UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

A web API service for calculating credit attributable to authors

Author:
Marc Burgess

Student Number:
BRGMAR019

8th October 2024

Minor Dissertation presented in partial fulfilment
of the requirements for the degree of Master of Science
in the Department of Statistical Sciences, University of Cape Town

Supervisor: Assoc. Prof. Co-Pierre Georg, EDHEC Business School

Declaration

I know the meaning of plagiarism and declare that all of the work in the dissertation, save for that which is properly acknowledged, is my own.

Signed by candidate

Signed: Marc Burgess – Date: 8th October 2024

Abstract

The academic project "Unicoin" is designed to use the Ethereum blockchain to provide researchers with a way to licence their work. This provides a relatively economically efficient way to receive compensation for novel ideas, but is limited to research that is commercialisable.

Foundational research is less commercialisable and is an environment where funding can be an issue. By considering the indirect impact of those providing the foundations upon which new research is built, it becomes possibly to more fairly allocate income and incentivise ongoing research. This research focusses on providing a means to allocate credit to the authors of works cited by research on the Unicoin platform by means of citation network analysis and the calculation of centrality measures.

In particular, I present a web service that is able to generate citation networks for a given piece of research and from them to efficiently calculate measures of the contribution of other papers to the research. I demonstrate this functionality with case studies. This system can be integrated into Unicoin in order to provide a fair allocation of income to all contributors.

Acknowledgements

I am extremely grateful to my supervisor, Assoc. Prof. Co-Pierre Georg, for his guidance, feedback, and patience over the course of this project, as well as to my employers at RGA Reinsurance for funding my studies and providing me with the time needed to complete them.

I am also grateful for the endless support of my parents and for them always giving me a place to spend some time and double down away from home.

I also thank my friends and colleagues for their feedback, advice and moral support along this journey.

Contents

Contents **iv**

List of Figures **vi**

1 Introduction **1**

2 Foundational research background **4**
2.1 The beginnings of research 4
2.2 Foundational research spending 6
2.3 Challenges in funding foundational research 6

3 The Unicoin solution **9**
3.1 Blockchains and smart contracts 9
3.2 Licensing on the blockchain 11
3.3 The existing Unicoin application 13
3.4 Unicoin user journey 16

4 A production system for allocating credit **18**
4.1 High level overview 18
4.2 Data 19
4.3 Technology stack 20
4.4 Allocating credit with centrality measures 23
4.5 Benchmarks and results 27
4.6 Effectively extending Unicoin 29

5 Conclusion and future directions **34**
5.1 Conclusion 34
5.2 Future work 35

Bibliography **37**

A API details **42**
A.1 Routes 42

A.2 Example HTTP session 43

B Comparison of graph-tool and NetworkX **48**

List of Figures

3.1 Unicoin licence lifecycle 12
3.2 Unicoin structure 14

4.1 Credit service stack 21
4.2 Example citation graph 24
4.3 Citation network visualisations 30

Chapter 1

Introduction

Foundational research, also known as basic or fundamental research, lays the groundwork for commercialisable scientific advances but scientists working in the field can struggle to obtain sufficient funding and attention relative to applied sciences. In this thesis, I propose a means for mitigating this by allowing authors to be allocated a share of income attributed to derived research. This consists of a software system that uses citation analysis to calculate credit attributable to published research; designed to be plugged into Unicoin, a project providing a decentralised marketplace for commercialisable ideas so that revenue for those ideas may be shared with the antecedent research they relied on.

Foundational research is often research carried out in an academic sense, without necessarily the expectation of immediate application (National Science Foundation, 1953, §6). In a sense, it can be described as the pursuit of knowledge for its own sake, for the benefit of society in general; and the results are expected to be published and shared widely. This particularly contrasts with applied and proprietary research which has the aim of solving an immediate problem.

Applied research that may yield immediate practical benefits often receives significant funding and recognition, but foundational research, which seeks to uncover knowledge without the promise of an immediate application, often struggles to obtain adequate resources and support (Petit, 2004). Finding additional means to stimulate foundational research is crucial to foster innovation, drive scientific progress, and ultimately improve human well-being.

Unicoin (Maree et al., 2019) is a "marketplace for ideas" that originated as a class academic project at UCT. It is designed as a distributed application with a web front end that provides the user interface and a back end built on the "Web3" technologies of blockchains and distributed storage. Unicoin offers a platform for academics to upload and distribute their research and optionally sell licences for commercialisation. Smart contracts on the Ethereum blockchain are used to handle payments and to implement the licences in the form of non-fungible tokens (NFTs).

The original Unicoin application was further developed in two theses. Maree (2020) in his thesis extended the original application with the goal of making the marketplace more economically efficient by using sealed bid auctions and offering the option of a Harberger tax[1]. Meiklejohn (2020) considered approaches for appropriately crediting cited works, and this thesis builds on that theme.

As much science is achieved by "standing on the shoulders of giants", it is also desirable that indirect contributions are recognised and receive fair compensation. This is particularly relevant to foundational research which may not provide directly commercialisable results of the sort that may be licenced through patents. Meiklejohn (2020) considered theoretical approaches to do this using citation analysis and centrality measures, but the current Unicoin application only provides a basic ability to assign credit to coauthors by manually entering shares.

In this thesis, my goal is to extend that work by providing a practical implementation of a system to automatically allocate a theoretically justified degree of credit to preceding work using the citation analysis approach. This system is designed in such a way that it may be easily integrated with the Unicoin application through an Application Programming Interface (API).

In the second chapter of this thesis, I provide a background on foundational research, with a focus on the challenges that researchers focussing outside of the applied sciences face. This is followed by a chapter describing the existing Unicoin application, its design, and current functionality, as well as the changes made to it outside the main focus of this work.

Chapter 4 covers the core work of this thesis by describing an application server built to provide an API with which Unicoin may interact in order to obtain measures of credit to attributed to cited works. The system obtains citation data from Elsevier's Scopus database and uses it to build a citation graph in a dedicated graph database. I describe how centrality measures can be calculated for each publication in the graph, and how the system can use the relative centralities of individual publications to calculate a measure of their influence and indirect contribution to other work.

The core of this system is written in Python, and I use existing open source libraries to communicate with Scopus, perform graph computations, and as a web service framework. I use additional software for the graph database and in order to allow multitasking capabilities so that multiple queries can run in parallel. Although this tool is built in the context of the Unicoin project, it is not inherently tied to it

[1]A Harberger tax is a tax levied on a self-declared valuation of an asset that is designed to both raise revenue and ensure efficient allocation of assets. Owners are obligated to accept any offer from another party to buy the asset at the self-declared price. Balancing the tax burden and the value of continuing to hold the asset encourages the owner to set a fair price (Posner & Weyl, 2018, ch. 1)

and could be used to allocate credit in other contexts, such as grant schemes.

In order to evaluate the approach, I provide demonstrations on three test cases.

All code written for this thesis is available on Github and the appendix provides documentation of the API and provides examples of its usage.

Chapter 2

Foundational research background

Foundational research, also known as basic research or fundamental research, is often research carried out in an academic sense, without necessarily the expectation of immediate application (National Science Foundation, 1953, §6). In a sense, it can be described as the pursuit of knowledge for its own sake, for the benefit of society in general; and the results are expected to be published and shared widely. This particularly contrasts with applied and proprietary research which has the aim of solving an immediate problem.

Lindberg (2007) emphasises the difference between technology and theoretical science: "it is one thing to know *how* to do things, another to know *why* they behave as they do". He provides the example of being able to engage in sophisticated carpentry without any understanding of the theoretical basis of stresses in timber. Knowledge of origins in the time of oral tradition was often not a case of cause and effect but rather a series of "decisive, isolated events by which the present order came into existence", in the form of creation myths.

2.1 The beginnings of research

Human innovation began with the development of technologies for obtaining the necessities of life and evolved to the modern paradigm of formal research in the time of the renaissance.

Lindberg (2007) argues that the invention of writing was a critical step in the evolution of understanding by "freezing what had hitherto been fluid" and "opening knowledge claims to the possibility of inspection, comparison and criticism" by virtue of having a formal record of different claims. The development of writing systems for numbers allowed cultures such as the Babylonians to develop the first abstract mathematical techniques, and the ability to make actual records of observations allowed them to recognise patterns in the movements of celestial objects so that astronomy and astrology could emerge.

In the Western tradition, Aristotle credited Thales of Miletus (c. sixth century BCE) as the first philosopher (Mansfeld, 1985). Thereafter many philosophical schools of thought profilerated in the Hellenistic world, although few have surviving writings. These philosophers supported themselves through various means – some charged fees to teach or had rich patrons, whereas at the other end of the scale the Cynics made a virtue of begging for food to survive. The Roman Empire proceeded to conquer Greece, but in the words of Horace "captive Greece captured, in turn, her uncivilised conquerers, and brought the arts to rustic Latium". (Horace, 14 B.C.E./2005, *Epistles II*, Epistle 1, Line 156). Roman society was eager to provide patronage to Greek scholars and teachers(Lindberg, 2007).

Many classical works owe their survival to the scholars of the Islamic Golden Age (ca. 8th–13th centuries) (Lindberg, 2007). The arrival of papermaking technology from China allowed the copying and dissemination of works to be significantly cheaper, and a common Arabic language allowed ideas to easily spread through the Islamic world. Scholars were again able to be well supported by patrons, and now also by writing and selling books.

In the classical period, the letter had become an established literary format, and it retained similar styles and formulas into the Middle Ages. Some stylistic manuals of the time recommended structures reminescent of modern scientific writing (Kronick, 2001). In addition to facilitating collaboration, letters provided an early means of establishing priority and were less costly than publishing pamphlets and books. This network of academic correspondents is sometimes referred to as an "invisible college".

By the mid-seventeenth century, these networks were evolving into learned societies and were proliferating in Europe. The Royal Society was one of them, and the first pure scientific journal was *Philosophical Transactions of the Royal Society*[1,2], originating as a private endeavour of the Society's secretary Henry Oldenburg in 1665 before later becoming an official publication (Kronick, 2001).

Until the twentieth century, research funding continued to be scarce and depended on private wealth, either on the part of the researcher themself or wealthy benefactors ("A Brief History of Research Funding in the United States", 2017). In the United States, major universities obtained funding through endowments of industrialists. Although patents had already been introduced by the fifteenth century (Harhoff et al., 2007), before the twentieth century patenting of academic research was not widespread, and indeed some argued that patents were harmful to the spirit of the discipline (Carter-Johnson, 2020). It had, however, begun to gain acceptance by the 1930s, and today is widespread.

[1]Originally *Philosophical Transactions, Giving some Account of the present Undertakings, Studies, and Labours of the Ingenious in many considerable parts of the World.*

[2]The French *Journal des sçavans* is another contender, but considered less focused on science.

2.2 Foundational research spending

Gross domestic expenditure on research and development (GERD) is highly dependent on the size of individual countries' economies, but as a proportion of GDP the largest spenders in 2020 were Israel (5.44%), South Korea (4.81%), Sweden (3.53%), Belgium (3.48%) and the United States (3.45%) (UNESCO Institute for Statistics (UIS), 2022). In absolute terms, the United States is the largest spender at 721 billion USD (2020 PPP), followed closely by China at 583 billion USD. The ten largest spenders made up 85% of 2.352 trillion USD global[3] research spend in 2020 (OECD, 2022). However, basic research spend is only a small share of the total research spend. For all the five countries listed above, basic research expenditure is less than 15% of the total GERD (OECD, 2022).

Binda et al. (2021) place the total South African research expenditure in the 2019/20 reference period at R34.5 billion. This is a decrease from R38.7 billion in 2017/18 and equates to a decrease from 0.76% to 0.62% of GDP.

