LIMOS
LABORATOIRE D'INFORMATIQUE, DE MODÉLISATION ET D'OPTIMISATION DES SYSTÈMES

UNIVERSITÉ Clermont Auvergne

Université Clermont Auvergne

Doctoral School **Sciences pour l'Ingénieur**

University Department **Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes**

Thesis defended by **Frédéric A. Hayek**

Defended on **December 18, 2025**

In order to become Doctor from Université Clermont Auvergne

Academic Field **Computer Science**

Speciality **Cyber Security**

Security Analysis of Distributed Ledgers

Thesis supervised by	Pascal Lafourcade	Supervisor
	Ariane Tichit	Co-Monitor

Committee members

Referees	Maria Potop-Butucaru	Professor at Sorbonne Université	
	Quentin Bramas	Associate Professor at Université de Strasbourg	
Examiners	Gérard Chalhoub	Professor at Université Clermont Auvergne	Committee President
	Romaric Ludinard	Associate Professor at IMT Atlantique	
Supervisors	Pascal Lafourcade	Professor at Université Clermont Auvergne	
	Ariane Tichit	Associate Professor at Université Clermont Auvergne	

Colophon

Doctoral dissertation entitled “Security Analysis of Distributed Ledgers”, written by Frédéric A. Hayek, completed on May 29, 2026, typeset with the document preparation system LaTeX and the yathesis class dedicated to theses prepared in France.

LIMOS
LABORATOIRE D'INFORMATIQUE, DE MODÉLISATION ET D'OPTIMISATION DES SYSTÈMES

UNIVERSITÉ Clermont Auvergne

Université Clermont Auvergne

Doctoral School **Sciences pour l'Ingénieur**

University Department **Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes**

Thesis defended by **Frédéric A. Hayek**

Defended on **December 18, 2025**

In order to become Doctor from Université Clermont Auvergne

Academic Field **Computer Science**

Speciality **Cyber Security**

Security Analysis of Distributed Ledgers

Thesis supervised by	Pascal Lafourcade	Supervisor
	Ariane Tichit	Co-Monitor

Committee members

Referees	Maria Potop-Butucaru	Professor at Sorbonne Université	
	Quentin Bramas	Associate Professor at Université de Strasbourg	
Examiners	Gérard Chalhoub	Professor at Université Clermont Auvergne	Committee President
	Romaric Ludinard	Associate Professor at IMT Atlantique	
Supervisors	Pascal Lafourcade	Professor at Université Clermont Auvergne	
	Ariane Tichit	Associate Professor at Université Clermont Auvergne	

LABORATOIRE D'INFORMATIQUE, DE MODÉLISATION ET D'OPTIMISATION DES SYSTÈMES

UNIVERSITÉ Clermont Auvergne

Université Clermont Auvergne

École doctorale **Sciences pour l'Ingénieur**

Unité de recherche **Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes**

Thèse présentée par **Frédéric A. Hayek**

Soutenue le **18 décembre 2025**

En vue de l'obtention du grade de docteur de l'Université Clermont Auvergne

Discipline **Informatique**

Spécialité **Cyber Sécurité**

Analyse de Sécurité des Registres Distribués

Thèse dirigée par	Pascal Lafourcade	directeur
	Ariane Tichit	co-encadrante

Composition du jury

Rapporteurs	Maria Potop-Butucaru	professeure à la Sorbonne Université	
	Quentin Bramas	mcf à l'Université de Strasbourg	
Examinateurs	Gérard Chalhoub	professeur à l'Université Clermont Auvergne	président du jury
	Romaric Ludinard	mcf à l'IMT Atlantique	
Directeurs de thèse	Pascal Lafourcade	professeur à l'Université Clermont Auvergne	
	Ariane Tichit	mcf à l'Université Clermont Auvergne	

Keywords: distributed ledgers, blockchains, consensus, cryptocurrency

Mots clés : registres distribués, blockchains, consensus, cryptomonnaies

This thesis has been prepared at

Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes

Web Site `https://limos.fr/`

To Brigitte and to Claude

To Greta, Roland, Jamil, Marie

To Béatrice, Randa, Wissam, Riva, Iman

To Géraldine, Wissam, Marc, Milad, Joséphine, Mario, Nathalie, Irina, Ralph, Pamela, Sary, Vera-Maria, Randa-Carmen, Edmond-Steve

To Makaio, Maxim, Marylou

A scientist is just a kid who never grew up.

Neil deGrass Tyson

Research is formalized curiosity. It is poking and prying with a purpose.

Zora Neale Hurston

Docendo discimus.
By teaching, we learn.

Seneca

Si le savoir ne nous rendait pas plus heureux, il ne servirait à rien.

Pierre-Marie Morel

Security Analysis of Distributed Ledgers

Abstract

Cryptography emerged from a fundamental recognition: secure communication cannot rely on trusted intermediaries or centralized authorities. This skepticism toward concentrated power finds parallel expression in the Austrian school of economics, particularly Friedrich Hayek's critique of government monetary monopolies and advocacy for competing currencies. At the intersection of cryptographic distrust and economic liberalism, Bitcoin introduced blockchain technology as a foundation for competing digital currencies secured by cryptography rather than central authorities, enabling decentralized consensus where trust emerges paradoxically from mutual distrust among network participants.

This thesis advances distributed ledger theory through two contributions. First, we address the digitalization of local currencies through four blockchain-based constructions: *Base-Local* establishes a foundational framework; *Geo-Limited* enforces geographical constraints via distance-bounding protocols; *Geo-Demurrage* introduces geographical demurrage where currency value decreases with distance; and *Uni-Local* demonstrates universal scalability while maintaining local spending incentives.

Second, we introduce *Proof of Behavior* (PoB), a consensus mechanism replacing energy-intensive computations with verifiable human actions. We resolve the dichotomy between decentralization and behavioral verification through a framework integrating Verifiable Delay Functions for sequential mining. The architecture includes a fuel-and-bounty transaction system ensuring size-neutral fees, temporal demurrage balancing money creation, and a checkpoint mechanism leveraging pBFT consensus with BLS signature aggregation to provide deterministic finality and enable storage pruning. We establish security guarantees through correspondence with the Bitcoin Backbone Protocol while identifying and mitigating PoB-specific attacks including selfish mining and concurrent sibling blocks. The prototype EcoMobiCoin validates feasibility by rewarding ecological mobility.

These contributions demonstrate that blockchain technology can serve socially aligned objectives while maintaining rigorous security and decentralization principles essential to trustless systems.

Keywords: distributed ledgers, blockchains, consensus, cryptocurrency

Analyse de Sécurité des Registres Distribués

Résumé

La cryptographie est née d'une constatation fondamentale : la sécurité des communications ne peut reposer sur des intermédiaires de confiance ou des autorités centralisées. Ce scepticisme à l'égard du pouvoir concentré trouve un écho dans l'école autrichienne d'économie, en particulier dans la critique de Friedrich Hayek à l'égard des monopoles monétaires gouvernementaux et son plaidoyer en faveur de la concurrence entre les monnaies. À la croisée de la méfiance cryptographique et du libéralisme économique, Bitcoin a introduit la blockchain comme fondement de monnaies numériques concurrentes sécurisées par la cryptographie plutôt que par des autorités centrales. La blockchain a permis un consensus où la confiance émerge paradoxalement de la méfiance mutuelle entre les participants au réseau.

Cette thèse fait progresser les registres distribués grâce à deux contributions. Premièrement, nous abordons la digitalisation des monnaies locales à travers quatre constructions basées sur la blockchain : *Base-Local* établit un cadre fondamental ; *Geo-Limited* impose des contraintes géographiques via des protocoles de limitation de distance ; *Geo-Demurrage* introduit une surestarie géographique ; et *Uni-Local* démontre une évolutivité universelle tout en maintenant les incitations à dépenser localement.

Deuxièmement, nous introduisons la preuve de comportement (*Proof of Behavior*, PoB), un mécanisme de consensus qui remplace les calculs énergivores par des actions humaines vérifiables. Nous résolvons la dichotomie entre décentralisation et vérification comportementale en intégrant des fonctions de délai vérifiables pour la preuve de travail séquentielle. L'architecture comprend un mécanisme de checkpoint tirant parti du consensus pBFT avec des agrégations de signatures. Nous établissons des garanties de sécurité grâce à la correspondance avec le protocole Bitcoin Backbone tout en identifiant et en atténuant les attaques spécifiques à PoB, notamment le minage égoïste et les blocs frères concurrents. Le prototype EcoMobiCoin valide la faisabilité en récompensant la mobilité écologique.

Ces contributions démontrent que la technologie blockchain peut servir des objectifs sociaux tout en maintenant les principes rigoureux de sécurité et de décentralisation essentiels aux systèmes sans confiance.

Mots clés : registres distribués, blockchains, consensus, cryptomonnaies

Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes

Acknowledgments

I would like to thank Maria Potop-Butucaru and Quentin Bramas for accepting to be the referees for this thesis.

I would like to thank Romaric Ludinard and Gérard Chalhoub for accepting to be examiners for this thesis.

I would like to thank Ariane Tichit who was very supportive throughout this experience.

Special thanks and acknowledgments go to my advisor Pascal Lafourcade for his mentoring and all he taught me, well beyond technical knowledge. You taught me how to read, how to review and how to write articles. You taught me how to best manage my time. You taught me how to do presentations and how to teach. You taught me to go straight to the point. And I thank you for your patience and your mentoring throughout these years.

I thank all the people who have been a part of Doct'Auvergne. Without this association, without these people, the outcome would have not been the same. You had a significant impact on my life and I will forever be grateful.

Contents at a Glance

Abstract xxiii

Acknowledgments xxv

Contents at a Glance xxvii

Contents xxix

List of Tables xxxi

List of Figures xxxiii

A Game of Trust 1

1 General Introduction 9

2 Local Cryptocurrency 29

3 Generic Blockchain on Generic Human Behavior 57

4 General Conclusion 145

Contents 151

Contents

Abstract **xxiii**

Acknowledgments **xxv**

Contents at a Glance **xxvii**

Contents **xxix**

List of Tables **xxxi**

List of Figures **xxxiii**

A Game of Trust **1**
In the beginning was the Word . . . 1
Put not your trust in princes . . . 3
David and Goliath . . . 4
Test all things; hold fast what is good . . . 5

1 General Introduction **9**
1.1 Introduction . . . 9
1.2 Cryptography . . . 10
1.2.1 Public Key Cryptography . . . 10
1.2.2 Hash Functions . . . 12
1.3 Money . . . 13
1.3.1 Money . . . 13
1.3.2 Demurrage . . . 16
1.3.3 Digital Currency . . . 17
1.4 Consensus . . . 19
1.4.1 Consensus . . . 19
1.4.2 Practical Byzantine Fault Tolerance (pBFT) . . . 20
1.5 Blockchains . . . 21
1.5.1 (De)centralization . . . 23
1.5.2 Governance . . . 25
1.6 Manuscript Outline and Contributions . . . 27

2 Local Cryptocurrency **29**
2.1 Introduction . . . 29
2.1.1 Challenges in Digitalization of Local Currencies . . . 31
2.1.2 Contributions . . . 31
2.1.3 Related Work . . . 33
2.1.4 Outline . . . 35
2.2 Economics . . . 36
2.2.1 Demurrage . . . 36
2.2.2 Foundations of Local Currencies . . . 39
2.2.3 Geographical Demurrage: A Spatial Approach to Local Currencies . . . 41
2.3 Distance Bounding . . . 45
2.4 Local Cryptocurrency . . . 47
2.4.1 Base-Local . . . 49
2.4.2 Geo-Limited . . . 50
2.4.3 Geo-Demurrage . . . 51
2.4.4 Uni-Local . . . 53
2.5 Conclusion . . . 54

3 Generic Blockchain on Generic Human Behavior **57**
3.1 Introduction . . . 57
3.1.1 Contributions . . . 59
3.1.2 Related Work . . . 60
3.1.3 Outline . . . 62
3.2 Foundational Concepts . . . 62
3.2.1 Verifiable Delay Functions . . . 62
3.2.2 BLS Signature Scheme . . . 63
3.2.3 Proof of Behavior . . . 65
3.3 System Architecture . . . 70
3.3.1 Block Structure . . . 70
3.3.2 Mining and Consensus Mechanism . . . 72
3.3.3 Transaction System . . . 75
3.3.4 Economic Mechanism . . . 85
3.4 Checkpoint Mechanism . . . 91
3.4.1 Motivation and Design Principles . . . 92
3.4.2 pBFT Consensus Protocol . . . 97
3.4.3 Data Pruning and Storage Optimization . . . 104
3.4.4 Economic Incentives . . . 106
3.5 System Analysis . . . 107
3.5.1 PoB Expiration . . . 107
3.5.2 Energy Consumption . . . 108
3.5.3 Monetary Policy Analysis . . . 110
3.6 Security Analysis . . . 115
3.6.1 Bitcoin Backbone Protocol Correspondance . . . 115
3.6.2 Special Attacks on PoB Blockchains . . . 119
3.6.3 Checkpoint Security Analysis . . . 134
3.7 Application and Implementation . . . 138
3.7.1 Applications of PoB Blockchains . . . 139
3.7.2 Prototype Implementation: EcoMobiCoin . . . 139
3.8 Conclusion . . . 141

4 General Conclusion **145**
4.1 Conclusion 145
4.1.1 Local Cryptocurrencies 145
4.1.2 Proof of Behavior 146
4.2 Perspectives 147
4.2.1 Perspectives on Local Cryptocurrencies 147
4.2.2 Perspectives on Proof of Behavior 148

Contents **151**

List of Tables

2.1 Four constructions properties. 32

3.1 Order of PoB added to Blockchain 67
3.2 Blockchain parameters 73
3.3 State of participant accounts before the transaction 78
3.4 State of participant accounts after the transaction 79
3.5 State of participant accounts before the transaction 79
3.6 State of participant accounts after the transaction 80
3.7 State of participant account before the redemption 82
3.8 State of participant account before the redemption 82
3.9 State of miner before the redemption 83
3.10 State of miner after the redemption 84
3.11 Accounts and Balances 94
3.12 Advantage of Merkle Trees 96
3.13 Comparison of light, full, and archive nodes: data kept vs. data pruned. Here h denotes the height of the latest finalized checkpoint, and exp is the PoB expiration window (in blocks). Full nodes retain all blocks after h to follow forks, while archive nodes store everything from genesis. 105
3.14 Proof of Work and Proof of Behavior correspondence 116
3.15 Comparison of assumptions in the Backbone correspondence (Section 3.6.1) and their validity in PoB under adversarial settings (Section 3.6.2). 120
3.16 Blockchain parameters (implementation prototype). 141

List of Figures

1.1 Cash Payment System 18
1.2 Simplified Digital Payment 18
1.3 Chronological flow of Chaum's e-cash protocol: (1) the user requests a blind signature on a coin, (2) the bank returns the signed coin, (3) the user spends it with a merchant, and (4) the merchant redeems it at the bank. 18
1.4 (De)centralization Spectrum. 23

2.1 Examples of Currency Value Fluctuation in Function of Distance. 44
2.2 Brands and Chaum Distance Bounding Protocol 47
2.3 Blockchain Miners 49
2.4 Transaction Structure 52

3.1 Illustration of BLS Signature 65
3.2 Example of PoB creation in metro transportation services 67
3.3 Example of PoB expiration 68
3.6 Simple standard transaction example illustration 78
3.7 Multi-party standard transaction example illustration 80
3.8 Redemption transaction example illustration 82
3.9 Coinbase transaction example illustration 84
3.10 Merkle Tree Construction Example 95
3.11 pBFT checkpoint creation process showing the three-phase protocol. Alice (w_0) serves as primary in view 0, broadcasting the checkpoint proposal to all witnesses. The protocol progresses through Pre-Prepare, Prepare, and Commit phases, requiring $2f = 66$ prepare messages and $2f + 1 = 67$ commit messages for finalization. Byzantine witnesses like Mallory (w_{13}) may send invalid or no messages (shown with dashed line), but the protocol succeeds with honest majority. 103
3.4 Full block structure 143
3.5 Mining workflow in the PoB blockchain. Users submit behaviors and transactions to a shared pool, while miners compute VDFs using their own PoBs, simulating a lottery process. A miner is selected for the next block. 144

A Game of Trust

Humans distinguish themselves as the sole species with advanced communication abilities, a trait that is arguably the cornerstone of our identity. This capability allows us to engage in cooperation on scales as vast as entire nations. Throughout history, as methods of communication have evolved, so too have the strategies to protect them. Cryptography, an ancient yet continually evolving discipline, has long been dedicated to ensuring the security of communications in settings where trust is uncertain. By nature, cryptographers embody an inherent skepticism, striving to maximize the security and reliability of communications by minimizing the need for trust in individual entities or even by minimizing the need for trust in physical components. This cautious outlook often extends into other facets of life, particularly with respect to authority. Similarly, the Austrian school of economics espouses a wariness towards governmental intervention, especially in economic matters. F. A. Hayek, a prominent advocate of this school, was particularly critical of government monopolies, including those over currency. At the intersection of these perspectives–cryptographic skepticism and economic distrust–Bitcoin was born as the first cryptocurrency. It represents a groundbreaking financial system in which no singular entity holds control, grounded in the principles of secure communication within a trustless environment.

Outline of the current chapter

In the beginning was the Word	**1**
Put not your trust in princes	**3**
David and Goliath	**4**
Test all things; hold fast what is good	**5**

In the beginning was the Word

We, as human beings, evolved and conquered the world thanks to our meticulous cooperation spanning inordinate numbers of individuals. Cooperation on such scales would not theoretically have been possible because of the constraints on group sizes [**dunbar1992neocortex**]. Dunbar's number is a suggested cognitive limit to the number of people with whom one can maintain stable social relationships–relationships in which an individual knows who each person is and how each person relates to every other person. When primate groups grow larger than their associated upper limit, relationships become unstable and the group fragments. However, human language evolved in the form of speech starting some 100'000 years ago [**tattersall2009language**] to allow cooperation among evergrowing groups [**dunbar1996grooming**]. Human language is so supple, it allows to talk about abstract and non-existent things. It allowed humans to construct and believe in very detailed shared myths which allowed cooperation among increasingly bigger

groups [**harari2014sapiens**]: nations, religions, juridical people (non-human persons such as corporations), and money.

The Sapir-Whorf hypothesis [**Sapir-Whorf**] suggests that the structure of a language shapes the speaker's worldview and cognition. Many studies show that language shapes thought [**boroditsky2001does**]. The language we speak shapes our perception of space and time [**boroditsky2010remembrances**, **fuhrman2010cross**], our perception of gendered things [**boroditsky2003sex**], and even of causality [**fausey2011dunnit**]. As the philospher Ludwig Wittgenstein put it: "The limits of my language are the limits of my world" [**tractatus**].

Language and its evolution [**fitch2010evolution**] did not shape the world on their own. The evolution of communication–the *modus operandi* of language–marked a significant turning point in shaping human societies and cultures. While language provided the framework for structured communication, it was the evolution of communication methods and tools that truly amplified the impact of language on the world. From the oral traditions of early human societies to the written languages that allowed for the preservation and dissemination of knowledge, advancements in communication have continually transformed how information is shared and understood.

The development of communication technologies, such as writing, the printing press, the telephone, and the internet, further revolutionized the way humans interact and exchange ideas, transcending geographical and cultural boundaries [**poe2010history**]. Each of these innovations not only made communication more efficient but also facilitated the spread of new ideas, social movements, and technological advancements, ultimately shaping societies and altering the course of history.

Securing communications has been a fundamental aspect of human interaction, evolving in tandem with advancements in communication methods across different dimensions, and nowhere has this been more pronounced than in the realm of warfare. In ancient times, the security of oral communications was entrusted to reliable messengers, chosen for their loyalty and discretion, to ensure that military strategies were delivered accurately and securely. As written communication gained prevalence, cryptography emerged as a vital tool, with codes and ciphers shielding strategic plans from enemy interception–such as the famed use of the Caesar cipher in Roman times [**kahn1996codebreakers**, **singh1999code**]. The necessity for secure military communications drove innovations that later found civilian applications. During World War II, the development of the Enigma machine [**kahn1996codebreakers**, **singh1999code**] by the Germans spurred advancements in cryptography and computational technology, directly influencing the birth of modern computing [**campbellkelly2004**]. The Global Positioning System (GPS), now a staple in civilian navigation and logistics, was initially developed for precise military targeting and troop movement [**bonnor2012history**]. The invention of the internet itself has roots in ARPANET, a project initiated by the U.S. Department of Defense to create a resilient and secure communication network [**leiner1999brief**].

As Gregory Aldrete put it: "The history of warfare is basically the history of technological change" [**aldrete**]. And this is true even beyond communication technologies: from the radar (air traffic control and weather forecasting) to drones (agriculture, filmmaking, and logistics) to the jet engine (modern commercial aviation) to kevlar (reinforced tires and sports equipment) to nuclear technology (source of power generation and medical applications such as cancer treatments). But these are out of our scope.

Cryptography's intrinsic secrecy is paramount in warfare due to its role in both securing one's communications and deciphering an adversary's strategies. By safeguarding sensitive information, cryptography acts as an unassailable shield against espionage, ensuring that military plans remain concealed from enemy interception. As sun Tzu articulated "These military devices, leading to victory, must not be divulged beforehand" [**tzu2008art**]. The clandestine nature of

cryptography is crucial as revealing its methods could empower adversaries to counteract or replicate the protections offered, thwarting strategic advantages. Sun Tzu adds, "If you know the enemy and know yourself, you need not fear the result of a hundred battles" [**tzu2008art**]. Cryptography embodies this principle by enabling forces to clandestinely understand adversaries while protecting their own vulnerabilities. As one of the most critical branches of warfare, it underpins the success and security of every strategy and operation, providing a decisive edge in military engagements. By mastering the art of cryptography, nations can navigate the complexities of modern warfare with the assurance that their communications and data are impervious to infiltration, maintaining the upper hand in conflicts.

Put not your trust in princes

Cryptography, as both an art and a science, is fundamentally about minimizing trust to maximize security. Rooted in historical contexts where the reliability of communication was in constant question, cryptography seeks to create systems that ensure confidentiality, integrity, and authenticity independently of the trustworthiness of individuals or institutions involved. By minimizing reliance on any single point of trust, cryptography remains resilient against breaches, regardless of the intentions or reliability of those involved in the communication chain.

One of the earliest examples emphasizing the need to minimize trust is the use of the Caesar [**kahn1996codebreakers**, **singh1999code**] cipher by Roman generals. Engaged in expansive military campaigns, Romans faced the challenge of sending sensitive information through potentially adversarial environments. Their solution was to use a simple encryption technique that shifted letters by a fixed number. While rudimentary, the Caesar cipher highlights the principle that the environment–comprising both attackers and unreliable messengers–cannot be relied upon for secure message delivery. By encrypting the message, the cipher assumed that anyone could be intercepting communications, thus safeguarding information from anyone not authorized to decrypt it.

Expanding this notion into more complex coordination problems, the Byzantine Generals Problem underscores another foundational cryptographic challenge: ensuring reliable consensus amidst uncertainty. In this theoretical scenario, a group of generals, each commanding part of an army encircling a city, must agree on a common plan of attack, despite being unable to trust that all messages will be reliably received. This dilemma points to the necessity of designing protocols that do not rely on the assumption that the message successfully reached the other side. Cryptographic techniques arising from this problem help mitigate the risks of network unreliability, ensuring that consensus can be achieved even if some communications fail.

Compounding these challenges is the generalized form of the Byzantine Generals Problem, which deals with the possibility of faulty or malicious entities within one's own ranks–a situation exemplified by the German Enigma machine during World War II [**kahn1996codebreakers**, **singh1999code**]. Although the encryption was robust, human operators often failed to follow protocols precisely, choosing predictable settings for convenience. Even worse, Hans-Thilo Schmidt, a Nazi soldier, sold sensitive Enigma documents to the French. These lapses provided critical vulnerabilities that the Allies exploited. This case emphasizes the importance of creating cryptographic systems that are resilient to insider threats and human error, acknowledging that not all allies can be relied upon to adhere strictly to protocols.

A more direct illustration of distrust in authority comes from the historical case of Eli Cohen, an Israeli spy who operated in Syria in the 1960s [**segev1986alone**]. Cohen masterfully infiltrated the highest echelons of the Syrian government, gathering intelligence that was pivotal for Israel's strategic advantage. His ability to operate undetected for several years highlights the

potential for deceit at the highest levels of authority. This case emphasizes the risk inherent in trusting authority figures or centralized entities without safeguards.

Even in scenarios where there is a completely trustworthy authority, reliance on a single entity creates a critical vulnerability known as a single point of failure. A single point of failure is a part of a system that, if compromised or fails, can cause the entire system to stop functioning. Trusting a centralized authority with sole control over data or processes means that any breach, mistake, or failure on their part could lead to catastrophic results. Historical precedents have shown repeatedly that concentrating power or control in one place significantly amplifies the risk of system-wide failure–whether due to corruption, incompetence, or malicious attacks.

Cryptographic solutions aim to avoid single points of failure by distributing trust. Techniques such as decentralized networks, distributed ledger technologies like blockchain, and consensus protocols are designed to function without relying on any singular authority. By decentralizing control, cryptographic systems ensure that no single actor holds all the power, thereby reducing the risks associated with failure or misuse of power. These innovations exemplify how cryptography promotes security and resilience by building systems where the failure or corruption of one component does not jeopardize the whole.

David and Goliath

In the field of cryptography, a professional deformation often occurs, fostering a mindset characterized by wariness and distrust. This natural evolution arises from the persistent challenge of ensuring security in environments where threats and vulnerabilities are omnipresent. Cryptographers must assume that any communication can be intercepted, any system can be exploited, and any trusted party can inadvertently or deliberately become a threat. This cautionary approach hones their skills in developing robust systems that resist tampering or surveillance, even from ostensibly reliable sources. As they work within the realm of protecting sensitive data, cryptographers learn to identify potential weaknesses and anticipate the intentions of adversaries, embedding skepticism into their professional ethos. Consequently, this cultivated vigilance not only advances cryptographers' innovative solutions to secure communications but also informs their broader perspective on authority and centralized power. They are particularly wary of governments and large institutions, recognizing the potential for these entities to misuse power, infringe on privacy, and compromise civil liberties. This deep-seated distrust drives cryptographers to champion decentralized systems that distribute trust and minimize dependency on singular entities, ensuring that no single authority can exert undue control or surveillance over individuals.

The surveillance and harassment of Martin Luther King Jr. during the civil rights movement is a stark illustration of how centralized powers can exploit surveillance to quash dissent [**FBI-MLK**]. The FBI, under J. Edgar Hoover, engaged in extensive monitoring of King, including wiretaps and covert infiltration of his inner circle. This overreach culminated in the infamous anonymous letter sent to King, which sought to intimidate him into abandoning his leadership by threatening to expose personal information about his sexual life and extramarital affairs and, shockingly, suggesting he take his own life. This misuse of surveillance underscores the dangers of unchecked authority and how it can be weaponized against individuals advocating for justice and change.

