


Research Paper Blockchain (RPB) : A Blockchain for Checking Previously Published and Concurrently Submitted Research Papers

Nicholas Kniveton and Navid Shaghghi 

Ethical, Pragmatic, and Intelligent Computing (EPIC) Research Laboratory
Department of Computer Science and Engineering (CSEN)
Santa Clara University (SCU), Santa Clara, California, USA
{nkniveton, nshaghghi}@scu.edu

Abstract—Thousands of technical conferences and peer reviewed journals around the world, each annually solicit hundreds of newly authored papers for indexing and publishing. Since no central indexing and publishing authority exists, each conference and journal maintains its own database of papers. It is thus relatively easy for authors to submit their papers to more than one conference or journal simultaneously despite being strictly prohibited by the various indexing authorities and publishers. A manual or even automated check of the hundreds of databases is unrealistic.

Blockchain technology, however, provides a viable solution to this long-standing problem. This paper explores a potential implementation of a Research Paper Blockchain (RPB) which stores encrypted copies of all papers submitted for publication to participating indexing and publishing authorities. Conference and journal publication chairs would attempt to add the submitted paper as a new block to RPB (uploaded through a graphical web front-end or an automated back-end interface) and thus check for the uniqueness of the submission. Based on the uniqueness, the system would either add the paper to or reject it from RPB and report a uniqueness score for the paper back to the publisher. If a paper was added, it would be time stamped as well as stamped with a percentage of uniqueness for future reference. Publishers could set their own uniqueness threshold for acceptance and thus guarantee the originality and freshness of a new paper submitted before wasting reviewer time and accepting that paper.

Index Terms—Blockchain, Double Publication, Double Submission, Originality, Plagiarism Detection, Research Paper

I. INTRODUCTION

THOUSANDS of technical conferences and peer reviewed journals around the world each annually solicit hundreds of newly authored papers for indexing and publishing. Since no central indexing and publishing authority exists, each conference and journal maintains their own database of papers. It is thus relatively easy for authors to submit their papers to more than one conference or journal simultaneously even though this is strictly prohibited by the various indexing authorities and publishers. A manual or even automated check of the hundreds of databases is unrealistic. And since the authors can simply pull their submissions from all the other conferences and journals after selecting the most highly ranked conference or journal that has accepted their paper, the authors

are able to cover their tracks with no one ever knowing that they had double submitted their paper and thus wasted hours of reviewers' time from each of the conferences or journals.

Furthermore, cases of plagiarism where an author has simply copied another already published paper to some unacceptable percentage can go undetected until someone finds both publications and cares enough to alert the publishers. Therefore, plagiarizers can easily get away with publishing plagiarized versions of existing papers in a vastly different, lesser known/read, or different language conference proceeding or journal with little chance of anyone running into both the original and plagiarized versions of the same paper.

This paper presents a viable remedy to these long-standing problems using blockchain technology. Section II will delineate a typical scenario where this issue arises. Section III will go over some background information about blockchain technology and the different ways in which they are validated and section IV covers the need for research paper validation prior to publication. section V will cover the design of a Research Paper Blockchain (RPB) under research and development at Santa Clara University's Ethical, Pragmatic, and Intelligent Computing (EPIC) laboratory and section VI will delineate RPB's implementation details. Lastly in sections VII and VIII, the current ongoing research and development of RPB, and some results are reported on respectively.

II. A CASE STUDY

Benjamin Carlisle describes an incident in which a blog post he authored was plagiarized and submitted to a research conference [1]. In particular, the article highlights the need for an automated system of checking with set protocols. When Benjamin identified the plagiarized content, the process to have the published work revoked was challenging. The plagiarized paper went through multiple human checks, but none of the reviewers recognized that much of the paper was not original.

Furthermore, there were multiple issues once the author requested the paper to be removed. Reviewers at the journal initially dismissed the claims of plagiarism as that would tarnish the reputation of the authors. Furthermore, the journal

was worried that retracting an article would be embarrassing to the journal, showing that their review process was inadequate.

