Benchmarking blockchain-based gene-drug interaction data sharing methods
A case study from the iDASH 2019 secure genome analysis competition blockchain track
Kuo, Tsung-Ting; Bath, Tyler; Ma, Shuaicheng; Pattengale, Nicholas; Yang, Meng; Cao, Yang; Hudson, Corey M.; Kim, Jihoon; Post, Kai; Xiong, Li; Ohno-Machado, Lucila

Published in:
International Journal of Medical Informatics

DOI:
10.1016/j.ijmedinf.2021.104559

Publication date:
2021

Document version
Publisher's PDF, also known as Version of record

Document license:
CC BY

Citation for published version (APA):
Benchmarking blockchain-based gene-drug interaction data sharing methods: A case study from the iDASH 2019 secure genome analysis competition blockchain track

Tsung-Ting Kuo\textsuperscript{a,*,1}, Tyler Bath\textsuperscript{a,1}, Shuaicheng Ma\textsuperscript{b,1}, Nicholas Pattengale\textsuperscript{c,1}, Meng Yang\textsuperscript{d,e,1}, Yang Cao\textsuperscript{f}, Corey M. Hudson\textsuperscript{c}, Jihoon Kim\textsuperscript{a}, Kai Post\textsuperscript{a}, Li Xiong\textsuperscript{b}, Lucila Ohno-Machado\textsuperscript{a,g}

\textsuperscript{a} UCSD Health Department of Biomedical Informatics, University of California San Diego, La Jolla, CA, USA
\textsuperscript{b} Department of Computer Science, Emory University, Atlanta, GA, USA
\textsuperscript{c} Sandia National Laboratories, Albuquerque, NM, USA
\textsuperscript{d} BGI-Shenzhen, Shenzhen, Guangdong, China
\textsuperscript{e} Bioinformatics Centre, Department of Biology, University of Copenhagen, Copenhagen, Denmark
\textsuperscript{f} Department of Social Informatics, Kyoto University, Yoshida-honmachi, Sakyo-ku, Kyoto, Japan
\textsuperscript{g} Division of Health Services Research & Development, VA San Diego Healthcare System, San Diego, CA, USA

ARTICLE INFO

Keywords:
Blockchain Distributed Ledger Technology
Pharmacogenetics
Gene-Drug Interaction
Data Sharing
Smart Contract

ABSTRACT

Background: Blockchain distributed ledger technology is just starting to be adopted in genomics and healthcare applications. Despite its increased prevalence in biomedical research applications, skepticism regarding the practicality of blockchain technology for real-world problems is still strong and there are few implementations beyond proof-of-concept. We focus on benchmarking blockchain strategies applied to distributed methods for sharing records of gene-drug interactions. We expect this type of sharing will expedite personalized medicine.

Basic Procedures: We generated gene-drug interaction test datasets using the Clinical Pharmacogenetics Implementation Consortium (CPIC) resource. We developed three blockchain-based methods to share patient records on gene-drug interactions: Query Index, Index Everything, and Dual-Scenario Indexing.

Main Findings: We achieved a runtime of about 60 s for importing 4,000 gene-drug interaction records from four sites, and about 0.5 s for a data retrieval query. Our results demonstrated that it is feasible to leverage blockchain as a new platform to share data among institutions.

Principal Conclusions: We show the benchmarking results of novel blockchain-based methods for institutions to share patient outcomes related to gene-drug interactions. Our findings support blockchain utilization in healthcare, genomic and biomedical applications. The source code is publicly available at https://github.com/tsungtingkuo/genedrug.