The survey also shows that the majority of South African research funding came from government sources (including university own funds). Government sources contributed R18.1 billion in 2017/18, R17.5 billion in 2018/19, and R19.4 billion in 2019/20. In 2017/18 funding by business nearly matched government at R16.1 billion, but by 2019/20 had reduced to R9.4 billion – less than 50% of the government amount.

The split of expenditures shows a similar pattern. Business R&D expenditure closely matches business funding amounts, so there has been a corresponding decrease from R15.9 billion to R10.7 billion. The majority of the remaining spending is done by the higher education sector (R13.0 billion in 2017/18; R14.2 billion in 2019/20), followed by science councils at roughly R6 billion.

In the South African context, from Binda et al. (2021) approximately 30% of research expenditure is allocated to basic research. This showed an increasing trend over the three financial years of the survey but more-so due to the decrease in total research spending than to increases in absolute basic research spend Spending is largely in the government sector (universities) – business is shown to only have put 6.5% of its expenditure towards basic research in 2019/20.

2.3 Challenges in funding foundational research

Although the importance of research for economic and social progress is widely understood, competition for resources means that foundational research, which may not have a direct link to the immediate needs of society, "is often the subject of

[3]Data consists of OECD countries plus Argentina, China, Romania, Russia, Singapore, South Africa and Taiwan

questioning as to its specific role" (Petit, 2004). In particular, decision makers and the general public without a scientific background may struggle to appreciate the importance. The long-term horizon of these research projects and lack of immediate practical applications can make it difficult for funders to see the value of the research, and as such a significant dialogue is required to improve this understanding. Politicians are also subject to swings in public opinion, which causes some disciplines to receive far more interest than others.

Cozzens (1997) describes difficulties in evaluating foundational research in the context of the US Government Performance and Results Act (GPRA), the requirements of which "closely resemble those of similar legislation in other countries". The purpose of GPRA is to "improve the efficiency and effectiveness of government programs by establishing a system to set goals for program performance and measure results", which it does by trying to measure short-term *outputs* and long-term *outcomes*. Cozzens says that although it is possible to measure the output of funding agencies in terms of activity – numbers of publications, proposals, reviews, and success rates – the *outcomes* are much less straightforward because outputs become part of a "pool of knowledge" that is made up of the work of many agencies, individuals, and private organisations from which results emerge irregularly and unpredictably. To track contributions through the system "they will end up spending more money tracking than they spent to support the research".

Where research is funded by industry, commercial interests can have a major impact on the directions research can take (Fabbri et al., 2018). It may also lead to bias in the design of the investigation and the reporting of results to obtain findings that are favourable to the sponsor. On the other hand, large industrial conglomerates with interests in many fields may actively encourage unfettered research due to the many realms in which discoveries may benefit them (Nelson, 1959).

The COVID-19 pandemic provides another example of how research funding priorities can rapidly change direction. The pandemic led to decreased income and in some cases led to the repurposing of existing grants to support COVID-19 research (Sohrabi et al., 2021). The impact was not felt evenly with fields that required on-site research more significantly affected than theoretical and computational work that could be performed remotely. Walker et al. (2022) indicates that 40% of researchers had their confidence in applying for grants undermined by the effects of the pandemic. Herman et al. (2021) found too that the effect was particularly acute for early career researchers, who are in the most financially precarious positions.

Finally, the application process for funding is fraught with difficulties. Researchers spend significant amounts of time writing grant proposals, and significant experience may be required to produce a successful application. Researchers may also find it difficult to identify all the potential sources of funding available to them. According

to von Hippel and von Hippel (2015), over 170 person hours are spent on the average proposal, yet the funding rates from major US government agencies are only slightly above 20%. They find that the effective funding rate for the 80% of proposals that are new or otherwise presently unfunded research is only 12%, because half of the available funds go to projects that were funded in the previous cycle. Their estimate is that just a small decrease in the funding rate below 20% might lead half of grant applicants to abandon federally funded research.

With foundational research in such a tenuous position despite the positive externalities it provides (Nelson, 1959), there is a strong need to incentivise work in these fields. The standard approach to incentivise research and invention is the patent. Although patents lead to the usual deadweight loss of a monopoly, they have the advantages over a system such as prizes of not needing to predict the benefits of particular research in advance (Harhoff et al., 2007). Patents, however, cannot generally be applied to foundational research as by most definitions patents protect inventions, not ideas.

In the rest of this thesis, I consider an approach for passing some of the benefit of patents to basic researchers who may have contributed key ideas to the later applied research. This offers a means to pass on some of the incentive of patents to the underlying foundational research.

Chapter 3

The Unicoin solution

Unicoin (Maree et al., 2019) is a web application originally developed as a class project for the University of Cape Town MPhil Financial Technology degree, with the objective of "[unlocking] the value of academic research papers [and enforcing the] data ownership of researchers to interested third parties".

Unicoin provides a platform for researchers to make their work publicly available, but also provides a marketplace for the work to be licenced for commercial usage.

Further work by Maree (2020) extends the original platform to allow for a decentralised sealed-bid auction mechanism and enables a Harberger tax to be implemented.

The Unicoin platform is built on a standard set of technologies. The core licencing and revenue distribution functionality consists of *smart contracts* on the Ethereum blockchain, written in Solidity.

3.1 Blockchains and smart contracts

A blockchain is a public distributed ledger that uses cryptographic techniques to build a secure and verifiable history of transactions. In this context, a smart contract is a computer program that is automatically run in association with a transaction on the blockchain. Together these provide a means to verify ownership of digital assets and to distribute funds in a decentralised way.

The concept of building a computer system that mutually suspicious actors can all trust was previously described in Chaum (1982). This system relied on the existence of secure physical "vaults".

Later work by Bayer et al. (1993) and Haber and Stornetta (1991) provides a means of recording an order of transactions that is secure in the sense that it is very difficult for an attacker to backdate a transaction. Their proposed approach provides an early example of a blockchain, albeit one that is generated by a central authority (but still publicly verifiable).

Blockchains as they are currently understood originated with the white paper by Nakamoto (2008) that introduces BitCoin and its associated protocol. Although there had been previous attempts to create a digital currency, they required a central authority to ensure that double spending did not occur. Nakamoto provided the first successful decentralised solution to this double-spending problem.

The BitCoin innovation depended on the concept of proof-of-work as proposed by Dwork and Naor (1993). A major implementation of this would be HashCash (Back, 2002). Proof-of-work here was designed as a way to reduce email spam by increasing the marginal cost of sending an email.

Proof-of-work relies on requiring participants to perform a computationally hard task in order to make it less feasible for malicious actors to abuse or influence a system. In HashCash and later BitCoin, the approach is to take some representation of the data that is to be secured, append a pseudorandom value to it (a "nonce") and then apply a secure one-way function to the combination. If the binary representation of the output starts with a given number of zeros (equivalent to an output value less than some integer), then the result is accepted. If not, another nonce must be chosen and the process repeated. This "brute-force" process become exponentially more difficult in the number of required zeros, and generates real costs in terms of computer power and electricity usage. As any attacker will have limited resources, this cost limits their ability to maliciously affect the system.

Although the original focus of BitCoin was as a currency, the first releases already included scripting functionality to allow for complex transactions; however, by design, this functionality is limited and not Turing Complete. Buterin (2014) later proposed Ethereum as an alternative that included support for much more powerful smart contracts through the Ethereum Virtual Machine (EVM). Ethereum is described as a "distributed state machine" rather than a distributed ledger, and instead of transactions a concept of "messages" is used. A message can represent a transfer of funds, but sending a message to the address associated with a smart contract calls that code.

As each new block is created and verified, every smart contract call that has been made is executed in order, and the results update the state to be stored in the block.

In 2022 Ethereum moved from proof-of-work to proof-of-stake (Kapengut & Mizrach, 2022). The goals of this change were to increase security and accessibility and to greatly reduce the amount of energy required by the network. In proof-of-stake the cost comes in the form of requiring verifiers to deposit a minimum number of Ethereum tokens (32, equivalent to approximately 57 000 USD in March 2023) into a special contract. This stake may be subject to being destroyed if a validator is identified to have behaved dishonestly. In order to mount an attack, an adversary would need to control 51% of staked ETH. As of March 2023 this would amount

to around 8.1 million ETH, or 14 billion USD – an amount extremely difficult to achieve and subject to enormous risk if the stake is destroyed.

3.2 Licensing on the blockchain

A key aim of the Unicoin platform is to provide a means for researchers to maintain ownership of their valuable intellectual property while providing a means for monetization. Blockchain technologies provide a secure way to record ownership and priority, and provide a natural way for compensation to be provided.

As described in Chapter 2, the standard method of publication continues to involve transferring copyright for the work to the publisher, despite the continuing growth of open-access models. Rights to reproduce the publication are, however, not the same as rights to make use of the research results.

Unicoin allows researchers to issue licences to make commercial use of their work. These licences are represented by non-fungible tokens (NFTs) on the Ethereum blockchain. Metadata associated with these tokens can store information such as purchase price, the licence owner, and whether the licence is still valid or has been revoked. Figure 3.1 describes the life cycle of such a licence as per Maree (2020).

Licence fees are paid on the blockchain. Using smart contracts allows this process to be automated and transparent, and the initial sale may optionally take the form of a sealed bid auction mediated by smart contracts. At the time that the licence sale is completed, the funds are deducted from the purchaser's account and immediately distributed to the accounts of the individuals registered as contributors on Unicoin, in proportion to their contribution. For the purposes of this thesis, this includes distribution to the authors of cited works, and even if those authors have not registered on the Unicoin platform, it is possible to hold the distributions in a form of escrow until they do. The Harberger tax, if applicable, is also automatically deducted and distributed on a regular basis, and the associated buyout process is automatically managed.

By using the blockchain, the licencing is completely transparent. Purchasers can identify whether any other licences have been issued that would affect the value of their investment, and all transactions can be verified. The Harberger tax, if applied, provides an ongoing passive income to researchers which may bring significant extra value to their work. Another benefit of using this decentralised approach to the transfer is that it reduces the amount of trust that participants need to place in the Unicoin platform as an intermediary. Even if Unicoin ceased to exist, regular payments could continue to be made and licences could still be transferred between owners.