These reasons show the need for a pre-publication verification algorithm that follows strict rules in order to eliminate the human factor in the decision to publish or retract an article for reasons other than scientific merit. Furthermore, by using such pre-publication verification technology many of the instances of article retractions would be eliminated as articles would be reviewed prior to publication, and plagiarized articles would never be published in the first place.

III. THE NEED FOR PAPER VERIFICATION

There are three important checks for paper verification: plagiarism, double submission, and duplicate publication.

A. Plagiarism

Plagiarism is defined as the representation or wrongful appropriation of another author's ideas, language, or work as one's own original work. With the plethora of online tools and software available today the checking of new work against already published work is often straightforward. However, due to the vast number of publications and databases which span almost all existing countries and languages, it is impossible to check each and every one of them before accepting a new contribution.

This can happen in both directions. It is possible that a researcher, having read a paper in a well-known journal or conference proceeding then replicates the paper and publishes it in an obscure journal in a different language and country which has loose or no copyright laws; and it is equally as likely that a researcher having read a paper published in an obscure journal or conference proceeding in some other country and language reproduces the paper and publishes it in a reputable journal and thus gains major recognition for work they did not originate and thus becomes a leader in that field. In the first scenario, the author(s) and/or publisher of the original work may never find the plagiarized version of their work, not have a method of recourse due to the lack of copyright laws in the country of the plagiarizer, or simply not care if the paper is plagiarized in such an obscure journal that results in no real loss in citations and readership. In the second scenario however, the original author(s) and/or publishers may not have the power to pursue any actions due to the wealth and strength of plagiarizers or the limitations faced by their country of origin due to international pressures such as sanctions just as an example.

Furthermore, how can publishers check work against papers which were never published? For instance, imagine a scenario in which an author submitted a paper for publication to a conference yet after the paper's acceptance, the author never registered for the conference and thus the paper was never actually published. What if then another individual, say a colleague or patent agent, having seen the work, reproduces the work and submits it to a different conference in an attempt to publish the work as his own original work? How would the second publisher be able to verify the originality of the work

since no existing tool can search the space of never published work?

B. Double Submission

After plagiarism, double submission is the biggest sin an author can commit. Double submission occurs when an author submits their paper to multiple conferences or journals simultaneously. Double submission is distinct from duplicate publication: while double submission can lead to duplicate publication, it often does not, making it nearly impossible to detect using existing tools. Double submission is notoriously difficult to catch as authors are often able to pull their work from conferences. A typical case might go as follows: an author writes a research paper, then submits it to multiple conferences or journals with relatively close submission deadlines. When the author receives a notification of acceptance from one of the conferences or journals, the author pulls the paper from the remaining conferences. Or, the author waits till all the acceptance/rejection notifications come in and then picks the most prestigious conference/journal to send the camera-ready version of the paper to and pulls the submission from all the other conferences/journals. This is not a case of plagiarism as the author is not reproducing someone else's work as their own but rather trying to increase the odds of having their paper published even if it is rejected by one or more publishers. Furthermore, since journal reviews often take way too long, as documented and rejected in [2], [3], and [4] for instance, it has been time and time again proposed that authors be allowed to simultaneously submit their papers to multiple journals.

Even though this may seem to be a good strategy it is never the less problematic as it is unethical and can lead to legal disputes. It is unethical for two reasons: First, each paper undergoes reviews by several peers. Those Peers are faculty and researchers who are taking time out of their own work and research in order to serve as reviewers. Instead of spending time reviewing duplicate work, these peers could be reviewing unique works produced by others that may be worthy of publication. Second, the editors of the particular conference proceedings or journal issue will most likely be tailoring and balancing the publication with all the papers which have been accepted by the reviewers. If an author's paper is accepted in multiple conferences, the author will pull the paper from all but one conference. Pulling a paper from a conference in the last minute has the potential to throw off the carefully crafted balance created by reviewers and conference proceedings editors, harming other authors and the conference as a whole.

Double submission can also lead to potential legal disputes if the author is unable to pull their accepted work, turning the scenario into a case of duplicate publication explored below.