For cryptographers, such historical examples are critical reminders of the need for robust privacy protections. These incidents reinforce their commitment to developing technologies that secure communications and personal data from prying eyes, even those of authoritative entities. Privacy, therefore, emerges as not only a shield for individual safety but a cornerstone

for safeguarding freedom of speech [**USConst**] and civil liberties. In the absence of privacy, movements like those led by King risk being undermined by the very systems meant to protect them. Hence, cryptographers emphasize decentralized, secure communication tools that ensure individuals can advocate for change without fear of surveillance or coercion by overreaching powers.

The right to privacy [**Right_to_privacy**] is a fundamental pillar in safeguarding individual autonomy and is essential for maintaining a free and open society. It provides the means by which individuals can control their personal information and decide what to share with the world, protecting them from intrusion and undue influence. This right to privacy is not only a personal safeguard but is also integral to the defense of free speech. Without privacy, individuals cannot freely express their thoughts and ideas without fear of surveillance or repercussion. This link between privacy and free speech is captured in the belief that for open dialogue to flourish, individuals must have the assurance that their communications are secure and private. Cypherpunks, a group of activist cryptographers, championed these principles, recognizing that privacy enables freedom of expression. Eric Hughes's "A Cypherpunk's Manifesto" [**hughes1997cypherpunk**] articulates this connection by advocating for systems that support anonymous interactions. This philosophy acknowledges that without privacy, freedom of speech is undermined, as individuals are less likely to voice dissenting opinions or engage in open discourse if they fear exposure or retaliation [**snowden2019permanent**]. Thus, cryptography's role in preserving privacy directly supports the freedom of speech, allowing diverse voices to be heard in a truly open society. Furthermore, "if privacy is outlawed, only the outlaws will have privacy." This statement serves as a cautionary warning that if privacy measures are curtailed, only those willing to defy the law, often with malicious intent, will maintain privacy, thus emphasizing the need for widespread, protected privacy rights.

Vitalik Buterin's insight that "cryptography is one of the very few fields where adversarial conflict continues to heavily favor the defender" [**Vitalik_PoS**] underscores the unique strategic advantage it provides in preserving personal freedom. While most technologies can be disproportionately wielded by those with resources and power, cryptography stands out as an equalizing force. Take for example how Edward Snowden describes how he encrypted all the data he had stolen from the NSA:

> "A little bit of math can accomplish what all the guns and barbed wire can't: a little bit of math can keep a secret."

The mathematical nature of cryptography–embodied in techniques like elliptic curve cryptography (ECC)–offers robust protection even against state-level actors, empowering individual citizens to secure their digital activities without needing vast resources. This capability allows any and all individuals to rival the intrusive powers of states in terms of securing personal data and communications. In no other domain can an average person wield a tool that effectively counters the surveillance apparatus of entire nations, making cryptography a quintessential defender of personal autonomy against overreaching authorities. This inherently democratic access to power aligns with cryptographers' aspirations to uphold individual rights, ensuring freedom and privacy can be preserved in the digital age.

Test all things; hold fast what is good

Cryptographers' inherent distrust of centralized authorities parallels the ideas of the Austrian school of economics, which, along with other ideologies, often exhibits a deep skepticism of government intervention and control. This skepticism arises from concerns about the misuse of

power, infringement on individual freedoms, and inefficiencies often associated with centralized systems [**hayek1944road**]. F. A. Hayek, a leading figure in Austrian economics, extended this skepticism to monetary policy, advocating for competitive currencies as a means to counteract the potential pitfalls of government monopolies on currency [**hayek1976denationalisation**]. Hayek proposed that allowing currencies to compete in a free market would lead to more stable and efficient financial systems, reducing the risks of inflation and mismanagement. His vision sought to decentralize economic power, reflecting a broader belief in the self-regulating potential of markets to preserve individual autonomy and economic integrity.

David Chaum's e-cash, devised in the 1980s, represented a groundbreaking approach to digital currency that offered the anonymity and privacy of cash transactions within a digital framework. Although transactions involved interaction with a bank, e-cash functioned as a distinct currency operating under different principles. Users could obtain digital coins from a bank that were uniquely encoded and spend them with complete anonymity, using cryptographic protocols that prevented either party from tracing the exchange. The e-cash system used "blind signatures," enabling the bank to validate the authenticity of the digital currency without accessing transaction details. This innovation maintained privacy while ensuring that each e-cash coin was unique and non-reproducible, preventing the problem of double spending. Though it did not garner widespread use, e-cash played a significant role in illustrating the potential of cryptography in secure digital transactions.

Bitcoin emerged in 2009 as an innovative realization of a decentralized and trustless currency model, fundamentally altering the approach to digital finance. This transformation is anchored in Bitcoin's blockchain technology–a transparent public ledger that records every transaction across a widespread network of computers. Within this system, a radical shift occurs: no single entity is trusted, so every participant independently verifies transactions. When a transaction is initiated, it is broadcast to the network, where multiple, independent participants confirm its validity through a consensus mechanism. This consensus ensures that all entries in the blockchain are accurate and tamper-proof.

The introduction of Bitcoin not only heralded a new era in digital currency but also triggered the development of a competitive marketplace for cryptocurrencies. Numerous alternative cryptocurrencies have since emerged, each offering unique features and improvements on Bitcoin's foundational design. Ethereum, for instance, expands the possibilities with smart contract functionality, while Litecoin offers quicker transaction processing. Privacy-focused cryptocurrencies like Monero and Z-Cash put a premium on anonymity, using sophisticated cryptographic techniques to obscure transaction details. This flourishing ecosystem of cryptocurrencies encourages continuous innovation, fostering an ever-diversifying financial landscape that challenges conventional banking norms. By offering a range of decentralized monetary options, these currencies empower users to select solutions tailored to their needs, echoing F. A. Hayek's vision for a competitive and decentralized currency market.

Within the Bitcoin system, trust arises paradoxically from the mutual distrust among entities on the network. Each participant on the blockchain network ensures the integrity of transactions by independently verifying their validity. When someone initiates a transaction, it is broadcast across the entire network, where multiple participants rigorously check and confirm its authenticity through a consensus mechanism. This lack of reliance on a single entity strengthens the network's security, as every participant must agree on the transaction details. For the average user who doesn't engage in direct verification, confidence in the system is rooted in this distributed validation process: because each entity diligently verifies every transaction, and because entities do not trust each other, users can trust the resulting consensus. Any wrongdoing would have been detected by the non-colluding majority. Thus, the average user can trust the overall system without needing to trust any individual actor specifically.

Beyond cryptocurrencies, the blockchain has proven to be a versatile technology with applications that extend far beyond digital currencies. At its core, a blockchain ensures that data is transparently and securely recorded, offering a tamper-proof system that can be applied across various industries. For example, in supply chain management, blockchains track and verify products at each stage of production, enhancing transparency and reducing fraud. In healthcare, blockchain technology securely manages patient records, ensuring data integrity while maintaining privacy. The realm of finance benefits from blockchain-based applications that streamline transactions, reduce costs, and eliminate inefficiencies associated with traditional banking systems. Additionally, blockchain has found uses in voting systems, offering secure and transparent electoral processes that can enhance democratic engagement. As the technology evolves, its capacity to provide decentralized solutions across diverse sectors highlights its potential to revolutionize how systems are designed, making processes more resilient, transparent, and efficient.

1

General Introduction

In this first chapter we introduce the setting of our work: decentralization. We introduce the concepts and the definitions used throughout the manuscript: from cryptography to blockchains. We also introduce the specific problems solved that are detailed in the subsequent chapters.

Outline of the current chapter

1.1 Introduction **9**
1.2 Cryptography **10**
1.2.1 Public Key Cryptography 10
1.2.2 Hash Functions 12
1.3 Money **13**
1.3.1 Money 13
1.3.2 Demurrage 16
1.3.3 Digital Currency 17
1.4 Consensus **19**
1.4.1 Consensus 19
1.4.2 Practical Byzantine Fault Tolerance (pBFT) 20
1.5 Blockchains **21**
1.5.1 (De)centralization 23
1.5.2 Governance 25
1.6 Manuscript Outline and Contributions **27**

1.1 Introduction

Communication is already difficult enough with only two parties involved, even when these parties are honest and willing. The involved parties have to speak the same language. They have to know the same words. They have to know the same expressions. They have to know the same culture. They have to interpret intonations the same way. And even then, miscommunication happens. It becomes exponentially more difficult when more people communicate together, *a fortiori* when the communication is not instantaneous and some entities may be malicious.

Imagine a set of unknown size of people who are communicating by postcards. Some may speak the same language, some may not. Some may be honest, some may be malicious. Some

may receive messages in a certain order, some may receive them in a different order, and some may receive a different set of messages. And their goal is to reach consensus on something. This is the setting we are working in.

In this work we show our contributions to the applications of consensus in such a setting.

This type of consensus is achieved using cryptography. The word "cryptography" itself comes from Ancient Greek, combining the words kryptós, meaning "hidden" or "secret", and graphein, meaning "to write". But cryptography grew larger than just hidden writing. Today cryprography consists of the study of secure communications. We assume the reader has familiarity with computer science, mathematics and cryptography.

Outline. First we define the cryptographic notions that will be used throughout this manuscript in Section 1.2. Chapter-specific notions will be defined within each chapter.

In Section 1.3 we talk about what constitutes a currency, and how cryptography can create a digital currency.

Consensus is defined in Section 1.4, along with primitive ways to achieve it when the set of users is predefined.

Blockchains are introduced in Section 1.5 to address consensus with unbounded numbers of unknown entities.

Finally we outline our contributions and their integration in this manuscript in Section 1.6.

1.2 Cryptography

Cryptography has many branches. We focus here only on the branches we deem useful for our contributions.

At its most basic level, cryptography is the art and science of keeping information secure. Imagine you want to send a secret message to a friend, but you're worried that someone might intercept and read it. Cryptography provides mathematical tools to scramble your message in such a way that only your intended recipient can understand it, while anyone else sees only meaningless gibberish.

There are two fundamental approaches to cryptography, which can be understood through bicycle lock analogies. The first, called *secret key cryptography* or symmetric cryptography, is like a combination lock on a bicycle. Both Alice and Bob know the same secret combination – say, 1234. Alice can use this combination to lock her bicycle, and Bob can use the same combination to unlock it later. This works perfectly when Alice and Bob can safely share the combination beforehand, but creates a problem: how does Alice securely share the combination with someone she's never met? If Alice shouts the combination across a crowded street, everyone will hear it.

The second approach, called *public key cryptography* or asymmetric cryptography, solves this problem using a different type of lock mechanism. Imagine Alice has a bicycle lock that clicks shut when pressed closed – anyone can close this lock without needing any special tool or code. However, once closed, it can only be opened with a specific key that Alice keeps. Alice gives copies of this lock to Bob and anyone else who wants to send her something securely. Now, Bob can put a message in a box, attach Alice's lock, and click it shut. But only Alice, with her private key, can open it.

In cryptography, we call Alice's physical key the *private key* (which she keeps secret) and the lock mechanism her *public key* (which everyone can use to "lock" messages for Alice). This revolutionary concept enables secure communication without prior secret sharing and forms the foundation of modern digital security, including blockchain technology.

1.2.1 Public Key Cryptography

Building on our bicycle lock analogy, public key cryptography works through mathematical trapdoor functions – calculations that are easy to perform in one direction but computationally infeasible to reverse without special information (the private key). For example, it's easy to multiply two large prime numbers together, but extremely difficult to factor the result back into its original primes without additional information.

Definition 1.1: Public Key Cryptosystem

A public key cryptosystem consists of three probabilistic polynomial-time algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ such that:

- $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$ is a randomized algorithm that takes a security parameter λ and outputs a public key pk and private key sk.
- $\mathsf{Enc}(pk, m) \to c$ is a randomized algorithm that takes a public key pk and message m, and outputs a ciphertext c.
- $\mathsf{Dec}(sk, c) \to m'$ is a deterministic algorithm that takes a private key sk and ciphertext c, and outputs a message m' or a special failure symbol $\perp$.

A secure public key cryptosystem must satisfy the following properties.

Property 1.2: Correctness

For all messages m and all key pairs $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$, we have $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m$.

Property 1.3: Semantic Security (IND-CPA) [GOLDWASSER1984270]

An adversary who sees a ciphertext learns nothing about the underlying plaintext beyond what they already knew. Formally, for any two messages m_0, m_1 of equal length chosen by a probabilistic polynomial-time adversary, if the adversary is given the encryption of one of them (chosen uniformly at random), they cannot determine which message was encrypted with probability significantly better than random guessing.

Property 1.4: Chosen Ciphertext Attack Security (IND-CCA2) [rackoff1991non]

Even if an adversary can decrypt ciphertexts of their choice (except for the challenge ciphertext), they still cannot break semantic security. This protects against active attacks where adversaries can manipulate ciphertexts and observe the decryption results.

In many cryptographic systems, a person's identity is fundamentally defined by their public key. Rather than using traditional identifiers like names or government-issued IDs, a user's public key becomes their cryptographic identity. Anyone can verify that messages or transactions came from the holder of the corresponding private key without needing to know anything about the person's real-world identity. This creates a pseudonymous system where users are identified by their cryptographic keys rather than personal information.

Digital Signatures

This concept of cryptographic identity leads naturally to digital signatures. In the distributed setting we are working in – where many people send each other postcards – it is crucial to have the ability to sign one's postcard. Without this ability anyone can send anyone else anything claiming to be just about anybody. This is achieved using digital signature schemes.

The idea is as follows: instead of Bob using Alice's public key to encrypt a message and only Alice with her private key can decrypt it; for a digital signature we have Alice that uses her own private key to encrypt, and Bob (and everybody else) can use Alice's public key to decrypt. The goal here is not to hide the information but rather to know its origin. And since nobody could have encrypted the message with Alice's private key except Alice, hence it must have been Alice who issued the message.

Definition 1.5: Digital Signature Scheme

A digital signature scheme consists of three probabilistic polynomial-time algorithms, denoted as KeyGen, Sign, and Verify, such that:

- $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$ is a randomized algorithm that takes a security parameter λ, and outputs the pair of public key pk and private key sk.
- $\mathsf{Sign}(sk, m) \to \sigma$ is a deterministic algorithm taking the private key sk and the to-be-signed message m and outputs a signature σ.
- $\mathsf{Verify}(pk, m, \sigma) \to \{0, 1\}$ is a deterministic algorithm taking the message m, its signature σ along with the public key pk and outputs 1 if σ is a valid signature for m under pk, or 0 otherwise.

A secure digital signature scheme must satisfy the following properties:

Property 1.6: Correctness

If $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and $\sigma \leftarrow \mathsf{Sign}(sk, m)$ for a given message m, then $\mathsf{Verify}(pk, m, \sigma) = 1$.

Property 1.7: Existential Unforgeability under Chosen Message Attack (EUF-CMA)

A digital signature scheme is secure if it is computationally infeasible for an adversary to forge a valid signature on any message, even after seeing signatures on messages of the adversary's choice.

The mathematical implementation works as follows: Alice generates a signature by using the Sign algorithm with her private key sk and the message m she wants to sign, producing $\sigma = \mathsf{Sign}(sk, m)$. She then publishes both the message m and signature σ along with her public key pk. Anyone can verify that Alice signed this specific message by running the Verify algorithm: $\mathsf{Verify}(pk, m, \sigma)$ returns 1 if the signature is valid and 0 otherwise. Crucially, computing a valid signature requires knowledge of Alice's private key, but verification only requires her public key, which can be shared publicly.

The security of digital signatures ensures that even if an adversary sees many valid message-signature pairs from Alice, they cannot forge Alice's signature on any new message. This property, called existential unforgeability under chosen message attack (EUF-CMA), guarantees

that signatures serve as unforgeable digital seals that authenticate both the sender and the integrity of the message.

1.2.2 Hash Functions

Cryptographic hash functions are fundamental primitives in modern cryptography. Informally, a hash function is a mathematical transformation that takes an input of arbitrary length and produces a fixed-length output called a *digest* or *hash value*. One may think of a hash function as a kind of "digital fingerprinting machine": no matter how large or small the input, the output always has the same length, and even the tiniest change in input radically alters the resulting fingerprint.

Everyday Intuition. To motivate the idea, consider a library that wishes to uniquely identify every book in its collection. Instead of recording every word of every book, the library assigns each book a short code computed by some deterministic procedure. This code is far smaller than the original book, yet uniquely represents it within the collection. Cryptographic hash functions serve an analogous purpose in computer science: they compress data of arbitrary size into fixed-size values that are practically unique to each input.

Definition 1.8: Hash Function [Menezes1996, Stinson2019]

A hash function is a deterministic algorithm

$$H : \{0,1\}^* \to \{0,1\}^n$$

that maps an input string of arbitrary length to a fixed-length output string of n bits.

Not all hash functions are suitable for cryptographic use. A *cryptographic hash function* must satisfy certain security properties that make it resistant to adversarial manipulation. The following three properties are standard [**Menezes1996**, **Bellare2005**, **Stinson2019**]:

Property 1.9: Preimage Resistance

Given a hash value $y \in \{0,1\}^n$, it is computationally infeasible to find any input $x \in \{0,1\}^*$ such that $H(x) = y$.

Property 1.10: Second Preimage Resistance

Given an input $x \in \{0,1\}^*$, it is computationally infeasible to find another input $x' \neq x$ such that $H(x) = H(x')$.

Property 1.11: Collision Resistance

It is computationally infeasible to find any pair of distinct inputs $x, x' \in \{0,1\}^*$ such that $H(x) = H(x')$.

These properties ensure that hash functions act as one-way, collision-free fingerprinting mechanisms. While collisions must exist in principle (because the input domain is infinite while the output range is finite), the requirement is that such collisions cannot be feasibly found with current computational resources.

1.3 Money

What is money? This apparently simple question has provoked centuries of debate across economics, politics, and–more recently–computer science. Although different schools of thought disagree on how money should be managed, they generally agree on what money is and what functions it performs. We first introduce the classical *monetary view*, then consider a computational restatement of it, and finally contrast the Keynesian and libertarian visions.

1.3.1 Money

The terms *money* and *currency* are often used interchangeably but are not strictly synonymous. *Money* refers to any asset that fulfills the functions of medium of exchange, store of value, unit of account, and standard of deferred payment [**jevons1875money**, **hicks1967critical**, **mankiw2018principles**]. *Currency*, by contrast, usually denotes the institutionalized form of money that circulates in daily life–banknotes, coins, or digital balances denominated in state-issued units. In other words, all currencies are money, but not all forms of money are currencies. Gold bars, for instance, can be money but are not typically currency in modern economies.

Importantly, money cannot be defined by government decree alone. As Carl Menger argued in his seminal essay on the origin of money [**menger1892origin**], money emerges spontaneously through the practices of individuals who adopt certain goods as media of exchange. A government may declare a token as *legal tender*, but unless people actually use it to conduct exchanges, it will not function as money. Conversely, history shows many cases where commodities or instruments became money long before governments recognized them [**graeber2011debt**].

The Monetary View

The classical view defines money by its functions rather than by its material form. This framework goes back to Jevons' *Money and the Mechanism of Exchange* [**jevons1875money**], refined by Hicks [**hicks1967critical**] and widely adopted in modern treatments such as Mankiw [**mankiw2018principles**].

Definition 1.12: Money

An asset qualifies as money if it fulfills the following four functions:

1. *Medium of exchange:* it facilitates trade by eliminating the need for barter;
2. *Store of value:* it preserves purchasing power over time;
3. *Unit of account:* it provides a common standard to measure heterogeneous goods;
4. *Standard of deferred payment:* it enables obligations to be denominated consistently across time.

Historically, money has taken diverse forms. Commodity monies such as salt, cattle, and shells are documented across early civilizations [**davies1994history**]. Gold and silver coinage dominated in the ancient and medieval world, where their intrinsic value limited inflation [**sargent2002smallchar**. Representative money, such as gold-backed banknotes, appeared in the nineteenth century [**eichengreen1996glob**. The Bretton Woods system (1944–1971) pegged currencies to the U.S. dollar, itself convertible into gold [**bordo1993bretton**]. Today's fiat currencies, from dollars to euros, lack intrinsic backing but retain their role through state authority and collective trust [**goodhart1998twoconcepts**].

The Computational Restatement

In the digital age, technologists have reframed this functional definition in informational terms. Elon Musk, for instance, has described money as a *database for resource allocation across time and space*. The analogy is instructive: each classical function corresponds to an operation in an abstract record-keeping system.

- *Medium of exchange:* every transaction is an update to the record. When Alice pays Bob, Alice's balance decreases and Bob's increases; the ledger (whether mental, paper, or digital) must reflect this change.
- *Store of value:* balances persist across time, much like durable records that can be inspected later.
- *Unit of account:* the record enforces a standardized format: all entries are denominated in the same unit, making values comparable.
- *Standard of deferred payment:* the record can register obligations to be settled later, such as debts or contracts, which remain as outstanding claims until they are updated.

Crucially, this computational restatement is not restricted to digital systems. Clay tablets in Mesopotamia already recorded debts and rations in the third millennium BC [**powell1996mesopotamia**, **hudson2004origins**]. Tally sticks in medieval England served as transferable debt instruments for centuries [**davies1994history**, **martin2013money**]. Even sealed jars of coins found in archaeological contexts represent primitive databases of stored value and claims [**peacock2011archaeology**]. In this sense, the computational view is a restatement of the monetary view in informational language, not a radical departure from it.

The Keynesian Vision

John Maynard Keynes introduced a radically different emphasis in the *General Theory of Employment, Interest and Money* [**Keynes**]. While agreeing on money's functions, Keynes stressed its macroeconomic role. Agents hold money not only for transactions but also out of *liquidity preference*: the desire to retain liquid assets for precaution or speculation.

From this perspective, money is not neutral. Variations in money supply and interest rates directly affect investment, employment, and output. Governments therefore must manage money actively. By expanding the money supply in recessions or contracting it during booms, central banks can stabilize aggregate demand and smooth business cycles. Without such management, Keynesians argue, economies risk unemployment and instability.

The Libertarian Vision

Classical liberal and libertarian economists accept the functions of money but reject the Keynesian prescription for active management. Milton Friedman, in *The Optimum Quantity of Money* [**friedman1969optimum**], proposed that discretionary monetary policy be replaced by simple rules, such as a constant growth rate of the money supply. In his view, central banks cannot outperform markets in predicting the future.

Friedrich Hayek went further in *Denationalisation of Money* [**hayek1976denationalisation**], arguing for a system of competing private currencies. Just as competition disciplines producers of goods, it would discipline issuers of money. Currencies that lost credibility through inflation or mismanagement would simply fall out of use.

In this vision, the danger lies not in leaving money to the market, but in entrusting it to political authorities. Inflation, devaluation, and monetary repression are viewed as consequences of state monopoly.

Synthesis

All schools of thought recognize the same core functions of money. The differences lie in governance. Keynesians see money as a policy instrument, to be managed for stability and employment. Libertarians see money as a spontaneous order, best disciplined by rules or competition. The computational view reframes the classical definition in terms of record-keeping, showing that the same functions can be implemented in clay, paper, or code.

With these perspectives in mind, we now turn to specific monetary designs, beginning with demurrage.

1.3.2 Demurrage

The concept of *demurrage* denotes the systematic depreciation of money balances through time.

Definition 1.13: Demurrage

Let $\delta \in (0,1)$ denote a periodic depreciation rate. A currency is said to be subject to *demurrage* if any monetary balance M_0 evolves according to

$$M_t = M_0(1-\delta)^t,$$

so that its value decreases over time unless actively circulated.

The notion is most closely associated with Silvio Gesell, who in *The Natural Economic Order* (1916) [**Gesell**] argued that money enjoys an artificial advantage over perishable goods: while commodities decay, money can be hoarded indefinitely. By attaching a carrying cost to money itself, Gesell sought to neutralize this asymmetry and foster a continuous flow of exchange. Keynes, though disagreeing with Gesell on many points, described him as an "unduly neglected prophet" in the *General Theory* (1936) [**Keynes**], acknowledging that demurrage provides a theoretical means of overcoming liquidity traps.

Experiments with demurrage have been historically limited but revealing. During the Great Depression, several communities issued "stamp scrip," banknotes that required the periodic purchase of adhesive stamps to retain validity. The best-known case was Wörgl, Austria (1932–33), where observers reported that the local scrip circulated rapidly, financing public works and stimulating trade. Yet such experiments were short-lived: national governments suppressed them, citing concerns over monetary sovereignty and the administrative burden of maintaining stamped currency. Later regional currencies such as the Chiemgauer in Germany incorporated demurrage in milder form [**lietaer2001**], while in the digital era projects like Freicoin explored how cryptographic protocols could enforce similar principles automatically [**freicoin**].

The intellectual appeal of demurrage lies in its challenge to the traditional functions of money. Whereas the classical definition emphasizes money's role as a store of value, demurrage reconfigures it primarily as a medium of exchange. By penalizing idle balances, it raises the relative cost of hoarding and channels purchasing power back into circulation. Advocates interpret this as a means to stabilize demand in downturns and to prevent speculative holding of currency. At the same time, the very mechanism that ensures circulation undermines one of money's defining roles: its capacity to preserve value across time. From this perspective, demurrage exposes a fundamental tension between liquidity and durability in monetary design.

The practical obstacles are equally significant. In physical form, demurrage has historically been cumbersome to administer–stamped notes required constant oversight, and public acceptance was fragile. In digital environments, however, the technical barriers largely vanish: a central bank digital currency could in principle deduct holding fees automatically, and cryptographic currencies can encode demurrage directly into their protocol rules. Yet implementability does not resolve the deeper questions of acceptability and legitimacy. For many users, a currency that predictably loses value is counterintuitive and politically controversial. The debate therefore continues to oscillate between its theoretical elegance and its practical limitations.

Demurrage thus illustrates how altering the temporal properties of money can reshape its economic function. It remains an unconventional and contested design, historically marginal yet periodically rediscovered in times of monetary experimentation. In the age of programmable money, its feasibility is greater than ever, but so are the questions it raises about the purposes that money ought to serve.

1.3.3 Digital Currency

Before turning to digital currencies, it is instructive to recall the distinctive features of physical cash.

Cash and Its Properties

Banknotes and coins are *bearer instruments*: ownership is established simply by physical possession, and transfer requires nothing more than handing them over. Several properties follow directly from this design:

- **Peer-to-peer immediacy.** Cash transactions involve only the parties present. No intermediary is required to approve or route the payment [**Goodell2021**].
- **Finality.** Once handed over, cash cannot be reversed electronically; the settlement is immediate and irreversible [**Goodell2021**].
- **Anonymity.** Cash payments leave no intrinsic ledger trail; unless a party voluntarily discloses them, external observers cannot reconstruct the transaction graph. In this sense, cash is untraceable by default [**HullSattath2021**, **Berg2020**].
- **Offline functionality.** Cash does not depend on electricity, networks, or digital infrastructure, which enables universal accessibility [**Goodell2021**].
- **Fungibility.** All units of the same denomination are interchangeable; a €10 note is accepted regardless of its serial number or history [**HullSattath2021**, **Berg2020**].