A downside is that it becomes difficult to implement contingent payments such

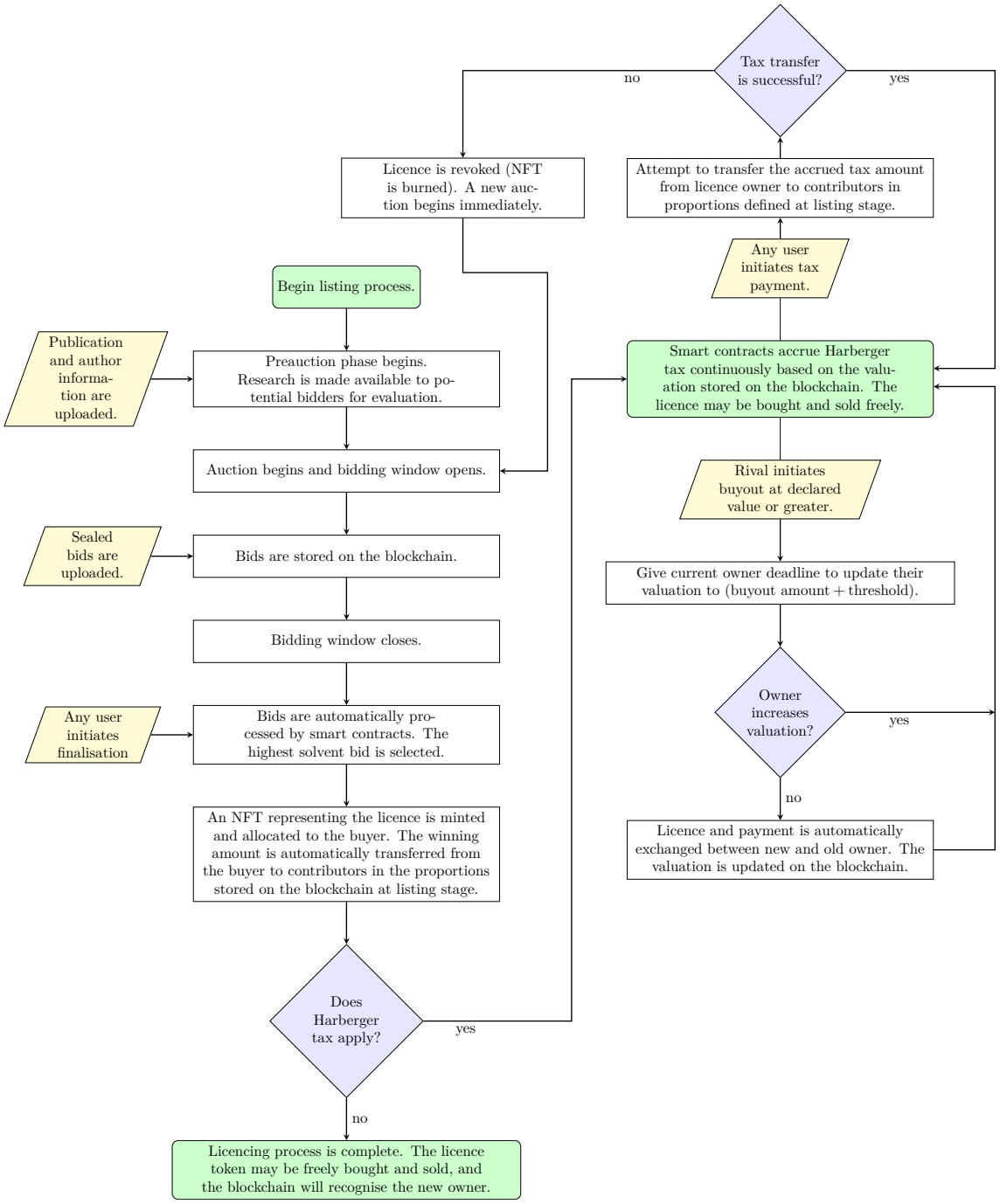


Figure 3.1: Overview of the lifecycle of a Unicoin licence sold at auction on the Ethereum blockchain. The licence is an NFT awarded based on a sealed-bid auction. A Harberger tax offers the potential for researchers to earn an ongoing income based on a fair valuation of the licence, encouraged by the potential for a forced buyout. All of the logic is encapsulated in smart contracts.

as royalties because smart contracts require specifying the amounts to be paid at the time the licence is minted. The token also adds limited value if individual terms and conditions need to be negotiated for the licence. At that point, a normal licence contract may provide more certainty and confidence given the novelty of NFTs.

3.3 The existing Unicoin application

Several technologies are brought together to provide the functionality described above. A user interface (UI) needs to be provided so that buyers and sellers can interact with smart contracts, and information, including user information and publication details, needs some form of storage.

Figure 3.2 shows the design of the Unicoin application built by Maree et al. (2019) and expanded by Maree (2020).

At a high level the components are:

1. A front-end web application. This is a single-page application (SPA) served using a Node.js web server.

2. A blockchain application consisting of the set of smart contracts that implement the ownership and payment system design.

3. Decentralised file storage implemented on the InterPlanetary File System (IPFS).

4. Third-party services for authentication and interaction with the Ethereum blockchain.

Front-end application

The front-end serves as the user interface for Unicoin. It provides the landing page and navigation to reach different functionality, and where that functionality cannot be done within the web browser, it makes requests to other services to do that work.

In order to build the front-end, the developers used version 2 of the Vue[1] framework with the Vue Material[2] component library. These use a combination of Javascript and HTML to define the interface in the form of a SPA. Vuex[3] is used to provide state management.

The front-end is built to be hosted using the Node.js[4] webserver software. Rather than a directly managed server instance, Netlify is used to provide serverless hosting

[1]https://v2.vuejs.org/
[2]https://www.creative-tim.com/vuematerial/
[3]https://vuex.vuejs.org
[4]https://nodejs.org

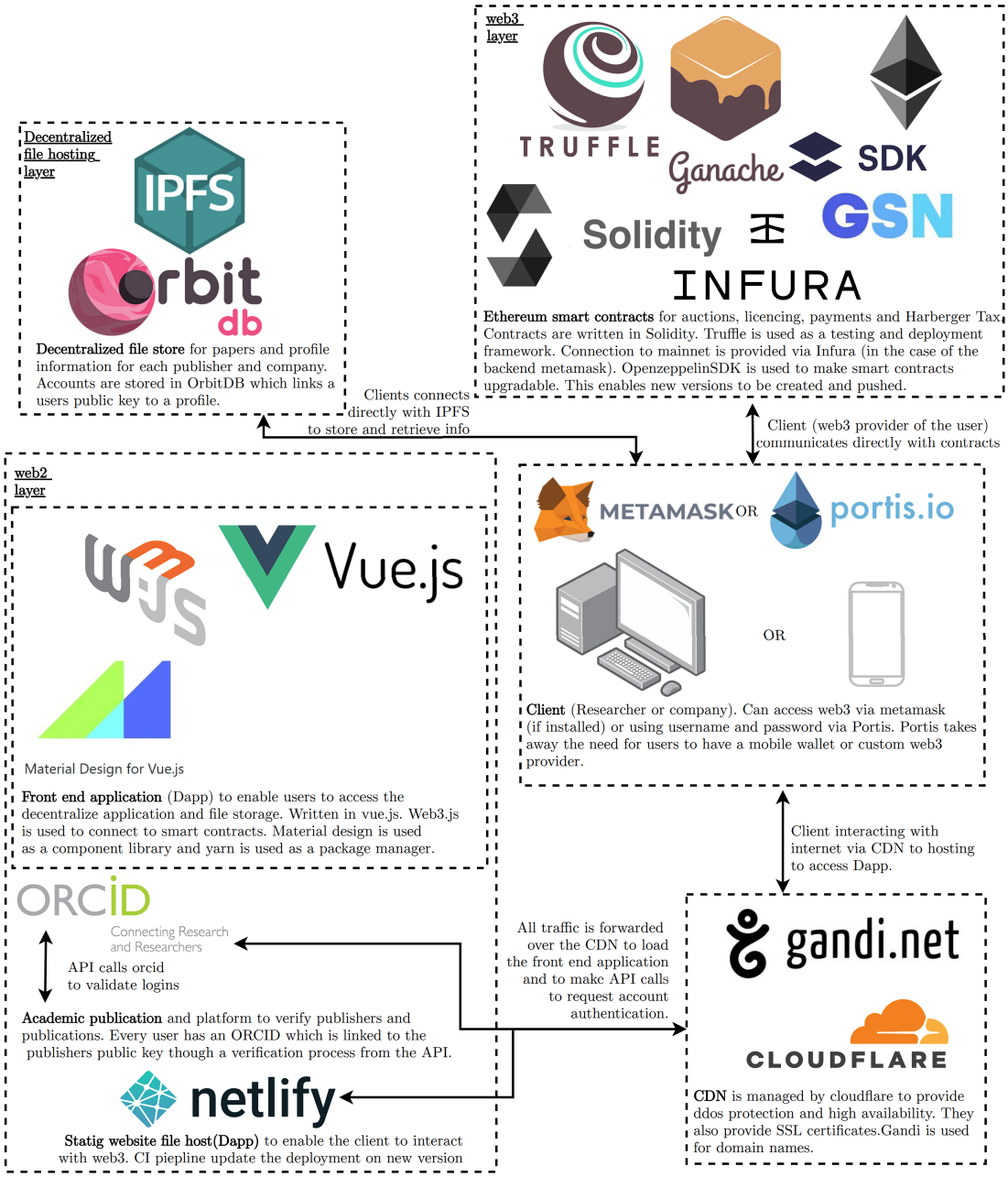


Figure 3.2: UniCoin structure, reproduced from Maree (2020). This figure shows the different technologies making up the Unicoin application, including the client, web-host, storage and blockchain interface.

with continuous integration - the developer needs to only push updated code to Github, and any updates can be automatically built and deployed.

Cloudflare is used as a content delivery network (CDN). In a production setting, a CDN helps to provide greater capacity and performance by transparently mirroring the same content at multiple locations.

Blockchain application

The business logic of Unicoin is implemented in the form of smart contracts on the Ethereum blockchain.

Different tokens are used to provide individual aspects of the functionality – ERC20 tokens provide the ability to transact using the Dai stablecoin in order to keep transaction fees lower than native ETH. ERC721 tokens are the non-fungible tokens used to represent the licences.

Unicoin uses the Ethereum Gas Station Network (GSN) in order to avoid users having to obtain ETH themselves to pay the gas for transactions. By using GSN, transactions are wrapped in *meta-transactions*, ultimately allowing the Unicoin registry to pay the fees on behalf of the users. The Dai stablecoin is also used instead of the native ETH cryptocurrency to keep transaction costs low.

The contracts themselves take the form of a *hub and spoke pattern*. Central contracts are used to manage different aspects of the application and maintain a separation of concerns. There are separate contracts to manage:

- User registration
- Publication details
- Licence issuance
- Payments
- Auctions
- Harberger tax.

A final contract acts as the hub to which all requests are sent and routed to the appropriate implementing contract.

The front-end expects to communicate with the smart contracts primarily through the use of the Web3 provider of the MetaMask smart wallet, but there are fallbacks to use the APIs provided by Infura.

Decentralised file storage

The Unicoin application needs to store user profiles and publication data. This is done using IPFS.

IPFS is a distributed file storage system that relies on a series of peers who volunteer to store data. When content is uploaded, it is given a unique content identifier (CID) which is based on a cryptographic hash of the data itself.

The content stored on IPFS can generally not be identified without the CID and cannot be deleted or modified. In order to update a document, a new version must be uploaded with a new CID. Data are cached on individual peers as they are retrieved, but this does have a limited lifespan as caches are regularly cleared so that data that have not been accessed by anyone for some time may become lost. Individual nodes can choose to "pin" specific content in order to retain it indefinitely.

Unicoin data for individual documents and users are serialised to JSON format for IPFS upload. For Unicoin's purpose, it is desirable for retention to be permanent.

Third party services

Unicoin uses third-party services in order to authenticate users and interact with IPFS.

For user authentication, Open Researcher and Contributor ID (ORCID) is used. ORCID is a service that allows researchers to obtain a unique identifier so that their contributions are recognised in the face of non-unique personal names, name changes, and differing cultural naming conventions. ORCID provides an authentication API, and through this Unicoin is able to associate a new Unicoin user with an ORCID which allows Unicoin to not have to implement its own authentication process.

Unicoin uses Infura to interact with IPFS. Generally, uploading and retrieving data from IPFS requires the user to have their own IPFS node. Infura provides a gateway API that can be used instead, which removes the need for Unicoin to manage its own node. Infura also offers a gateway to interact with the Ethereum blockchain, which can be used as a fallback when the Unicoin user does not have the MetaMask wallet installed.

3.4 Unicoin user journey

The existing Unicoin interface as of Maree (2020) provides three core functionalities – user registration, publication listing, and licence purchasing.

The first step for any user is to register. Users may register either as academics planning to list their publications or as companies intending to buy licences to listed work. Academic researchers are expected to link their profile to ORCID in order to

provide a form of verification. Their user details will be obtained from their ORCID profile. For both academics and companies their profile is then linked to a wallet in order to interact with the Blockchain. The primary interface for this is MetaMask.