C. Duplicate Publication

Duplicate publication can occur with or without double submission. In the case of double submission, an author submits work that has already been published to a second publisher. When an author's paper is published, the author

typically transfers an exclusive copyright of the paper to the publisher [5]. Since only one publisher can own the paper, the submission of a paper to multiple publishers could result in multiple exclusive owners of the work, creating a legal battle as to who is the true owner. Even though electronic publication has enhanced the chances of duplicate publication being detected, the tools and methods are no further reliable than plagiarism detection tools and methods. Mainly meaning that it is detectable after the fact or at best if one copy is already fully published and available before the second copy is submitted for publication. No detection mechanism exists for if both copies are under publication at the same time.

But, it should also be noted that duplicate publication is not always a problem and hence its detection alone is not enough. For example as noted by Janie Morse, editor of one of Sage publications' journals, some exceptions are the publication of a translation of an already published article into another language, a republishing of an article in an anniversary issue, or an invited republication of a particularly meritorious article in a book or special collection provided the author does not fail to provide, or the new publisher does not fail to obtain, copyright release from whoever holds the copyright - which usually is the publisher of the original article, and that the article is published with appropriate acknowledgement to the original source [6].

IV. OTHER PROPOSED SOLUTIONS

Even though various software solutions (such as Turnitin [7], Unicheck [8], Grammarly [9], etc.) for detecting plagiarism and double publication are utilized, unfortunately they are only applicable after the fact. No preventative technological solutions for those issues as well as the nefarious act of double submission have been proposed except for [10] in 2018 which the authors do also note as such, as part of the scarce results of their literature review. Their proposed solution relies on a central system with Application Programming Interface (API) access to all participating publishers' editorial systems and the sharing of the attributes - such as authors' names and email addresses, abstract, etc. - that are collected by the editorial system during the manuscript submission.

An obvious issue with that proposed system, which is not missed by the authors, is that it would require infrastructure not currently in existence. It would be a very difficult task to require publishers to create APIs for use with the system, let alone the fact that the data they would be sharing through their APIs would have Personally Identifiable Information (PII) that cannot be obfuscation for obvious reasons. Therefore, such a system would require an enormous security consideration and infrastructure. The labor, infrastructure, and maintenance costs of which would far outweigh the potential benefits of such a system.

Hence, the current best strategies in use so far have been to hold training sessions for graduate students and other researchers early in their career [2], for scientists to employ conscious efforts to ensure that plagiarism does not creep into any scientific work of theirs [11], adding measures to

punish redundant publication and duplicate submission in author guidelines [12], and for conference submission systems to require authors to confirm that their submission conforms to the conference and society rule for double submission and plagiarism [2].

Any system with real potential, would have to work without exposing PII and be operable without the need for new infrastructure created by the publishers. The use of hashing and a shared blockchain with the ability for manual submission by journal editors and/or conference Technical Program Committee (TPC) members edges the development of such preventative technology closer to reality.

V. TECHNICAL BACKGROUND

A blockchain is a decentralized, peer to peer method of data storage that is highly resistant to modification. Users upload their data into a system where participants of the peer to peer blockchain network compile the data into a block and then add it to a chain of blocks. [13] The main methodology for validating submitted blocks is the Proof of Work protocol.

In a Proof of Work system, each time a block is to be added to the chain, block validators, known as "miners", compete with each other to solve cryptography problems that are very difficult to solve but have solutions that are very easy to verify. [14]. Solving each problem requires a large amount of computational power, known as "work". Whoever solves the problem first is allowed to place the new block on the chain and is thus rewarded for their efforts with new cryptocurrency coins, or fractions thereof, based on the difficulty of the work. Since anyone can easily verify the miner's solution, users of the blockchain can consider the new block to be valid and thus the data within it to be trustworthy due to the work that was required to add the block.

This allows users of the blockchain network to know that their data is stored both securely and permanently without the need for a centralized arbiter of trust such as a bank, data warehouse company such as Google, or government agency [15].

RPB uses a Proof of Work protocol due to its known effectiveness and security.

VI. DESIGN

RPB consists of two main parts: a blockchain that stores the data and a verification algorithm that analyzes the data.