These properties make cash a natural benchmark for monetary design, see Figure 1.1. By contrast, conventional digital payment schemes–such as bank transfers, credit cards, or mobile wallets–are inherently mediated systems. They rely on intermediaries who maintain transaction logs, enforce compliance, and often retain discretionary control over reversals or censorship. See Figure 1.2 where the entity labeled "bank" should be understood as a simplification: in practice, this role may be fulfilled by a network of financial institutions, including correspondent banks, payment processors, and clearing houses, each of which must authorize and relay the transaction. While this architecture provides oversight and fraud prevention, it also concentrates power in a small number of actors who can unilaterally deny service, freeze accounts, or restrict capital flows.

The complexity of this structure becomes most visible in cross-border transactions. An international wire transfer, for example, typically involves multiple correspondent banks, each applying its own compliance checks, fees, and settlement procedures. Routing a payment from a European bank to an Asian merchant may require traversing several jurisdictions, exposing the transaction to differing legal frameworks, anti-money-laundering controls, and currency conversion steps. As a consequence, international transfers that should be near-instant in a purely digital medium can take several days to settle, with costs that accumulate at each stage. This illustrates the tension between the technical possibility of instantaneous digital communication and the institutional reality of centralized monetary infrastructures.

The challenge of digital currency design is therefore not only technical but also architectural: how to approximate the desirable properties of cash–finality, anonymity, fungibility–in a medium that avoids the bottlenecks and vulnerabilities of centralized intermediaries. This question motivates the emergence of electronic cash (e-cash), which seeks to recreate the peer-to-peer qualities of cash within a cryptographic framework.

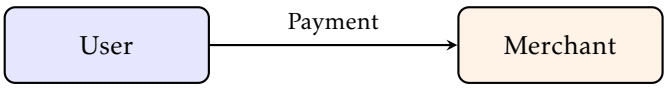


Figure 1.1: Cash Payment System

Figure 1.2: Simplified Digital Payment

e-Cash

One of the earliest and most influential attempts to replicate the properties of physical cash in a purely digital form was David Chaum's proposal of *Untraceable Electronic Cash* (often abbreviated as *e-cash*) [**chaum1983blind**, **e-cash**]. To understand the significance of this idea, it is helpful to recall the key characteristics of physical banknotes and coins: they are easy to transfer from hand to hand, they do not require the presence of a central authority for every transaction, and – most importantly – they preserve a strong form of privacy. When Alice hands a $20 bill to Bob, no external entity (such as a bank or government) automatically learns of this exchange, and neither Alice nor Bob needs to seek permission for the transaction to be valid.

Chaum's e-cash sought to preserve these same properties in the digital world. His central innovation was the use of *blind signatures*, a cryptographic primitive that allows a bank to "sign" a digital coin for a user without learning the coin's exact content. One may think of blind signatures as the digital analogue of placing a piece of paper inside a carbon-lined envelope: the bank stamps the outside of the envelope with its seal, and when the user later removes the paper, it carries a valid signature – yet the bank never saw what was written on it. This ingenious idea allowed users to withdraw anonymous "digital banknotes" from their bank and later spend them with merchants, all without the bank being able to link a withdrawal to a particular payment. e-cash is illustrated in Figure 1.3.

In practice, Chaum's protocol worked as follows. A user generated a random serial number for a digital coin, "blinded" it using a mathematical transformation, and sent it to the bank for signing. The bank verified that the user had sufficient funds, signed the blinded value, and

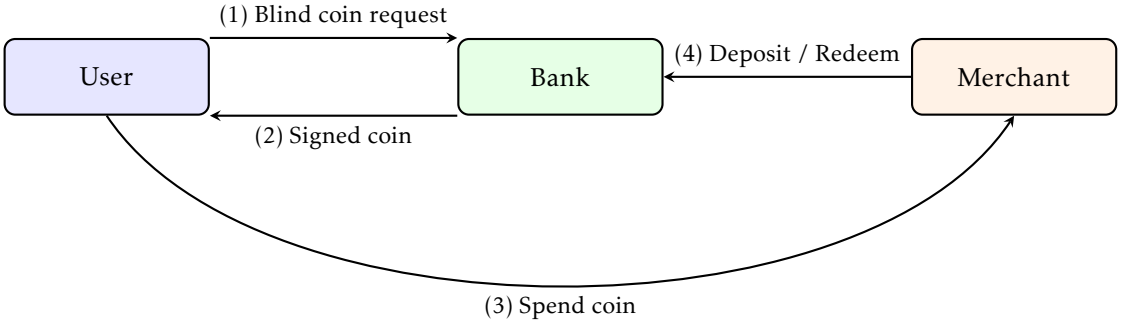


Figure 1.3: Chronological flow of Chaum's e-cash protocol: (1) the user requests a blind signature on a coin, (2) the bank returns the signed coin, (3) the user spends it with a merchant, and (4) the merchant redeems it at the bank.

deducted the amount from the user's account. Once unblinded, the signed coin became valid electronic cash that could be given to a merchant, who could then deposit it at the bank for redemption. The bank could check the coin's validity (by verifying its signature and ensuring the serial number had not been spent before) but could not determine which customer had originally withdrawn it.

This scheme successfully captured the *untraceability* of cash in a digital setting. However, it had two important limitations:

1. It relied on a *trusted central authority* (the bank) to issue and redeem coins, which meant that users still had to place absolute trust in a single entity. Unlike physical cash that can circulate independently once issued, Chaum's e-cash always required the bank's cooperation to validate transactions.

2. Its very strength – the anonymity of payments – was perceived as a weakness by regulators and financial institutions. From their perspective, an untraceable payment system could facilitate money laundering or tax evasion.

To use an analogy, Chaum's e-cash was like a digital version of casino chips: they could be spent anonymously inside the casino (between players and dealers), but ultimately, they had to be cashed out at the cashier's desk. The cashier (the bank) thus remained the bottleneck and the point of trust.

While Chaum's proposal was a milestone in cryptographic research and inspired a long line of anonymous payment systems, it failed to gain widespread adoption. The reliance on a central authority conflicted with the desire for decentralization, and the strong anonymity conflicted with financial regulation. These tensions partly explain why e-cash systems remained mostly academic prototypes throughout the 1980s and 1990s.

The paradigm shifted dramatically with the introduction of blockchains with Bitcoin in 2008 [**Bitcoin**], which abandoned the requirement of a central issuing bank altogether. Instead, Bitcoin uses a distributed consensus protocol to track ownership and prevent double-spending, while offering pseudonymity rather than full untraceability. In this sense, Bitcoin can be viewed as both a continuation and a departure from Chaum's vision: it realized the dream of digital cash but replaced blind signatures and central issuance with cryptographic proofs and decentralized consensus.

1.4 Consensus

Blockchain technology solves the postcard consensus problem. First we formally define what consensus is. Then we introduce an algorithm to solve a weaker form of the postcard consensus problem.

1.4.1 Consensus

The postcard consensus problem is an analogy for a computer network consensus. Instead of people, we talk about *nodes* achieving *consensus* in the *message passing model* [**Wattenhofer2019**].

Definition 1.14: Node

We call a single actor in the system *node*. In the postcard scenario the people are the nodes. In a computer network the computers are the nodes. In the classical client-server model both the server and the client are nodes. And so on. If not stated otherwise, the total number of nodes in the system is n.

Definition 1.15: Consensus

There are n nodes, of which at most f might crash, *i.e.*, at least $n - f$ are correct. Node i starts with an input value v_i. The nodes must decide for one of those values, satisfying the following properties:

- **Agreement:** All correct nodes decide for the same value.
- **Termination:** All correct nodes terminate in finite time.
- **Validity:** The decision value must be the input value of a *node*.

Definition 1.16: Message Passing Model

In the *message passing model* we study distributed systems that consist of a set of nodes. Each node can perform local computations, and can send messages to every other node.

1.4.2 Practical Byzantine Fault Tolerance (pBFT)

The Byzantine Generals Problem, introduced by Lamport, Shostak, and Pease [**byzantine_generals**], captures the difficulty of reaching agreement in a distributed setting where some participants may behave arbitrarily or maliciously. Imagine generals of a Byzantine army who must decide whether to attack or retreat, but where some generals may be traitors who send contradictory messages. The loyal generals must nevertheless reach a common decision. This metaphor illustrates the essence of *Byzantine faults*: arbitrary deviations from the protocol, including lying, collusion, and inconsistency.

System Model. Practical Byzantine Fault Tolerance (pBFT), due to Castro and Liskov [**pbft**], solves this problem in real networks. We consider a set R of n replicas (or participants), among which at most f may be Byzantine. Consensus is possible provided that

$$n \geq 3f + 1$$

One replica is designated as the *primary* (or leader), while the others act as *backups*. A particular assignment of primary and backups is called a *view*.

Protocol Overview. The protocol proceeds in three stages, amplifying agreement at each step:

1. **Pre-prepare.** The primary proposes a value to be decided upon and broadcasts it with a sequence number to all backups.
2. **Prepare.** Each backup checks the proposal. If valid, it echoes it to all others. When a replica collects $2f+1$ matching proposals, it knows that a supermajority has seen the same value.
3. **Commit.** Replicas then confirm their readiness to act on the proposal by broadcasting a commit message. Once $2f+1$ commits are received, the value is accepted as the decision.

Why a Commit Stage? One might wonder: if $2f+1$ replicas already agreed during the prepare stage, why not stop there? The reason is subtle. Two different groups of $2f+1$ prepares could overlap in only f participants, and if those overlapping participants are Byzantine, they might equivocate and cause conflicting outcomes. The commit stage guarantees that there is always at least one honest participant in the overlap of any two decision sets, ensuring consistency across the system.

Postcard Analogy. Suppose four friends must agree on which postcard to paste into their album, knowing that one of them might be mischievous. The process unfolds as follows:

- The leader friend (primary) proposes a card and shows it to the group (pre-prepare).
- Each honest friend copies this proposal and circulates it among the others (prepare). If at least three friends agree on the same card, they appear ready to proceed.
- To be absolutely sure, each friend then sends a "commit" postcard stating, "I am ready to glue this card into the album." Only when two friends receive at least two such commit postcards do they actually glue the card.

This extra step ensures that no subgroup can be tricked into gluing a different card. The album remains consistent, even if one friend acts maliciously.

Discussion. pBFT thus guarantees both agreement and termination under Byzantine faults, provided $n \geq 3f+1$ and the network is only *partially synchronous*: message delays can be arbitrary but are eventually bounded. These properties make pBFT particularly attractive for systems with a modest and fixed number of participants that require strong consistency. Its quadratic communication overhead, however, limits scalability in very large open networks.

1.5 Blockchains

The paradigm of blockchains emerged with the publication of the Bitcoin whitepaper by the pseudonymous Satoshi Nakamoto in 2008 [**Bitcoin**]. Bitcoin represented the first practical system to achieve consensus in an open, permissionless setting, *i.e.*, among a potentially unbounded set of participants who may join or leave at will and without prior authentication. Earlier consensus protocols such as Paxos or practical Byzantine Fault Tolerance (pBFT) presupposed a

fixed and known set of participants. Bitcoin, by contrast, resolved the long-standing problem of *achieving agreement in an adversarial network without central coordination*, introducing a system that simultaneously functioned as a digital currency and as a trustless consensus protocol.

At a high level, the Bitcoin system maintains a distributed ledger of transactions – called the *blockchain*. Each transaction represents the transfer of ownership of some units of currency (bitcoins) from one participant to another, authenticated by digital signatures. Transactions are grouped into *blocks*, and blocks are linked together sequentially in a chain by means of cryptographic hash pointers. This construction guarantees that altering any past block would invalidate all subsequent blocks, thereby ensuring immutability.

The crucial innovation lies in the way Bitcoin reaches consensus on which block should be added next. Nakamoto introduced the *Proof-of-Work* (PoW) mechanism: participants, called *miners*, repeatedly evaluate a cryptographic hash function until they find an output below a global difficulty target. This costly computation produces a block header that proves the miner expended computational effort. The protocol stipulates that the valid blockchain is the longest (or more precisely, the heaviest) chain of valid blocks, thus giving the network a simple total ordering rule. Nodes adopt whichever chain has accumulated the most Proof-of-Work, which probabilistically guarantees convergence to a single history.

Formally, one may view a blockchain as a distributed state machine replicated across a network of nodes. A transaction represents a request to update the state. Consensus ensures that all honest nodes apply the same sequence of state transitions, even in the presence of adversaries.

Definition 1.17: Blockchain Ledger

Let $\mathcal{T}$ denote the set of all valid transactions. A blockchain ledger is a sequence of blocks $(B_0, B_1, \ldots, B_m)$ such that:

1. Each block B_i contains a finite ordered list of transactions $(t_1, \ldots, t_k) \subseteq \mathcal{T}$.
2. Each block header includes a cryptographic hash of B_{i-1}, thereby forming a hash chain.
3. Every transaction appears in at most one block of the ledger.

Definition 1.18: Consensus via Nakamoto Proof-of-Work

Let $\mathcal{C}$ be the set of all valid candidate chains. Each node adopts a chain $C \in \mathcal{C}$ according to the following rule:

$$C^* = \arg\max_{C \in \mathcal{C}} \mathsf{Work}(C),$$

where $\mathsf{Work}(C)$ denotes the total computational difficulty accumulated in C. Consensus is achieved when all honest nodes select the same C^* with overwhelming probability.

This framework provides two essential guarantees, often restated in terms of consensus:

- *Agreement (safety)*: All honest nodes adopt the same blockchain prefix.
- *Termination (liveness)*: New transactions from honest participants are eventually included in the chain.

In this sense, Bitcoin simultaneously introduced (i) a decentralized currency, and (ii) a general method to achieve consensus in open networks. The blockchain data structure, combined with PoW-based consensus, inaugurated a new field of research and applications far beyond digital money.

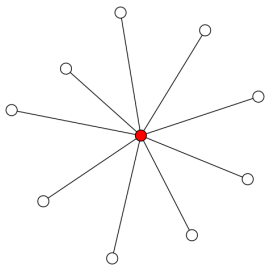
From a broader perspective, Vitalik Buterin later captured the essence of the blockchain idea in more general terms [**BC_value**]:

> "A blockchain is a magic computer that anyone can upload programs to and leave the programs to self-execute, where the current and all previous states of every program are always publicly visible, and which carries a very strong cryptoeconomically secured guarantee that programs running on the chain will continue to execute in exactly the way that the blockchain protocol specifies."

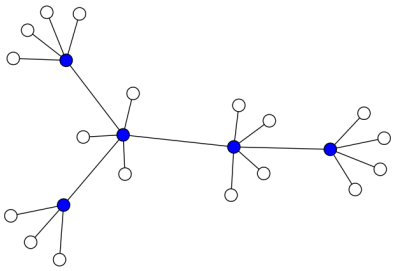
Buterin stresses that this definition deliberately avoids financially-charged terms like *ledger* or *transactions*, and does not privilege any specific consensus mechanism. Instead, the emphasis is on the blockchain as a *general-purpose cryptoeconomic infrastructure*: decentralized, secured by cryptography and incentives, and providing reliable execution of arbitrary state transition functions.

1.5.1 (De)centralization

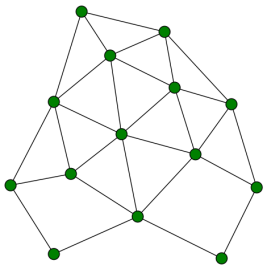
The terms *centralized*, *decentralized*, and *distributed* are widely used in the blockchain literature and beyond. They are often illustrated with diagrams such as those in Figure 1.4, adapted from [**baran1964distributed**]. While these images are instructive, they can also be misleading if taken too literally. In fact, Figures 1.4b and 1.4c are often inverted on the internet. Moreover, as Baran himself emphasized, the terminology is not standardized and should be understood as describing points along a *spectrum* rather than discrete categories.



(a) Centralized.



(b) Decentralized.



(c) Distributed.

Figure 1.4: (De)centralization Spectrum.

Intuitive motivation. To gain intuition, think of a network of people connected by communication links. In a *centralized* network, everyone communicates through a single hub: if the hub

fails, the entire system collapses. In a *distributed* network, everyone is directly connected to everyone else: no single failure can disconnect the group. Between these extremes lie *decentralized* networks, where there are multiple hubs or partial redundancies: the system can tolerate some failures, but not all.

From this perspective, decentralization is not an all-or-nothing property but a matter of degree: how many failures can the network withstand before breaking apart?

Graph-theoretic formalization. We can make this intuition precise using graph theory. Consider a network as a graph $G = (V, E)$, where the vertices V represent entities (computers, people, or organizations) and the edges E represent communication links.

Definition 1.19: Vertex Cut

vertex cut (or separating set) of a connected graph G is a set of vertices whose removal disconnects G, i.e., makes it impossible for some vertices to reach others.

Definition 1.20: Vertex Connectivity

The *vertex connectivity* $\kappa(G)$ of a non-complete graph G is the size of its smallest vertex cut. Equivalently, it is the minimum number of vertices that must be removed to disconnect the graph. A graph is said to be *k-vertex-connected* (or simply *k-connected*) if $\kappa(G) \geq k$.

Intuitively, $\kappa(G)$ measures the network's robustness to node failures. A large value of $\kappa(G)$ indicates that the system can lose many participants without fragmenting. At the extreme, the complete graph on n vertices, denoted K_n, has $\kappa(K_n) = n - 1$, because one would need to remove all but one node to disconnect it.

Beyond graph structure. While this graph-theoretic perspective helps quantify architectural robustness, it captures only one dimension of (de)centralization: the physical pattern of interconnections. As Vitalik Buterin [**Vitalik_Decentralization**] points out, decentralization has at least three distinct aspects:

- *Architectural (de)centralization*, concerning the network's topology and fault tolerance, as formalized above.
- *Political (de)centralization*, concerning who holds decision-making power. A system may be architecturally decentralized but politically centralized if a single actor controls most nodes.
- *Logical (de)centralization*, concerning whether the system behaves as one monolithic computer or as multiple independent subsystems.

These three axes are independent. A system may be decentralized in one dimension but centralized in another. Thinking of decentralization in this multidimensional way helps avoid oversimplification and highlights the trade-offs involved in system design.

These three types of decentralization are independent. They can be perceived as the axes of a three-dimensional space. The notion of logical centralization is subtle. A system is said to be *logically centralized* if, by design, it maintains a single authoritative state that all participants agree upon. Conversely, it is *logically decentralized* if it naturally supports multiple independent states that can evolve separately.

Illustrative examples. With this clarification, we can revisit common examples across the three axes of decentralization. For simplicity, we classify each example in binary terms (centralized vs. decentralized) along each axis.

- **Languages.**
  - Architecturally decentralized: anyone can communicate with anyone else.
  - Politically decentralized: no single authority controls how people speak.
  - Logically decentralized: languages diverge naturally. If humanity were split in two, each group would develop its own variant.
- **Corporations.**
  - Architecturally centralized: decisions often must pass through specific hierarchical nodes (e.g., managers, the CEO).
  - Politically centralized: ultimate decision power is concentrated in executives or shareholders.
  - Logically centralized: the corporation functions as one unified entity; splitting it in half does not produce two valid corporations, but two new organizations.
- **BitTorrent.**
  - Architecturally decentralized: peers can connect freely.
  - Politically decentralized: no central actor controls the network.
  - Logically decentralized: each swarm (per file) is independent; different groups can share different files without needing global coordination.
- **Blockchains.**
  - Architecturally decentralized: any node can communicate with any other.
  - Politically decentralized: no single entity controls who can participate.
  - Logically centralized: by design, there is only one authoritative blockchain state. All nodes replicate the same state. Forks (see Section 1.5.2) are not intended as independent systems but as temporary disagreements to be resolved.

Synthesis. These examples highlight that decentralization is multidimensional. A system may be architecturally decentralized but politically centralized, or politically decentralized but logically centralized. Thinking in terms of these three axes helps prevent oversimplified claims that a system is "decentralized" or "centralized" without qualification.

Notes on Bribing and Collusion. Bribing is a form of collusion. It can be done in a very discreet manner, especially in the blockchain space [**Vitalik_Collusion**]. It's not a bribe, it's a "staking pool" that "shares dividends". More obfuscated bribes exist too. A cryptocurrency exchange can very easily offer good services and low prices to its clients while using client's deposited coins to participate in various coin-voting systems. Detecting or stopping bribing is a difficult case to solve.

1.5.2 Governance

Consensus mechanisms ensure that a network of participants can agree on a common sequence of transactions. Yet beyond this lies a more fundamental meta-question: who decides how the rules of the system themselves should evolve? In traditional societies, constitutions and legal frameworks provide explicit processes for modifying laws and institutions. In decentralized blockchain systems, by contrast, there is no single authority to dictate upgrades or resolve disputes. This problem is generally referred to as *blockchain governance*.

Definition 1.21: Blockchain Governance

Blockchain governance is the collection of social, economic, and technical processes by which a decentralized community coordinates decisions about the evolution, parameters, and maintenance of its protocol.

Introduction to Governance. Governance can thus be understood as *consensus about consensus*. Whereas consensus protocols decide on the next block to be added to the chain, governance processes decide how the protocol itself changes over time. One may think of this as the distinction between writing entries in a ledger and deciding on whether to change the ledger's format, add new columns, or replace it entirely. By analogy, if consensus is the day-to-day activity of writing laws and applying them, governance is the constitutional framework that prescribes how laws may be created, amended, or repealed. Different blockchains embody different constitutional traditions, ranging from informal deliberation, as in Bitcoin, to highly codified on-chain voting, as in Tezos [**goodman2014tezos**].

Models of Governance. Two broad models of governance are usually distinguished. In *off-chain governance*, decisions emerge from human deliberation, precedent, and informal coordination. Developers propose changes–for instance through Bitcoin Improvement Proposals (BIPs) [**bip2**]–which are then debated on mailing lists, forums, and conferences. Miners and users ultimately signal their approval or rejection by adopting the corresponding software updates. Bitcoin's upgrade process is the paradigmatic case of such off-chain governance [**narayanan2016bitcoin**]. By contrast, *on-chain governance* embeds decision-making procedures directly into the protocol itself. Token holders may propose, debate, and vote on upgrades, with outcomes automatically enforced by smart contracts. The design of Tezos illustrates this approach: amendments are submitted to token holders, approved through formalized voting cycles, and, if accepted, integrated into the protocol without requiring any informal agreement [**goodman2014tezos**].

Stakeholders and Power. The governance process involves a heterogeneous set of stakeholders, whose incentives and powers are rarely aligned. Core developers often enjoy disproportionate influence, since they design and maintain the protocol code, yet their authority depends on community trust. Miners or validators play a decisive role by choosing which software version to run; their economic incentives, shaped by block rewards and transaction fees, affect their stance toward upgrades [**bitcoin_backbone**]. Token holders, in systems with explicit voting rights, exercise formal authority, while in other systems they express preferences indirectly through markets and delegation. Node operators, even without voting rights, exert influence by determining which chain to follow in case of disagreement. Application developers, whose software depends on protocol stability, lobby for or against changes that may affect compatibility. Finally, end users have the ultimate power of adoption: unpopular forks or contentious upgrades

can simply fail if the majority of users refuse to migrate. Governance, in practice, emerges from a negotiation among all these actors, with no single group enjoying unqualified sovereignty.

Upgrades and Forks. Moments of governance are most visible when a protocol is upgraded. Minor changes, such as adjusting block size or gas limits, can be introduced gradually. More dramatic changes are expressed through *forks*.

Definition 1.22: Fork

A fork occurs when nodes disagree on the protocol rules, producing two chains that share a common history but diverge thereafter [**gao2019fork**].

Forks can be classified according to their compatibility. A *soft fork* tightens the rules while preserving backward compatibility: nodes that do not upgrade still recognize the new blocks as valid, though they may not be able to use all new features. The activation of Segregated Witness (SegWit) in Bitcoin is a well-known example. A *hard fork*, by contrast, introduces new rules that are incompatible with older versions of the software. Non-upgraded nodes reject new blocks, leading to a permanent divergence of the chain. The Ethereum/Ethereum Classic split following the DAO hack is the paradigmatic illustration of a hard fork and its social, economic, and political consequences [**atzei2017dao**]. By analogy with constitutional law, a soft fork resembles an amendment to an existing constitution, whereas a hard fork resembles secession or revolution: the community divides into two sovereign orders, each enforcing its own incompatible rules.

Incentives and Sanctions. The sustainability of governance depends on incentive alignment. Proof-of-Stake systems, for example, incorporate explicit sanctions such as slashing, which penalize validators who equivocate or remain offline [**buterin2017casper**]. Token-based rewards encourage participation in governance, while reputation systems [**blind_trustless**] attempt to reward long-term contributors. Yet each of these mechanisms carries risks: token-based voting may lead to plutocracy, slashing may discourage risk-taking [**buterin2017casper**], and reputation systems remain vulnerable to manipulation and Sybil attacks [**douceur2002sybil**].

Challenges and Open Questions. Blockchain governance faces persistent challenges reminiscent of those encountered in political theory. Scalability is a major concern: large communities risk paralysis, and tiered or delegated structures attempt to balance inclusivity with efficiency. Wealth concentration raises questions of fairness, leading to proposals such as quadratic voting [**lalley2018quadratic**]. Voter apathy remains endemic, with participation rates often low unless explicit incentives are provided. Regulatory uncertainty further complicates governance, as governance tokens may fall under securities law and different jurisdictions impose conflicting requirements [**sec2019framework**]. Above all, blockchain communities must constantly navigate the tension between innovation and stability: protocols must evolve to remain relevant, yet excessive change can undermine trust and backward compatibility.

Synthesis. In sum, blockchain governance can be seen as a distributed form of constitutional politics, with its own actors, institutions, conflicts, and compromises. Consensus mechanisms secure agreement about blocks, but governance mechanisms secure agreement about rules–and when they fail, the blockchain itself records the fracture through the indelible trace of a fork.

1.6 Manuscript Outline and Contributions

This manuscript is structured around two main contributions, each addressing a different dimension of consensus and decentralization. We conclude with a synthesis of these results. The first contribution made is not addressed in this manuscript [**SignCons**].

Privacy-Preserving Permissioned Blockchains. The first contribution, published in [**SignCons**], develops a generic framework for privacy-preserving private permissioned blockchains. We proposed *SignCons*, a Byzantine Fault Tolerant blockchain inspired by Hyperledger Fabric and the Practical Byzantine Fault Tolerance (pBFT) protocol. A central challenge addressed is the apparent incompatibility between permissioning–which restricts participation to authorized users–and privacy–which requires unlinkability between transactions and their issuers. Our work resolves this tension by introducing several composable constructions providing varying degrees of anonymity and pseudonymity for clients, endorsers, and orderers. These constructions rely on blind signatures, group signatures, and ring signatures (as well as their linkable variants), enabling different trade-offs between privacy, accountability, and efficiency. We proved that all protocols maintain safety and liveness, and formally established their privacy guarantees. This contribution demonstrates that strong privacy can be achieved even in restricted blockchain settings, thus extending the design space for enterprise and consortium blockchains.