After registering, an academic can choose to list a publication on Unicoin. In order to do this a copy of the publication must be uploaded and key information provided including title, abstract, and keywords. The next step is to specify contributors. In the existing Unicoin application, each contributor and the share of revenue they are to receive is manually input. The publication may then be listed either at a fixed price or using a sealed-bid auction. In the latter case, the Harberger tax may be turned on. At the end of the listing process, the transaction must be mined on the Ethereum blockchain, again using the MetaMask interface.

Licences may be purchased on the publication marketplace. The marketplace lists all publications that are available to purchase a licence. All publications uploaded to Unicoin are available for non-commercial usage under a Creative Commons licence and may be downloaded here. Voluntary donations can also be made. Where an auction is in progress, bids may be made. After the bidding window of an auction has closed, any user may choose to finalise the auction, in which case the winning bidder is identified and an NFT licence is automatically issued to the wallet associated with their account. Payment to all contributors also happens automatically at this point.

A page is available to the academic who originally listed the publication where they are able to view details about the winning bid and to claim any Harberger tax due to them.

Chapter 4

A production system for allocating credit

I have covered the reasons that a new approach is needed to incentivise foundational research, and described the Unicoin system that attempts to solve some of these problems.

An extension of this is to allocate credit to indirect contributors – authors of publications that have contributed to another's work. This would potentially incentivise research with the promise of long-term rewards from the future impact of the work on others. Meiklejohn (2020) introduced the theoretical approach of applying centrality measures for attributing credit on the Unicoin platform, and in this chapter I cover the details of those measures and provide a practical implementation of a system that can be "plugged into" to the Unicoin front-end system to allow it to take advantage of them.

4.1 High level overview

This thesis contributes to the Unicoin project by providing a means to distribute credit to the authors of publications that may have made a significant contribution to another's work. This credit allocation would be equivalent to including these indirect contributors in the payments made when the licence is issued and when Harberger taxes are paid, as described in Section 3.2.

In order to calculate this credit allocation, I obtain 3rd-party citation data and store and process it in order to build a citation graph. Citation data is obtained by making automated requests to a proprietary citation and publication metadata store via a web API. I use a graph database engine to persist the data that has been returned to disk and to model the citation and authorship relationships within the results in order to form an explicit citation network that can be interrogated.

As a second step, I use dedicated network analysis library to build calculate centrality measures for (the subset of) the graph under consideration. The graph database is queried to obtain separate lists of nodes and edges that are converted into an in-memory graph. The network analysis library uses this in-memory structure in order to efficiently calculate centrality measures describing the relative influence of individual documents in the citation graph. By considering the relative centralities of a publication and the publications that it cites, I can calculate a measure of attributable credit.

The above functionality combined into a single application designed to service the greater Unicoin system. I provide a programmatic interface by which the rest of the system can communicate with this service in order to make requests for calculations and to retrieve results. This includes a multitasking system so that multiple requests may be handled at the same time.

4.2 Data

For this project, I source data from Scopus, a database of metadata for academic works produced by Elsevier. I used the Python package pybliometrics (Rose & Kitchin, 2019, Version 3.4.0) to query the REST API provided by Scopus.

Scopus has content records dating at least to 1823, and has cited references dating back to 1970 (Beatty, 2017). As of 2017 Scopus claims to have indexed more than 66 million items from nearly 23 000 serial publications and 140 000 books.

This project takes the approach of walking the edges of the citation network from a selected paper. It retrieves lists of the documents that are referenced by, and cite, the selected paper and the process is repeated recursively for each of the items in the new lists. The user can configure the depth of this recursion.

In order to prevent abuse, Scopus imposes limits on access. To access the Scopus API, a key is required, and each key is limited in the amount of requests that may be made in a given time period (10 000 requests per week for the Abstract Retrieval API used for this project[1]). It is, however, possible to request multiple API keys, each with their own limit, and the pybliometrics package is able to automatically rotate through them as each becomes exhausted.

The above, combined with the limited rate at which requests can be made and responses received, places limits on the size of the graph that may be practically interrogated. For this reason, the default recursion depth is set to two levels.

Although Scopus provides a convenient method to obtain data, it is necessary to be aware of some of its limitations. First, it is proprietary and it is possible that free access may be revoked at any point. In terms of the data itself, Scopus, though

[1]See https://dev.elsevier.com/api_key_settings.html

large, does not cover all publications and has limited coverage prior to 1970 (Beatty, 2017). Also, like most data sources, it is imperfect and can contain duplicate or invalid records. Although this is not expected to have a major impact on results, it is something to be aware of.

4.3 Technology stack

I implement the credit allocation function using a back-end service to be called via a Representational State Transfer Application Programming Interface (REST API). The API provides a simple protocol that allows the client (the Unicoin application) to make a request for the server to perform the calculations and then to retrieve the results separately.

The code layer of the implementation is written in Python. I select Python as an implementation language because it provides an environment for rapid prototyping and has a wide ecosystem of packages providing functionality for a given use-case.

REST is an API architectural style originally proposed by Fielding (2000, ch. 5) for client-server communications that emphasises a stateless loose coupling of components. This involves defining a layer of abstractions such that all server resources, including actions, are reached via Hypertext Transfer Protocol (HTTP) requests to Universal Resource Identifiers (URIs) uniquely identifying each resource. A *representation* of the state of the resource may then be returned, in this case in the form of Javascript Object Notation (JSON).

The service itself consists of four distinct components:

1. Server that provides the client facing REST API. It performs some operations itself and passes others off to separate processes to be performed asynchronously.

2. Graph database that stores a set of documents that have already been processed and the edges connecting those documents.

3. Task queue which allows the server to hand over long-running tasks in order that it may continue to respond to new requests in the interim.

4. In-memory data store to be used for message-passing between the task queue and its child tasks.

The interactions between these components are shown in Figure 4.1.

REST server

The server provides the interface to the "back-end" functionality of the Unicoin application, which cannot be performed within the web application loaded within

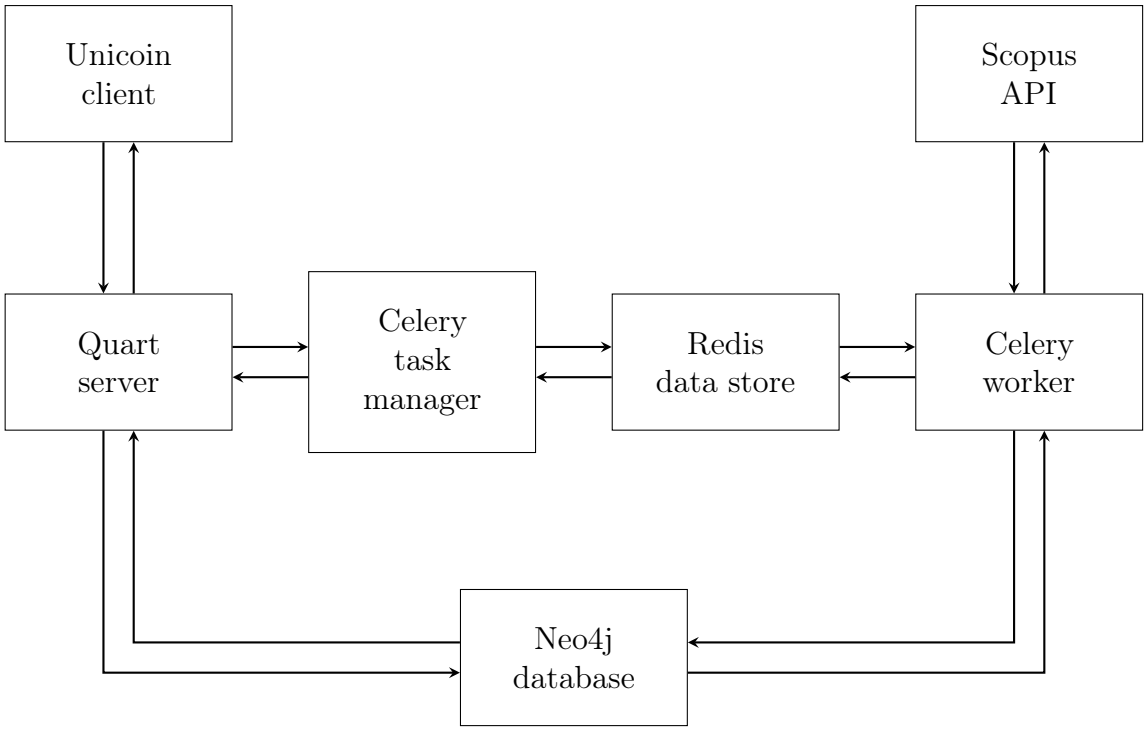


Figure 4.1: The data flow of the credit service stack. The Quart server receives the requests from the client. For simple requests, it may query the database directly. For more complex requests, it uses the Celery framework to create a separate task. Celery creates a worker process and communicates with it using Redis. The worker thread is able to retrieve data using the Scopus API and to store results in the database, which it may again retrieve at the calculation stage.

the user's browser. I use the Python Quart framework (Jones, 2022, Version 0.17.0) to provide client-facing server functionality. The authors of Quart describe it as a "web microframework" that is an evolution of another Python framework, Flask.

The server receives queries from the client and prepares the response in the correct format. For simple requests, it may query the database itself, but for resource intensive operations, the work is delegated to the task queue to be performed asynchronously. The client may send API requests to query the status and results the tasks, in which case the server in turn queries the task queue and prepares a response with the status.

Details of the API provided are in Appendix A.

Graph Database

I use the Neo4j graph database (Neo4j Inc. & Neo4j Contributors, 2022, Version 4.4.15) to store the citation network and author relationships.

Although it is possible to download citation data on-demand and perform the network centrality calculations purely in-memory, this has limitations. The process of requesting article metadata from Scopus is slow, so storing results provides a useful caching mechanism when multiple requests are made covering overlapping

parts of the citation network. Storing the results also makes the platform more extensible, so that ultimately it will be possible to include data sources other than Scopus and potentially provide a single source of truth for related projects.

I chose to use a graph database in particular as it is a natural fit to the network structure being analysed. According to Robinson et al. (2015, pp. 4–10) a graph database is "an online database management system, with Create, Read, Update and Delete (CRUD) methods that expose a graph data model" and "relationships are first-class citizens of the graph data model". This contrasts with standard relational databases, where relationships are implied by concepts such as foreign keys. Using the graph database then allows me to more closely represent the actual data. However, this technology is significantly less mature than traditional databases. There is not yet a standardised query language, although an ISO standardisation effort is currently underway (Hare et al., 2019).

My decision to use Neo4j in particular was driven by its open-source nature and because of its Python driver. It is also relatively mature and easy to set up, and it has an expressive query language known as Cypher. The down-sides that I have identified are that it is not 100% open-source (certain features are only available with an enterprise licence), and that performance is worse than alternatives on some benchmarks (e.g. Rawal, 2017; Tencent Cloud Team, 2020).

Task queue and data store

I use the Celery task queueing protocol (Solem & Celery Contributors, 2022, Version 5.2.7) for Python to allow the server to handle multiple requests simultaneously. The Redis in-memory data store serves as a message-passing system for Celery.

Python has limitations for concurrent programming because multi-threading is limited by the use of the "Global Interpreter Lock" - a global *mutex*[2] that allows only one processing thread to have control of the interpreter at a time (Beazley, 2009).

For a project of this nature, this places limitations on the ability to service multiple CPU-intensive requests at the same time. The task queue provides a means to distribute work across separate threads and ultimately different machines. This introduces significant room for scalability of the solution by allowing arbitrary parallelisation (including across multiple machines), whilst at the same time allows programming and maintenance to remain simple by retaining all of the source code within a single Python project.