1. Introduction

1.1. Gene-drug interaction

Genetic variation is known to affect drug response. Presence of specific genetic variants can result in variability of drug efficacy and adverse drug reactions (ADR) through alternate pharmacokinetic (PK) and pharmacodynamic (PD) pathways. One such example is warfarin, an anticoagulant commonly used to prevent or treat blood clots. It is notoriously challenging to correctly adjust warfarin doses due to inter-patient variability resulting from both clinical data (e.g., age, sex, race, body mass index, conditions, and other medications) and genetics (e.g., variants in VKORC1, CYP2C9, and CYP4F2 genes) [1]. While patients with AA genotype in SNP rs9923231 of the VKORC1 gene are sensitive to warfarin and require lower doses, those with AG or GG genotypes are less sensitive. Complications arising from inadequate warfarin dosing constitute some of the most common ADRs reported to the Food and Drug Administration (FDA) [2]. For this reason, warfarin has been added to the FDA list of drugs with pharmacogenomics labeling; the recent list has 304 unique drugs [3].
Gene-drug relationship data are very important for clinicians and researchers. There are several publicly available gene-drug interaction datasets, such as the one produced by the Clinical Pharmacogenetics Implementation Consortium (CPIC) [4]. Based on these datasets, researchers may evaluate and investigate interactions for their associations with specific patient outcomes (e.g., improved, unchanged, or deteriorated), suspected gene-outcome-relations (e.g., yes, or no), and serious side-effects (e.g., yes, or no). However, these evaluation results may be silenced within an institution. A mechanism for institutions to share the evaluation results of the gene-drug interactions they obtained locally could help speed up research.

With the advance of sequencing technology, genetic testing is becoming more available, making pharmacogenetic-based drug dosing more viable in clinical practice. CPIC is one such effort to provide peer-reviewed, updated, and evidence-based guidelines for gene-drug pairs. However, a level 1 quality guideline in CPIC requires consistent evidence, with large sample sizes in well-designed and well-conducted studies. Gathering sufficient and high-quality evidence of gene-drug outcomes is a daunting task due to technical, economic, administrative, and ethical reasons.

1.2. Traditional methods and threat models

Intuitively, we can adopt a centralized method that uses a central server and collects the evaluation results (Fig. 1A) via a traditional software program performing logging/querying operations (Fig. 2A). However, this setting could introduce multiple threats. As shown in previous studies [5–7], a central server and traditional program can present the barriers/challenges listed below:

(i) Single-point-of-failure (e.g., the whole system stops working when the server stops due to a routine maintenance or a malicious attack).
(ii) Mutable data (e.g., the information on the server may be altered by the “root” user).
(iii) Unverifiable data source (e.g., the sources of the evaluation results may also be changed on the central server).
(iv) Non-transparent software (e.g., unspecified changes and thus inconsistent code).
(v) Alterable programs (e.g., the deployed program can still be altered locally).

1.3. Blockchain smart contracts

To overcome these issues, we consider a decentralized architecture to solve the above-mentioned risks brought by a central server and traditional program. This architecture enables consistent and large-scale evidence gathering from multiple participating hospitals and individuals. Among the decentralized data storage methods, blockchain [8–11] is one of the more promising candidates (Fig. 1B). The latest blockchain platforms, such as Ethereum [9], Hyperledger Fabric [12], or R3 Corda [13], support smart contracts, (Fig. 2B) which are computer programs running on blockchain [14]. The desired technical properties of blockchain with smart contracts [14–16] include:

(i) No single-point-of-failure (i.e., it is peer-to-peer).
(ii) Immutable data (i.e., it is very difficult to change the data on the chain).
(iii) Data provenance (i.e., the source of data is confirmed and therefore cannot be falsified).
(iv) Transparent software (e.g., each software change can be verified and confirmed).
(v) Unchangeable program code (e.g., the deployed program is not alterable, and new versions of the program are recorded and visible to all nodes) [17].

Therefore, using smart contracts on blockchain to store and query patient outcomes related to gene-drug data pairs could further improve the transparency and immutability of the software among the participating institutions.