A. Blockchain

RPB uses off the shelf, existing blockchain technology that has been proven secure and reliable. RPB's proof of concept uses the bitcoin blockchain, a well known system that utilizes a proof of work consensus mechanism [16], for its simplicity and security. However, for the working product, a more elastic, scalable, and efficient solution was needed. The security and reliability provided by proof of work based blockchains comes at a cost of high energy usage and a lack of elasticity. Proof of work blockchains have only one method of verification and it is very challenging to make changes.

Changes that seem minor often require what is known as a "fork", where a completely new blockchain path is created from the old chain. Due to the changing nature of conferences and publications, creating a fork every time a minor change is needed is not a viable option. In order to allow for elasticity and scalability, RPB needed a different consensus mechanism. Mechanisms designed for enterprises meet these requirements but they are typically complex and difficult to implement. One such mechanism is the Ethereum based Quorum Enterprise Blockchain Client [17]. Quorum allows for both elasticity and scalability that RPB needs, however, implementing a Quorum based blockchain solely for RPB is not feasible. To design the RPB, a commercial blockchain service that uses Quorum was chosen. Microsoft Azure Blockchain Service met the requirements for RPB, and a Quorum based blockchain using the RAFT consensus protocol was created. Using Azure allows RPB to make changes as needed and scale the product without sacrificing its efficiency. The Control of RPB's blockchain, should it be adopted by publishers, would be through a consortium of research conferences and journal publishers, such as the Association for Computing Machinery (ACM), European Alliance for Innovation (EAI), International Academy, Research, and Industry Association (IARIA), Institute of Electrical and Electronics Engineers (IEEE), Springer, etc. for computer science and engineering manuscripts for example.

B. Verification Algorithm

The second component of the RPB, the verification process, consists of the following protocol: 1. Verify that the "transaction", which includes the uploaded material, meets the general parameters of RPB (uploader, file type, etc). 2. Run an algorithm that compares the current paper to all others in the chain. The algorithm would iterate through all papers previously added to the chain, comparing the new paper to the old ones. Whenever a matching paper is discovered, the forger would note the transaction ID of the matching paper and the percentage of similarity. 3. Once all parameters are met and the matching algorithms are complete, the miner would compile the paper into a block and add it to the chain.

VII. IMPLEMENTATION

RPB is implemented in three phases: a proof of concept using an existing blockchain product for documents, a custom prototype built on a private blockchain, and a working product running on the Microsoft Azure Blockchain Service.

A. Proof of Concept

To show that RPB is viable, a proof of concept was needed that showed the two core parts of the system could be implemented: uploading papers to a blockchain and verifying the uniqueness of papers.

1) *Blocksign*: Creating a blockchain that accepts large research papers is not an easy task, so existing products were explored. A product produced by a company called Blocksign that verifies signatures on PDF documents by storing a hash

of the document on a blockchain [18] was chosen. Blocksign piggy-backed their service off the pre-existing blockchain developed for Bitcoin. Using a function known as OP Return that is attached to the script for every bitcoin transaction, users can input up to 40 bytes of data that will be added to the bitcoin transaction, which will in turn be added to the blockchain. Blocksign utilizes OP Return to store the hash of a PDF document. Blocksign did not perform similarity comparisons between papers, which RPB must do, but Blocksign provides a means to upload papers to a blockchain. When a Blocksign user signs a paper, they upload it to the Blocksign website, which hashes the entire contents of the signed paper. The output hash is then added to a bitcoin transaction to be stored on the blockchain. Later on, a user can verify the document, its signatures, and the time it was signed. Since the hash was encoded on the blockchain, the user can trust the timestamp and signatures.

2) *Verification Algorithm*: To create RPB's proof of concept, a comparison algorithm that could run on top of Blocksign's services was created. The algorithm took the text of the documents that had been uploaded and hashed them using a SHA-1 hashing algorithm. While SHA-1 is not considered secure compared to SHA-256, for the purposes of a proof-of-concept SHA-1 is appropriate as it is simple and easy to implement. Next, the algorithm compared the hashes of the two in order to verify if the paper was unique or whether it was copied. This allows the algorithm to detect when two documents were identical, indicating plagiarism.