Celery requires a mechanism for passing messages between the workers and the task queue itself. Celery supports a few different options, and I chose to use Redis,

[2]From *mutual exclusion*, a mutex is a system used to prevent consistency violations caused by multiple threads of execution attempting to access the same resource at the same time.

which is an in memory data store (Salvatore & Redis Contributors, 2022) that is easy to set up and very fast.

Celery does not fully support Windows operating system natively, but may be used with the Windows Subsystem for Linux (WSL). Redis is distributed as a Docker image.

Worker tasks

All time consuming and resource intensive tasks are offloaded from the API server to Celery tasks.

The most time-consuming task is to generate the citation network. This is IO-bound – it is limited by the rate at which Scopus can be queried and has relatively low impact on CPU and memory usage.

Python is an interpreted language and, as such, is much slower than a compiled language, which is relevant for CPU-bound tasks. Where Python is inefficient for heavy computations, there are a number of packages available that call code written in compiled languages such as C and C++ to mitigate this.

The heaviest computations that are performed are to calculate centrality after the citation graph has already been built, and these calculations are performed using the Python graph-tool package (Peixoto, 2014, Version 2.45) which leverages native code and data structures for efficiency. I originally used NetworkX (Hagberg et al., 2008) as done by Meiklejohn, however, its memory footprint was orders of magnitude higher, which caused failures for large networks on the test machine. (See Appendix B for a comparison).

4.4 Allocating credit with centrality measures

Publications are related to each other by the action of citation. This suggests representing these relationships in the form of a directed graph (digraph) with the publications as nodes and each citation as a relationship. In this form, it is possible to measure the importance of a given paper using the concept of centrality.

Jackson (2010) describes centrality measures as a tool for comparing nodes of a network and for enabling one to better understand how a given node relates to the network as a whole.

Figure 4.2 shows an example graph. Paper 1 cites Paper 2. Paper 2 cites Papers 3, 4, and 6.

The graph can also be represented in the form of an adjacency matrix. In the

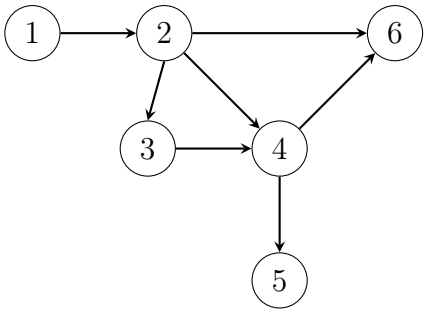


Figure 4.2: An example citation graph. Nodes represent papers, and edges represent citations, such that the arrow points from the citing paper to the cited paper. Paper 1 cites Paper 2, which in turn cites Papers 3, 4, and 6. Papers 5 and 6 are leaf nodes that do not cite other papers.

general case for such a matrix $\mathbf{G} = [g_{ij}]$, the elements g_{ij} are given by

$$g_{ij} = \begin{cases} 1 & \text{if there is a (directed) edge from node } i \text{ to node } j \\ 0 & \text{otherwise} \end{cases}$$

.

In case of an undirected graph, the matrix will be symmetric with $g_{ij} = g_{ji}$ because the edges are bidirectional.

Below is the adjacency matrix corresponding to the citation graph in Figure 4.2:

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

A simple measure of centrality is degree centrality (Jackson, 2010). The degree of a node is the number of edges connected to it, and in a digraph this may be split into in-degree and out-degree for incoming and outgoing edges, respectively. In a citation network, in-degree is most relevant for assessing importance because we are interested in the number of times the publication has been cited.

For a graph defined by the adjacency matrix $\mathbf{G}$ with n nodes, let the in-degree of node i be $d_i(G)$. Then, the in-degree centrality is given by

$$\frac{d_i(\mathbf{G})}{n-1} \tag{4.1}$$

.

This centrality measure ranges over $[0, 1]$. In a citation network, 0 would mean no citations and 1 would mean the paper is cited by every other paper.

There is a more sophisticated class of measures that allows for the transitive importance of the papers citing the current paper, as introduced by Bonacich (1972, 1987) and Katz (1953).

Katz Prestige can be defined recursively as:

$$P_i^K = \sum_{j \neq i} g_{ij} \frac{P_j^K(\mathbf{G})}{d_j(\mathbf{G})} \tag{4.2}$$

.

That is, the Katz Prestige of a node is equal to the sum of the normalised Prestiges of its neighbours, where the normalisation is by the degree of the node. An intuition for normalisation is that a node with many connections only gives a fraction of its "time" to each of its connections (Jackson, 2010). In the case of the citation network, this could mean that if a paper references many other papers, the importance of each individual reference is lower.

The type of degree used affects the results of the Katz Prestige. If $d_i(\mathbf{G})$ refers to total degree or out-degree rather than in-degree then the solution reduces to $P_i^K(\mathbf{G}) = d_i(\mathbf{G})$ - that is, Katz Prestige is equal to the degree. For a citation network, out-degree makes more intuitive sense as a normalisation factor than in-degree – the number of times a paper is cited should only increase its importance. As such, Katz Prestige adds limited additional value for this problem.

Bonacich (1972) introduced eigenvector centrality, a related measure that avoids the issue of reducing to degree centrality. It is effectively an unweighted version of Katz Prestige and satisfies

$$\lambda C_i^e(\mathbf{G}) = \sum_j g_{ij} C_j^e(\mathbf{G}) \tag{4.3}$$

where $C_i^e(\mathbf{G})$ is the eigenvector centrality of node i. In matrix form, with $\mathbf{G}$ again representing the adjacency matrix, this is

$$\lambda C^e(\mathbf{G}) = \mathbf{G} C^e(\mathbf{G}) \tag{4.4}$$

which makes it clear that $C^e(\mathbf{G})$ is an eigenvector of $\mathbf{G}$. By convention, the normalised all-positive eigenvector associated with the largest eigenvalue is used (which is known to exist by the Perron–Frobenius theorem).

Both Katz Prestige and eigenvector centrality have a potential problem in the context of a citation network. A citation network is unlikely to be *strongly connected* because we do not expect to see cycles in the citations. This implies that the adjacency matrix may be upper triangular and that it may not be possible to solve for eigenvectors.

Katz (1953) introduced a further measure known as Katz Centrality. Katz Centrality measures the number of walks that originate at a node, with shorter walks having a greater weighting than long walks.

The adjacency matrix $\mathbf{G}$ gives the number of walks of length 1 between any two nodes. There is no more than 1 walk of length 1 between two nodes, but $\mathbf{G}^k$ describes how many walks of length k exist between any pair. If $\mathbb{1}$ is an $n \times 1$ column vector of 1's then $\mathbf{G}^k\mathbb{1}$ is the total number of walks of length k that start at each node (and $\mathbf{G}\mathbb{1}$ gives the degrees of each node).

The Katz Centrality is then defined, for a weighting factor a as

$$P^{K2}(\mathbf{G}, a) = \sum_{k=0}^{n} a^k \mathbf{G}^k \mathbb{1} \tag{4.5}$$

$$= \left(\sum_{k=1}^{n} a^k \mathbf{G} \right) a\mathbf{G}\mathbb{1} \tag{4.6}$$

$$= (\mathbb{I} - a\mathbf{G})^{-1} a\mathbf{G}\mathbb{1} \tag{4.7}$$

where $\mathbb{I}$ is the $n \times n$ identity matrix. This converges for small enough a. Katz introduced a to represent the probability that a link is "effective". A possible interpretation in the case of a citation network is the probability that the citation is relevant, or a positive citation.

Bonacich (1987) introduced a modified version of the above. The Bonacich centrality may be given by

$$Ce^B(\mathbf{G}, a, b) = (\mathbb{I} - b\mathbf{G})^{-1} a\mathbf{G}\mathbb{1} \tag{4.8}$$

with b small enough to ensure convergence. In this case $a\mathbf{G}\mathbb{1}$ gives an initial value for the importance of each node, and the walks originating from it are weighted by b in the same fashion as Katz Centrality.

A final important measure is PageRank, which was introduced by Page et al. (1999) and served as the original basis for the rankings on the Google search engine. The original measure is essentially a modification of Katz Prestige:

$$R(\mathbf{G}, i) = c \sum_{j \neq i} g_{ij} \frac{R(\mathbf{G}, j)}{d_j(\mathbf{G})} + cE(\mathbf{G}, i) \tag{4.9}$$

$$= c \sum_{j \in B_i} \frac{R(\mathbf{G}, j)}{d_j(\mathbf{G})} + cE(\mathbf{G}, i) \tag{4.10}$$

where B_i is the set of nodes linking to node i, c is some normalization factor and $E(\mathbf{G}, i)$ is known as a "source of rank".

This modification solves the issue that Katz Prestige has with graphs that are not strongly connected. In the web search context, PageRank models the probability that a web surfer randomly following links will end up at node i, and the E term represents a probability of "jumping" to the node without needing to follow a link. In the context of citation analysis, it might represent the possibility of reaching an

article by virtue of opening a textbook, or having a working relationship with an author.

The centrality measures given above cannot be used directly as a measure of the credit to be allocated to different papers because the scale of the results is essentially arbitrary. In addition, older papers will have had more time to obtain citations, leading to bias in the results.

It is possible to account for the age of each paper by applying a discount factor to their centrality scores based on age. Labelling the paper of interest Paper 0, the centrality scores of the other papers in the network are multiplied by $\delta^{T_0 - T_i}$ for some δ such that $0 < \delta < 1$, and with T_0 the year of publication of Paper 0, and T_i the year of publication of Paper i. This approach follows Meiklejohn with the slight modification of using T_0 as a baseline instead of a fixed reference year.

Fang (2018) and Meiklejohn (2020) discuss approaches to calculating attributable credit based on centrality measures. Letting the set of papers that are cited by Paper 0 be generation -1 or G_{-1} (see Hu et al., 2011) with $|G_{-1}|$ elements, their average discounted centrality score is:

$$\overline{Y_{-1}} = \frac{1}{|G_{-1}|} \sum_{i \in G_{-1}} Y_i \tag{4.11}$$

and the credit attributable to Paper 0 can then be obtained as

$$\theta_0 = \frac{Y_0}{Y_0 + \overline{Y_{-1}}} \tag{4.12}$$

For this thesis I calculate the above measure for in-degree, eigenvector, Katz, and Pagerank centralities.

It should be noted that there are some criticisms of using citation analysis to assign credit and associate value with research. Cozzens (1997) points out that "negative citations" can skew results when a work has been widely cited from a critical perspective. They also note that average citation counts vary by discipline. The calculations discussed in this section effectively normalise for this within a specific discipline, but there may be distortions where different fields fall within the same network. Lastly, according to Cozzens theoretical work tends to have fewer citations than experimental results. This may cause foundational research heavy in theory to not be allocated sufficient credit.

4.5 Benchmarks and results

In order for this system to meet its design goals, desirable results are credit measures that behave as expected, and an acceptable level of performance in order to allow

the calculations to be performed frequently and without long delays for the user.

To test this, I used three papers coauthored by South African actuary Rob Dorrington. This follows the approach of Meiklejohn, which in turn was based on that of Tol (2011). The test cases were as follows:

1. *Rapid mortality surveillance using a national population register to monitor excess deaths during SARS-CoV-2 pandemic in South Africa*, a paper published in 2022 and authored by Dorrington, Moultrie, Laubscher, Groenewald and Bradshaw (DOI 10.1186/s41118-021-00134-6). This paper had 5 citations on Scopus at the time of writing, and in addition cites only a small number of documents available on Scopus .

2. *Sentinel surveillance of sexually transmitted infections in South Africa: A review* by Johnson, Coetzee, and Dorrington (DOI 10.1136/sti.2004.013904). This is an older paper published in 2005 with a moderate number of citations (94).