Blockchain has been proposed in various healthcare, genomic and biomedical applications [5–7], such as medical record management [18–20], dynamic consent in biobanking [21], genomic data access logging [22,23], pharmaceutical supply chain [24–26], and privacy-preserving predictive modeling [27–33]. Meanwhile, applications in pharmacogenetics are still limited [19]. While blockchain has been the underlying infrastructure for crypto-currencies such as Bitcoin [8] for more than a decade, the design and usability of blockchain have yet to be well-understood in health sciences as they currently are in the world of finance.

Although the idea of adopting blockchain and smart contracts for sharing gene-drug evaluation results may conceptually be feasible, practical issues in implementing such a system have yet to be investigated. Many blockchain-based solutions are still in early stages [23,34] and the resources to support blockchain and smart contract developers are also scarce [35,36]. Therefore, we aim at benchmarking the potential of a decentralized gene-drug system on blockchain, with smart contracts.

2. Material and methods

2.1. Competition

University of California San Diego (UCSD) adopted a community-based approach to benchmarking, and organized Track 1 of the iDASH Secure Genome Analysis Competition in 2019 [17]. There were 30 teams from 11 countries, including China, Germany, India, Japan, Luxembourg, Netherlands, Singapore, Switzerland, Turkey, United Kingdom, and the USA. The development phase lasted three months, after which five teams submitted solutions. We requested that each

![Fig. 1. Architecture of storing the patient outcome of gene-drug pairs. A. Centralized architecture (central server) where the centralized gene-drug outcome server can lead to a single-point-of-failure. The central server can change the records from other sites and can even modify the source of evaluation results. B. Decentralized architecture (blockchain) without a central server that can eliminate the possibility of a single point-of-failure. By adopting blockchain technology, the data are immutable and source-verifiable.](image-url)
solution be able to store all patient outcomes for gene-drug pair records on-chain (i.e., no off-chain local storage of data was allowed). For querying the records, the solution was required to support searching records by any combination of gene name, variant number, and drug name. Results had to contain counts and percentages of outcomes, suspected-gene-outcome-relations, and serious-side-effects. The solution was also required to make the records searchable from any site (e.g., Institution 1 should be able to search any record from Institution 2 and so on).

Existing blockchain and smart contract studies have demonstrated their features and advantages, such as immutability/robustness [8,37,38], either by mathematical proof or empirical analyses, along with thorough comparisons with centralized or redundant solutions [27,29,32,39,40]. In this competition, we aimed to demonstrate the feasibility of adopting blockchain and smart contracts to share patient outcomes related to gene-drug interactions among institutions. Of the five submitted solutions, one was unable to complete within the competition timeline and another published their results separately [41]. Therefore, in this study, we focus on the benchmarking and comparison of three solutions.

The blockchain platform we selected based on prior review [15] was Ethereum [9], which is an open source platform that supports smart contracts and that is maintained by the community. We configured the Ethereum blockchain network as a permissioned network, so that the evaluations could be executed independently of the public blockchain, and the testing environment would not be tied to the concept of cryptocurrency. We adopted the Proof-of-Authority (PoA) consensus protocol using the Clique algorithm [42], which is suitable for permissioned networks that do not need intensive computation like the one needed for the Proof-of-Work (PoW) Ethash algorithm [37] to secure the network. Compared to other platforms (e.g., Hyperledger Fabric [12] or R3 Corda [13]) that also support smart contracts, Ethereum does not require additional ordering or notary services, thus it is appropriate for our purpose. We adopted Solidity [43], one of the most popular smart contract languages running on Ethereum, to implement the solutions.

### 2.2. Data

The dataset for benchmarking was generated using the gene-drug relationship data from CPIC [4]. Each record contained the following six fields (Table 1): gene name, variant number, drug name, outcome, suspected gene outcome relation, and serious side effect. First, we obtained 127 unique gene names and 226 unique drug names from CPIC and randomly chose one gene name and one drug name as a pair to generate a record. Next, for each record, we selected a variant number [1–99], an outcome status [Improved, Unchanged, Deteriorated], a suspected gene outcome relation [Yes, No], and a serious side effect [Yes, No], all randomly. For the development process the teams were provided with four patient outcomes of gene-drug pair files, each of which with 10,000 records representing the observed patient outcome for a gene-drug pair from four institutions. During the evaluation process we utilized 200 and 1,000 records from each of the four sites.