3) *Proof of Concept Results*: Using the verification algorithm comparing the hashes of two papers stored on blocksign, the proof of concept was able to successfully recognize when two papers were identical and when they were unique. However, due to the nature of hashing functions, the comparison was atomic. One slight change in the document would yield a completely different hash, meaning that a user could simply change one letter in the document to fool the system. A malicious actor for instance could submit a paper to a conference, and then change only one letter or word before submitting it to another. When the second conference uploads the paper to RPB, the verification algorithm would tell the user that there are no matches to the uploaded paper and hence accept the paper as a new block on the blockchain. For this reason, the use of the bitcoin blockchain is not ideal for the final version of our product and a more robust system needed to be developed.

B. Prototype

While the proof of concept showed the user whether or not two papers are identical, this information was not particularly useful because since a hashing function outputs a completely different string for each unique input, the hashes of two papers were completely different even if the papers differed by only a letter or word. Therefore, the verification algorithm would allow a malicious actor to write a paper, submit it to a conference, and then change only one letter or word before submitting it to another. When the second publisher uploads the paper, the prototype verification algorithm would tell the

user that there are no matches to the uploaded paper. For this reason, the use of the bitcoin blockchain is not ideal for the final version of our product. RPB would need to use a system that goes beyond a simple hash of the complete document. To accomplish this. Thus a custom blockchain was implemented that uses blocks that have room for plain text so that the system can determine what degree of similarity the papers share rather than whether they are identical.

1) *Blockchain Template*: An IMB python blockchain template [19] was used to create a proof-of-work blockchain in python and then modified to store chunks of plain text inside the blocks. The blockchain did not include a cryptocurrency as that feature is not necessary for the operation of RPB. The goal of the prototype was to show that papers can be added to a blockchain and compared in a non-atomic manner. The details of creating the blockchain are unimportant to report in this paper as it was created from an existing template that uses the proof of work consensus mechanism. Proof of work blockchains are generally well understood and the security of blockchain technology has been proven through extensive research and real world use. Therefore instead, this subsection focuses on the modifications we made to the standard template to accommodate the features of RPB. As with any blockchain, the entire blocks in the RPB prototype are still hashed but RPB's custom chain allows for large amounts of data to be stored in each block. Each block has a size of 100kb, allowing for approximately 100,000 characters of text (minus the small amount used for the block's code and hash) to be added to the block. This size was chosen to allow for nearly all papers to fit on a single block, easing comparisons. This size can be manipulated based on the specific use case RPB is being used for. RPB stored the text data in JSON format, which is the format typically used for storing digital transactions. Each paper to be added to the blockchain was treated like a "transaction" with a date, time, and uploader identification. In the "new_transaction" section of the template, the plain text data was inserted from the uploaded paper into the blockchain block. Once inserted, each block was added to the chain as it normally would.

2) *Comparison Algorithm*: RPB's custom comparison algorithm reads the data inside each "transaction", which in this case is the text, and compares it to the data in another transaction. A python text comparison library [20] was used to compare the data in each transaction. After the comparison is run, a percentage of similarity is outputted to the user. This allows the user to decide whether or not the paper needs further investigation. RPB is intended to have a variable cut off point, as percentages of similarity may differ based on different use cases. For example, if a paper consists of large amounts of quoted and cited text, a high percentage of similarity would be expected. Another case in which two original reports are being compared might require a very low percentage of similarity. Hence, RPB allows the user to determine what is acceptable for them by adjusting the similarity level acceptable.

3) *Prototype Results*: Using the newly developed proof of work blockchain and verification algorithm, RPB was robustly

tested. Multiple papers with varying degrees of similarity were uploaded to the prototype blockchain and then processed to reveal degrees of similarity. The prototype was tested at multiple different acceptable levels of similarity, and each time the prototype successfully accepted the papers that fell below the threshold and rejected the ones above the threshold. The results showed that RPB was a working concept and merited further development. The results also showed the challenges of using blockchain in an environment that varies consistently. To make adjustments to RPB, the source code had to be changed each time, requiring the blockchain to start over again. Restarting a local blockchain is not an issue, however, if changes needed to be made to RPB after it had been deployed to a large number of conferences and publishers, changing source code and restarting the chain would not be an option. For this reason, it became clear that RPB be best implemented using a commercial service that allows for active management of the blockchain.