3. *Life Expectancies of South African Adults Starting Antiretroviral Treatment: Collaborative Analysis of Cohort Studies* by Johnson, Mossong, Dorrington and others (DOI 10.1371/journal.pmed.1001418). This is a 2013 paper with 302 citations on Scopus.

The credit attributable to each paper is shown in Table 4.1. Scores behave in the expected way: the paper with fewest citations retains the lowest proportion of credit, and the paper with the highest number retains the largest. The results using eigenvector centrality are degenerate. This is consistent with the discussion in Section 4.4 – in a directed acyclic graph, there are no nonzero solutions to the eigenvector problem. This is consistent with Meiklejohn's results.

Table 4.2 shows the amount of time required to generate the graph and calculate the credit attributable. These numbers are illustrative – in particular, the graph generation is highly dependent on network conditions and is susceptible to being interrupted due to Scopus errors. The vast majority of time is spent querying the Scopus API to obtain the structure of the citation network. Even for the newest paper with few citations, this took in excess of 30 minutes. By comparison, computing the centrality measures having already stored the graph took a fraction of a second.

The construction time might be improved with a higher-speed internet connection and by utilising more worker processes, however, Scopus will continue to provide a limit on the maximum possible performance of this approach.

For completeness, Figure 4.3 shows the details of the citation graphs that have been generated. Figure 4.3(a) shows the network of papers directly citing, and being cited by Paper 1, and their own antecedents and descendents. These may also be

	Centrality measure			
Document	In-degree	Eigenvector	Katz	Pagerank
Paper 1	0.218	0	0.363	0.107
Paper 2	0.590	0	0.687	0.439
Paper 3	0.659	0.028	0.744	0.494

Table 4.1: Credit attributable to papers 1, 2 and 3 as calculated by the system and returned by the REST API, based on each centrality measure.

Document	Documents in graph	Construction time (min)	Memory required for centrality calculation (MB)	Centrality calculation time (s)
Paper 1	473	35	0.55	0.11
Paper 2	7874	79	10.53	1.73
Paper 3	22 411	253	28.74	5.19

Table 4.2: Performance statistics for the credit-attribution process. This table shows the number of nodes in the resulting citation graph, the time required to build the graph using the Scopus API starting from a blank database, and the time and memory required to perform the centrality calculations within the Python process.

described as generations -2, -1, 0, 1 and 2 (see Hu et al., 2011). Figure 4.3(b) includes all papers citing and cited by generations -1, 1, and 1. These additional papers influence the centrality scores that are calculated despite not providing a citation path to and from the paper of interest.

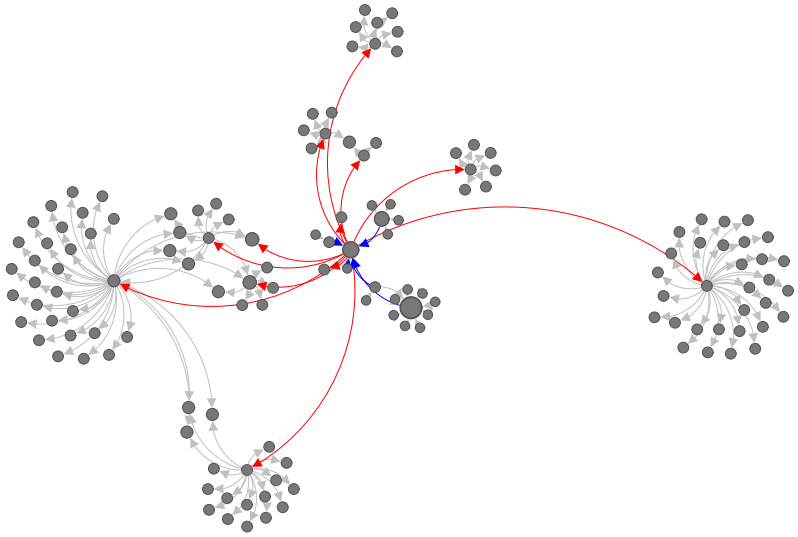
Figures 4.3(c)/(d) and 4.3(e)/(f) show the same pairs of plots for Paper 2 and Paper 3 respectively.

4.6 Effectively extending Unicoin

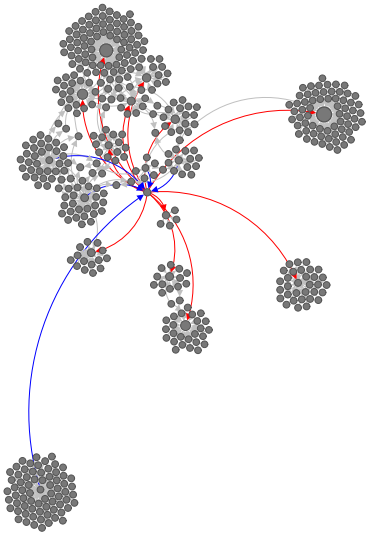
The system I present here is designed to integrate into Unicoin as described in Chapter 3.

Specifically, the initial point at which credit allocations will be made is in the publication listing stage. The existing listing process requires authors to manually specify how credit is to be shared among research collaborators. This work would not replace that step, but would require an additional step to be added where the share of credit due to the paper being listed versus its antecedent research as determined by the system in this thesis. Due to the lengthy process of building the citation tree from Scopus data, it would be necessary to perform this step asyncronously as a background task rather than requiring the user to wait to complete the listing.

The resulting centrality measures could be stored with the rest of the publication details using IPFS, and as part of the smart contract so that the share of revenue allocated to other research is distributed separately. A more complicated task will be

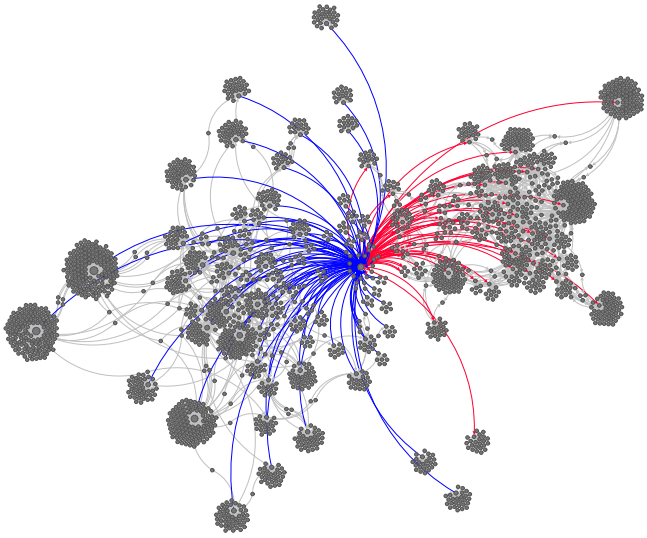


(a) Papers within two generations of Paper 1

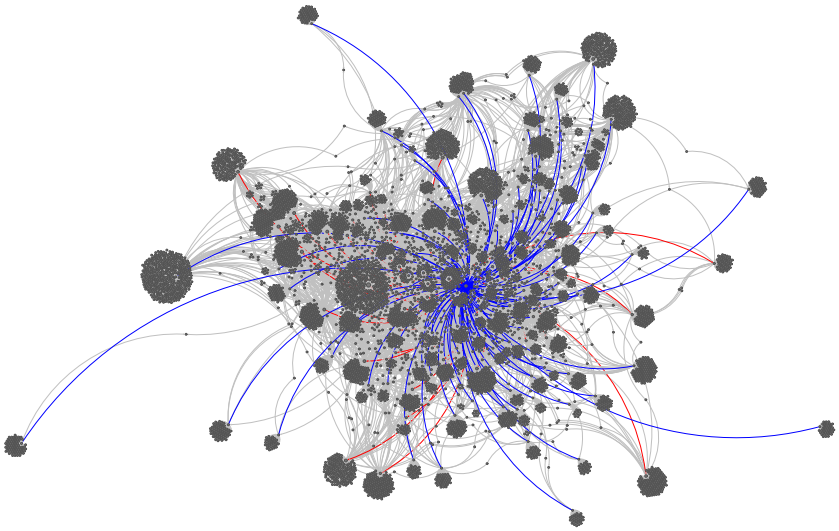


(b) All papers within citation network of Paper 1

Figure 4.3: Citation networks for papers 1, 2, and 3. Subfigures (a), (c) and (e) show all papers that can be reached from the relevant paper within two generations on a directed walk; that is, papers that directly and indirectly cite the paper of interest and papers that it directly and indirectly cites. (b), (d) and (f) show all papers within a maximum distance of 2 from the paper. The edges in blue are citations of the paper of interest, and the edges in red are toward the papers it cites.

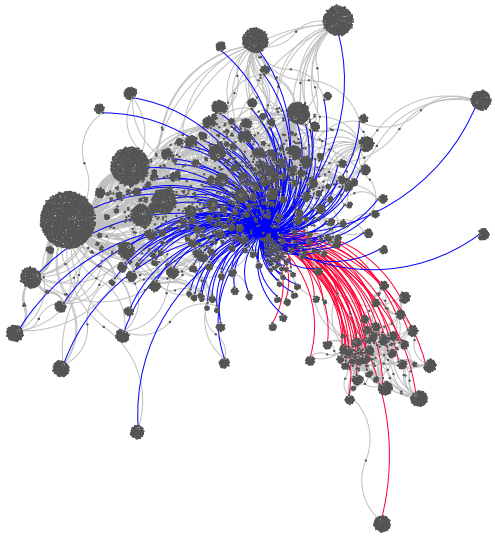


(c) Papers within two generations of Paper 2

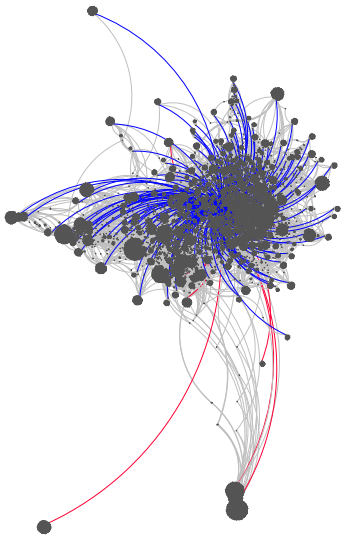


(d) All papers within citation network of Paper 2

Figure 4.3: (Continued from page 30) Citation network visualisations.



(e) Papers within two generations of Paper 3



(f) All papers within citation network of Paper 3

Figure 4.3: (Continued from page 31) Citation network visualisations.

distributing that revenue, since not all cited papers that are allocated credit will be registered on Unicoin. Ideally those distributions should also be shared with further generations of the citation graph too.

Effectively integrating this tool into the Unicoin application will mean that every time a licence is purchased or a Harberger tax payment is made, researchers who indirectly contributed to the work will receive some compensation for their effort, providing an ongoing incentive to continue to work on foundational research.

Chapter 5

Conclusion and future directions

5.1 Conclusion

I have demonstrated an automated system that provides a means to extend the Unicoin "marketplace for ideas" to distribute credit and, therefore, income to the authors of foundational research.

Foundational research, though crucial, often faces an uphill battle to justify its existence to policy makers and the general public in the face of unpredictable future benefits and a general lack of awareness. As the COVID-19 pandemic showed, it is also susceptible to disruption in the face of major unexpected events. Providing more freedom and certainty in their research direction would serve to attract and retain more researchers, and one way to do this is to provide alternative funding mechanisms.

This thesis described Unicoin, which aims to optimally commercialise ideas by allowing academics to directly licence their work and to disseminate it without the limitations of traditional academic journals and the poor efficiency of the current patents market. The work here builds on this by working towards allowing work that may not directly commercialisable to benefit by receiving a share of the income received in the market by research that it has indirectly contributed towards. It also provides a means to generate an ongoing passive income which is a long-term reward. As such, foundational research is incentivised because there is a reward even when it is not possible to licence the results and when there is little certainty over the next source of funding.