### 2.3. Methods overview

We developed three methods to solve the distributed data sharing problem: Query Index (hashing-based mapping), Index Everything (comprehensive mapping), and Dual-Scenario Indexing (complete/wildcard mapping). The three solutions were developed by the following three teams, respectively: Emory Team, formed by members from Emory University and Kyoto University (1st place winner of the competition), Team Genigma from Sandia National Laboratories (2nd place), and Omics for all from BGI-Shenzhen (Honorable Mention). The details of these promising solutions are introduced in the following subsections.

### 2.4. Query Index

The first method, Query Index, was a domain knowledge-based approach to implement a storage and query efficient solution. The following two kinds of domain knowledge in the gene-drug interaction data sharing were utilized in the design of an efficient solution: (1) the query output is the accumulated statistics of the gene-drug interaction data, and (2) the amount of unique gene-drug relations (i.e., approximately 106 in CPIC specification) is much smaller than the amount of raw gene-drug interaction records. This implementation utilized the above two facts, stored the statistical information of all unique gene-drug relations (i.e., gene-variant-drug triples) in an upper-bounded size array and cached all indices in a hash table for fast insertion and query. Fig. 3 illustrates an example of the array and hash table data structure of Query Index. Every gene-variant-drug triple could be invoked in 8 different types of queries (i.e., a query specifying gene name, drug name, and variant number and 7 queries with wildcard characters in different fields). For example, the result of GBA-nicotine-74 will be returned in query (GBA, nicotine, 74), query (GBA, *, 74), query (GBA, nicotine, *), and so on. Based on this small number of query fields, a key-value hash table was built to support all possible queries. In the hash table, the keys were gene-variant-drug tuples and their wildcard alternatives, and the values were the indices of the actual information in the
Data*: The complete data structure is \{ gene name, variant, drug name, total outcome count, improved count, unchanged count, deteriorated count, suspected relation count, side effect count \}.

<table>
<thead>
<tr>
<th>Array</th>
<th>Hash Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data*</td>
<td>PROC, 60, aspirin</td>
</tr>
<tr>
<td>Index</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 3. Example of two records for the Query Index method.

array. Upon receiving a query request, the Query Index method first found the matching index list in the hash table if the record existed, then traversed the indices to retrieve the actual information from the array. For the insertion, with the help of the hash table, the method could locate the index of the gene-variant-drug tuple in the array in \(O(1)\) time and update the counts. If the record did not exist, the method would append the record at the end of the array and insert corresponding entries in the hash table.

2.5. Index Everything

The second method, Index Everything, was a straightforward implementation approach. Since there were only a few hundred distinct genes and drugs, a unique 8-bit unsigned integer (\(\text{uint8}\)) value was assigned to each distinct gene (respectively, drug) value. These values were assigned lazily, i.e., the next available ascending value was assigned upon the first insert containing that gene or drug. As such, a unique 24-bit unsigned integer (\(\text{uint24}\)) could be trivially derived for each gene-variant-drug triple, specifically by concatenating the corresponding three \(\text{uint8s}\). Thus, for any observation, this \(\text{uint24}\) derived by concatenation was used as an index into various outcome counts stored in the Solidity mapping structures. This indexing/storage scheme is illustrated in Fig. 4. The two query modalities (entryExists and query) implementations were similarly straightforward. Specifically, given the wildcard value (\(\text{**}\)) in any position, all possible values were searched for that position, expressed as a tuple for which any non-wildcard search value collapsed the specific dimension.

\[
\text{mapping} (\text{uint24} \Rightarrow \text{uint}) , \quad \text{e.g., seriousSideEffectCounts}
\]

