conforti, R., Dumas, M., Rosa, M.L.: Split Miner: discovering accurate and simple business process models from event logs. In: International Conference on Data Mining, pp. 1–10. IEEE (2017)
11. Bertolino, A., Marchetti, E., Morichetta, A.: Adequate monitoring of service compositions. In: 9th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, pp. 59–69 (2013)
12. Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: On the role of fitness, precision, generalization and simplicity in process discovery. In: Meersman, R., et al. (eds.) OTM 2012. LNCS, vol. 7565, pp. 305–322. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33606-5_19
13. Casino, F., Dasaklis, T.K., Patsakis, C.: A systematic literature review of blockchain-based applications: current status, classification and open issues. Telematics Inform. **36**, 55–81 (2019)
14. Corradini, F., Morichetta, A., Polini, A., Re, B., Tiezzi, F.: Collaboration vs. choreography conformance in BPMN 2.0: from theory to practice. In: 22nd International Enterprise Distributed Object Computing Conference, pp. 95–104. IEEE (2018)
15. Corradini, F., Fornari, F., Polini, A., Re, B., Tiezzi, F.: A formal approach to modeling and verification of business process collaborations. Sci. Comput. Program. **166**, 35–70 (2018)
16. Corradini, F., Fornari, F., Polini, A., Re, B., Tiezzi, F., Vandin, A.: BproVe: a formal verification framework for business process models. In: Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering, ASE 2017, Urbana, IL, USA, 30 October–03 November 2017, pp. 217–228 (2017)
17. Doganata, Y., Curbera, F.: Effect of using automated auditing tools on detecting compliance failures in unmanaged processes. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 310–326. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03848-8_21
18. Ghose, A., Koliadis, G.: Auditing business process compliance. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, vol. 4749, pp. 169–180. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74974-5_14

19. Holotiuk, F., Pisani, F., Moormann, J.: The impact of blockchain technology on business models in the payments industry. In: Towards Thought Leadership in Digital Transformation: 13. Internationale Tagung Wirtschaftsinformatik, pp. 12–15 (2017)
20. Jans, M., Alles, M.G., Vasarhelyi, M.A.: The case for process mining in auditing: sources of value added and areas of application. Int. J. Accounting Inf. Syst. **14**(1), 1–20 (2013)
21. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from event logs containing infrequent behaviour. In: Lohmann, N., Song, M., Wohed, P. (eds.) BPM 2013. LNBIP, vol. 171, pp. 66–78. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06257-0_6
22. Leemans, S.J., Fahland, D., van der Aalst, W.M.: Discovering block-structured process models from event logs - a constructive approach. Petri Nets **7927**, 311–329 (2013)
23. Leng, K., Bi, Y., Jing, L., Fu, H., Nieuwenhuyse, I.V.: Research on agricultural supply chain system with double chain architecture based on blockchain technology. Future Gener. Comp. Syst. **86**, 641–649 (2018)
24. Mannhardt, F., de Leoni, M., Reijers, H.A., van der Aalst, W.M.P.: Data-driven process discovery - revealing conditional infrequent behavior from event logs. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 545–560. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_34
25. Myers, D., Suriadi, S., Rad, K., Foo, E.: Anomaly detection for industrial control systems using process mining. Comput. Secur. **78**, 103–125 (2018)
26. Nakamoto, S., et al.: Bitcoin: A peer-to-peer electronic cash system (2008)
27. OMG: Business process model and notation (2011)
28. OMG: XES standard definition (2019)
29. Ramezani, E., Fahland, D., van der Aalst, W.M.P.: Where did i misbehave? diagnostic information in compliance checking. In: Barros, A., Gal, A., Kindler, E. (eds.) BPM 2012. LNCS, vol. 7481, pp. 262–278. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32885-5_21
30. Rozinat, A., de Medeiros, A.K.A., Günther, C.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The need for a process mining evaluation framework in research and practice. In: ter Hofstede, A., Benatallah, B., Paik, H.-Y. (eds.) BPM 2007. LNCS, vol. 4928, pp. 84–89. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78238-4_10
31. Samavi, R., Consens, M.P.: Publishing privacy logs to facilitate transparency and accountability. J. Web Semant. **50**, 1–20 (2018)
32. Sutton, A., Samavi, R.: Blockchain enabled privacy audit logs. In: d'Amato, C., et al. (eds.) ISWC 2017. LNCS, vol. 10587, pp. 645–660. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68288-4_38
33. Wood, G.: Ethereum: A secure decentralised generalised transaction ledger. Technical report, Ethereum Project Yellow Paper 151 (2014)
34. Zerbino, P., Aloini, D., Dulmin, R., Mininno, V.: Process-mining-enabled audit of information systems: methodology and an application. Expert Syst. Appl. **110**, 80–92 (2018)