Local Cryptocurrencies. In Chapter 2, we focus on the applications of blockchain technology to the design of local currencies, i.e., currencies whose circulation is geographically restricted. While at first sight the idea of a "local cryptocurrency" may appear contradictory, we show how to construct one and argue for its socio-economic benefits, including resilience and community empowerment. These results were published in [**LCoin**].

Human Behavior as Consensus. Chapter 3 addresses the consensus mechanism itself. We investigate how human behavior can be used as a building block for consensus, thereby enabling systems that automatically reward certain classes of desirable actions. This exploration opens the way toward consensus mechanisms that embed social incentives directly in their protocol design. A preliminary version of these results appeared in [**BehaviorChain**].

Finally, Chapter 4 synthesizes these contributions. Taken together, they highlight the interplay between privacy, locality, and human incentives in the broader design of decentralized systems.

2

Local Cryptocurrency

This chapter presents a novel approach to the implementation of local currencies using blockchains, introducing the concept of geographical demurrage as a mechanism for incentivizing localized economic activity. Local currencies, traditionally confined to specific geographical areas to stimulate local economies, have faced numerous challenges in their transition to digital formats. We address these limitations by proposing four distinct blockchain constructions: (i) *Base-Local* for a baseline cryptocurrency that emulates traditional local paper currencies; (ii) *Geo-Limited*, as an enhanced model incorporating geographical restrictions on transactions; (iii) *Geo-Demurrage*, as an advanced design that implements geographical demurrage, where currency value diminishes in function of the distance between points of reception and expenditure; (iv) *Uni-Local*, which maintains geographical demurrage while eliminating strict geographical boundaries. This represents a paradigm shift in local currency design, inherently encouraging localized economic activity without the need for explicit geographical constraints. These contributions have been published in [**LCoin**].

Outline of the current chapter

2.1 Introduction **29**
- 2.1.1 Challenges in Digitalization of Local Currencies . . . 31
- 2.1.2 Contributions . . . 31
- 2.1.3 Related Work . . . 33
- 2.1.4 Outline . . . 35

2.2 Economics **36**
- 2.2.1 Demurrage . . . 36
- 2.2.2 Foundations of Local Currencies . . . 39
- 2.2.3 Geographical Demurrage: A Spatial Approach to Local Currencies . 41

2.3 Distance Bounding **45**

2.4 Local Cryptocurrency **47**
- 2.4.1 Base-Local . . . 49
- 2.4.2 Geo-Limited . . . 50
- 2.4.3 Geo-Demurrage . . . 51
- 2.4.4 Uni-Local . . . 53

2.5 Conclusion **54**

2.1 Introduction

The concept of currency encompasses a diverse array of monetary instruments, each serving distinct economic and social functions. Among these varied forms, local currencies, also known as community or complementary currencies, represent a unique and increasingly significant category [**CC_classifying**]. These specialized monetary instruments are designed to operate within defined geographical areas [**CC_impact**] or communities, embodying a rich historical precedent dating back to the scrip currencies of the Great Depression era. In recent decades, local currencies have experienced a resurgence as tools for fostering local economic resilience and community development.

The theoretical foundations of local currencies draw from diverse economic schools of thought, including Silvio Gesell's work on demurrage and concepts of endogenous money creation [**Gesell**]. At their core, local currencies aim to address several key economic challenges:

1. Monetary Leakage: By encouraging the circulation of wealth within a defined local economy, these currencies seek to mitigate the outflow of monetary resources from a community.

2. Velocity of Money: Local currency systems often incorporate mechanisms to increase the velocity of money circulation, potentially stimulating local economic activity.

3. Community Cohesion: Beyond their economic function, local currencies serve as a social technology, fostering community ties and promoting local identity.

4. Economic Resilience: In times of macroeconomic instability, local currencies can provide a buffer against external economic shocks, offering a degree of local economic autonomy.

The implementation of local currencies has taken various forms, ranging from physical scrip to sophisticated digital systems. Notable examples include the WIR Bank in Switzerland, operational since 1934 [**WIR**], which facilitates a complementary currency system for small and medium-sized enterprises; the Chiemgauer in Germany [**chiemgauer**, **chiemgauer**_2], a regional currency incorporating a demurrage mechanism to encourage rapid circulation; the Bristol Pound in the UK [**bristol**], which has successfully integrated with local government services and digital payment systems; and the Eusko in the Pays Basque in France [**eusko**], the largest local currency in France by volume.

These systems typically are backed by national currency reserves. Their legal status varies significantly across jurisdictions, with some countries explicitly recognizing and regulating local currencies, while others operate in a legal grey area.

The ongoing digitalization of economic systems has precipitated a divergence in approaches to local currency modernization. Some initiatives have embraced digital transformation, while others remain hesitant to transition from traditional formats. Those local currencies that have ventured into the digital realm have predominantly adopted centralized payment system architectures. While this approach facilitates precise tracking and analysis of currency usage, it introduces significant vulnerabilities, including single points of failure, potential infringements on user privacy, and the concentration of control over the digital financial ecosystem.

Recent technological advancements, particularly in distributed ledger technologies, have opened new avenues for the implementation of local currencies. These developments offer potential solutions to longstanding challenges in scalability, security, and interoperability, while also introducing new complexities in terms of regulatory compliance and economic governance. However, the application of blockchain technology to local currencies has been met with reticence from many local institutions, primarily due to the association of cryptocurrencies with illicit activities and concerns over perceived lack of centralized control.

The study of local currencies intersects with broader academic discourses on monetary theory, economic geography, and social capital formation, representing a fertile ground for interdisciplinary research. As we progress into an era of increasing monetary digitalization and potential central bank digital currencies (CBDCs), the role and implementation of local currencies are subject to reevaluation.

In light of these challenges and opportunities, this chapter proposes a comprehensive framework for the implementation of local cryptocurrencies. Central to this framework is the ability to verify locality or distance, a crucial component in maintaining the geographical integrity of local currency systems. To this end, we examine various protocols developed for location and distance verification, which form a critical foundation for our proposed models.

This chapter seeks to contribute to the evolving discourse on local currencies by proposing novel mechanisms for their implementation in the digital age, with a particular focus on leveraging blockchain technology and introducing the concept of geographical demurrage. Through this exploration, we aim to address both the potential benefits and the complex challenges inherent in the digitalization of local currencies, potentially reshaping how communities interact with and benefit from localized economic systems.

2.1.1 Challenges in Digitalization of Local Currencies

The transition of local currencies from physical to digital formats presents significant challenges, particularly concerning digital centralization. This shift introduces a paradigm that, while offering potential benefits, raises critical concerns across various aspects of the currency system.

The technological infrastructure of centralized digital systems is by definition architecturally centralized. It creates single points of failure, making the entire currency vulnerable to technical glitches, cyberattacks, or deliberate manipulation. Unlike physical currencies, where distribution mitigates risk, a centralized digital infrastructure concentrates this risk, potentially jeopardizing the entire local economy. This centralization extends to data control and privacy issues, as these systems necessitate the collection and storage of vast amounts of transaction data and personal information. Such concentration of data not only creates an attractive target for malicious actors but also raises serious privacy concerns, contradicting the often community-centric, trust-based nature of local currencies.

Governance and autonomy present another significant challenge in centralized digital currencies. These systems may concentrate decision-making power in the hands of a few technical experts or administrators, *i.e.*, political centralization, potentially undermining the democratic and participatory ethos that often underpins local currency initiatives. This centralization of authority can alienate community members and contradict the fundamental principles of local economic empowerment. Similarly, in terms of economic design, centralized control over the digital currency's parameters (such as issuance, circulation limits, and exchange rates) could lead to manipulation or mismanagement, further contradicting the goal of local economic autonomy that many local currencies aim to achieve.

The issue of trust and transparency is paramount in digital local currencies. Centralization can obscure the operations of the currency system, potentially eroding trust among users. The lack of transparency in a centralized system contradicts the principles of openness often associated with local currency initiatives, which typically rely on community trust and participation for their success.

Addressing these centralization concerns is crucial for the successful digitalization of local currencies. Solutions should focus on architectural and political decentralization, while enhancing transperancy. The challenge lies in harnessing the benefits of digital technologies while preserving the decentralized, community-centric nature that is fundamental to the philosophy

of local currencies.

2.1.2 Contributions

Blockchain technology presents a compelling solution to the centralization challenges faced by digital local currencies. By providing an architecturally decentralized infrastructure, blockchain eliminates single points of failure, enhancing system resilience and security. The technology enables politically decentralized governance models, allowing for community participation in decision-making, which aligns closely with the democratic ethos of local currencies. Its cryptographic foundations offer robust data privacy while maintaining transaction transparency, striking a balance between openness and user protection. The inherent transparency of blockchain fosters trust by allowing all participants to verify transactions and system operations, while its immutability ensures long-term accountability. These features collectively contribute to increased community engagement, as the system's participatory nature fosters a sense of shared ownership and responsibility. While blockchain implementation comes with its own set of challenges, including technical complexity and the need for community education, it offers a playground for creating digital local currencies that are secure, transparent, and truly community-driven, potentially overcoming many of the centralization issues that threaten the core principles of local economic initiatives.

At the core of our contributions is the introduction of geographical demurrage as a mechanism for incentivizing localized economic activity in digital currencies. This concept represents a significant departure from traditional temporal demurrage systems, offering a more nuanced and flexible tool for currency design. By formalizing the relationship between value depreciation and spatial distance, we provide a theoretical foundation for a new class of location-aware digital currencies.

Our four-tier model progresses from *Base-Local*, a baseline blockchain for cryptocurrency emulating traditional local currencies, to an advanced and boundary-free system. Importantly, these constructions are designed with genericity in mind, allowing for modulable parameters of the blockchain, most prominently the consensus mechanism and its implications for block confirmation time, as well as block size and throughput. This flexibility enables adaptation to various local economic contexts and technological constraints. In our second model, *Geo-Limited*, we introduce a novel approach to enforcing geographical restrictions by utilizing local shops as Certificate Authorities. These shops produce certificates proving that clients wishing to receive funds are indeed within the dedicated area, verifying proximity through distance bounding protocols. The third model, *Geo-Demurrage* introduces the concept of geographical demurrage, where clients pay for transacting over distances – a feature we believe to be the first of its kind in cryptocurrency design. This geographical demurrage serves a dual purpose: it can be used to maintain the institution's working capital and/or incentivize shops to join the network. This progression culminates in the design of *Uni-Local*, a blockchain hosting *LCoin*, a universal local cryptocurrency that maintains the principle of encouraging local spending without imposing strict geographical limitations. Uni-Local represents a paradigm shift in the conceptualization of local currencies, offering a scalable solution that could potentially operate on a global scale while still promoting localized economic activity. Table 2.1 summarizes each construction's characteristics.

To address the technical challenges of implementing these models, we propose a novel application of distance bounding protocols in the context of local cryptocurrencies. By adapting these cryptographic primitives to verify transaction localities, we provide a secure and efficient method for implementing geographical restrictions and demurrage in digital currencies. This integration addresses key security challenges in location-based financial systems.

Table 2.1: Four constructions properties.

	Restricted Area	Geographical Demurrage
Base-Local	X	X
Geo-Limited	✓	X
Geo-Demurrage	✓	✓
Uni-Local	X	✓

Looking towards the future, we identify key areas for further research in the field of digital local currencies. These include the potential integration with central bank digital currencies (CBDCs), the application of machine learning for optimizing demurrage functions, and the exploration of cross-border applications of local currency principles. We also highlight the need for further investigation into privacy-preserving mechanisms for location-aware currencies, addressing a critical concern in the adoption of digital local currencies.

Through these contributions, our work not only advances the theoretical understanding of local currencies in the digital age but also provides practical frameworks for their implementation. By addressing fundamental challenges in the digitalization of local currencies and proposing innovative solutions, this chapter aims to significantly impact the field of monetary design and local economic development. Our holistic approach, combining economic theory, cryptographic techniques, and blockchain technology, offers a robust foundation for the next generation of local currencies, potentially reshaping how communities interact with and benefit from localized economic systems.

2.1.3 Related Work

The integration of blockchain technology into local currency systems has been met with significant challenges and reservations [**blockchain_eusko**]. The primary concerns stem from the inherent vulnerabilities of decentralized blockchain architectures, such as the potential for 51% attacks in Nakamoto-style consensus mechanisms and nothing-at-stake attacks in proof-of-stake systems. These theoretical vulnerabilities have obscured the risk-benefit analysis for many local currency initiatives [**blockchain_mlc**]. Nevertheless, there is a growing trend towards blockchain adoption in the realm of local currencies [**blockchain_avenir**]. This section examines some prominent examples that illustrate different approaches to blockchain integration in local currency systems.

Colu is an Israeli enterprise that has positioned itself as a bridge between traditional local currencies and blockchain-based cryptocurrencies. Their business model is predicated on facilitating this integration, and they have successfully implemented their system in various global locations. Colu's architecture leverages the Ethereum blockchain, with their proprietary smartphone application serving as an intermediary between local economies and the blockchain infrastructure.

The operational model of Colu involves collaboration with local authorities to implement bespoke local currencies. For each participating region, Colu creates a dedicated ERC20 token (referred to as a "local Colu"), pegged at a 1:1 ratio with the relevant sovereign currency. Local merchants can opt into the system, enabling them to receive local Colu transactions through the application. Colu's fee structure for merchants is designed to be more competitive than traditional credit card fees.

Initially, Colu launched its own Colu Local Network coin on Ethereum through an Initial

Coin Offering (ICO). However, the system's architecture reveals a high degree of centralization, with Colu maintaining sole control over transaction processing. The primary advantage of this ERC20-based structure appears to be the immutability of transaction records. It is worth noting that this centralized approach may conflict with the fundamental objective of local currencies, which is to ensure circulation within a specific geographical area. The fee structure, where charges to local merchants are remitted to Colu, potentially contradicts this principle by facilitating capital outflow from the local economy.

As a result, Colu falls in the Base-Local category of our construction. However it is politically centralized since it uses a privately devolopped application.

The Léman currency system presents an alternative approach to blockchain integration in local currencies. Operating in the transnational region surrounding Lake Geneva (Lac Léman), this system comprises two distinct currencies: the Swiss Léman, pegged to the Swiss Franc, and the French Léman, pegged to the Euro. Both currencies exist in physical and digital formats.

To support the digital iteration of the Léman, the Léman Foundation developed a proprietary blockchain called Com'Chain, which is derivative of the Ethereum blockchain architecture. The Com'Chain system employs a permissioned mining model, where potential miners must obtain approval from the Léman Foundation. This accreditation process is designed to ensure that miners have a vested interest in the currency's success and the local economy's well-being.

The Léman Foundation has also developed a smartphone application to facilitate user interaction with the Com'Chain network. Notably, only merchants registered with the Léman Foundation are authorized to receive transactions within this system. This information was primarily obtained through direct interviews with representatives of the Léman Foundation.

These case studies illustrate the diverse approaches to integrating blockchain technology with local currency systems. While they demonstrate the potential for blockchain to enhance transparency and efficiency in local currency operations, they also highlight the ongoing challenges in balancing decentralization, security, and the core principles of local economic development. The contrasting models of Colu and Léman provide valuable insights into the trade-offs and considerations involved in designing blockchain-based local currency systems.

As a result, for the same reasons as Colu, Léman falls in the Base-Local category of our constructions. It also faces political centralization challenges, since the Com'Chain's miners must be vetted by the Léman foundation.

CityCoins.co is a project that allows creating local cryptocurrency [**CityCoins**, **Vitalik_Crypto_Cities**]. In this paradigm, coins are continuously created with no hard limit. Miners are rewarded in local cryptocurrency for their work. One specificity of CityCoins is that 70% of block reward are awarded to the miner while the remaining 30% are awarded to the city government. However CityCoins made the decision of allowing independent individuals or organizations to create a CityCoin token without the backing of the city council. Only two cities embraced CityCoins: Miami and New York City. CityCoins is built on Stacks [**Stacks_whitepaper**, **Stacks_website**], a layer 1 blockchain running smart contracts on a Bitcoin-like blockchain. It utilizes Proof-of-Transfer (PoX). The project seems to have died out since exchanges are de-listing CityCoins and there has been no update on their website and their X account since February 2023.

Reno, Nevada, USA has emerged as a pioneer in integrating blockchain technology into various aspects of city governance and operations [**Reno**, **Vitalik_Crypto_Cities**]. Under the leadership of Mayor Hillary Schieve, the city's blockchain initiatives span a wide range of applications, from innovative funding methods to improved record-keeping systems. One of Reno's notable projects is the "Biggest Little Blockchain," a first-of-its-kind blockchain record-keeping system

aimed at increasing transparency and accountability. Initially focusing on the city's Register of Historic Places, this system, developed in partnership with BlockApps, aims to streamline the approval process for modifications to historic buildings and provide public access to accurate, immutable records. The city plans to expand this system to encompass other municipal records such as maintenance work, permitting, and licensing. Additionally, Reno has explored more well-known blockchain applications, including Mayor Schieve's proposal to sell an NFT to save the "Space Whale" sculpture. These efforts, along with the city's exploration of blockchain-secured processes for casinos and voting systems, and the potential creation of a Reno DAO governed by "Reno coins," demonstrate the city's comprehensive approach to leveraging blockchain technology. By embracing these innovative technologies, Reno aims to enhance governmental efficiency, transparency, and community engagement, while also potentially attracting tech-savvy businesses and residents to the city.

Although the city of Reno is a pioneer in terms of blockchain adoption, it does not do so for implementing its own local currency.

CityDAO is one of the most radical experiments in blockchain-based city governance [**CityDAO**, **Vitalik_Crypto_Cities**]. It is a Decentralized Autonomous Organization (DAO) with legal status under Wyoming's DAO law, aiming to create entirely new cities from scratch. More precisely, its goal is to "build a city of the future where everything is on-chain, making the physical world decentralized, transparent, immutable and permissionless". Its purpose is to simplify all of the city's bureaucracy both in terms of transparency and in terms of speed. CityDAO purchased its first plot of land in Wyoming. The project plans to expand by acquiring more land plots to build cities governed by a DAO, utilizing innovative economic ideas such as Harberger taxes[1] for land allocation and resource management, as well as quadratic voting[2] which it eventually implemented. CityDAO's governance model is notably progressive, avoiding coin-based voting in favor of a system based on "citizen" NFTs, with discussions about implementing one-person-one-vote systems using Proof-of-Humanity verification [**Proof_of_Humanity**]. Proof of Humanity is a system combining webs of trust with reverse Turing tests and dispute resolution to create a Sybil-proof list of humans. CityDAO is currently in its early stages, crowdfunding through the sale of these NFTs on platforms like OpenSea. CityDAO represents a bold attempt to reimagine city creation and governance using blockchain technology, aiming to create more open, participatory, and innovative urban environments.

CityDAO is a very ambitious project that goes beyond local cryptocurrency. CityDAO's local cryptocurrency falls into the Base-Local category of our constructions.

Cities present an ideal environment for implementing blockchain projects due to their unique characteristics and potential for innovation [**Vitalik_Crypto_Cities**]. Local governments often exhibit greater flexibility and willingness to experiment compared to national entities, making cities perfect testbeds for novel blockchain applications. The diverse needs and challenges of urban areas provide opportunities for tailored blockchain solutions that can address specific local issues. Cities' compact economic ecosystems are well-suited for adopting cryptocurrencies and blockchain-based financial systems. Moreover, blockchain technology can significantly enhance urban governance by improving transparency, accountability, and efficiency in various municipal processes, from record-keeping to voting. It also offers innovative ways to align residents'

[1] Also known as Common Ownership Self-assessed Tax (COST), is a type of property tax that aims to improve societal welfare by optimising for both investment and allocative efficiency of private property. It proposes a new kind of "partial ownership", halfway between private ownership and common ownership [**Harberger_tax**].

[2] Quadratic voting is a rated voting method procedure where voters express the degree of their preferences [**quadratic_voting**].

interests with city development, potentially revolutionizing concepts like community investment and participation. The technology enables more dynamic and responsive governance models, such as real-time voting on local matters. Importantly, city-level blockchain experiments carry lower risks than national implementations, as they operate under broader regulatory frameworks and allow easier opt-out options for residents. This combination of factors makes cities a sweet spot for blockchain innovation, offering a balance of impactful application, manageable risk, and adaptability that can drive meaningful urban transformation.

2.1.4 Outline

This chapter is structured to provide a comprehensive exploration of our proposed framework for local cryptocurrencies, progressing from foundational concepts to advanced implementations. The organization is as follows:

Section 2.2 elucidates pertinent monetary economics principles, with a particular focus on the novel concept of geographical demurrage. This section establishes the theoretical underpinnings necessary for understanding the economic implications of our proposed models.

Section 2.3 presents a formal definition and analysis of distance bounding protocols. These cryptographic primitives are crucial for the implementation of location verification mechanisms, which are integral to the functioning of geographical demurrage in our proposed systems.

Section 2.4.1 introduces our first generic local cryptocurrency model. This baseline model emulates traditional local paper currencies, providing a digital analogue without incorporating geographical demurrage. It serves as a foundation for understanding the core components of a blockchain-based local currency system.

Section 2.4.2 builds upon the baseline model by introducing mechanisms to restrict monetary circulation within a designated geographical area. This enhancement addresses one of the key challenges in digital local currency implementation: maintaining the local nature of the currency in a borderless digital environment.

Section 2.4.3 presents our third model, which incorporates the concept of geographical demurrage into the restricted circulation model. This section explores how the integration of distance-based value depreciation can create novel incentive structures for local economic activity.

Section 2.4.4 expands on the previous models by proposing a universal local cryptocurrency. This advanced model maintains the geographical demurrage mechanism while lifting strict geographical restrictions, potentially allowing for broader adoption while still incentivizing local spending.

Finally, Section 2.5 concludes the chapter by summarizing our key findings, discussing the implications of our proposed models, and outlining potential avenues for future research in the field of local cryptocurrencies.

Through this structured approach, we aim to provide a comprehensive and rigorous examination of the potential for blockchain technology to revolutionize local currency systems.

2.2 Economics

This section provides a comprehensive overview of the economic principles underpinning local currencies and introduces the novel concept of geographical demurrage. We explore the theoretical foundations that inform our proposed models for local cryptocurrencies, drawing from both established economic theories and emerging concepts in monetary economics.

2.2.1 Demurrage

The Velocity of Money is defined as the rate at which money is exchanged within an economy, plays a crucial role in local currency design. In the equation of exchange [**Macroeconomics**], $M \cdot V_T = P \cdot T$ (where M is the money supply, V_T is the velocity of money for all transactions in a given time frame, P is the price level, and T is the aggregate real value of transactions in a given time frame), local currencies aim to increase V_T within a defined geographical area. This increased velocity can potentially stimulate local economic activity and enhance the multiplier effect of local spending.

Demurrage is an economic concept and monetary mechanism that imposes a carrying cost or holding fee on currency or other forms of money. Popularized by the economist Silvio Gesell [**Gesell**] in the early 20th century, demurrage is designed to encourage the circulation of money and discourage its hoarding. In essence, it functions as a negative interest rate on held money, effectively reducing its value over time if not used or exchanged.

The fundamental principle behind demurrage is that money should serve primarily as a medium of exchange rather than a store of value. By implementing a cost for holding currency, demurrage aims to accelerate the velocity of money within an economy, potentially stimulating economic activity and investment.

In its most basic form, demurrage can be implemented through a stamp scrip system, where currency notes must be periodically stamped or marked to maintain their validity, with each stamp requiring a small fee. More modern proposals have suggested electronic systems that could automatically deduct a small percentage from digital currency balances at regular intervals.

The theoretical benefits of demurrage include:

1. Increased circulation of money: By discouraging hoarding, demurrage can lead to more frequent transactions and a higher velocity of money.

2. Stimulation of investment: As holding cash becomes costly, individuals and businesses are incentivized to invest in productive assets or consumption.

3. Reduction of wealth inequality: Demurrage can potentially erode large cash holdings more significantly than smaller ones, leading to a redistribution effect.

4. Stabilization of economic cycles: By encouraging consistent spending and investment, demurrage might help smooth out boom-bust cycles.

5. Mitigation of deflation: In deflationary environments, demurrage can counteract the incentive to hoard money as its purchasing power increases.

However, demurrage also faces significant criticisms and challenges:

1. Implementation difficulties: Applying demurrage to a modern, complex financial system presents substantial logistical and technological challenges.

2. Potential for capital flight: In a globalized economy, demurrage might encourage the movement of capital to non-demurrage currencies or assets.

3. Impact on savings: Critics argue that demurrage could discourage prudent saving behavior, potentially leading to financial instability for individuals and institutions.

4. Complexity in economic planning: Demurrage introduces an additional variable into financial decision-making, potentially complicating economic forecasting and planning.

5. Political and social resistance: The concept of deliberately devaluing currency may face significant opposition from the public and financial institutions.

While the concept of demurrage may seem abstract, its principles have manifested in various forms throughout history and across different economic landscapes.

Inflation acts as a subtle form of demurrage on fiat currency. As the general price level of goods and services rises over time, the purchasing power of money decreases. This erosion of value encourages spending or investment rather than hoarding cash. Central banks often target a low, stable inflation rate to maintain economic stability while still providing this gentle push towards circulation.

Vouchers. Although not presented as such, limited time vouchers also implement demurrage. They impose a time-based cost on asset holders, encouraging circulation and discouraging hoarding. They share similarities with traditional currency demurrage in value depreciation, time-bound utility and their influence on consumer behavior. However, important distinctions exist, such as the rate of depreciation, scope of application, recirculation potential, and legal status. Vouchers typically experience a sudden loss of value at expiration. Vouchers also operate within a limited economic context that is different from that of local currencies.

Negative interest rates are implemented by some central banks. They effectively charge account holders for storing money. This policy turns traditional banking on its head, as depositors pay the bank to hold their funds rather than earning interest. The aim is to stimulate economic activity by encouraging spending and investment. Like demurrage, negative rates penalize the holding of money and push for its circulation.

Storage costs for commodities. Holding physical commodities like oil, grain, or precious metals incurs storage costs, which can be viewed as a form of demurrage. These expenses, which may include warehouse rental, security, and insurance, accumulate over time. As a result, commodity holders are incentivized to sell or use their assets rather than store them indefinitely. This dynamic influences market behavior and pricing strategies in commodity markets.

Parking meters impose a time-based cost on the use of a parking space, functioning as a localized form of demurrage. By charging for the duration of parking, they encourage turnover and efficient use of limited urban space. This system discourages long-term occupation of valuable parking spots and promotes circulation, aligning with demurrage principles in a specific, practical context.

Library late fees serve as a demurrage-like mechanism in the context of public resource management. By imposing penalties for keeping books beyond their due date, libraries encourage the timely return and circulation of their collections. This system ensures fair access to resources for all patrons and maintains the efficiency of the library's operations. Late fees thus act as a form of demurrage on the temporary ownership of borrowed items.

Perishable goods markets inherently incorporate demurrage-like effects due to the natural depreciation of products. Fresh produce, dairy, and other time-sensitive items lose value rapidly as they approach expiration. This creates urgency for sellers to move inventory quickly and for buyers to consume goods promptly. The perishable nature of these goods effectively imposes a time-based cost on holding them, mirroring demurrage principles in a biological context.