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>6653732</td>
<td>3</td>
</tr>
<tr>
<td>1651834</td>
<td>7</td>
</tr>
<tr>
<td>29980</td>
<td>15</td>
</tr>
<tr>
<td>772346</td>
<td>21</td>
</tr>
</tbody>
</table>

Fig. 4. Visual depiction of the scheme of the Index Everything method (\(\|\) denotes integer concatenation) on the left, and an example mapping data structure counting side effects for each unique gene/variant/drug triple on the right. Structures like the one on the right exist for all observation categories: improved, unchanged, deteriorated, suspected relation, and side effect.

2.6. Dual-Scenario Indexing

The third method, Dual-Scenario Indexing, adopted a special data structure to store gene-drug relationship data. It was also assumed here that query operations (such as query and entryExists) were more frequently invoked than insert operations, thus the team focused on query performance optimizations. Two different data structures were used to support the precise search with all three given inputs (gene name, variant number and drug name) and the search with wildcard inputs under two scenarios: complete (i.e., gene-variant-drug) and wildcard searches. For the complete search scenario, a mapping structure named geneData mapping was used to store all \(\text{GeneDrugRelation}\) items with a key that was the concatenation of gene name A, variant number B and drug name E. Therefore, the geneData map could easily support all queries with “ABE” inputs. For the wildcard search scenario, the team built a special mapping structure \(\text{GeneDrugRelationKeyMapping}\) with keys of wildcard search strings (e.g., “AB”\(^*\)) and values of the complete search strings (e.g., “ABE”, the keys of the geneData data structure). The algorithm then pre-generated all possible combinations of geneData mapping keys for each wildcard input, and stored these combinations into the \(\text{GeneDrugRelationKeyMapping}\) data structure. For querying, the algorithm first searched \(\text{GeneDrugRelationKeyMapping}\) by “AB”\(^*\) to get all geneData keys (e.g., “ABE” and others) that correspond to \(\text{GeneDrugRelation}\) items with A and B. Then, it searched geneData mapping to get the detailed \(\text{GeneDrugRelation}\) items. An example explaining how \(\text{GeneDrugRelationKeyMapping}\) supports wildcard query operations is shown in Fig. 5.

3. Results

3.1. Evaluation

To evaluate the solutions, we inserted the two datasets (i.e., 200 and 1,000) to the blockchain either 1 or 200 records at a time to simulate different insertion speed and generated 60 queries to compute the query time required by each solution. Our evaluation criteria specified that: (a) a solution must complete the insertion of all records, (b) a solution must provide 100% correct query results, and (c) the speed of insertion and query is the most important feature, followed by storage and memory cost, and then scalability. Therefore, after checking the completeness and correctness of the solutions, we measured the insertion time, query time, disk storage, and memory usage, and then normalized these measurements to raw scores from 0 to 100. The raw scores were then weighted-summed to a subtotal score (insertion time = 35%, query time = 35%, disk usage = 15%, and memory usage = 15%). Next, the subtotal scores were weighted-summed to an overall score, with the weights corresponding to the number of test records (i.e., 200 and 1,000) to account for scalability. Finally, the overall scores for inserting 1 and 200 records at a time were averaged to generate the final scores.

The compute environment for evaluation was iDASH 2.0 [44], a Health Insurance Portability and Accountability Act (HIPAA) compliant platform based on Amazon Web Services (AWS) and supported by the UCSD Health Information Services and Department of Biomedical Informatics. We set up 24 Virtual Machines (VMs) to evaluate the solutions. Each VM had 2 CPU cores, 8 GB of RAM and 100 GB of storage; Ubuntu was the operating system.