C. Working Product

1) *Commercial Blockchain Service Selection*: After reviewing the results from the prototype and reviewing different consensus mechanisms, an enterprise block-chain service was pursued. Multiple services were reviewed, but it was decided to run RPB on Microsoft Azure's Blockchain Service. Azure allows RPB to be implemented in a manner that is scalable, secure, efficient, and easily adaptable to different organizations. Azure allows the uploading of RPB's comparison algorithm to the service and running it on top of Microsoft's Blockchain Workbench. Azure's Blockchain is an Ethereum based system that uses the RAFT consensus mechanism [21]. Through Ethereum's Raft mechanism, RPB will be able to grow and expand. By using Azure, RPB is much more scalable and will utilize a blockchain mechanism that is backed by a reputable organization.

2) *Implementing RPB using Azure*: After selecting Azure Blockchain Service, RPB's verification algorithm was converted into a JSON based application that could run on Azure. Azure was configured to run RPB, and then the original python code was translated as needed and uploaded into the Blockchain Workbench. The implementation process in Azure was simple and RPB was able to quickly run live.

3) *Azure Results*: After successfully implementing the blockchain in Azure, multiple tests were performed. Ten papers of varying degrees of similarity from 0 to 100, in increments of 10 percent, were uploaded to the chain and tested using RPB. RPB successfully identified the similarity percentages between the papers and either accepted or rejected them based on the configured threshold. The final results clearly displayed that RPB was a viable, functional product.

VIII. WORK IN PROGRESS

1) *A Better Verification Algorithm*: While RPB's working product results showed that the Azure based app worked well, they also revealed room for improvement. The comparison algorithm used in RPB compares content in a quantitative

way, disregarding the qualitative components of written works. Therefore, the algorithm has the potential to miss plagiarism that an author attempted to mask by replacing words with synonyms or rewriting the ideas of others without appropriate credit. To help solve these more complex cases, the RPB team is currently developing an algorithm that uses natural language processing to better detect plagiarism and reduce false negatives. For example, the new algorithm can recognize text in quotation marks that is cited properly, allowing this text to be removed from the tally of "copied" or "unoriginal" text. Furthermore, the algorithm can track groups of words and phrases, allowing for better detection of more complex cases of plagiarism.

2) *Graphical User Interface*: Currently, papers are uploaded to RPB through an online terminal. While this method is effective, it is not ideal for users who are unfamiliar with terminal commands. The RPB team is currently constructing a web based graphical user interface that allows conferences to more easily upload papers and access results.

By implementing these two changes, RPB's ease of use and effectiveness will be increased.

IX. CONCLUSION

The implementation of the RPB proof of concept showed that RPB is able to add papers to a blockchain and verify whether the papers match. However, more importantly, the results revealed the need for the further development. While RPB's prototype successfully revealed whether two papers had any differences, the use of a hash limited the scope of the prototype's verification system. Since a hashing function outputs a completely different string for each unique input, the hashes of two papers were completely different even if the papers differed by only a letter or word. Therefore, the verification algorithm would allow a malicious actor to write a paper, submit it to a conference, and then change only one letter or word before submitting it to another. When the second publisher uploads the paper, the prototype verification algorithm would tell the user that there are no matches to the uploaded paper. For this reason, the use of the bitcoin blockchain is not ideal for the final version of our product.

RPB's prototype solved the issues revealed in the proof of concept. The prototype results proved that RPB is functional and able to perform comparisons between papers that identify differences beyond a binary "unique" or "not unique" comparison. Files with 0, 33, 66, and 100 percent similarity were tested via the RPB prototype and the prototype performed as expected, revealing to the user the proper percentages of similarity. However, the percentage of similarity was both manipulable and had a high number of false positives. By performing superficial changes to a paper, such as replacing words with synonyms and changing the order of the text, RPB identified papers as unique, where deep down they were very similar. Furthermore, papers that had a high number of quotes were often identified as not unique, even when they had original content. These results encouraged the implementation of

a more complex comparison algorithm using natural language processing for the final product.