Software licenses with expiration dates. Time-limited software licenses embody demurrage-like characteristics in the digital realm. These licenses grant users the right to access software for a specific period, after which the software becomes unusable without renewal. This model creates a depreciating asset, as the value of the license diminishes over time. It encourages regular payments and updates, aligning with software companies' goals of maintaining steady revenue streams and ensuring users have access to the latest versions.

Frequent flyer miles with expiration dates. Many loyalty programs, particularly airline frequent flyer schemes, implement point expiration policies that mirror demurrage effects. Miles or points accumulated by customers lose value over time, eventually becoming worthless if unused. This system incentivizes regular engagement with the brand and encourages customers to redeem their rewards. The expiration policy helps manage the liability of outstanding points for companies while promoting active participation in the loyalty program.

Planned obsolescence creates an effect similar to demurrage on durable goods, while not a direct financial mechanism. Products designed to become outdated or non-functional after a certain period effectively depreciate in value over time. This strategy encourages consumers to replace items regularly, driving ongoing sales for manufacturers. While controversial, planned obsolescence aligns with demurrage principles by discouraging long-term hoarding of goods and promoting continuous economic activity through repeated purchases.

2.2.2 Foundations of Local Currencies

Local currencies are designed to achieve specific economic objectives within their designated areas. By encouraging the use of a currency that can only be spent locally, these systems aim to increase the local economic multiplier effect, potentially leading to increased local employment and economic resilience. Local currencies also seek to reduce the outflow of monetary resources from a community, a phenomenon known as economic leakage. By keeping money circulating within a defined area, they aim to enhance local economic self-sufficiency. In times of economic downturn, local currencies can act as a counter-cyclical measure, providing liquidity and facilitating exchange when national currency may be scarce.

A local currency, in economic terms, can be rigorously defined as a complementary monetary instrument with spatially constrained circulation, typically operating within a delineated sub-national geographic area or specific community network. This medium of exchange, while not possessing legal tender status, functions parallel to the national fiat currency, often with the explicit aim of fostering localized economic activity, enhancing community resilience, or achieving specific socio-economic objectives. The issuance, management, and governance of such currencies are generally decentralized, often overseen by non-governmental entities or community organizations. Their economic properties include limited convertibility, potentially restricted supply mechanisms, and circulation velocities that may differ significantly from national currencies due to their targeted use and limited acceptance. The value of local currencies may be pegged to national currencies, backed by specific goods or services, or based on mutual credit systems. Their implementation can be analyzed through the lens of monetary theory, examining their impact on local money supply, price levels, and economic indicators such as employment and business activity within their circulation zone.

The Wörgl Experiment: A Historical Case Study. The concept of demurrage in local currencies finds a seminal historical precedent in the Wörgl experiment [**worgl**], a monetary initiative implemented in the Austrian town of Wörgl from July 31, 1932, to September 1, 1933. This case

study provides empirical evidence for the potential efficacy of demurrage-based local currencies and offers valuable insights into their economic impact.

Amidst the Great Depression, Wörgl, like many communities, faced severe economic challenges, including high unemployment and decreased economic activity. Under the leadership of Mayor Michael Unterguggenberger, the town issued its own currency, known as "Certified Compensation Bills" or "Wörgl Schillings." These notes were subject to a monthly demurrage fee [**worgl_demurrage**] of 1%, or 12% annually, implemented through the requirement of affixing a stamp worth 1% of the note's face value each month to maintain its validity.

The implementation of the Wörgl currency resulted in several notable economic outcomes. The velocity of the Wörgl scrip was estimated to be much higher than that of the national currency, leading to a significant increase in local transactions. The town was able to initiate and complete various public works projects, reducing unemployment substantially. Neighboring communities began to express interest in adopting similar systems, indicating the potential for regional economic revitalization.

Despite its apparent success, the Wörgl experiment was short-lived, terminated by the Austrian National Bank after 13 months. Critics have raised several points. Questions remain about the applicability of such a system on a larger scale or over a longer period. Economic Distortions: The artificial acceleration of money velocity may lead to suboptimal economic decisions in the long term. Legal and Regulatory Challenges: The experiment's termination highlights the potential conflicts between local currency initiatives and national monetary policy.

The Wörgl experiment continues to inform contemporary local currency designs, including digital implementations. It demonstrates the potential power of well-designed local currencies to address economic challenges, while also highlighting the need for careful consideration of broader economic and regulatory contexts.

Dual Currency Systems are governed by two main laws: Gresham's law and Thiers' law [**GreshamVSThiers**]. Gresham's Law ("bad money drives out good") comes into play under specific economic conditions, primarily when two forms of money are legally recognized as valid for transactions at the same face value, but one possesses a higher intrinsic or perceived value than the other. This scenario typically occurs when there's a fixed exchange rate between the two forms of money, and people have the freedom to choose which to use in transactions and which to hold. The law is most applicable when the "good" money (the more valuable form) is in limited supply relative to the "bad" money, and people are aware of the value difference. Historical examples include bimetallic standards where gold and silver coins circulated together, or situations with debased coinage alongside older, purer coins. In modern contexts, Gresham's Law can manifest in countries with unstable economies where people prefer to save in foreign currencies while spending local currency, or in some cryptocurrency scenarios where stable coins and more volatile cryptocurrencies coexist. However, it is crucial to note that Gresham's Law does not universally apply to all situations where different forms of money coexist; the specific economic and legal context is vital for its application.

Thiers' Law ("good money drives out bad") operates under conditions that contrast with those of Gresham's Law, primarily in free market environments where there's no legal obligation to accept different forms of money at fixed exchange rates. This principle comes into play when individuals and businesses have the liberty to choose which currency to accept for transactions, and one form of money is generally perceived as more stable, reliable, or valuable than others. Thiers' Law is most applicable in scenarios with floating exchange rates and relatively stable economic environments that allow for rational decision-making based on each currency's merits. It requires that people have access to information about the relative values and stability of different currencies. This law is often observed in international trade, countries with weak

currencies, cryptocurrency adoption, well-designed local currency initiatives, and instances of unofficial dollarization. Essentially, Thiers' Law suggests that in free market conditions, "good money" (more stable or valuable) will be preferred and circulated, while "bad money" (less stable or valuable) will be avoided. This principle can significantly influence the success or failure of alternative currencies, including local currencies and cryptocurrencies, by affecting their adoption and circulation in competition with established national currencies.

In the context of local currencies, the local currency is pegged to the sovereign currency while at the same time there is no legal obligation to accept the local currency. We are thus in a scenario where both Gresham's Law and Thiers' Law are applicable. Intrinsically, the sovereign currency is more valuable than the local currency since they have the same purchasing value and the former is more widely accepted than the latter. Nevertheless, local currencies are still circulating because of their perceived value. In fact, local currency designers can influence which law predominates through policy decisions. For example, implementing demurrage encourages Gresham's Law-like behavior, while focusing on unique benefits and ease of use might promote Thiers' Law dynamics.

Legal Framework. In practice, the implementation of local currencies is subject to stringent legal and regulatory frameworks in many jurisdictions. Numerous nations prohibit or severely restrict the use of alternative currencies to maintain monetary sovereignty and economic stability. When local currencies are sanctioned, it is typically under the condition that they do not directly compete with the state's official currency and adhere to a comprehensive set of rules and regulations.

The establishment of a complementary local currency generally requires the formation of a dedicated institution responsible for its management and oversight. Participation in the local currency system is often restricted to businesses that formally affiliate with this governing institution and agree to abide by a specific charter or code of conduct.

A critical regulatory requirement frequently imposed on local currencies is the maintenance of a fixed exchange rate with the sovereign currency. This peg serves to stabilize the local currency's value and mitigate potential economic disruptions. Additionally, to ensure fiscal responsibility and prevent inflationary pressures, many regulatory frameworks mandate that the managing institution maintain 100% reserves for the local currency in circulation, effectively prohibiting fractional reserve practices.

An interesting feature of many local currency systems is the unidirectional convertibility mechanism. While users can typically acquire local currency units by depositing equivalent amounts of sovereign currency, the reverse conversion is often prohibited for individual users. This design aims to encourage local circulation and prevent rapid outflows of currency from the community. However, recognizing the practical needs of businesses to engage in external transactions and meet tax obligations, a provision is often made allowing registered businesses to convert local currency back to sovereign currency, albeit usually subject to a conversion fee. These fees serve as a primary source of operational funding for the managing institution.

It is important to note that the enforcement of these regulations, particularly concerning paper-based local currencies, can present significant challenges. The potential for unofficial transactions between non-affiliated businesses and currency holders exists, potentially circumventing the established system. Some local currency initiatives have opted to waive conversion fees, at least initially, to promote adoption and circulation.

This complex interplay of regulatory requirements, economic design, and practical considerations underscores the intricate nature of implementing and maintaining local currency systems within the broader context of national monetary policy and economic governance.

2.2.3 Geographical Demurrage: A Spatial Approach to Local Currencies

In the following subsection, we present the innovative concept of "geographical demurrage" as an extension of traditional demurrage principles. This approach introduces a spatial dimension to currency valuation, where the value of a unit of currency decreases not with time, but with the distance it travels from its point of origin or last transaction. Consequently, the definition of local currency must be revisited to better suit our proposed local cryptocurrencies.

Geographical Demurrage: A Novel Approach. We introduce the concept of geographical demurrage as an innovative extension of traditional demurrage principles. In this model, the value of currency decreases not with time, but with the distance it travels from its point of origin or last transaction. Formally, we define the value function as follows.

Definition 2.1: Geographical Demurrage

Let $d \geq 0$ denote the geographical distance between the point of receipt x and the point of expenditure y. A currency is said to be subject to *geographical demurrage* if the value $V_{v_0}(d)$ of an initial balance v_0 spent at distance d is given by

$$V_{v_0}(d) = v_0 \cdot f(d),$$

where $f : \mathbb{R}_{\geq 0} \to [k, 1]$ is a monotone decreasing function such that $f(0) = 1$ and $\lim_{d\to\infty} f(d) = k$, with $0 \leq k < 1$.
Equivalently, the geographical demurrage rate is defined as

$$\delta(d) = 1 - f(d),$$

which is strictly increasing with $\delta(0) = 0$ and $\lim_{d\to\infty} \delta(d) = 1 - k$.

This formulation ensures that no demurrage is applied when a coin is spent at its point of receipt, while the demurrage increases monotonically with distance while being bounded by k.

This demurrage function can take various forms depending on the specific economic objectives and geographical characteristics of the local currency system.

Example 2.2

Take for instance the following demurrage functions:

1. Linear demurrage: $f(d) = \max(1 - \alpha d, k)$, where α is the rate of value decrease per unit distance.
2. Quadratic demurrage: $f(d) = \max(1 - \alpha d^2 - \beta d, k)$, with $\alpha > 0$ and $\beta \geq 0$.
3. Inverse demurrage: $f(d) = k + \frac{1}{\frac{1}{1-k} + \alpha d}$, with $\alpha > 0$.
4. Exponential demurrage: $f(d) = k + (1 - k)^{1+\alpha d}$, with $\alpha > 0$.
5. All-or-nothing demurrage:

$$f(d) = \begin{cases} 1, & \text{if } d < \alpha \\ k, & \text{if } d \geq \alpha \end{cases}$$

with $\theta > 0$.

6. Step demurrage: $f(d) = \max\left(\frac{\left\lceil \frac{\alpha - d}{\beta} \right\rceil}{\frac{\alpha}{\beta}}, k\right)$ with $\alpha > \beta > 0$.

Examples of curves using the aforementioned forms are presented in Figure 2.1. The parameters are chosen as follows. General parameters: $v_0 = 100, k = 0.2$. Linear: $\alpha = 0.01$. Quadratic: $\alpha = 0.00025, \beta = 0$. Inverse: $\alpha = 0.5$. Exponential: $\alpha = 0.5$. All-or-norhing: $\alpha = 50$. Step: $\alpha = 100, \beta = 10$. Notice how the values never drop below 20. This is due to the lower bound on demurrage defined as $k = 0.2$.

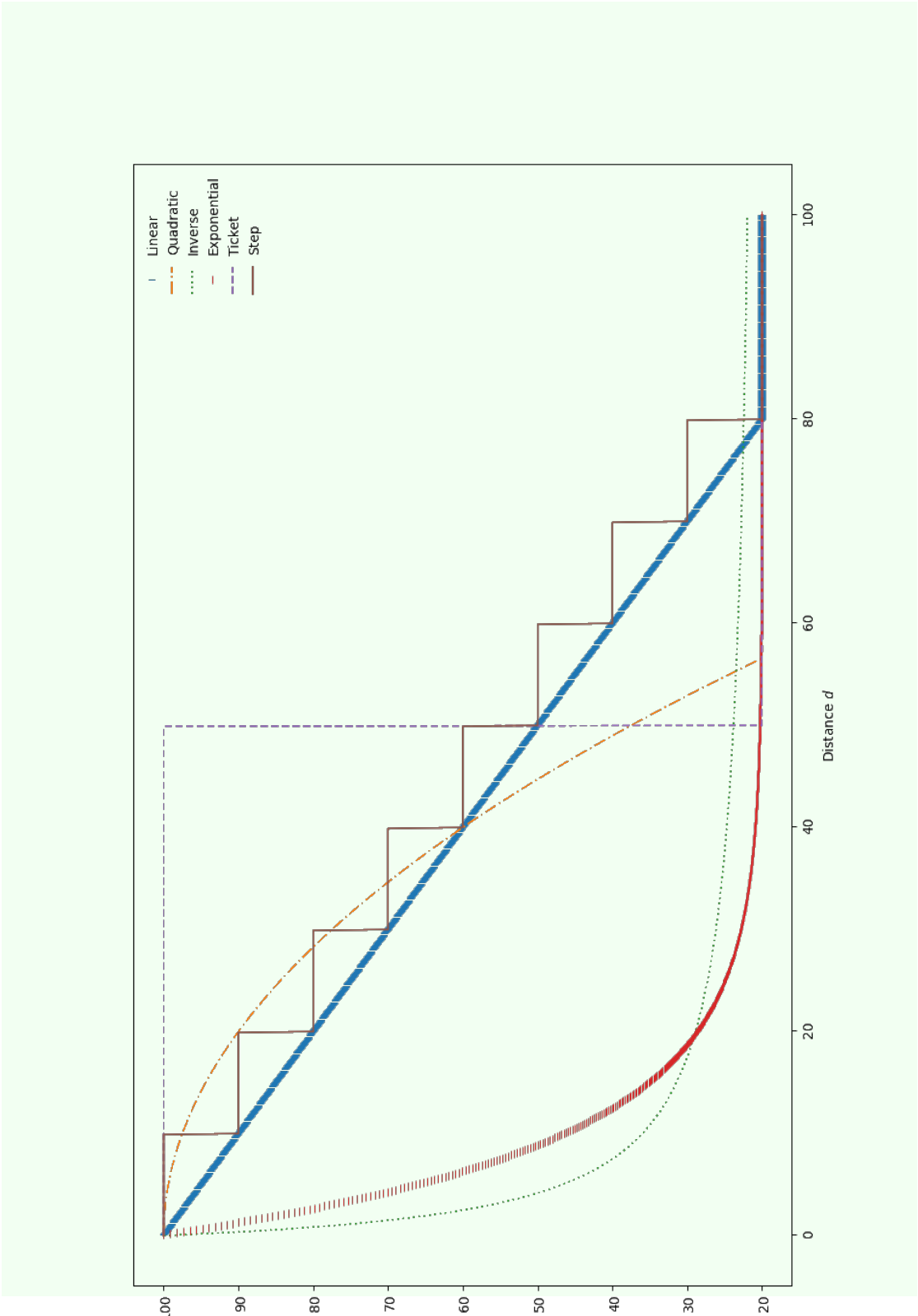

Linear
Quadratic
Inverse
Exponential
Ticket
Step
Distance d
0
20
40
60
80
100
100
90
80
70
60
50
40
30
20

Figure 2.1: Examples of Currency Value Fluctuation in Function of Distance.
The choice of function and parameters allows for fine-tuning of the local currency's behavior to achieve desired economic outcomes. The selection of the functional form and associated parameters enables precise calibration of the local currency's behavioral characteristics, facilitating the attainment of specific economic objectives. Upon closer examination, it becomes evident that a convex function $f(d)$ proves counterproductive, as it imposes disproportionately severe penalties on the sender for each incremental unit of distance. Conversely, concave functions offer the sender a degree of flexibility in local expenditure before implementing an increasingly stringent penalty structure. This non-linear approach to distance-based penalties allows for a more nuanced economic incentive structure, potentially fostering localized economic activity while still discouraging transactions beyond a certain geographical threshold. The careful design of this function can thus serve as a powerful tool in shaping local economic dynamics, balancing the need for local economic stimulation with the broader goals of the currency system.

The introduction of geographical demurrage has several potential economic implications. By creating a gradient of currency value based on distance, geographical demurrage provides a direct economic incentive for local spending. This could lead to the formation of economically optimal transaction radii, potentially reshaping local economic geographies. Unlike traditional local currencies with fixed geographical boundaries, a system with geographical demurrage allows for more fluid and economically responsive local currency zones. The effective range of the currency is determined by economic factors rather than arbitrary administrative boundaries. While designed to encourage local spending, a well-calibrated geographical demurrage system could also facilitate beneficial economic interactions between neighboring communities, potentially leading to more robust regional economic networks.

Determining the optimal demurrage function and parameters to achieve desired economic outcomes without creating unintended distortions is a complex task requiring careful economic modeling and potentially adaptive mechanisms. The novel nature of geographical demurrage may present challenges in terms of regulatory compliance and legal recognition, particularly in cross-border scenarios.

Redefining Local Currency. A local currency is conventionally defined as a medium of exchange whose circulation is predominantly confined to a specific, contiguous geographic area that is substantially smaller than the territory of the issuing sovereign state. This definition can be formalized through the lens of geographical demurrage: Let $D(x,y)$ represent the demurrage rate if received at location x and spent at location y, and S denote the issuing sovereign state area. Then:

$$D(x,y) = \begin{cases} 0, & \text{if } y \in S \\ 1, & \text{if } y \notin S \end{cases} \tag{2.1}$$

Where 0 indicates no value loss and 1 represents complete value loss (100% demurrage).

Critique of Traditional Definition: It is important to note that the absolute restriction of a currency's circulation to a predefined geographical sphere is, in practice, unenforceable, particularly for physical currencies. The possibility of extra-territorial exchange between willing parties renders strict geographical constraints theoretically and practically untenable.

Proposed Broader Definition: In light of these considerations, we propose a more generalized and robust definition:

Definition 2.3: Local Currency

A *local currency* is a medium of exchange characterized by a non-uniform geographical demurrage function.

Formally, let C be a currency and $D_C(x, y)$ be its demurrage function if a coin is received at location x and spent at location y. C is a local currency if and only if:

$$\exists x_1, y_1, x_2, y_2 \in S : D_C(x_1, y_1) \neq D_C(x_2, y_2) \tag{2.2}$$

Where x_1, x_2, y_1 and y_2 represent geographical locations and S denote the issuing sovereign state area.

This definition encompasses currencies with varying degrees of geographical preference, from those with binary demurrage (as in the traditional concept) to those with more nuanced, continuous demurrage functions over space as seen in Figure 2.1. In such scenarios we have $D(x, y) = 1 - f(\|\vec{xy}\|)$. It allows for a more flexible representation of local currencies which suits our local currency constructions.

2.3 Distance Bounding

Distance bounding protocols, introduced by Brands and Chaum [**distance_bounding**], constitute a class of cryptographic mechanisms designed to establish an *upper bound* on the physical distance between two communicating entities. These protocols are essential in applications where geographical proximity is a prerequisite for access or transactional privileges. Their relevance to local cryptocurrencies lies in their ability to provide cryptographic assurances that a transaction recipient is indeed located within a defined spatial domain.

At a fundamental level, distance bounding protocols exploit the invariance of the speed of light, a principle rooted in Einstein's theory of special relativity [**special_relativity**]. Because no information can propagate faster than light, round-trip timing measurements of challenge–response exchanges provide a secure upper bound on the physical separation between a prover (the party claiming proximity) and a verifier (the party performing the verification).

Definition 2.4: Distance Bounding Protocols

A *distance bounding protocol* is a cryptographic protocol between a prover P and a verifier V designed to establish an upper bound on their physical separation. Formally, the verifier obtains a bound

$$d(P, V) \leq c \cdot \Delta t,$$

where c is the speed of light in vacuum and Δt the measured round-trip delay. Security requires time-criticality of responses, unpredictability of challenges, and cryptographic binding of the session transcript to prevent replay and relay attacks.

In practice, most distance bounding protocols follow a canonical structure consisting of four phases:

1. **Setup Phase**: P and V exchange initial cryptographic material, such as keys or nonces.

2. **Commitment Phase**: P commits to information that will be revealed later to bind responses to the current session.

3. **Rapid Bit Exchange Phase**: V sends a sequence of one-bit challenges, to which P must respond instantaneously. V records the precise round-trip time of each challenge–response pair.

4. **Verification Phase**: V computes the distance bound and verifies both the commitment and the integrity of the session.

The security of distance bounding protocols relies on several core principles: *time-criticality* (responses must occur within strict temporal windows), *challenge unpredictability* (preventing pre-computation of responses), *bit-wise independence* (prohibiting parallel processing of responses), and *cryptographic binding* (ensuring session integrity).

These properties collectively provide resilience against various attacks, including [**distance_bounding_survey**, **distance_bounding_survey_pascal**]:

Impersonation fraud is an attack where an adversary acting alone purports to be a legitimate prover.

Distance fraud is an attack where a dishonest prover purports to be in the neighborhood of the verifier. He cheats without help of other entities located in the neighborhood.

Mafia fraud is an attack where an adversary defeats a distance bounding protocol using a man-in-the-middle between the verifier and an honest prover located outside the neighborhood.

Terrorist fraud is an attack where an adversary defeats a distance bounding protocol using a man-in-the-middle between the verifier and a dishonest prover located outside of the neighborhood under the following circumstances. The dishonest prover actively helps the adversary to maximize her current attack success probability but without giving her any advantage for future man-in-the-middle attacks. (In such attacks, the man-in-the-middle would attempt to pass the distance bounding protocol as a valid prover/tag that the man-in-the-middle does not represent/possess.)

Distance hijacking is an attack where a dishonest prover aims to convince a verifier that he is located within the verifier's neighborhood, abusing some other provers who are indeed in the verifier's neighborhood.

Surveys [**distance_bounding_survey_pascal**, **distance_bounding_survey**] show different distance bounding protocols resisting differently to the different identified attacks. In this chapter we suppose the use of a distance bounding protocol resistant to all aforementioned attacks.

Many technologies allow devices in proximity to communicate directly, such as NFC [**NFC_survey**] or RFID [**RFID_survey**]. These technologies rely on Radio Frequency (RF). They do not prevent attacks such as relay attacks for they are only communication protocols [**RFID-NFC_overview**]. In this context, distance bounding protocols were invented to prevent such attacks from happening.

Example of Distance Bounding Protocol. The first distance bounding protocol, introduced by Brands and Chaum, concerns two participants: the prover $\mathcal{P}$ is the entity claiming to be within a certain distance of the verifier; and the verifier $\mathcal{V}$ is the entity trying to verify the proximity of the prover. During the setup phase, $\mathcal{P}$ and $\mathcal{V}$ agree on a commitment scheme and a number of

rounds k for the rapid bit exchange. The next three phases (commitment, rapid bit exchange and verification phases) are illustrated in Figure 2.2. This protocol defends only against distance fraud and mafia fraud.

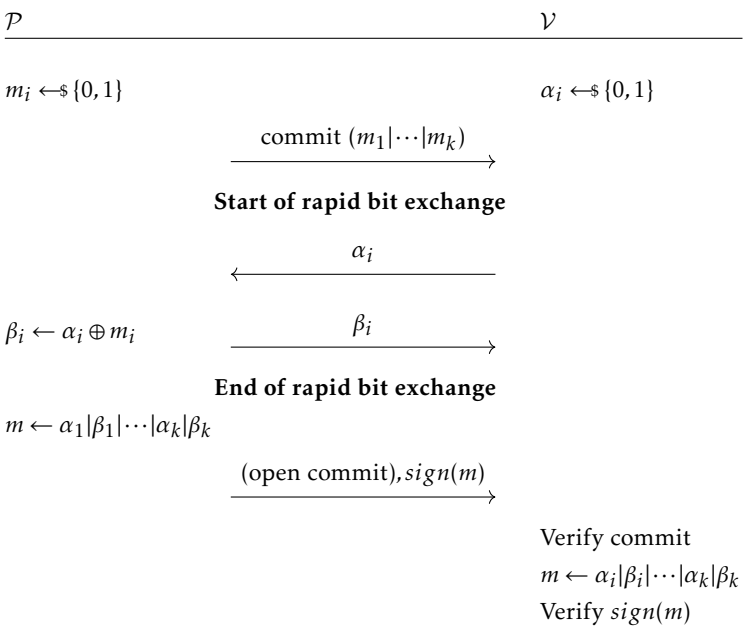


Figure 2.2: Brands and Chaum Distance Bounding Protocol

2.4 Local Cryptocurrency

In this section, we first detail the blockchain underlying all four construction before delineating the four distinct typologies of localized cryptographic currencies. Base-Local emulates the functional characteristics of traditional paper-based local currencies. Geo-Limited variant incorporates geographical constraints, effectively circumscribing the circulation of monetary units within a predefined spatial domain. Geo-Demurrage builds upon the geographical restriction, further augmenting it with the implementation of a spatially-dependent demurrage mechanism. Finally, we propose the fourth and final construction, Uni-Local, that retains the geographical demurrage feature while eschewing spatial restrictions, thereby engendering a paradoxical construct: a universally applicable local cryptocurrency. These four proposed local cryptocurrency models share fundamental structural elements. The initial typology serves as the foundational archetype, from which the subsequent three variants are derived through progressive modifications and enhancements.

Blockchain

In our proposed framework, we posit that the local currency institution assumes primary responsibility for the project, analogous to the management of local paper currency. This entity

secures legal authorization, drafts a comprehensive charter, and extends invitations to local merchants for participation.

The foundational blockchain architecture is uniform across all local cryptocurrency implementations. We advocate for a permissioned blockchain system, wherein the miner cohort is comprised of a subset of affiliated local merchants as illustrated in Figure 2.3. Any merchant within this network retains the option to assume mining responsibilities. The consensus mechanism is left to the discretion of the institution, aligning with the blockchain's and local cryptocurrency's inherent flexibility. While any consensus protocol is theoretically viable, the permissioned nature of the blockchain renders Byzantine Fault Tolerant (BFT) consensus mechanisms, such as Algorand's *BA*⋆, particularly advantageous. Depending on the specific blockchain infrastructure, prospective miners may need to formally declare their intent to participate in the mining process to existing network participants.

Privacy considerations are not a primary focus of this work. They remain intersting and deserve to be delved into more deeply in future research.