3.2. Measurement results and final scores

Results and the scores are summarized in Table 2 and Fig. 6, respectively. As shown in the tables, inserting 200 records at a time reduced insertion time per record significantly. Also, while the insertion time increased linearly with the number of records in the test data, query times were more consistent, which could reflect the blockchain characteristic that writing is relatively slow (because it requires consensus block creation), while reading is fast (only local blocks are searched).
The required disk space (<40 MB) and memory (<300 MB) were relatively small. In terms of final scores, the Query Index method performed the best, followed by the Index Everything method. The Dual-Scenario Indexing method used more memory, and its insertion/query time and disk usage were comparable with those of other solutions.

3.3. Comparison of the three proposed methods

To further understand the differences between our three proposed methods, we analyzed the results in Table 2 for each of our proposed methods as follows. The storage usage for all solutions is similar (approximately 20–35 MB) and negligible when considering modern storage devices (e.g., 100 GB in our experiments). Therefore, our analysis focused on the other three measurements (i.e., runtime of insertion, runtime of query, and memory usage).

(1) Query Index. This method constructed a hash table for the queries and exhibited superior run time of query (23–24 s for 60 queries, or about 0.5 s per query, the fastest in all different scenarios regardless of the number of records per insertion). It also had relatively small memory usage (like the best solution, Index Everything, in all scenarios). For the runtime of insertion, it performed better when one record at a time was inserted, while it was comparatively slower when multiple records were inserted at a time.

![An example of four geneDrugRelation items](image)

<table>
<thead>
<tr>
<th>geneDrugRelation item</th>
<th>geneName</th>
<th>variantNumber</th>
<th>drugName</th>
<th>Other values</th>
</tr>
</thead>
<tbody>
<tr>
<td>GeneDrugRelationABE</td>
<td>A</td>
<td>B</td>
<td>E</td>
<td>......</td>
</tr>
<tr>
<td>GeneDrugRelationABF</td>
<td>A</td>
<td>B</td>
<td>F</td>
<td>......</td>
</tr>
<tr>
<td>GeneDrugRelationACE</td>
<td>A</td>
<td>C</td>
<td>E</td>
<td>......</td>
</tr>
<tr>
<td>GeneDrugRelationACF</td>
<td>A</td>
<td>C</td>
<td>F</td>
<td>......</td>
</tr>
</tbody>
</table>

![Key data store structure of the Dual-Scenario Indexing method.](image)
Index Everything. This approach indexed all possible queries ahead in a mapping table and performed extremely well when multiple records at a time were inserted (only 24–42% of the time used by the other two methods). It also used the least memory in all combination scenarios. However, this method required more insertion time when one record at a time was inserted. Also, the query time was slightly slower than that for the Query Index method.

Dual-Scenario Indexing. This solution created two mapping structures to store the complete and wildcard queries and provided the shortest insertion time when one record at a time was inserted. The runtimes of insertion for multiple records at a time were comparable to those for the Query Index method. It required more time to query and more memory usage when compared to the other two methods.

To summarize, different methods can be more suitable for different applications and scenarios. To achieve a fast insertion time, Index Everything (inserting multiple records at a time) and the Dual-Scenario Indexing (inserting one record at a time) would be more appropriate. To optimize query time, Query Index would be the best method. To preserve memory usage, both Index Everything and Query Index approaches could be considered.

4. Discussion

To benchmark and understand the potential of the decentralized gene-drug relationship sharing system on blockchain with smart contracts, we developed three methods: Query Index, Index Everything, and Dual-Scenario Indexing. These methods applied different techniques (hash, comprehensive, and complete/wildcard mapping) to index the queries. The concepts of the proposed methods were straightforward, and we demonstrated their feasibility. Our results can serve as the basis for future researchers to improve their blockchain-based solutions in different applications (e.g., requiring faster insertion time, needing shorter query time, or preferring smaller memory usage).