I have demonstrated that it is possible to build a system providing a general API to read citation data for an arbitrary published paper from third-party sources and compile it into a complete citation graph from which centrality measures may be calculated. This system is able to convert these centrality measures into measures of the degree to which credit for research outputs should be allocated to cited works and return that information for use in the larger Unicoin application or other contexts.

I have shown through examples that this system has good local performance characteristics; however, it is limited by the rate at which the third party service may be queried, and that it produces measures of credit-attributable that behave reasonably given the number of citations in each of the example networks. This provides a practical means by which a share of the income received on the Unicoin platform or elsewhere may be allocated towards the contributing research of others.

5.2 Future work

In this thesis, I present a system for calculating a measure of credit that may be attributed to an arbitrary work of research based on its influence within a citation network. This work opens some avenues for further research.

An immediate continuation of the work is to continue to update the Unicoin application itself to fully integrate the work here. A consideration is that the approach that I have taken here is to use a "Web2.0" approach of a central application server and database which is contrary to the decentralised approach of the original Unicoin project. This was necessary due to the impracticality of performing compute-intensive tasks directly on the EVM but future work may be able to investigate alternative solutions.

Another area of further work is to consider how best to efficiently handle the continuous evolution of the citation network. Over time existing papers will gain more citations as will the rest of the papers in its network, and as such it would be appropriate to recalculate the credit attributable over time. It would be valuable to investigate how frequently these updates should be made, balancing the frequency of network changes against the resources required to perform the updates. Integrating sources other than Scopus, including directly processing uploaded papers, would also be valuable.

There are open questions about how to allocate credit from a paper to individual contributors. Unicoin currently uses a manual approach of specifying contributors and the share of income that should be distributed to each. Further work may consider ways to allocate this more fairly and possibly include informal contributors (see e.g. Rose & Georg, 2021)

Unicoin and the system I have described here do not have substantial checks to guard against bad actors and innocent errors. The actions taken by the smart contracts may be provably correct, but they cannot verify the integrity of the original inputs. Potential problems include users selling licences to research that is not their own and researchers failing to sufficiently cite other authors to prevent sharing income with them. Simple ways to solve these might respectively be manually vetting each listing and relying on the standard research peer review process, but these may

add significant overheads and conflict with Unicoin's decentralised ethos. Additional research may be able to identify novel solutions for this, including tackling it from a behavioural economics perspective.

Finally, recent developments in the realm of AI could be an avenue to explore. Cutting edge large language models (LLMs) (Radford et al., 2019) may provide an avenue for more subtle ways to identify influences on a particular piece of research. It may also help identify informal contributors, as discussed in Rose and Georg (2021), as well as allow a better understanding of the other sources of funding that have already supported a publication, such as private and government grants.

Bibliography

Back, A. (2002, August 1). *Hashcash - a denial of service counter-measure* (White Paper). http://www.hashcash.org/papers/hashcash.pdf

Bayer, D., Haber, S., & Stornetta, W. S. (1993). Improving the Efficiency and Reliability of Digital Time-Stamping. In R. Capocelli, A. De Santis & U. Vaccaro (Eds.), *Sequences II* (pp. 329–334). Springer New York.

Beatty, S. (2017, February 14). *Cited references in Scopus go back to 1970: A quick look at the impact on h-index*. Scopus Blog. Retrieved December 26, 2022, from https://blog.scopus.com/posts/cited-references-in-scopus-go-back-to-1970-a-quick-look-at-the-impact-on-h-index

Beazley, D. (2009, May 14). *Inside the Python GIL*. Retrieved December 4, 2022, from http://www.dabeaz.com/python/GIL.pdf

Binda, L., Clayford, M., Kasongo, A., Kondlo, L., Mudavanhu, P., Mathekga, J., Molotja, N., Mahlaela, A., Mavi, S., Mhlanga, N., Mustapha, N., Ralphs, G., Sass, T., Saunders, N., Ramoroka, K., Sigenu, V., Sithole, M., Slater, A., Vlotman, N., ... Zondi, L. (2021, December). *South African national survey of research and experimental development 2019/2020* (Statistical Report). Centre for Science, Technology and Innovation Indicators. Retrieved December 21, 2022, from https://www.dst.gov.za/images/RD_StatisticalReport2019-20__WEB.pdf

Bonacich, P. (1972). Factoring and weighting approaches to status scores and clique identification. *The Journal of Mathematical Sociology*, *2*(1), 113–120. https://doi.org/10.1080/0022250X.1972.9989806

Bonacich, P. (1987). Power and Centrality: A Family of Measures. *American Journal of Sociology*, *92*(5).

A Brief History of Research Funding in the United States. (2017). In R. J. Trew (Ed.), *Get Funded: An Insider's Guide to Building An Academic Research Program* (pp. 9–24). Cambridge University Press. https://doi.org/10.1017/9781107705869.003

Buterin, V. (2014). *Ethereum: A next-generation smart contract and decentralized application platform* (White Paper). Retrieved January 10, 2023, from

https://ethereum.org/669c9e2e2027310b6b3cdce6e1c52962/Ethereum_Whitepaper_-_Buterin_2014.pdf

Carter-Johnson, J. (2020, February 6). Chapter 2: University technology transfer structure and intellectual property policies. In *Research Handbook on Intellectual Property and Technology Transfer*. Edward Elgar Publishing. https://doi.org/10.4337/9781788116633.00011

Chaum, D. L. (1982). *Computer systems established, maintained, and trusted by mutually suspicious groups* [Doctoral dissertation, University of California, Berkeley]. Retrieved January 10, 2023, from https://chaum.com/wp-content/uploads/2022/02/chaum_dissertation.pdf

Cozzens, S. E. (1997). The knowledge pool: Measurement challenges in evaluating fundamental research programs. *Evaluation and Program Planning*, *20*(1), 77–89. https://doi.org/10.1016/S0149-7189(96)00038-9

Dwork, C., & Naor, M. (1993). Pricing via Processing or Combatting Junk Mail. In E. F. Brickell (Ed.), *Advances in Cryptology — CRYPTO' 92* (pp. 139–147). Springer Berlin Heidelberg.

Fabbri, A., Lai, A., Grundy, Q., & Bero, L. A. (2018). The Influence of Industry Sponsorship on the Research Agenda: A Scoping Review. *American journal of public health*, *108*(11), e9–e16. https://doi.org/10.2105/AJPH.2018.304677

Fang, H. (2018). A Discussion of citations from the perspective of the contribution of the cited paper to the citing paper. *Journal of the Association for Information Science and Technology*, *69*(12), 1513–1520.

Fielding, R. T. (2000, January). *Architectural styles and the design of network-based software architectures* [Doctoral dissertation, University of California, Irvine]. Retrieved December 21, 2022, from https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

Fielding, R. T., Nottingham, M., & Reschke, J. (2022, June). *HTTP Semantics* (Internet Request for Comment No. 9110). Retrieved January 29, 2023, from https://www.rfc-editor.org/rfc/rfc9110.html

Haber, S., & Stornetta, W. S. (1991). How to time-stamp a digital document. *Journal of Cryptology*, *3*(2), 99–111. https://doi.org/10.1007/BF00196791

Hagberg, A. A., Schult, D. A., & Swart, P. J. (2008). Exploring network structure, dynamics, and function using NetworkX. In G. Varoquaux, T. Vaught & J. Millman (Eds.), *Proceedings of the 7th Python in Science Conference* (pp. 11–15).

Hare, K. W., Lee, V., Plantikow, S., van Rest, O., & Michels, J. (2019, March 5). *SQL and GQL*. Berlin. Retrieved December 26, 2022, from https://www.gqlstandards.org/resources/papers-and-slides

Harhoff, D., Hall, B., von Graevenitz, G., Hoisl, K., Wagner, S., Gambardella, A., & Giuri, P. (2007, July 8). *The Strategic Use of Patents and Its Implications for Enterprise and Competition Policy*. European Commission. https://books.google.co.za/books?id=0i_FzQEACAAJ

Herman, E., Nicholas, D., Watkinson, A., Rodríguez-Bravo, B., Abrizah, A., Boukacem-Zeghmouri, C., Jamali, H. R., Sims, D., Allard, S., Tenopir, C., Xu, J., ÅšwigoÅ„, M., Serbina, G., & Cannon, L. P. (2021). The impact of the pandemic on early career researchers: What we already know from the internationally published literature. *Profesional de la información*, *30*(2). https://doi.org/10.3145/epi.2021.mar.08

Hu, X., Rousseau, R., & Chen, J. (2011). On the definition of forward and backward citation generations. *Journal of Informetrics*, *5*(1), 27–36. https://doi.org/10.1016/j.joi.2010.07.004

Jackson, M. O. (2010). *Social and economic networks*. Princeton University Press.

Jones, P. (2022, March 26). *Quart* (Version 0.17.0). https://quart.palletsprojects.com

Kapengut, E., & Mizrach, B. (2022). *An event study of the ethereum transition to proof-of-stake*. https://doi.org/10.48550/ARXIV.2210.13655

Katz, L. (1953). A new status index derived from sociometric analysis. *Psychometrika*, *18*(1), 39–43. https://doi.org/10.1007/BF02289026

Kline, A., Trans. (2005). *Horace: The Epistles Book II: Epistle I*. Poetry in Translation. Retrieved February 18, 2023, from https://www.poetryintranslation.com/PITBR/Latin/HoraceEpistlesBkIIEpI.php
(Original work published 14 B.C.E.)

Kronick, D. A. (2001). The Commerce of Letters: Networks and "Invisible Colleges" in Seventeenth- and Eighteenth-Century Europe. *The Library Quarterly: Information, Community, Policy*, *71*(1), 28–43. Retrieved February 16, 2023, from http://www.jstor.org/stable/4309484

Lindberg, D. C. (2007). *The beginnings of western science: The European scientific tradition in philosophical, religious, and institutional context, prehistory to A .D. 1450* (2nd ed.). The University of Chicago Press.

Mansfeld, J. (1985). Aristotle and Others on Thales, or the Beginnings of Natural Philosophy (With Some Remarks on Xenophanes). *Mnemosyne*, *38*(1/2), 109–129. Retrieved February 18, 2023, from http://www.jstor.org/stable/4431379

Maree, C. (2020). *Auctions and mechanism design for decentralized marketplaces* [Master's thesis, University of Cape Town].

Maree, C., Mandlate, H., & Meiklejohn, L. (2019). *UniCoin* [MPhil Group Project]. University of Cape Town. https://github.com/unicoinlicences/unicoindapp

Meiklejohn, L. (2020). *How to attribute credit if you must* [Master's thesis, University of Cape Town].

Nakamoto, S. (2008, October). *Bitcoin: A Peer-to-Peer Electronic Cash System* (White Paper). Retrieved January 9, 2023, from https://bitcoin.org/bitcoin.pdf

National Science Foundation. (1953). *The Third Annual Report of the National Science Foundation* (Annual Report No. 3). National Science Foundation. https://www.nsf.gov/pubs/1953/annualreports/ar_1953_sec6.pdf

Nelson, R. R. (1959). The Simple Economics of Basic Scientific Research. *Journal of Political Economy*, *67*(3), 297–306. Retrieved March 25, 2023, from http://www.jstor.org/stable/1827448

Neo4j Inc. & Neo4j Contributors. (2022, November 25). *Neo4j Graph Database* (Version 4.4.15). https://neo4j.com

OECD. (2022). *Basic research expenditure as a percentage of GDP*. https://www.oecd-ilibrary.org/content/component/94e2068d-en

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999, November). *The PageRank citation ranking: Bringing order to the web.* (No. 1999-66). Stanford InfoLab. http://ilpubs.stanford.edu:8090/422/
Previous number = SIDL-WP-1999-0120.