Currency Issuance and Transactions. The institution serves as the primary point of exchange for clients to acquire local currency. In this process, the institution issues local currency tokens in exchange for sovereign currency, which may be tendered in either physical or digital form. Acting as a custodian, the institution secures these fiat funds while concurrently initiating a blockchain transaction to mint an equivalent quantum of cryptocurrency tokens. These tokens are then cryptographically associated with the client's public key. In this capacity, the institution effectively functions as a centralized exchange platform, bridging the fiat and crypto ecosystems.

Conversely, when affiliated merchants seek to exchange local cryptocurrency for sovereign currency, the institution may impose a nominal fee to generate operational capital. Upon execution of such a buyback, the corresponding quantity of cryptocurrency tokens is irreversibly removed from circulation through a burning mechanism on the blockchain.

Transactions within the system adhere to the unspent transaction output (UTXO) model: the output of one transaction serves as the input for a subsequent transaction, authenticated by the sender's digital signature and bound to the recipient's public key. While privacy considerations are not addressed in this iteration, transactions can accommodate multiple inputs and one or two outputs. Each input represents a discrete coin, with the first output designated for the beneficiary and the second, if applicable, returning change to the sender.

Fees. The implementation of a fee mechanism in the underlying blockchain infrastructure for local cryptocurrencies necessitates a nuanced approach that balances network sustainability, economic incentives, and the fundamental objectives of local currency systems. The incorporation of transaction fees serves multiple purposes: primarily, it acts as an incentive structure for network participants (specifically, the consortium of local merchants acting as miners) to maintain and secure the blockchain, thereby ensuring its continued operation and integrity. Additionally, a well-calibrated fee structure can serve as a deterrent against network spam and potential denial-of-service attacks, contributing to the overall robustness of the system.

However, the introduction of fees must be judiciously considered within the context of local economic ecosystems, where the promotion of accessibility and the enhancement of monetary velocity are often paramount. The imposition of substantial fees could potentially create friction in the system, particularly for microtransactions which are frequently prevalent in local economies. This friction could, in turn, impede adoption and utilization of the local cryptocurrency, thereby undermining its efficacy as a tool for local economic stimulation.

Several approaches warrant consideration. A minimal fee structure, where transaction costs are kept significantly below those of traditional payment systems or major cryptocurrencies,

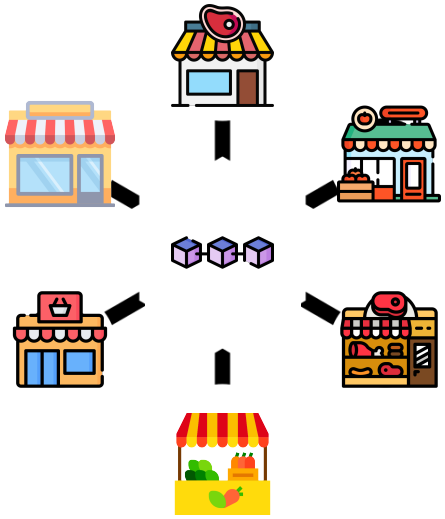


Figure 2.3: Blockchain Miners

could strike a balance between network sustainability and transaction fluidity. Alternatively, a dynamic fee model could be implemented, where fees are adjusted based on network congestion, transaction size, or other relevant parameters. This approach could help optimize network resources while maintaining accessibility for smaller transactions.

Another avenue to explore is a fee-less model. The viability of a fee-less model is particularly pronounced in the context of local cryptocurrencies due to the inherently constrained scale of these systems. Unlike global cryptocurrencies that must contend with vast networks and potential spam attacks, local systems operate within a more limited and defined scope. This reduced scale allows for alternative methods of network regulation and maintenance that do not rely on transaction fees. One such method is the implementation, as we do, of a permissioned blockchain architecture, where network participants (local merchants acting as nodes) are vetted and approved. This approach significantly mitigates the risk of malicious activities that transaction fees are often designed to deter in larger, permissionless systems. The trusted nature of participants in a local system obviates the need for economic disincentives against network abuse.

For the aforementioned reasons, we choose to go forward with a fee-less model.

2.4.1 Base-Local

The initial construct, denoted as *Base-Local*, represents the fundamental implementation of the previously delineated architecture, without of any supplementary modifications or enhancements. This elemental configuration mimics other local cryptocurrencies. It maintains fidelity to the original design principles without the incorporation of additional features or optimizations.

Discussion

This initial construction serves as a foundational model for subsequent iterations. Two notable characteristics emerge: the absence of geographical restrictions on currency circulation and the

implicit trust placed in affiliated merchants.

Unrestricted Area. The current design allows for transactions to occur beyond the designated geographical sphere of the local currency. Moreover, there are no mechanisms preventing unaffiliated entities from generating public/private key pairs and accepting payments in the local currency. While these entities cannot exchange the local currency for sovereign currency through the institution, such practices, though potentially at odds with the currency's intended purpose, remain legal provided all tax obligations are met in sovereign currency. This limitation is not unique to our digital implementation and is also present in physical local currency systems.

Trust in Shops. The proposed system necessitates a degree of trust in affiliated merchants, as collusion among them could potentially compromise the entire cryptocurrency ecosystem. While this trust requirement is analogous to that of physical local currencies, the potential consequences of betrayal in the digital realm are more severe. Nevertheless, our solution offers significant advantages over centralized digital local currencies. It is architecturally and politically decentralized, eliminating single points of failure and requiring a majority of merchants to act maliciously in concert to compromise users' funds. This stands in stark contrast to conventional digital currencies, which demand implicit trust in a single entity overseeing all transactions.

2.4.2 Geo-Limited

This iteration of the local cryptocurrency builds upon the preceding model, incorporating an additional layer to constrain transactions within a specified geographical sphere. This is achieved through the implementation of distance bounding protocols to issue certificates of proximity.

Proximity Certification. The recipient of a transaction must always be within the delimited area. To ascertain the recipient's location at the time of transaction, we employ certificates issued by designated Certificate Authorities (CAs). In our framework, the role of CAs is restricted to affiliated merchants, minimizing the trust requirements as local currency users inherently place confidence in these entities by virtue of their participation in the network Prior to issuing a proximity certificate, CAs execute a distance bounding protocol with the client to verify their physical proximity. The choice of the distance bounding protocol is left for the implementation. However, it must defend againts: distance fraud, mafia fraud and terrorist fraud. Impersonation attacks do not interest us, since even if the attacker purports to be a valid prover, the attacker will not able to spend the prover's coins. And distance hijacking attacks do not interst us either, since in our trust model we do trust the shops.

The resulting certificate of proximity encompasses the client's public key, the CA's location, and a timestamp. For transactions where the recipient is an affiliated merchant, the merchant can self-issue a certificate without the need for a distance bounding protocol. In peer-to-peer transfers, the recipient must engage with a CA to undergo a distance bounding protocol. The CA then issues a proximity certificate to the recipient, who forwards it to the sender. The sender is then allocated a predetermined window of time to execute the transaction. Let τ represent the validity period of the certificate; miners are programmed to reject certificates older than τ when incorporating transactions into the blockchain. We assume temporal synchronization among CAs to maintain consistsency in certificate validation.

Discussion

This enhanced model enables the enforcement of geographical constraints on monetary circulation, a desirable feature in local currencies that has hitherto been challenging to implement. The system relies on trusting affiliated merchants in their capacity as Certificate Authorities. This trust assumption is not problematic, as it is subordinate to the more critical trust already placed in these entities for mining, maintaining, and securing the blockchain.

In the context of blockchain operations, we assume that a majority of merchants are not malicious or colluding. For the certificate issuance process, we extend this trust to assume that no merchants collude with clients to issue fraudulent proximity certificates. This tiered trust model aligns with the inherent security requirements of the system while providing a robust mechanism for geographical constraint enforcement.

2.4.3 Geo-Demurrage

This iteration introduces a sophisticated geographical demurrage mechanism into the blockchain-based transactions. The demurrage is applied to coins upon transfer, with the amount calculated based on the spatial displacement between the point of receipt by the sender and the point of disbursement to the receiver.

Point of Attachment. To keep track of the distance travelled by coins, each coin is associated with a *Point of Attachment* (PoA), representing its current geographical location. Upon creation, a coin's PoA is initialized to the location of the issuing institution. Subsequently, when a coin is transferred, its PoA is updated to the recipient's location at the time of transaction. The recipient's location must be explicitly specified alongside their public key in the transaction data. This location is in fact the proximity certificate issued by a CA. So in practice, a PoA is always the same address as one of the CAs.

Geographical Demurrage Function. We reprise the value function from Definition 2.1: $V_{v_0}(d) = v_0 \cdot f(d)$ where $V_{v_0}(\cdot)$ is a non-negative strictly decreasing function with $V_{v_0}(0) = v_0$. We reprise $f(d)$ as a monotonic decreasing function of distance, with $f(0) = 1$ and $\lim_{d\to\infty} f(d) = k$, where $0 \le k < 1$. We reprise the demurrage function $\delta(d) = 1 - f(d)$ that maps a distance to its corresponding geographical demurrage rate which is a strictly increasing function with the property $\delta(0) = 0$ and $\lim_{d\to\infty} \delta(d) = 1 - k$.

Example 2.5

Consider a sender S and a receiver R, with points x, y, and z in space. If S received amount n of currency at point x and wishes to transfer $m < n$ to R at a later time, where S is at point y and R is at point z, the transaction incurs a demurrage of $\delta(\|\vec{xz}\|)$. In other words, if S pays m in total, then R will receive $V_m(\|\vec{xz}\|)$. For R to receive m, then S should pay r such that: $V_r(\|\vec{xz}\|) = m$.

Example 2.6

Let Alice buy coins from the institution (whether online or in-person). These coins' location is set to be the institution's headquarters' location. When Alice spends part of these coins at a shop, demurrage is computed based on the distance between the institution's headquarters and the shop. When Alice sends money to Bob, Bob has to go into a registered shop (or the institution's headquarters) and do a distance bounding

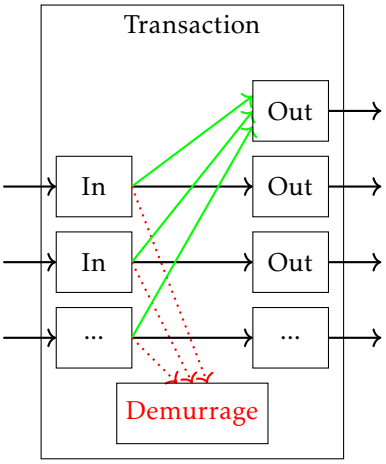


Figure 2.4: Transaction Structure

protocol to prove he is in the close vicinity of this location. This can be done using NFC-based distance bounding protocols [**distance_bounding_NFC**]. Then Alice sends Bob coins on which demurrage is applied based on the distance between the institution's headquarters and the shop Bob went to.

Transaction Structure. Transactions can involve multiple inputs and outputs. Each input represents a distinct coin, while the outputs comprise the beneficiary's received amount and any change returned to the sender. This structure allows the sender to source the transaction amount from multiple coins at different locations. The transaction structure is formally defined by Equations 2.3 and 2.4 where out_0 is the total amount sent to the beneficiary, $out_{0,i}$ is the contribution from coin i to the beneficiary, in_i is the input amount of coin i, out_i is the change returned for coin i, Δ_i is the distance between the location of coin i and the recipient's location, and $\delta(\cdot)$ is the demurrage function. Notice the greater or equal sign ($\geq$) in Equation 2.4, this permits the output to not be greater than the input (considering demurrage). Ideally they should be equal, But in some cases the true output is not a multiplier of the minimal subdivision of the currency, in which case it is problematic. The greater or equal sign here solves this problem. The transaction structure is illustrated in Figure 2.4. Notice how the demurrage is similar to the fees in other cryptocurrencies: it is necessary in order to pursue transactions.

$$\sum_{i=1}^{n} out_{0,i} = out_0 \tag{2.3}$$

$$\forall i \in \{1, \cdots, n\}, (in_i - out_i) \cdot (1 - \delta(\Delta_i)) \geq out_{0,i} \tag{2.4}$$

Example 2.7

Alice sends two seperate coins in one transaction to Bob. The first coin is worth 10 units of currency and the second is worth 20. She sends all of the first but keeps 8 units of the second's 20. So $in_1 = 10, out_1 = 0$ and $in_2 = 20, out_2 = 8$ Let the distance from the first coin to the recipient's location be $\Delta_1 = 5$ km and the distance from the second coin to the recipient's location be $\Delta_2 = 10$. Let $\delta(x) = 1 - \max(1 - 0.05 \cdot d, 0)$. Then from the first coin's 10 units, Bob will get: $out_{0,1} = (in_1 - out_1) \cdot (1 - \delta(\Delta_1)) = (10 - 0) \cdot (1 - (1 - \max(1 - 0.05 \cdot 5, 0))) = 7.5$. From the second coin's 20 units, Bob will get: $out_{0,2} = (in_2 - out_2) \cdot (1 - \delta(\Delta_2)) = (20 - 8) \cdot (1 - (1 - \max(1 - 0.05 \cdot 10, 0))) = 6$. In total Bob will receive: $out_0 = \sum_{i=1}^{2} out_{0,i} = 7.5 + 6 = 13.5$.

Demurrage Allocation Strategies. We propose two strategies for allocating the money generated by geographical demurrage. The first strategy benefits the institution while the second benefits the miners. Naturally nothing prevents from implementing a hybrid strategy where the demurrage is shared.

In the first approach, the demurrage is effectively destroyed, thus reducing the total currency supply. As the institution is obligated to maintain a sovereign currency reserve equivalent to the local currency in circulation, this reduction allows the institution to release a portion of its reserve, which can be utilized as operating capital.

Alternatively, the demurrage could be allocated to the miners (*i.e.*, the participating merchants) as a reward for their role in maintaining the blockchain. This model incentivizes transparency and trustworthiness among merchants, as their income is directly tied to the secure operation of the blockchain. Furthermore, this compensation could offset potential losses incurred during the reconversion of local currency to sovereign currency, potentially neutralizing the financial impact of participating in the local currency system.

A hybrid model example could be one where part of the demurrage is destroyed (*i.e.* benefitting the institution) and the rest is distributed to the miners (local merchants). These sophisticated mechanisms for geographical demurrage and its allocation serve to reinforce the local nature of the currency while providing sustainable economic incentives for key participants in the system.

2.4.4 Uni-Local

In this final iteration, we present a refined construct that synthesizes elements from previous models while introducing novel features. This universal local cryptocurrency, which we designate as LCoin, eschews the geographic constraints imposed in Geo-Limited and Geo-Demurrage while retaining the geographical demurrage mechanism from Geo-Demurrage. Alternatively, this can be conceptualized as an augmentation of Base-Local with the addition of geographical demurrage.

The transaction structure remains consistent with that of Geo-Demurrage, adhering to the format illustrated in Figure 2.4 and governed by Equations 2.3 and 2.4.

Discussion

The universal local cryptocurrency paradigm deliberately relinquishes geographic restrictions, a departure from conventional local currency models. While this expansive approach might ostensibly challenge the currency's "local" designation, we posit that it retains its local essence

through its inherent mechanism of incentivizing proximal transactions. The geographical demurrage feature persistently encourages localized spending, albeit within a potentially unlimited geographical scope. This universal model of local cryptocurrency presents significant scalability potential, necessitating the implementation of highly efficient consensus mechanisms such as Algorand's *BA*⋆ protocol to accommodate potential growth. In contrast to previous constructions where a centralized institution was responsible for blockchain establishment, charter formulation, and coin creation within a localized context, the universal model introduces new complexities stemming from its non-local nature. This shift from a centralized, trusted format to a more distributed model presents both opportunities and challenges. The implementation of such a universal local cryptocurrency necessitates the participation of diverse merchants across disparate geographical locations within a unified network. A critical consideration is the establishment of multiple exchange platforms for LCoin acquisition. The absence of proximal exchange options (institutions to buy coins from nearby) could potentially negate the benefits for users, as they would incur substantial geographical demurrage fees for local transactions.

As the network expands geographically, maintaining trust becomes increasingly challenging. While trust in local merchants is often predicated on familiarity and frequent interactions, extending this trust to a network predominantly composed of unfamiliar entities presents a significant hurdle. We anticipate that such a theoretical universal local cryptocurrency may naturally develop boundaries, potentially aligning with existing political borders, as this would facilitate legal enforcement within established judicial frameworks, thereby fostering greater trust. A potential drawback of the universal local cryptocurrency model lies in its scalability. Should it grow to encompass a substantial portion of a sovereign state's territory, it might be perceived as a competitor to the national currency, potentially prompting regulatory intervention or cessation of the project by state authorities.

2.5 Conclusion

This chapter presents a comprehensive exploration of local cryptocurrencies, offering novel approaches to address longstanding challenges in the implementation and operation of local currency systems. Through a series of increasingly sophisticated models, we have demonstrated the potential for blockchain technology to revolutionize the concept of local currencies, enhancing their effectiveness and expanding their potential impact. Our journey began with the introduction of a generic cryptocurrency model, Base-Local, that emulates traditional local paper currencies, providing a digital analogue with enhanced security and transparency. This is interesting for easily building a local cryptocurrency. Building upon this foundation, we developed a more advanced model, Geo-Limited, that successfully restricts currency circulation to a designated geographical sphere – a feature long desired but previously unattainable in both physical and digital local currency systems. This geographical constraint was achieved through the innovative application of distance bounding protocols, with local merchants serving as Certificate Authorities to verify proximity. A significant contribution of this work is the introduction of the concept of geographical demurrage. This novel mechanism, which applies a value depreciation based on the distance a coin travels from its point of origin, provides a powerful tool for incentivizing local economic activity. The integration of geographical demurrage into our local cryptocurrency model renders Geo-Demurrage, and it not only encourages localized spending but also offers a sustainable method for system maintenance and merchant incentivization. This is perhaps the best version that should be implemented by local institutions for it patches the issues with local cryptocurrencies. The culmination of our research is the proposal of a universal local cryptocurrency, which we have named LCoin, based on the Uni-Local framework.

This final model retains the geographical demurrage feature while lifting strict geographical restrictions, creating a paradoxical yet potentially transformative concept: a local currency with global applicability. LCoin represents a paradigm shift in local currency design, inherently encouraging localized economic activity without the need for explicit geographical constraints.

Throughout our exploration, we have carefully considered the economic implications, technical challenges, and trust requirements associated with each model. Our approach balances the need for decentralization and security inherent in blockchain systems with the specific requirements of local currency operations, including the need for trusted local entities and compliance with legal and regulatory frameworks. While this research provides a robust foundation for the development of blockchain-based local currencies, it also opens up several avenues for future investigation. Privacy considerations, which were not a primary focus of this work, represent a critical area for further research. The development of privacy-preserving mechanisms that are compatible with the geographical verification requirements of our models presents a significant challenge. The scalability and broader economic implications of a potentially global local currency system, as represented by LCoin, also warrant deeper examination. As these systems grow, questions of governance, economic impact, and regulatory compliance will become increasingly complex and important. In conclusion, this chapter contributes to the evolving discourse on local currencies by proposing innovative mechanisms for their implementation in the digital age. By leveraging blockchain technology and introducing concepts such as geographical demurrage, we have presented a framework that addresses many of the limitations of traditional local currencies while opening up new possibilities for local economic development and community engagement. As we move forward, the continued refinement and practical implementation of these models have the potential to significantly reshape how communities interact with and benefit from localized economic systems in an increasingly digital world.

3

Generic Blockchain on Generic Human Behavior

This chapter introduces a novel blockchain and cryptocurrency system based on Proof of Behavior (PoB), designed to reward specific human behaviors while maintaining a permissionless and energy-efficient structure. The proposed system utilizes Verifiable Delay Functions (VDFs) to achieve consensus, significantly reducing energy consumption compared to traditional Proof of Work systems. The chapter details the blockchain's structure, including a non-biased transaction fee system and the implementation of demurrage to encourage currency circulation. Security analysis draws parallels with Bitcoin's backbone protocol, demonstrating the system's robustness. We present potential applications, focusing on "EcoMobiCoin" and its application for rewarding eco-friendly mobility choices. The goal is to create incentive-based cryptocurrencies to promote desirable behaviors while maintaining the decentralized nature of blockchain technology. This work has benefitted of publication [**BehaviorChain**], though it has been thoroughly improved upon.

Outline of the current chapter

3.1 Introduction **57**
- 3.1.1 Contributions . . . 59
- 3.1.2 Related Work . . . 60
- 3.1.3 Outline . . . 62

3.2 Foundational Concepts **62**
- 3.2.1 Verifiable Delay Functions . . . 62
- 3.2.2 BLS Signature Scheme . . . 63
- 3.2.3 Proof of Behavior . . . 65

3.3 System Architecture **70**
- 3.3.1 Block Structure . . . 70
- 3.3.2 Mining and Consensus Mechanism . . . 72
- 3.3.3 Transaction System . . . 75
- 3.3.4 Economic Mechanism . . . 85

3.4 Checkpoint Mechanism **91**
- 3.4.1 Motivation and Design Principles . . . 92
- 3.4.2 pBFT Consensus Protocol . . . 97
- 3.4.3 Data Pruning and Storage Optimization . . . 104
- 3.4.4 Economic Incentives . . . 106

3.5 System Analysis **107**
3.5.1 PoB Expiration 107
3.5.2 Energy Consumption 108
3.5.3 Monetary Policy Analysis 110
3.6 Security Analysis **115**
3.6.1 Bitcoin Backbone Protocol Correspondance 115
3.6.2 Special Attacks on PoB Blockchains 119
3.6.3 Checkpoint Security Analysis 134
3.7 Application and Implementation **138**
3.7.1 Applications of PoB Blockchains 139
3.7.2 Prototype Implementation: EcoMobiCoin 139
3.8 Conclusion **141**

3.1 Introduction

The emergence of "using blockchains for good" represents a significant evolution in the blockchain ecosystem, reflecting a broader shift towards leveraging technology for social and environmental benefit. This concept developed as the potential of blockchain's core attributes – decentralization, transparency, immutability, and programmability – was recognized beyond financial applications. As the technology matured, innovators began exploring its application to complex societal issues, aligning with global initiatives like the UN's Sustainable Development Goals. This shift was partly driven by a desire to counter criticisms of blockchain's energy consumption and perceived lack of real-world utility. The movement gained momentum through the creation of social impact tokens, integration with IoT and AI, and adoption by corporations, governments, and NGOs for various social initiatives. Academic research, the development of more energy-efficient consensus mechanisms, and the concept of tokenizing social impact further legitimized and expanded this field. Community-driven projects demonstrated blockchain's potential for grassroots change, while the combination with other emerging technologies opened new possibilities for creating efficient systems for social good. This evolution towards "blockchain for good" signifies a crucial shift in the technology's perception and utilization, reflecting a growing trend in the tech industry to create solutions that not only drive innovation but also contribute positively to society and the environment. In this chapter we address two aspects of "using blockchain for good": (i) incentivize specific human behavior with cryptocurrency rewards; and (ii) mitigating blockchains' excessive energy consumption.

The endeavor to link human behavior with blockchain-based reward systems presents a multifaceted challenge, primarily manifested in the issues of centralization and data verification. The centralization problem stems from the inherent contradiction between the decentralized ethos of blockchain technology and the centralized mechanisms often employed to validate and verify human actions. Many projects in this domain, such as Sweatcoin [**sweatcoin**, **sweatcoin_short**, **sweatcoin_long**] for fitness tracking or SolarCoin [**solarcoin**] for renewable energy production, rely on centralized authorities or proprietary algorithms to authenticate user-reported behaviors. This centralization not only undermines the fundamental principles of blockchain technology but also introduces single points of failure and potential manipulation. The reliance on centralized validation mechanisms creates a paradoxical situation where a purportedly decentralized system becomes dependent on centralized arbiters, thereby negating many of the advantages that blockchain technology offers, such as trustlessness and disintermediation. Closely intertwined with the centralization issue is the challenge of data verification and integrity. The authentication of human behaviors in a decentralized, tamper-resistant manner presents significant technical

and logistical hurdles. Many existing systems rely on easily manipulable data sources, such as GPS coordinates for location-based activities or self-reported metrics for various behaviors. The vulnerability of these data sources to falsification or manipulation severely compromises the integrity of the entire reward system. Moreover, the need for robust verification mechanisms often conflicts with user privacy concerns, creating a tension between the desire for accurate data and the protection of personal information. This dilemma is further exacerbated in blockchain systems, where data immutability means that any erroneous or fraudulent information, once recorded, becomes permanently enshrined in the ledger. The challenge, therefore, lies in developing verification protocols that are simultaneously decentralized, resistant to manipulation, respectful of user privacy, and capable of producing irrefutable proofs of behavior that can be consensually validated by the network participants without relying on trusted intermediaries.

The energy consumption of blockchain networks, particularly those employing Proof of Work (PoW) consensus mechanisms, has emerged as a critical concern in the technological and environmental spheres. Bitcoin, as the progenitor and most prominent example of blockchain technology, exemplifies this issue with its staggering energy requirements. Recent analyses indicate that Bitcoin's annual energy consumption is comparable to that of entire nations [**BTC_Power**], a scale that underscores the magnitude of this technological phenomenon's environmental impact. This prodigious energy utilization is seen by many as a major hindrance in blockchain democratization and acceptance.

The stark reality of PoW's energy intensity has catalyzed exploration into alternative consensus mechanisms, with Proof of Stake (PoS) emerging as a leading contender. PoS protocols promise drastically reduced energy consumption by eliminating the computational race inherent to PoW. However, this transition is not without its challenges and potential trade-offs, including concerns about centralization and security that require careful consideration. The environmental implications of blockchain technology extend beyond mere energy consumption to encompass the generation of electronic waste. The rapid obsolescence of specialized mining hardware, driven by the competitive nature of PoW mining, contributes to a growing e-waste problem. This aspect of blockchain's environmental impact, while less discussed than energy consumption, represents a significant ecological concern, particularly given the toxic materials often present in electronic components and the challenges associated with their proper disposal or recycling.

Furthermore, the energy consumption issue has profound implications for the scalability and public perception of blockchain technology. As these networks grow in size and adoption, their energy requirements tend to increase proportionally, raising questions about the long-term viability and sustainability of current blockchain architectures. This scaling problem not only poses technical challenges but also significantly impacts public opinion. The narrative of blockchain as an energy-intensive, environmentally detrimental technology has gained traction in public discourse, potentially hampering broader adoption and integration of blockchain solutions across various sectors. This perception challenge is particularly acute given the increasing global emphasis on sustainability and corporate social responsibility, potentially limiting blockchain's application in environmentally conscious industries or regions with strict energy regulations.

Our aim in this chapter is to propose a solution to overcome the aforementioned challenges: we propose a generic blockchain mined by using Proof of Behavior (PoB) [**PoB**]. PoB aims to replace PoW by a specific human behavior – preferably an ecological one – to avoid consuming energy in hash computations on the one hand, and to incentivize and further the development of eco-responsible habits on the other.