Although the speed of logging and querying gene-drug outcome records on blockchain via smart contract is not comparable with that of a traditional database and may limit the real-world applications, we believe the benefits of our proposed solution (i.e., no single-point-of-failure, immutable data, guaranteed data provenance, transparent software, and an unchangeable program) are important to the sharing of the gene-drug evaluation results. Our work also provides a contribution to the broader perspective of benchmarking blockchain platforms for non-

---

Table 2

Results of each solution with different combination scenarios of records in test data (i.e., 200 versus 1,000) and number of records inserted at a time (i.e., 1 versus 200). The runtime of Query is the time to execute 60 different queries. Note: A software update of the Dual-Scenario Indexing (marked with ‘*’) was applied after the competition deadline to produce correct results with performance no worse than that of the original submission on one record per insert, and a negligible increase in insertion speed on 200 records per insert; measured query speed increased in all cases since the correct results had smaller size.

<table>
<thead>
<tr>
<th>Number of Total Records per Site</th>
<th>Number of Records per Insert</th>
<th>Solution</th>
<th>Complete</th>
<th>Correct</th>
<th>Runtime of Insertion (s)</th>
<th>Runtime of Query (s)</th>
<th>Storage Usage (MB)</th>
<th>Memory Usage (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>1</td>
<td>Query Index</td>
<td>Yes</td>
<td>Yes</td>
<td>212.75</td>
<td>23.00</td>
<td>21.18</td>
<td>56.54</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Index Everything</td>
<td>Yes</td>
<td>Yes</td>
<td>226.75</td>
<td>28.00</td>
<td>21.23</td>
<td>56.44</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dual-Scenario Indexing</td>
<td>Yes</td>
<td>Yes*</td>
<td>203.00</td>
<td>30.00</td>
<td>21.19</td>
<td>106.11</td>
</tr>
<tr>
<td>200</td>
<td></td>
<td>Query Index</td>
<td>Yes</td>
<td>Yes</td>
<td>13.25</td>
<td>23.00</td>
<td>19.49</td>
<td>90.32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Index Everything</td>
<td>Yes</td>
<td>Yes</td>
<td>4.75</td>
<td>28.00</td>
<td>19.46</td>
<td>73.26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dual-Scenario Indexing</td>
<td>Yes</td>
<td>Yes*</td>
<td>11.25</td>
<td>29.50</td>
<td>19.50</td>
<td>106.06</td>
</tr>
<tr>
<td>1,000</td>
<td>1</td>
<td>Query Index</td>
<td>Yes</td>
<td>Yes</td>
<td>1006.75</td>
<td>24.00</td>
<td>31.44</td>
<td>59.41</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Index Everything</td>
<td>Yes</td>
<td>Yes</td>
<td>1157.25</td>
<td>29.00</td>
<td>32.98</td>
<td>58.87</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dual-Scenario Indexing</td>
<td>Yes</td>
<td>Yes*</td>
<td>1003.50</td>
<td>53.00</td>
<td>31.36</td>
<td>226.14</td>
</tr>
<tr>
<td>200</td>
<td></td>
<td>Query Index</td>
<td>Yes</td>
<td>Yes</td>
<td>51.50</td>
<td>24.00</td>
<td>24.99</td>
<td>110.10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Index Everything</td>
<td>Yes</td>
<td>Yes</td>
<td>12.25</td>
<td>29.00</td>
<td>24.09</td>
<td>103.72</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dual-Scenario Indexing</td>
<td>Yes</td>
<td>Yes*</td>
<td>49.75</td>
<td>53.00</td>
<td>25.11</td>
<td>225.73</td>
</tr>
</tbody>
</table>

Fig. 6. Final scores for each solution. The results were weighted based on the number of records in the test data (i.e., 200 records in red and 1,000 records in blue) and were averaged from the results of inserting 1 or 200 records at a time. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)
healthcare applications and implementations [45,46]. During the development and evaluation of solutions, we identified that the rapidly evolving blockchain and smart contract platform could create challenges. Looking at the example of Ethereum, the platform is implemented in using the GO programming language and has had more than 150 releases since its first release in 2014 (i.e., about 2 weeks per release on average) [47]. Therefore, the performance of our methods may be improved when the underlying blockchain platform becomes more mature.