Peixoto, T. P. (2014). The graph-tool python library. *figshare*. https://doi.org/10.6084/m9.figshare.1164194

Petit, J.-C. (2004). Why do we need fundamental research? *European Review*, *12*(2), 191–207. https://doi.org/10.1017/S1062798704000195

Posner, E. A., & Weyl, E. G. (2018). *Radical Markets : Uprooting Capitalism and Democracy for a Just Society*. Princeton University Press. http://ezproxy.uct.ac.za/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=1628442&site=ehost-live

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners. Retrieved September 28, 2024, from https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

Rawal, P. (2017, January 23). *Neo4j vs Dgraph - The numbers speak for themselves.* Dgraph Blog. Retrieved December 20, 2022, from https://dgraph.io/blog/post/benchmark-neo4j/

Robinson, I., Webber, J., & Eifrem, E. (2015). *Graph databases: New opportunities for connected data* (2nd. Edition). O'Reilly Media, Inc.

Rose, M. E., & Georg, C.-P. (2021). What 5,000 acknowledgements tell us about informal collaboration in financial economics. *Research Policy*, *50*(6). https://doi.org/10.2139/ssrn.2709107

Rose, M. E., & Kitchin, J. R. (2019). Pybliometrics: Scriptable bibliometrics using a Python interface to Scopus. *SoftwareX*, *10*. https://doi.org/10.1016/j.softx.2019.100263.

Salvatore, S., & Redis Contributors. (2022, April 27). *Redis* (Version 7.0.0). https://redis.io

Sohrabi, C., Mathew, G., Franchi, T., Kerwan, A., Griffin, M., Soleil C Del Mundo, J., Ali, S. A., Agha, M., & Agha, R. (2021). Impact of the coronavirus (COVID-19) pandemic on scientific research and implications for clinical academic training – A review. *International Journal of Surgery*, *86*, 57–63. https://doi.org/10.1016/j.ijsu.2020.12.008

Solem, A., & Celery Contributors. (2022, May 25). *Celery - distributed task queue* (Version 5.2.7). https://docs.celeryq.dev/en/stable/index.html

Tencent Cloud Team. (2020, August 14). *Graph database performance comparison: Neo4j vs NebulaGraph vs JanusGraph*. Retrieved December 20, 2022, from https://www.nebula-graph.io/posts/performance-comparison-neo4j-janusgraph-nebula-graph

Tol, R. (2011). Credit where credit's due: Accounting for co-authorship in citation counts. *Scientometrics*, *89*(1), 291–299. https://EconPapers.repec.org/RePEc:spr:scient:v:89:y:2011:i:1:d:10.1007_s11192-011-0451-5

UNESCO Institute for Statistics (UIS). (2022). *UIS.Stat Research and experimental development (full dataset)*. Retrieved October 30, 2022, from http://data.uis.unesco.org/Index.aspx?DataSetCode=SCN_DS&lang=en

von Hippel, T., & von Hippel, C. (2015). To apply or not to apply: A survey analysis of grant writing costs and benefits. *PLoS ONE*, *10*(3). https://doi.org/10.1371/journal.pone.0118494

Walker, J., Brewster, C., Fontinha, R., Haak-Saheem, W., Benigni, S., Lamperti, F., & Ribaudo, D. (2022). The unintended consequences of the pandemic on non-pandemic research activities. *Research Policy*, *51*(1), 104369. https://doi.org/10.1016/j.respol.2021.104369

Appendix A

API details

A.1 Routes

The server provides the following endpoints (known as "routes" in the Quart context) that serve as the resources in the REST API:

- `/AddByDoi/` and `/AddByEid/` are used to initiate a search on Scopus for a particular document, using either its DOI or EID (internal identifier used by Scopus) respectively.

- `/buildgraph/` is used to initiate a job to process the citation network around a particular document. The caller can provide DOI, EID, or the universally unique identifier (UUID) used by the graph database to identify a document that has already been added. The caller can optionally provide the depth of the graph to build. The response provides an identifier for the job that can be used to query status and obtain results.

- `/buildgraph/status/<taskid>` allows the caller to query the status of a job to determine whether it is still running or has completed.

- `/buildgraph/result/<taskid>` returns the UUID of the document in the graph database, as well as the time it took for the job to complete. If the job is not complete, or failed, it will only return the status of the job.

- `/centrality/` is used to initiate a job to calculate the centrality measures for a document and its surrounding citation graph. It expects a document UUID to be provided – the graph should already have been built before calling this.

- `/centrality/status/<taskid>` functions identically to the corresponding `buildgraph` endpoint.

- `/centrality/result/<taskid>` returns a JSON object containing the calculated centrality measures for every document within the graph associated with the task.
- `/credit/` is used to initiate a job to calculate the credit measures for a document based on each of the centrality figures. It implicitly performs the centrality calculation and as such has the same requirements as the `centrality` task.
- `/credit/status/<taskid>` functions identically to the corresponding `buildgraph` and `centrality` endpoints.
- `/credit/result/<taskid>` returns a JSON object containing the calculated credit measures for the document.

The interface is also designed to follow IETF RFC 9110 HTTP semantics (Fielding et al., 2022) for requests and responses.

A.2 Example HTTP session

This appendix demonstrates an example client-server interaction that builds the citation graph for a paper and then calculates the centrality measures is given below. The Postman API testing tool was used to make the requests in this example. Client requests are given in blue and responses from the server are in red. Linebreaks have been added for readability.

Initial request to build graph

```
POST /buildgraph HTTP/1.1
Content-Type: application/json
User-Agent: PostmanRuntime/7.30.0
Accept: */*
Postman-Token: bab9fb66-6e7a-43dd-82e7-8c7e3b356bcf
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Content-Length: 45
Referer: http://127.0.0.1:5000/buildgraph
Host: 127.0.0.1:5000

{
"doi": "10.1186/s41118-021-00134-6"
}
```

Per RFC 9110 the server returns with a 202 status code to indicate that the job has been accepted and uses the location header to give the address to use to check for status updates. The task ID is also provided in the response body.

```
HTTP/1.1 202 Accepted
content-type: application/json
content-length: 50
location: /buildgraph/status/14a41d64-2fac-4503-8eb6-ebe740ce131d
date: Sun, 29 Jan 2023 12:54:10 GMT
server: hypercorn-h11

{"task_id":"14a41d64-2fac-4503-8eb6-ebe740ce131d"}
```

Checking status of job in progress

```
GET /buildgraph/status/14a41d64-2fac-4503-8eb6-ebe740ce131d HTTP/1.1
User-Agent: PostmanRuntime/7.30.0
Accept: */*
Postman-Token: 0cdbc513-6b29-44ee-8857-f7ae44dbf91f
Host: 127.0.0.1:5000
Accept-Encoding: gzip, deflate, br
Connection: keep-alive


HTTP/1.1 200 OK
content-type: application/json
content-length: 23
date: Sun, 29 Jan 2023 12:56:26 GMT
server: hypercorn-h11

{"state":"IN PROGRESS"}
```

Checking status of a completed job

```
GET /buildgraph/status/14a41d64-2fac-4503-8eb6-ebe740ce131d HTTP/1.1
User-Agent: PostmanRuntime/7.30.0
Accept: */*
Postman-Token: 97da7daf-2cfc-4b8a-9487-8514af3ca0e0
Host: 127.0.0.1:5000
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

```
HTTP/1.1 303
content-type: application/json
content-length: 86
location: /buildgraph/result/14a41d64-2fac-4503-8eb6-ebe740ce131d
date: Sun, 29 Jan 2023 13:13:08 GMT
server: hypercorn-h11

{"result":"/buildgraph/result/ \\
    14a41d64-2fac-4503-8eb6-ebe740ce131d","state":"SUCCESS"}
```

The 303 status code implies a redirect, such that the next request should be to the address in the location response header for the result.

Requesting result of completed job

```
GET /buildgraph/result/14a41d64-2fac-4503-8eb6-ebe740ce131d HTTP/1.1
User-Agent: PostmanRuntime/7.30.0
Accept: */*
Postman-Token: 97da7daf-2cfc-4b8a-9487-8514af3ca0e0
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://127.0.0.1:5000/buildgraph/status/ \\
    14a41d64-2fac-4503-8eb6-ebe740ce131d
Host: 127.0.0.1:5000


HTTP/1.1 200 OK
content-type: application/json
content-length: 117
date: Sun, 29 Jan 2023 13:13:08 GMT
server: hypercorn-h11

{"task_result":{"elapsed":249.12086719699437, \\
    "result":"0260291a-51d5-478b-9734-b2389ac3fda1"}, \\
    "task_state":"SUCCESS"}
```

The result endpoint returns the elapsed time, and the database identifier for the paper in the original request.

Requesting calculation of credit attributable

```
POST /credit HTTP/1.1
Content-Type: application/json
```

```
User-Agent: PostmanRuntime/7.30.0
Accept: */*
Postman-Token: c5d06c52-4476-4364-83a5-54d10ac7a61c
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Content-Length: 56
Referer: http://127.0.0.1:5000/credit
Host: 127.0.0.1:5000

{
"uuid": "0260291a-51d5-478b-9734-b2389ac3fda1"
}
```

```
HTTP/1.1 202 Accepted
content-type: application/json
content-length: 50
location: /credit/status/05b5de0e-037f-45da-a12a-2308c1c95e05
date: Sun, 29 Jan 2023 13:45:09 GMT
server: hypercorn-h11

{"task_id":"05b5de0e-037f-45da-a12a-2308c1c95e05"}
```

Requesting status and result of credit calculation

```
GET /credit/status/3a3f0d15-ae78-44f4-a20a-388b8d7724b7 HTTP/1.1
User-Agent: PostmanRuntime/7.30.0
Accept: */*
Postman-Token: 85196128-7612-4835-bd27-98eee4754356
Host: 127.0.0.1:5000
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

```
HTTP/1.1 303
content-type: application/json
content-length: 82
location: /credit/result/3a3f0d15-ae78-44f4-a20a-388b8d7724b7
date: Sun, 29 Jan 2023 17:39:48 GMT
server: hypercorn-h11
```

```
GET /credit/result/3a3f0d15-ae78-44f4-a20a-388b8d7724b7 HTTP/1.1
User-Agent: PostmanRuntime/7.30.0
Accept: */*
Postman-Token: 85196128-7612-4835-bd27-98eee4754356
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://127.0.0.1:5000/credit/status/ \\
    3a3f0d15-ae78-44f4-a20a-388b8d7724b7
Host: 127.0.0.1:5000


HTTP/1.1 200 OK
content-type: application/json
content-length: 204
date: Sun, 29 Jan 2023 17:39:48 GMT
server: hypercorn-h11

{"task_result":{"elapsed":16.781157707999228, \\
    "result":{"eigenvector":-1.389611034381612e-10, \\
              "in_degree":0.2179065090878315, \\
              "katz":0.3631717401173076, \\
              "pagerank":0.1066386278744122}}, \\
    "task_state":"SUCCESS"}
```

Appendix B

Comparison of graph-tool and NetworkX

			Centrality calculation			
			graph-tool		NetworkX	
Document	Nodes in graph	Construction time (min)	Memory (MB)	Time (s)	Memory (MB)	Time (s)
Paper 1	473	35	0.55	0.11	9.03	0.39
Paper 2	7874	79	10.53	1.73	1958.42	5.09
Paper 3	22 411	253	28.74	5.19	n/a	n/a

Table B.1: Performance statistics for graph generation (obtaining data from Scopus) and calculation of credit attributable for the three example papers of section 4.5. For calculation of credit attributable graph-tool and NetworkX are compared. NetworkX statistics are not available for Paper 3 because the amount of memory required to be allocated is so large that the operating system terminates the Python process.