3.1.1 Contributions

Our research significantly extends the concept of Proof of Behavior (PoB) introduced in [**PoB**], providing a comprehensive framework for its implementation in blockchain technology. We present a formal definition of a generic PoB, encompassing all necessary elements for its verification and impact assessment. This formalization serves as a foundation for a novel consensus mechanism that integrates PoB with well-established cryptographic primitives.

The proposed blockchain architecture is designed to be energy-efficient while simultaneously promoting specific behaviors inherent to the PoB. By incorporating Verifiable Delay Functions (VDFs) [**VDF0**], we achieve a substantial reduction in energy consumption, limiting it to a maximum of a few CPUs per miner. This approach stands in stark contrast to the unlimited parallelization characteristic of PoW based consensus mechanisms, offering a more sustainable alternative.

Our system innovates in its reward structure, directly incentivizing desired behaviors through the blockchain's native cryptocurrency. We introduce an optional mechanism allowing miners to incorporate and reward the PoBs of non-miners, fostering an inclusive ecosystem where all participants exhibiting the target behavior can be compensated.

We have devised a novel transaction fee structure that mitigates miner bias towards transaction size. This system requires specific conditions to prevent exploitation, fundamentally altering the traditional fee model: senders pay a price proportional to the transaction size, while miners receive a fixed reward regardless of transaction magnitude. We call it the fuel and bounty system. Additionally, we propose the integration of temporal demurrage, a concept advocated by economists like Silvio Gesell [**Gesell**], causing coins to depreciate over time. We demonstrate how this can be seamlessly incorporated into our non-biased transaction model.

The versatility of this cryptocurrency framework opens up numerous applications, with potential benefits for many existing projects. As a concrete implementation, we detail *EcoMobiCoin*, a cryptocurrency mined through ecologically sound mobility choices. To the best of our knowledge, this represents the first comprehensive framework for utilizing generic behaviors in blockchain consensus and reward mechanisms, marking a significant advancement in the field of behavior-incentivized cryptocurrencies.

3.1.2 Related Work

The field of blockchain technology has witnessed significant interest in creating cryptocurrencies that incentivize specific human behaviors. This section examines existing approaches across different domains, analyzing their architectural choices and limitations to contextualize our contributions.

Core Behavior-Incentive Cryptocurrencies

Sweat/Sweatcoin represents the most developed example of behavior-based cryptocurrency [**sweatcoin**, **sweatcoin_short**, **sweatcoin_long**]. Built on the Near blockchain, Sweat rewards physical activities like walking, biking, and swimming, with only the first 5,000 daily steps earning Sweat tokens (additional steps earn the centralized Sweatcoin token). The system implements demurrage as a non-activity fee and relies exclusively on SweatCo Ltd as its Movement Validator at launch. Critically, verification depends on analyzing raw data from recording devices, making it vulnerable to data synthesis attacks. This exemplifies the core challenge our work addresses: achieving tamper-proof behavior verification in a decentralized system.

Ecocoin illustrates the centralization pitfalls common in this domain [**ecocoin**]. While claiming to be a cryptocurrency rewarding ecological behaviors (green commuting, low energy consumption, sustainable food choices), it relies entirely on human inspectors or certified vendors for validation. The project provides no specification of its underlying distributed ledger, offers no security guarantees, and appears abandoned since 2019. Notably, ecocoins are backed by trees held in escrow, representing an interesting but ultimately centralized approach to value backing.

Solarcoin demonstrates sector-specific behavior incentivization, rewarding solar energy production at 1 solarcoin per 1 MWh [**solarcoin**]. Originally using its own blockchain, it migrated to the Energy Web Chain [**energy_web_chain**] with Proof of Authority consensus. The system's fundamental flaw lies in requiring proof submission to the centralized Solarcoin Foundation, which then rewards producers – a clear violation of blockchain's decentralization principles.

Broader Behavior-Incentive Landscape

Several other projects explore similar concepts with varying approaches. Clinicoin operates a permissioned blockchain connecting health activity loggers, validators, and healthcare providers [**clinicoin**]. Multiple fitness-focused platforms (Lympo, Bitwalking, EarthMiles, Movecoin, Gizer, Step n) implement variations of activity-to-token conversion, typically using centralized validation or existing blockchain infrastructures rather than novel consensus mechanisms.

The environmental sector shows particular activity, with numerous carbon trading platforms (Climatecoin, CarbonX, Nori, ClimateTrade, Earth Dollar) and energy-focused projects (WePower, Power Ledger, Verv, EnergiToken, CoolBitX) emerging. However, these generally focus on trading existing environmental assets rather than creating new consensus mechanisms for behavior verification. Plastic Bank represents an interesting physical-to-digital bridge, establishing recycling centers where plastic waste exchanges for digital tokens.

Alternative Consensus Mechanisms

Freicoin provides crucial precedent for our demurrage implementation [**freicoin**]. It implements temporal demurrage by charging holding fees that increase over time, encouraging circulation and preventing hoarding. This demonstrates the feasibility of integrating economic mechanisms that discourage accumulation within cryptocurrency systems.

Primecoin attempts to replace PoW's "wasteful" computation with useful mathematical work – specifically finding Cunningham and bi-twin prime chains [**primecoin**, **primecoin_paper**]. While conceptually appealing, unsuccessful mining attempts still result in discarded work, and the system doesn't address behavior incentivization. This highlights the challenge of replacing PoW while maintaining both utility and decentralization.

Elapsed Time consensus protocols (Proof of Elapsed Time [**PoET**], Proof of Luck [**PoL**]) rely on Trusted Execution Environments to enforce waiting periods before mining rights. While energy-efficient, they require specialized hardware and trusted components, limiting true permissionlessness.

Critical Analysis and Research Gaps

This survey reveals fundamental unresolved challenges in behavior-incentive cryptocurrencies:

1. **The Verification Paradox:** Projects achieving true decentralization (like Primecoin) do not incentivize specific behaviors, while behavior-focused projects (like Sweat, Solarcoin) rely on centralized validation. This creates a false dichotomy our work seeks to resolve.

2. **Data Integrity vs. Privacy:** Systems using GPS traces or self-reported metrics face manipulation vulnerabilities, while privacy-preserving approaches struggle with behavior verification. Existing solutions typically sacrifice one for the other.

3. **Permissionlessness vs. Energy Efficiency:** Most energy-efficient consensus mechanisms require trusted validators or predetermined participant sets, undermining blockchain's core value proposition of open participation.

4. **Generic vs. Specific Approaches:** Current systems are typically designed for single behavior types, lacking the flexibility to adapt to diverse behavioral incentivization needs.

Our Proof of Behavior framework addresses these gaps by providing a generic, energy-efficient consensus mechanism that maintains permissionlessness while enabling tamper-resistant behavior verification across diverse application domains.

Positioning of our Contribution

The evolution of behavior-incentive cryptocurrencies reveals a clear developmental trajectory from centralized systems claiming blockchain benefits (Ecocoin) to real blockchains with centralized behavior validation (Sweat, Solarcoin), yet no existing system simultaneously achieves the four critical properties required for robust behavior incentivization: permissionlessness, energy efficiency, decentralized verification, and behavioral genericity. While Primecoin maintains permissionlessness but sacrifices energy efficiency, and Sweat achieves efficiency but requires centralized validation, our PoB framework represents the first system to integrate behavior verification directly into the consensus mechanism itself rather than layering validation over existing blockchain architectures. This fundamental architectural innovation enables unique capabilities impossible in existing systems: cross-domain behavior integration within a single blockchain, trustless validator networks without reliance on centralized authorities, and adaptive incentive structures through configurable behavior valuation mechanisms. Furthermore, our VDF-based approach inverts the traditional behavior-first, blockchain-second design philosophy by creating a consensus-first, behavior-agnostic framework where energy efficiency emerges from sequential computation constraints rather than trusted authority delegation. This positions our work as the logical culmination of the field's progression toward systems that preserve blockchain's core decentralization principles while enabling sophisticated behavior incentivization across arbitrary domains.

3.1.3 Outline

The structure of this chapter is organized as follows:

Section 3.2 defines key cryptographic primitives. It defines Verifiable Delay Functions in Section 3.2.1 and then defines Boneh–Lynn–Shacham signatures in Section 3.2.2. Along with this chapter's flagship Proof of Behavior in Section 3.2.3, these notions will be the foundation to building our blockchain.

Section 3.3 proposes a novel blockchain architecture that incorporates the PoB as its core consensus mechanism. This section details the intricate interplay between PoB and blockchain dynamics, elucidating the system's structure, transaction mechanisms, and block validation processes.

Section 3.4 defines checkpoints and although it adds a layer on top of the blockchain, it optimizes space and security.

Section 3.5 analyzes in detail the ramifications of our PoB-based blockchain and its constructions. It explores the blockchain's energy consumption, the implications of our special fee mechanism and the benefits of temporal demurrage in the case of a PoB-based blockchain.

Section 3.6 is devoted to a thorough security analysis of the PoB blockchain. We examine the system's resilience against various attack vectors, analyze its consistency and liveness properties, and compare its security guarantees with established blockchain protocols.

Section 3.7 explores the practical implications of our PoB blockchain, with a primary focus on our flagship application, *EcoMobiCoin* and the current state of our implementation efforts. We discuss the practical challenges encountered, the solutions devised, and provide insights into the system's performance and scalability based on preliminary results.

The chapter concludes in Section 3.8 with a comprehensive synthesis of our findings, a critical evaluation of the proposed system's strengths and limitations, and a discussion of future research directions in the realm of behavior-incentivized blockchain technologies.

3.2 Foundational Concepts

In this section we introduce and formally define Verifiable Delay Functions and Boneh–Lynn–Shacham signatures. Then we define the Proof of Behavior. We will have everything in place to construct our blockchain in the subsequent section.

3.2.1 Verifiable Delay Functions

Verifiable Delay Functions (VDFs) [**VDF0**] are cryptographic primitives that have gained significant attention in recent years due to their potential applications in blockchain technology, distributed systems, and other areas of computer science. A VDF is a function that takes a specified amount of time to compute, even on a parallel computer, but can be verified quickly. More formally:

Definition 3.1: VDF Scheme

A *VDF* scheme consists of three algorithms, Setup, Eval and Verify_{VDF}, such that:

$\mathsf{Setup}(\lambda, t) \to pp = (ek, vk)$ is a randomized algorithm that takes a security parameter λ and a desired puzzle difficulty t and produces public parameters pp that consist of an evaluation key ek and a verification key vk. We require Setup to be polynomial time in λ. By convention, the public parameters specify an input space $\mathbb{X}$ and an output space $\mathbb{Y}$. We assume that $\mathbb{X}$ is efficiently sampleable. Setup might need secret randomness, leading to a scheme requiring a trusted setup. For meaningful security, the puzzle difficulty t is restricted to be sub-exponentially sized in λ.

$\mathsf{Eval}(ek, x) \to (y, \pi)$ takes as input the public parameter ek and $x \in \mathbb{X}$ and produces an output $y \in \mathbb{Y}$ and a (possibly empty) proof π. Eval may use random bits to generate the proof π but not to compute y. For all pp generated by $\mathsf{Setup}(\lambda, t)$ and all $x \in \mathbb{X}$, algorithm $(eval, ek, x)$ must run in parallel time t with $poly(\log(t), \lambda)$ processors.

$\mathsf{Verify}_{VDF}(vk, x, y, \pi) \to \{Accept, Reject\}$ is a deterministic algorithm that takes as inputs the public parameter vk, an entry $x \in \mathbb{X}$, the supposed corresponding output $y \in \mathbb{Y}$ and the supposed corresponding proof π and outputs *Accept* or *Reject*. Verify_{VDF}

must run in total time polynomial in $\log(t)$ and λ.

The following characteristics stem from the definition:

Sequentiality: Any honest user can generate the pair $(y, \pi) \leftarrow \mathsf{Eval}(pp, x)$ in t sequential steps, while any parallel-machine adversary with a polynomial number of processors cannot distinguish the output y from random values in significantly fewer steps.

Efficient verifiability: The algorithm Verify_{VDF} is aimed to be as fast as possible for honest users to run. In particular, the total time should be of the order $O(polylog(t))$.

Uniqueness: For every input x, it is difficult to obtain a value y such that $\mathsf{Verify}_{VDF}(pp, x, y, \pi) \rightarrow Accept$, but $y \neq \mathsf{Eval}(pp, x)$.

Two possible constructions are Wesolowski's construction [**VDF2**] and Pietrzak's construction [**VDF1**]. They are both efficient and simple, though the first is a little more efficient and the latter a little simpler. A detailed comparison and analysis of both constructions can be found in [**VDF_survey**]. Both constructions are not post-quantum secure.

3.2.2 BLS Signature Scheme

The previous discussion established that every finalized checkpoint must carry collective attestation from a large set of witnesses, yet storing individual signatures would create prohibitive overhead. What is needed, therefore, is a cryptographic tool that allows multiple signatures on the same message to be combined into a single compact proof without sacrificing verifiability. Aggregate signature schemes provide exactly this functionality through its unique aggregation property.

Definition 3.2: Aggregate Signature Scheme [boneh2003aggregate]

n *aggregate signature scheme* allows multiple signatures, possibly generated by distinct signers on (the same or different) messages, to be compressed into a single short signature. Formally, it consists of four polynomial-time algorithms:

- $\mathsf{KeyGen}(\lambda) \rightarrow (sk, pk)$: on input a security parameter λ, output a secret key sk and corresponding public key pk.
- $\mathsf{Sign}(sk, m) \rightarrow \sigma$: on input a secret key sk and a message $m \in \{0,1\}^*$, output a signature σ.
- $\mathsf{Aggregate}(\{\sigma_i\}_{i=1}^n) \rightarrow \sigma^*$: on input a set of signatures $\sigma_1, \ldots, \sigma_n$, output an aggregate signature σ^*.
- $\mathsf{Verify}(\{pk_i, m_i\}_{i=1}^n, \sigma^*) \rightarrow \{0,1\}$: on input a set of public keys and their corresponding signed messages, together with an aggregate signature σ^*, output 1 if and only if all signatures are valid.

Property 3.3: Correctness

Correctness requires that for any key pairs $(sk_i, pk_i) \leftarrow \mathsf{KeyGen}(\lambda)$ and messages m_i, if $\sigma_i \leftarrow \mathsf{Sign}(sk_i, m_i)$ for all i, then

$$\mathsf{Verify}(\{(pk_i, m_i)\}_{i=1}^n, \mathsf{Aggregate}(\{\sigma_i\}_{i=1}^n)) = 1.$$

BLS Aggregate Signatures. The Boneh–Lynn–Shacham (BLS) signature scheme [**boneh2001short**] is the canonical example of an aggregate signature scheme [**boneh2003aggregate**]. It operates over bilinear pairings on elliptic curves and produces constant-size signatures. Its distinctive feature is that signatures from different signers on the same message can be combined multiplicatively into a single short aggregate signature. Verification of the aggregate requires only one pairing equation, provided the verifier knows the set of participating public keys. This aggregation property makes BLS signatures particularly suitable for consensus protocols and checkpointing mechanisms (Section 3.4), where hundreds of individual signatures must be compressed into one compact proof without sacrificing verifiability.

Example 3.4

Suppose three miners M_1, M_2, and M_3 have to establish a checkpoint. Each miner holds a secret key sk_i with corresponding public key $pk_i = g^{sk_i}$.

1. Each miner computes a signature on the same checkpoint message m (*e.g.*, the state root at height h):
$$\sigma_i = H(m)^{sk_i}, \quad i \in \{1, 2, 3\}.$$
2. The individual signatures $\sigma_1, \sigma_2, \sigma_3$ are combined into a single aggregated signature:
$$\sigma = \sigma_1 \cdot \sigma_2 \cdot \sigma_3.$$
3. To verify that all three witnesses signed the message, a verifier only needs to check:
$$e(\sigma, g) \stackrel{?}{=} e(H(m),\ pk_1 \cdot pk_2 \cdot pk_3).$$

This aggregation means that instead of carrying and verifying three separate signatures, the system stores only one short signature σ and performs a single pairing check. In large-scale scenarios with hundreds of witnesses, this drastically reduces communication and storage overhead, while still proving that a supermajority signed the checkpoint.
Note that this aggregation does not require a trusted party: any node can multiply the signatures once they have been collected. However, the checkpoint must also record the set of witnesses who signed, since the verifier must know which public keys to use in the verification equation. An illustration is provided in Figure 3.1.

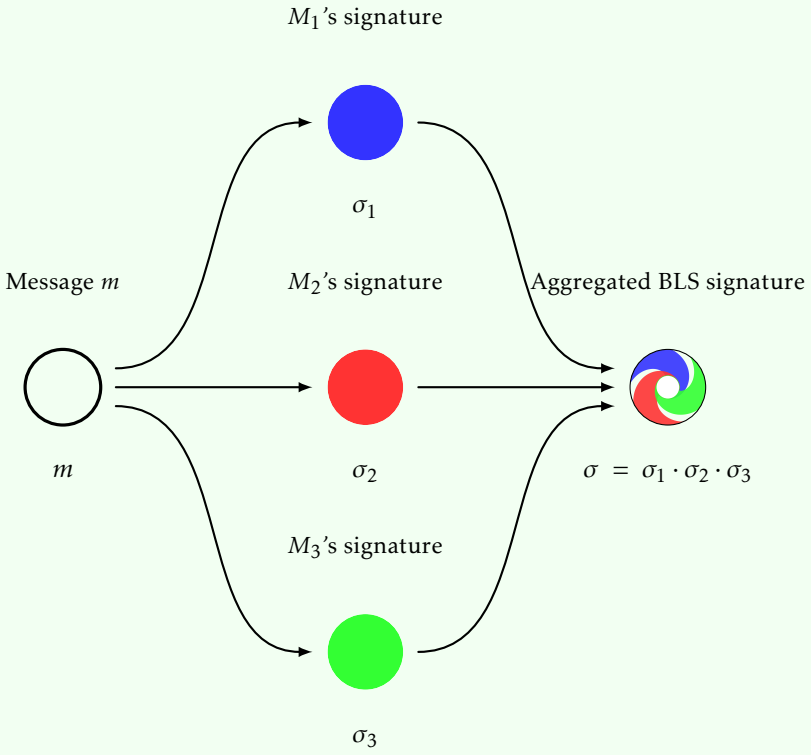


Figure 3.1: Illustration of BLS Signature

3.2.3 Proof of Behavior

Our generic blockchain is mined using a *Proof of Behavior* (PoB) [**PoB**]. The blockchain's genericity consists of the type of behavior the blockchain seeks to promote. It could adapt for instance to walking, using public transportation, recycling, carpooling...

Whatever the type of behavior, a PoB is caracterized by: (i) **who** did the behavior; (ii) **when** it was done; and **what** was the actual behavior itself.

To authenticate a behavior, *i.e.*, to make it a *Proof* of Behavior, these information must be signed together by a validator. This signature binds the **who** with the **when** with the **what** and prevents falsification of the behavior. The validator could be operated by a human actor or by an automatic verification mechanism such as a Trusted Platform Module.

Definition 3.5: Proof of Behavior

A *Proof of Behavior (PoB)* is a tuple $\pi = (b \| \sigma_b)$ where b is the tuple $b = (pk, ts, data)$ containing: pk a public key representing the identity of the producer (**who**); ts the timestamp of the behavior (**when**); and $data$ some auxiliary data about the behavior itself (**what**). σ_b is the signature of b by a given validator able to verify that the behavior included in $data$ was done at timestamp ts by pk.

We need a function to verify the proof. It should take as input a PoB and output whether it is valid or invalid. However we choose to add to this function another role: valuation of the PoB's

impact. We call this function Quantify.

Definition 3.6: Quantify

Quantify(π) $\rightarrow v$ is a deterministic algorithm that takes as input the PoB π, and outputs a non-negative number $v \geq 0$ that represents the value of said behavior.
The Quantify function serves dual roles: (i) verification, by distinguishing valid ($v > 0$) from invalid ($v = 0$) behaviors; (ii) valuation, by assigning varying degrees of importance to different behaviors when $v > 0$.
For certain types of PoB, PoBs may be of equal importance. In these cases, Quantify outputs only either 0 (invalid) or 1 (valid). In this case the PoB is *non quantifiable*.

Example 3.7

Let us consider the behavior type to be mobility using public transportation. Let users tap their card on the terminal during entrance only. Such a terminal can be seen as a proxy of the public transportation organization. When the user taps their card on the terminal, the terminal, acting as the validator, creates a PoB $\pi = (b||\sigma_b)$ with $b = (pk, ts, data)$ where pk is the public key of the user, ts is a timestamp generated by the terminal's internal clock at the exact moment of card tap, and $data$ is data about the journey being travelled.
Crucially, the timestamp ts is not provided by the user but generated by the validator (terminal) itself, ensuring temporal integrity. The terminal then cryptographically signs the entire tuple b, binding the user's identity, the validator-generated timestamp, and the journey data together. This prevents any party from later claiming the behavior occurred at a different time.
To verify and quantify the proof, we plug it into Quantify which should output 1 if the PoB is valid and 0 otherwise. The PoB scheme here is non quantifiable. Suppose however that the user is required to tap his card at entry *and* exit points of the network (which is the case in many places, such as the Paris RER network), then the PoB scheme is quantifiable and the Quantify function could for example output a number proportional to the distance covered.
This is illustrated in Figure 3.2.

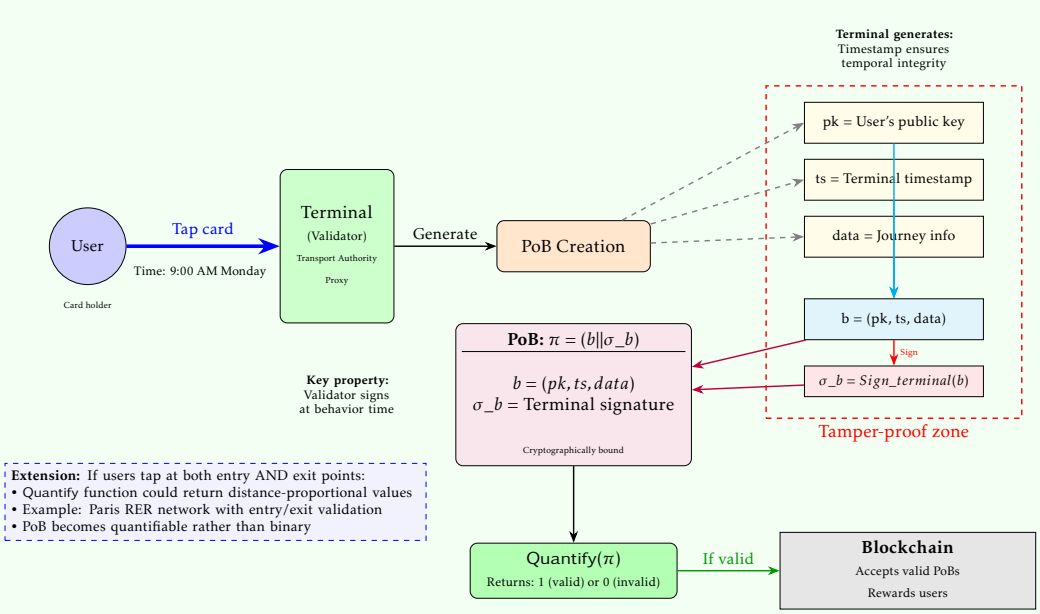


Figure 3.2: Example of PoB creation in metro transportation services

Definition 3.8: PoB Expiration

Let π be a Proof of Behavior with timestamp ts_π, and let ts_{newest} denote the timestamp of the newest PoB in the blockchain (that may be different from the most recently added PoB to the blockchain). Given a predetermined expiration threshold $\text{exp} \in \mathbb{R}^+$, we say that π is *expired* if and only if:

$$ts_{\text{newest}} - ts_\pi > \text{exp}$$

PoB timestamps are compared exclusively against other PoB timestamps within the blockchain, not against miners' local clocks. This design choice eliminates the necessity for global time synchronization between network participants while ensuring deterministic validation of expiration status across all nodes.

Example 3.9

Let exp = 6 hours (expiration threshold).
Consider a blockchain where PoBs are added in the sequence in Table 3.1:

Table 3.1: Order of PoB added to Blockchain

Block	PoB Added	PoB Timestamp	Behavior
B_1	π_A	8:00 AM Monday	Alice takes metro
B_2	π_B	3:00 PM Monday	Bob rides bike
B_3	π_C	9:00 AM Monday	Carol takes bus
B_4	π_D	10:00 AM Tuesday	Dan walks
B_5	π_E	7:00 AM Monday	Eve carpools

Let us do a step-by-step expiration analysis. An illustration is provided in Figure 3.3.

After Block B_1:

- Newest PoB timestamp: 8:00 AM Monday (π_A)
- No PoBs can be expired (only one PoB exists)

After Block B_2:

- Newest PoB timestamp: 3:00 PM Monday (π_B)
- Check π_A: |3:00 PM Mon - 8:00 AM Mon| = 7 hours > 6 hours
- π_A is now expired!

After Block B_3:

- Newest PoB timestamp: **still** 3:00 PM Monday (π_B)
- Note: π_C (9:00 AM Monday) is older than π_B
- Check π_C: |3:00 PM Mon - 9:00 AM Mon| = 6 hours $\not>$ 6 hours
- π_C is valid (exactly at threshold)
- π_A remains expired

After Block B_4:

- Newest PoB timestamp: 10:00 AM Tuesday (π_D)
- Now the newest timestamp has changed!
- Check all PoBs against new newest:
  - π_A: |10:00 AM Tue - 8:00 AM Mon| = 26 hours > 6 hours → expired
  - π_B: |10:00 AM Tue - 3:00 PM Mon| = 19 hours > 6 hours → expired
  - π_C: |10:00 AM Tue - 9:00 AM Mon| = 25 hours > 6 hours → expired

After Block B_5:

- Block B_5 attempts to add π_E (7:00 AM Monday)
- Current newest PoB: still π_D at 10:00 AM Tuesday
- Check π_E: |10:00 AM Tue - 7:00 AM Mon| = 27 hours > 6 hours
- π_E is rejected - it's already expired upon submission!

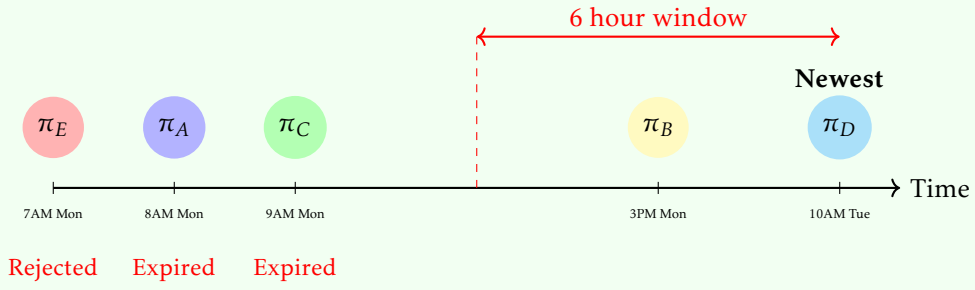


Figure 3.3: Example of PoB expiration

More generally, for any PoB π with timestamp ts_π:

$$\text{Status}(\pi) = \begin{cases} \text{Valid} & \text{if } ts_{\text{newest}} - ts_\pi \leq \text{exp} \\ \text{Expired} & \text{if } ts_{\text{newest}} - ts_\pi > \text{exp} \end{cases}$$

where $ts_{\text{newest}} = \max\{ts_{\pi'} : \pi' \in \text{Blockchain}\}$
The key insights of our PoB expiration system:

1. **Dynamic Expiration:** A PoB that was valid can become expired when a newer PoB is added (see π_A after B_2)
2. **Order Independence:** PoBs do not need to be added chronologically. π_C (9:00 AM) was added after π_B (3:00 PM)
3. **Relative Timestamps:** Expiration depends only on PoB timestamps, not miner clocks or block creation times
4. **Rejection at Entry:** PoBs that are already expired relative to the newest timestamp are rejected (see π_E)

Trust Model Discussion

The trust model represents the most critical challenge in behavior-based blockchains: how do we verify that a claimed behavior actually occurred without relying on centralized authorities?