Our observations are limited to the results based on Ethereum smart contract implementation using PoA consensus protocol. Although the general concept of the simulated evaluation for the pharmacogenetics gene-drug sharing application can be adopted by using other blockchain platforms such as Hyperledger Fabric and R3 Corda, more experiments need to be conducted to compare the speed and scalability of different blockchain platform options. Also, evaluations on a larger dataset and more blockchain nodes can further reveal the scalability performance of this application.

Moving forward, this benchmark study only simulated multiple-site record sharing, and real deployments of the suggested solutions can be the next step. For example, the implementations can be packaged into Docker [48] image files to simplify the process of adopting our proposed approaches. Additionally, our benchmarking is limited to evaluating the performance of our methods on pharmacogenetics data; investigating other aspects of blockchain (e.g., governance, adjudication, and permission controls) could also extend this study.

5. Conclusion

We demonstrated that sharing gene-drug interaction data using smart contracts on blockchain technology is feasible. Specifically, we can store 4,000 gene-drug evaluation results from 4 sites within 1 min and query all these pairs within 0.5 s. We believe these results can serve as benchmarks for future blockchain-based healthcare, genomic and biomedical applications.

Summary table

<table>
<thead>
<tr>
<th>What was already known on the topic</th>
<th>• Adoption of blockchain distributed ledger technology for genomics and healthcare applications is on the rise</th>
</tr>
</thead>
<tbody>
<tr>
<td>What this study added to our knowledge</td>
<td>• Blockchain and smart contracts are increasingly being used in biomedical research applications</td>
</tr>
<tr>
<td></td>
<td>• Benchmarking results of novel blockchain-based methods for institutions to share patient outcomes of gene-drug interactions may promote data sharing and thus enable personalized medicine</td>
</tr>
<tr>
<td></td>
<td>• The results can eventually support future blockchain-based healthcare, genomic and biomedical applications</td>
</tr>
</tbody>
</table>

CRediT authorship contribution statement

Tsung-Ting Kuo: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Resources, Software, Validation, Visualization, Writing – original draft. Tyler Bath: Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – review & editing. Shuacheng Ma: Investigation, Methodology, Software, Visualization, Writing – original draft. Nicholas Pattengale: Meng Yang: Yang Cao: Investigation, Methodology, Software, Visualization, Writing – review & editing. Kai Post: Investigation, Methodology, Software, Validation, Visualization, Writing – review & editing. Li Xiong: Investigation, Methodology, Software, Visualization, Writing – review & editing. Lucila Ohno-Machado: Investigation, Methodology, Project administration, Resources, Supervision, Visualization, Writing – review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors would like to acknowledge the support from PlatON International Limited. We would also like to acknowledge Dr. Heidi J. Sofia, Dr. Haixu Tang, Dr. Xiaofeng Wang, Dr. Xiaojian Jiang, Dr. Arif Harmanci, and Dr. Miran Kim for their support in co-organizing the competition and the workshop. The use of the integrating Data for Analysis, Anonymization, and SHaring (iDASH) 2.0 Amazon Web Services (AWS) cloud network is supported by Dr. Michael Hogarth, Andrew Greaves and Jit Bhattacharya.

Funding

The competition is funded by the U.S. National Institutes of Health (NIH) (R13HG009072). T.-T. Kuo is partly funded by the National Human Genome Research Institute (NHGRI) of the U.S. NIH under Award Number R00HG009680, the U.S. NIH (R01HL136835 and R01GM118699), and UCSD Academic Senate Research Grant RG084150. L. Ohno-Machado is funded by the U.S. NIH (R01GM118609, R01HL136835, R01HG011066). The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript. C. M. Hudson and N. Pattengale are partly supported by the Laboratory Directed Research and Development program at Sandia National Laboratories, a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

References