ACKNOWLEDGMENT

Many thanks are due to Santa Clara university's blockchain working group led by Professor Ahmed Amer of the Computer Science and Engineering (CSE) department for their belief in the project and help in ideation and brainstorming. Also, thanks to SCU's Frugal Innovation Hub for their recognition of the project's humanitarian goal and their continued support. And lastly to the department of Mathematics and Computer Science (MCS) of the Collage of Arts and Sciences as well as the department of Computer Science and Engineering (CSEN) of the school of Engineering at Santa Clara University for their support of the project.

REFERENCES

- [1] A. A. McCook, "Authors retract much-debated blockchain paper from f1000," May 2017. [Online]. Available: <https://retractionwatch.com/2017/05/24/authors-retract-much-debated-blockchain-paper-f1000/>
- [2] H. Schulzrinne, "Double submissions: Publishing misconduct or just effective dissemination?" *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 3, p. 40–42, Jun. 2009. [Online]. Available: <https://doi.org/10.1145/1568613.1568622>
- [3] U. Cem, "Multiple submission, duplicate submission and duplicate publication," *Balkan medical journal*, vol. 30, no. 1, p. 1, 2013.
- [4] S. Pressman, "Simultaneous multiple journal submissions: The case against," *American Journal of Economics and Sociology*, vol. 53, no. 3, pp. 316–333, 1994.
- [5] P. Samuelson, "Self-plagiarism or fair use," *Communications of the ACM*, vol. 37, no. 8, pp. 21–25, 1994.
- [6] J. M. Morse, "Duplicate publication," 2007.
- [7] Turnitin, LLC, "Empower students to do their best, original work," 2020. [Online]. Available: <https://www.turnitin.com>
- [8] Unicheck, "Plagiarism checker that prefers results over numbers," 2017. [Online]. Available: <https://unicheck.com>
- [9] Grammarly Inc., "Plagiarism checker by grammarly," 2021. [Online]. Available: <https://www.grammarly.com/plagiarism-checker>
- [10] M. Kolhar, A. Alameen, and S. B. B. AlMudara, "A proposal to detect the double submission of a manuscript sent for review," *Science and Engineering Ethics*, vol. 24, pp. 1315–1329, 2018.
- [11] S. S. Khadihar, "The plague of plagiarism: Prevention and cure!!!" 2018.
- [12] J. Wei, "The countermeasures and thinking on redundant publication and duplicate submission," *Journal of Liaoning Normal University (Natural Science Edition)*, no. 3, p. 26, 2012.
- [13] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized business review*, 2008.
- [14] R. Wottenhofer, *Blockchain Science*, 3rd ed. Inverted Forest Publishing, 2019.
- [15] T. D. Washington, *Blockchain Technology*. Content Arcade Publishing, 2019.
- [16] "How does bitcoin work?" [Online]. Available: <https://bitcoin.org/en/how-it-works>
- [17] Quorum, "Enterprise ethereum client." [Online]. Available: <http://docs.guorum.com/en/latest/>
- [18] K. Cruz, "Blocksign: Signing documents on the blockchain," 2014. [Online]. Available: <https://bitcoinmagazine.com/articles/blocksign-signing-documents-on-the-blockchain-1416508388>
- [19] S. Kansal, "Develop a blockchain application from scratch in python," Jan 2020. [Online]. Available: <https://developer.ibm.com/technologies/blockchain/tutorials/develop-a-blockchain-application-from-scratch-in-python/>
- [20] "7.4. difflib - helpers for computing deltas." [Online]. Available: <https://docs.python.org/2/library/difflib.html>
- [21] M. Iansiti and K. R. Lakhani, "The truth about blockchain," Aug 2019. [Online]. Available: <https://hbr.org/2017/01/the-truth-about-blockchain>