The Centralization Problem. Consider our public transportation example. If only the Paris Metro authority can validate metro rides, then only Parisians can participate in that aspect of the blockchain. Worse, if someone wants to create a global mobility blockchain, they would need cooperation from every transportation authority worldwide – an impossible coordination challenge. This creates fragmented ecosystems where London has its own blockchain, Tokyo has another, and they cannot interoperate due to mutual distrust between authorities. Our framework supports multiple approaches to solve this trust problem:

Hardware-Based Validation: Trusted Platform Modules (TPMs) built into smartphones or IoT devices can cryptographically sign behavior data. For example, a smartphone's TPM could verify that GPS movement patterns are genuine and have not been spoofed by location-faking apps. The TPM acts as a neutral validator that no single authority controls.

Distributed Validator Networks: Multiple independent validators can collectively verify behaviors. For instance, ride-sharing companies (Uber, Lyft), public transit authorities, and independent mobility verification services could form a validator network. A behavior is only accepted if multiple validators from different organizations agree it's legitimate.

Cross-Verification Systems: Different data sources can validate each other. A claimed bike ride could be verified by combining GPS traces, accelerometer data showing pedaling motion, and heart rate increases consistent with physical exercise. No single data source is trusted alone.

Implementation Flexibility. The key insight is that trust models are implementation choices, not protocol limitations. A conservative implementation might require government transportation authorities as validators for maximum legal compliance. An innovative implementation might use only decentralized hardware validation for maximum censorship resistance. A pragmatic implementation might use hybrid approaches: government validators for official transit systems, but allow TPM-validated walking and cycling for broader participation.

This flexibility enables the same underlying blockchain protocol to serve both regulated environments (where authorities want control) and open environments (where users want freedom), depending on how the specific PoB instance is configured.

3.3 System Architecture

Having established the foundational cryptographic primitives – Verifiable Delay Functions, BLS signatures, and Proof of Behavior – we now present the comprehensive system architecture that integrates these components into a coherent blockchain protocol. This section details how PoB consensus operates in practice, describing the block structure that accommodates behavioral data, the mining mechanism that replaces computational competition with behavior-based selection, and the transaction system that includes our novel fuel and bounty fee structure. We also introduce the economic mechanisms, including demurrage implementation and reward distribution, that align network incentives with desired behavioral outcomes. Together, these architectural components create a permissionless, energy-efficient blockchain that directly incentivizes specific human behaviors while maintaining the security guarantees essential for decentralized operation.

3.3.1 Block Structure

The block architecture in our PoB blockchain represents a sophisticated integration of traditional blockchain elements with behavior-specific components. Unlike conventional blockchains that primarily contain transaction data, our blocks must accommodate Proof of Behavior records, VDF computations, and multiple transaction types while maintaining efficient verification and storage properties. The design carefully balances the need to include rich behavioral data with practical constraints on block size and validation complexity. This section formalizes the block structure, establishes validity criteria that ensure behavioral authenticity, and introduces allocation mechanisms that prevent any single content type from monopolizing blockchain resources. The resulting architecture creates a robust foundation for behavior-incentivized consensus while preserving the essential properties of decentralized ledger systems. The block architecture in the PoB blockchain represents a sophisticated amalgamation of cryptographic primitives and behavioral incentives.

Definition 3.10: Block

Each block B is defined as a tuple:

$$B = (data_B, \sigma_{data_B})$$

where $data_B$ encapsulates the block's content and σ_{data_B} represents the miner's cryptographic signature over this content. The $data_B$ component is further decomposed into:

$$data_B = (\pi_m, y, \pi_{VDF}, Tx_{miner}, (\pi_i, Tx_i)_{i \in I}, (Tx_j)_{j \in J})$$

Here, π_m denotes the miner's PoB, which itself is structured as $\pi_m = (b \| \sigma_b)$, where $b = (pk_m, ts, data_{PoB})$. The components y and π_{VDF} represent the output and proof of the VDF, respectively. Tx_{miner} is the coinbase transaction rewarding the miner, $(\pi_i, Tx_i)_{i \in I}$ is the set of included PoB proofs and their corresponding redemption transactions, and $(Tx_j)_{j \in J}$ represents the set of standard transactions.

An illustration of a full block is given in Figure 3.4.

The next sections define and explain the structure of a block from top to bottom. Firstly in Section 3.3.2 we explain the top part of the block, specifically how the miner's PoB and along with a VDF grant the miner mining rights. Then in Section 3.3.3 we introduce and define all different types of transactions. And finally in Section 3.3.4 we address the economic mechanisms used to reward PoB producers and to reward miners and the application of demurrage.

Block Size Constraints. The size of each block must be carefully limited to ensure network efficiency and prevent denial-of-service attacks. These constraints are particularly critical in PoB blockchains because behavioral data – including timestamps, location traces, and cryptographic signatures from validators – requires significantly more storage space than simple financial transactions.

The precise size limit depends on the specific PoB implementation and must balance several competing factors: behavioral data complexity, network propagation speed, validation overhead, and storage scalability. The maximum block size is therefore a blockchain-specific parameter that must be set during initialization.

To prevent either behavioral incentives or economic transactions from monopolizing blockchain resources, each block reserves fixed quotas for different content types:

- PoB allocation: Fixed percentage reserved for Proof of Behavior data and corresponding reward transactions.
- Transaction allocation: Fixed percentage reserved for standard currency transactions and coinbase rewards.

By reserving fixed quotas for each content type within a block, the protocol removes opportunities for miners to manipulate space allocation, while keeping the consensus rules straightforward. This design prevents adversaries from exploiting allocation mechanisms to crowd out either transactions or PoB records, and it allows every miner to verify block composition through deterministic, easily checkable rules rather than subjective judgments about which entries deserve priority.

Within each allocation, content follows simple ordering rules:

- PoB section: Prioritized by quantification value, then timestamp order.
- Transaction section: Strictly first-come-first-served, maintaining fee system neutrality.

This fixed allocation necessarily sacrifices some efficiency – during periods of low behavioral activity, unused PoB space cannot accommodate additional transactions, and vice versa. This design choice prioritizes system integrity over optimal resource utilization.

However, the benefits substantially outweigh the costs. The predictable allocation guarantees that neither use case can be completely crowded out by the other, ensuring the blockchain fulfills both its behavioral incentivization mission and its role as a transaction ledger. Moreover, the simplicity reduces implementation complexity and potential attack surfaces, contributing to overall system robustness.

Definition 3.11: Block Validity Criteria

A block B is valid if and only if:

1. All transactions in B must be valid.
2. $\forall \pi \in PoB_{list}, (\mathsf{Quantify}(\pi) > 0) \wedge (ts_{block} - ts_{\pi} \leq exp) \wedge (\pi \notin Blockchain)$ where ts_{block} is the the newest timestamp in the blockchain and ts_{π} is the timestamp of the PoB.
3. The recipient of the coinbase transaction Tx_{miner} must be the account associated with pk_m.
4. $\mathsf{Verify}_{sign}(pk_m, data_B, \sigma_{data_B}) \rightarrow Accept$.
5. Let $v \leftarrow \mathsf{Quantify}(\pi_m) \neq 0$ and $t \leftarrow \frac{H(B \| \pi_m) mod \delta}{v}$.
   Then, $Setup(\lambda, t) \rightarrow (ek, vk)$ and $\mathsf{Verify}_{VDF}(vk, B_{prev}, y, \pi_{VDF}) \rightarrow Accept$.

These criteria ensure the integrity, authenticity, and consistency of each block within the blockchain. The first criterion validates all included transactions. The second ensures that all PoB proofs are valid, recent, and not previously included. The third criterion verifies that the miner claiming the block reward is indeed the one who performed the necessary PoB. The fourth criterion ensures the block's cryptographic integrity through the miner's signature. Finally, the fifth criterion validates the correctness of the VDF computation, which is integral to the blockchain's consensus mechanism.

3.3.2 Mining and Consensus Mechanism

Our mining protocol diverges significantly from traditional approaches. Rather than competing through raw computational power, miners compete for mining rights based on their quantifiable behaviors. The probability of a miner being selected as the next block producer is proportional to the magnitude of their behavior, as quantified by their Proof of Behavior (PoB).

Definition 3.12: VDF Delay Parameter

To extend the blockchain from a given block B, a miner with a PoB π_{miner} must compute a VDF with a delay parameter t. This parameter is derived from the PoB and is defined as:

$$t = \frac{H(B \| \pi_{miner}) \mod \delta}{\mathsf{Quantify}(\pi_{miner})}$$

where $H(\cdot)$ denotes a cryptographic hash function, δ represents a difficulty parameter, and $\mathsf{Quantify}(\cdot)$ is a function that assigns a numerical value to a given PoB.

The blockchain's operational parameters, enumerated in Table 3.2, are integral to its functioning. These parameters, with the exception of the difficulty parameter δ, are immutable post-blockchain instantiation. Any modification to these fixed parameters necessitates a fork in the blockchain, a process elucidated in Section 3.4.

The difficulty parameter δ is unique in its mutability, being subject to periodic adjustments to maintain consistent block production rates in the face of fluctuating network conditions.

Table 3.2: Blockchain parameters

Symbol	Description
exp	PoB expiration window
δ	Mining difficulty parameter
P	Difficulty adjustment period (in blocks)
Δt	Desired average block time
f_r	Transaction fee (fuel) rate
b	Fixed miner bounty per transaction
k	Miner's reward rate for own PoB
k'	Non-miner's reward rate for own PoB
k''	Miner's reward rate for including others' PoBs
min_{cur}	Minimal subdivision of currency
d_{block}	Demurrage rate per block
d_{day}	Demurrage rate per day
μ	Target expected blocks per checkpoint
ε	Epoch length (blocks between checkpoints)
Λ	Network synchrony bound (message delay upper bound)
T_{view}	pBFT view-change timeout deadline
θ	Checkpoint witness collateral

This innovative consensus mechanism effectively aligns the incentives of miners with the desired behaviors, creating a symbiotic relationship between the blockchain's security and the promotion of specific actions within the network's ecosystem.

Mining Dynamics. The inclusion of the previous block B as an input to the hash function introduces a crucial element of unpredictability into the mining process. This design choice ensures that the duration of the VDF for a given PoB remains indeterminate until the preceding block is known, effectively preventing premature initiation of VDF computations.

The relationship between the $\mathsf{Quantify}(\pi_{miner})$ function and the VDF delay parameter t introduces a bias towards more significant behaviors. PoBs with higher quantification values have a higher probability of yielding smaller t values, resulting in shorter VDF computation times. This mechanism aligns the mining advantage with the magnitude of the demonstrated behavior, reinforcing the incentive structure of the system.

As the number of active miners increases, the probability distribution of VDF durations shifts, with a decreased likelihood of all PoBs resulting in prolonged VDF computations. To maintain system stability and desired block generation rates, the difficulty parameter δ undergoes periodic adjustments. These adjustments occur at predefined intervals of P blocks.

The cryptographic hash function employed in this system acts as a pseudo-random function with a substantial output space (minimum 2^{256}). The operation $\mod \delta$ serves to constrain this output to a more manageable range of $\delta + 1$, allowing for fine-tuned control over mining difficulty.

For a target average mining time Δ_t, the adjustment of δ after every P blocks is computed as follows:

$$\delta \leftarrow \delta \cdot \frac{\Delta_t}{\overline{\Delta_t}} \tag{3.1}$$

where $\overline{\Delta_t}$ represents the observed average mining time over the preceding P blocks. To prevent extreme fluctuations, we propose implementing upper and lower bounds on the magnitude of δ adjustments, similar to the approach employed in the Bitcoin protocol [**Bitcoin**]. These bounds could be set at a minimum of one-quarter and a maximum of four times the previous δ value.

It is noteworthy that the VDF computation is independent of the current block's contents. While this design choice precludes parallelization of mining through simultaneous computation of multiple VDFs with varying block contents, it does introduce the potential for a miner to generate multiple valid sibling blocks concurrently. The security implications of this characteristic are examined in depth in Section 3.6.

Figure 3.5 provides a schematic representation of the mining process. Participants generate PoBs and transactions (denoted as Tx), which are aggregated in a common pool. Miners then utilize their personal PoBs to compute the VDF, with the first miner to complete this computation earning the right to publish a new block.

This mining mechanism effectively integrates behavior-based incentives with cryptographic primitives, creating a unique consensus protocol that aligns network security with the promotion of desired behaviors.

Relative advantage of VDF durations. It is important to emphasize that a shorter VDF duration does not guarantee that a miner will win the race to produce the next block. Suppose miner M_1 obtains a delay t_1 and miner M_2 obtains a delay t_2 with $t_1 < t_2$. If M_2 started evaluating its VDF strictly before M_1, it may complete the computation earlier despite having the longer delay. Consequently, the mining process does not reduce to a deterministic ranking by delay parameter alone, but instead depends jointly on the *relative values of t* and the *starting times of VDF computations*. This interaction ensures that the advantage conferred by higher quantification values is only probabilistic: shorter delays increase the likelihood of winning but never provide certainty.

Blockchain Forks. Forks in the PoB blockchain occur when multiple miners simultaneously produce valid blocks, creating competing chain extensions. This phenomenon manifests in several distinct scenarios within our consensus mechanism.

The most common fork scenario arises when multiple miners complete their VDF computations within a narrow time window and broadcast their blocks nearly simultaneously. Since each miner computes their VDF using their own PoB π_{miner}, different miners will generally have different delay parameters $t = \frac{H(B\|\pi_{miner}) \mod \delta}{Quantify(\pi_{miner})}$, leading to varying computation times. However, the probabilistic nature of this process means that occasionally, multiple miners may finish their VDFs close enough in time that their blocks propagate through the network before other miners become aware of competing solutions.

Another fork scenario emerges from network latency and partitioning effects. When a miner completes a VDF and broadcasts their block, network delays may prevent other miners from receiving this information before they complete their own computations. This creates temporary forks that must be resolved through the chain selection mechanism.

Main Chain Determination. In accordance with the principles established in the Bitcoin protocol [**Bitcoin**], the determination of the main chain in our system follows a cumulative difficulty rule. Specifically, the main chain is defined as the subset of the blockchain that possesses the highest aggregate difficulty. This aggregate is computed as the sum of the difficulty parameters δ associated with each constituent block in the chain.

Definition 3.13: Main Chain

Formally, for a set of competing chains $C = \{C_1, C_2, \ldots, C_n\}$, where each chain C_i consists of blocks $\{B_{i,1}, B_{i,2}, \ldots, B_{i,m_i}\}$, the main chain C_{main} is determined by:

$$C_{main} = \arg\max_{C_i \in C} \sum_{j=1}^{m_i} \delta_{i,j}$$

where $\delta_{i,j}$ represents the difficulty parameter of block $B_{i,j}$.

This criterion ensures that the main chain not only reflects the longest sequence of blocks but also represents the chain that has required the most cumulative computational effort to produce, as measured by the difficulty parameters. This approach provides resilience against potential attacks and maintains consistency with the underlying Proof of Behavior consensus mechanism.

Note that the possibility of a miner with a longer delay completing first due to an earlier start does not interfere with the chain selection rule. The determination of the main chain depends only on cumulative difficulty (Definition 3.13), and is thus insulated from variations in VDF completion order. This choice is deliberate: had we relied on cumulative delays as the selection criterion, the outcome could be biased by such timing effects and even exploited through block withholding strategies (Section 3.6.2). By using cumulative difficulty instead, the protocol guarantees that transient advantages in VDF completion do not affect long-term chain quality or security.

The VDF-based nature of our system naturally creates a lottery-like mechanism where miners with shorter delay parameters have higher probabilities of mining the next block. An alternative chain selection rule could prioritize chains with the shortest cumulative delay parameters, which would more directly reflect this lottery characteristic. However, we deliberately chose cumulative difficulty (δ values) as our main chain criterion to prevent a critical attack vector. Under a shortest-delay rule, an attacker could mine a block with an exceptionally short delay parameter, withhold it from publication, and potentially mine additional blocks on top of it using selfish mining strategies. This private chain could then be published to reorganize the blockchain and enable double-spending attacks, as the network would be forced to accept the chain with the shortest cumulative delays regardless of when it was revealed. The cumulative difficulty approach provides stronger security guarantees against such strategic block withholding attacks.

3.3.3 Transaction System

The transaction framework of our blockchain encompasses three distinct categories, each serving a specific function within the ecosystem:

1. *Standard Transactions:* These facilitate the transfer of coins between network participants, serving as the primary means of value exchange within the system.
2. *Redemption Transactions:* These are responsible for allocating rewards to the producers of valid PoBs, thereby incentivizing the behaviors central to the system's design.
3. *Coinbase Transactions:* These transactions are unique to each block and serve to reward the miner responsible for block production, encompassing both the block reward and any transaction bounties.

Each of these transaction types plays a crucial role in maintaining the economic incentives and operational integrity of the blockchain. Detailed specifications for each transaction type will be elucidated in the subsequent sections.

Standard Transactions

Standard transactions facilitate value transfer between network participants through direct account balance modifications. Unlike UTXO-based systems that track individual transaction outputs, our model maintains a global state of account balances, enabling more efficient storage and simpler checkpoint operations.

Fuel and Bounty System. To address the limitations of traditional transaction fee mechanisms, we introduce a novel fuel and bounty system that achieves two primary objectives: (i) incentivizing miners to include transactions in blocks, and (ii) establishing a non-discriminatory system that does not inherently favor larger transactions over smaller ones.

Definition 3.14: Fuel and Bounty System

A fuel and bounty system is characterized by the tuple $(f_r, b, min_{sent}, min_{cur})$ where:

- $f_r \in]0, 1[$ is the fuel rate, representing the proportion of transaction value burned
- $b \in \mathbb{N}$ is the fixed bounty paid to miners per included transaction
- $min_{sent} \in \mathbb{N}$ is the minimum permissible transaction amount
- $min_{cur} \in \mathbb{N}$ is the smallest unit of the cryptocurrency

subject to the constraint:

$$f_r \cdot min_{sent} \geq b \geq min_{cur}$$

This inequality ensures economic viability by preventing miners from profiting through self-generated spam transactions.

When a user initiates a transaction of value V, the system operates as follows:

1. The sender pays a fuel cost of $F = V \cdot f_r$, which is permanently removed from circulation. It is effectively "burned".
2. The transaction outputs can total at most $V - F = V(1 - f_r)$.
3. The miner who includes the transaction receives a fixed bounty b.

This design decouples the miner's incentive from the transaction size, as the miner receives the same reward b regardless of whether the transaction transfers min_{sent} or a much larger amount.

Each standard transaction is structured to support multiple senders and recipients while incorporating the fuel and bounty mechanism. The transaction explicitly specifies the accounts from which funds are withdrawn and the amounts transferred to recipient accounts, with fuel being permanently burned from the total value transferred.

Definition 3.15: Standard Transaction

A standard transaction $\mathcal{T}$ is defined as a tuple:

$$\mathcal{T} = (\{(S_i, v_i, \mathrm{ctr}_i, \sigma_i)\}_{i=1}^{n}, \{(R_j, r_j)\}_{j=1}^{m})$$

where:

- $\{(S_i, v_i, \text{ctr}_i, \sigma_i)\}_{i=1}^{n}$ represents the set of n sender accounts, with:
  - S_i being the public key identifying sender account i
  - $v_i \geq \min_{sent}$ being the amount withdrawn from sender i's account
  - ctr_i being the next sequential transaction number for sender i's account
  - σ_i being the cryptographic signature authorizing the withdrawal: $\sigma_i = \text{Sign}_{sk_i}(\mathcal{T}_{\text{data}})$, where $\mathcal{T}_{\text{data}}$ includes all senders, recipients, amounts, and counters
- $\{(R_j, r_j)\}_{j=1}^{m}$ represents the set of m recipient accounts, with:
  - R_j being the public key identifying recipient account j
  - $r_j > 0$ being the amount credited to recipient j's account

Counter Mechanism for Replay Protection. Each account in the blockchain maintains its own counter, starting at 0 for new accounts. The counter represents the number of transactions that account has sent (not received). When an account participates as a sender in a transaction, it must provide the next sequential counter value.

Key properties of the counter system:

- Each sender account has its own independent counter
- Counters must be used sequentially (no gaps allowed)
- Receiving funds does not affect an account's counter
- In multi-sender transactions, each sender uses their own next counter
- A transaction is rejected if any sender's counter is incorrect

Note that the fuel amount is not explicitly stored in the transaction structure. Instead, it is deterministically computed from the transaction values using the system's fuel rate f_r. The fuel burned equals $f_r \cdot \sum_{i=1}^{n} v_i$ and is automatically calculated during transaction validation.

Definition 3.16: Transaction Validity

A standard transaction $\mathcal{T}$ is valid if and only if:

$$\begin{aligned}
&\text{(Balance Check)} && \forall i \in [1,n], && \text{Balance}(S_i) \geq v_i \\
&\text{(Conservation)} && \sum_{j=1}^{m} r_j \leq (1 - f_r) \cdot \sum_{i=1}^{n} v_i \\
&\text{(Minimum Value)} && \forall j \in [1,m], && \exists k \in \mathbb{N}^*, r_j = k \cdot \min_{cur} \\
&\text{(Authorization)} && \forall i \in [1,n], && \text{Verify}(S_i, \mathcal{T}_{\text{data}}, \sigma_i) = \text{Accept} \\
&\text{(Counter Validity)} && \forall i \in [1,n], && \text{ctr}_i = \text{CurrentCtr}(S_i) + 1
\end{aligned}$$

The counter validity check ensures that each sender is using exactly the next expected counter for their account. If sender S_i has previously sent k transactions, their current counter is k, and they must use counter $k+1$ for their next transaction.

Example 3.17: Simple Transfer

Alice wishes to send 50 coins to Bob. Let Alice's account balance have 150 coins and Bob's account have 30 coins. Their respective counters are 6 and 12.
The accounts' states before the transaction are shown in Table 3.3.

Table 3.3: State of participant accounts before the transaction

Participant	Role	Balance	Counter
Alice	Sender 1	150	6
Bob	Receiver 1	30	12

The transaction $\mathcal{T}_1$ is structured as:

- **Senders**:
$$\{(S_{\text{Alice}}, 100, \mathbf{7}, \sigma_{\text{Alice}})\}$$
- **Recipients**:
$$\{(R_{\text{Bob}}, 45)\}$$

The transaction execution (with $f_r = 0.1$) does the following (as illustrated in Figure 3.6):

- Alice must use counter = 7 (her current counter 6 + 1)
- Fuel automatically burned: $0.1 \cdot 50 = 5$ coins
- Alice's balance: $150 - 50 = 100$ coins, counter becomes 8
- Bob's balance: $30 + 45 = 75$ coins, counter remains 3 (unchanged as receiver)
- 5 coins are permanently removed from circulation
- Miner receives fixed bounty b

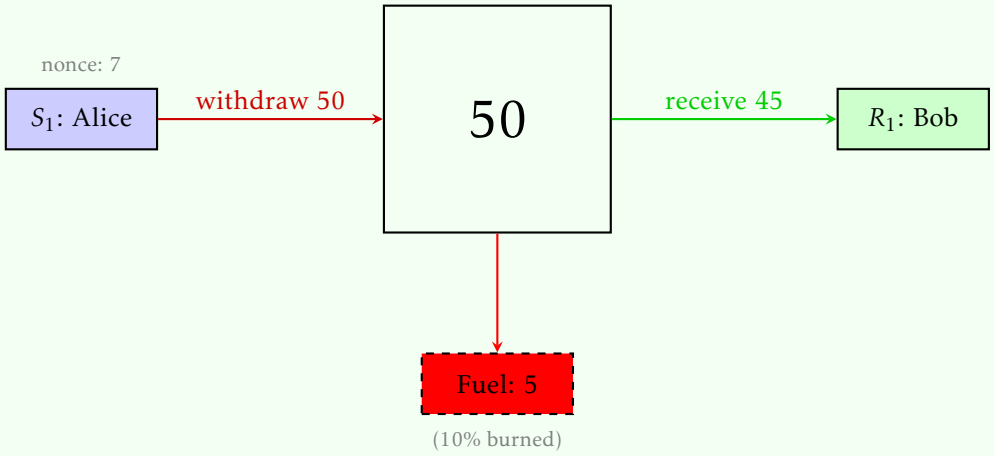


Figure 3.6: Simple standard transaction example illustration

The accounts' states after the transaction are shown in Table 3.4.

Table 3.4: State of participant accounts after the transaction

Participant	Role	Balance	Counter
Alice	Sender 1	100	7
Bob	Receiver 1	75	12

Note that Bob's counter does not change when receiving funds. Only when Bob sends a transaction will he need to use counter 4.

Example 3.18: Multi-Party Payment: Ride-Sharing

Consider a ride-sharing scenario where three passengers (Alice, Bob, and Carol) share a ride with driver Dave. Alice, Bob and Carol have different destinations and they split the cost of the ride according to their usage. The total fare is 60 coins. Alice pays 10, Bob pays 20 and Carol pays 30.
The accounts' states before the transaction are shown in Table 3.5.

Table 3.5: State of participant accounts before the transaction

Participant	Role	Balance	Counter
Alice	Sender 1	100	7
Bob	Sender 2	75	12
Carol	Sender 3	150	3
Dave	Recipient 1	500	45

The transaction $\mathcal{T}_2$ is structured as:

- **Senders**:

$$\{(S_{\text{Alice}}, 10, \mathbf{8}, \sigma_{\text{Alice}}),\\ (S_{\text{Bob}}, 20, \mathbf{13}, \sigma_{\text{Bob}}),\\ (S_{\text{Carol}}, 30, \mathbf{4}, \sigma_{\text{Carol}})\}$$

- **Recipients**:

$$\{(R_{\text{Dave}}, 54)\}$$

The transaction execution (with $f_r = 0.1$) does the following (as illustrated in Figure 3.7):

- Each sender must use their own next counter:
  - Alice uses counter 8 (= 7 + 1)
  - Bob uses counter 13 (= 12 + 1)
  - Carol uses counter 4 (= 3 + 1)
- Total input: $10 + 20 + 30 = 60$ coins
- Fuel automatically burned: $0.1 \cdot 60 = 6$ coins

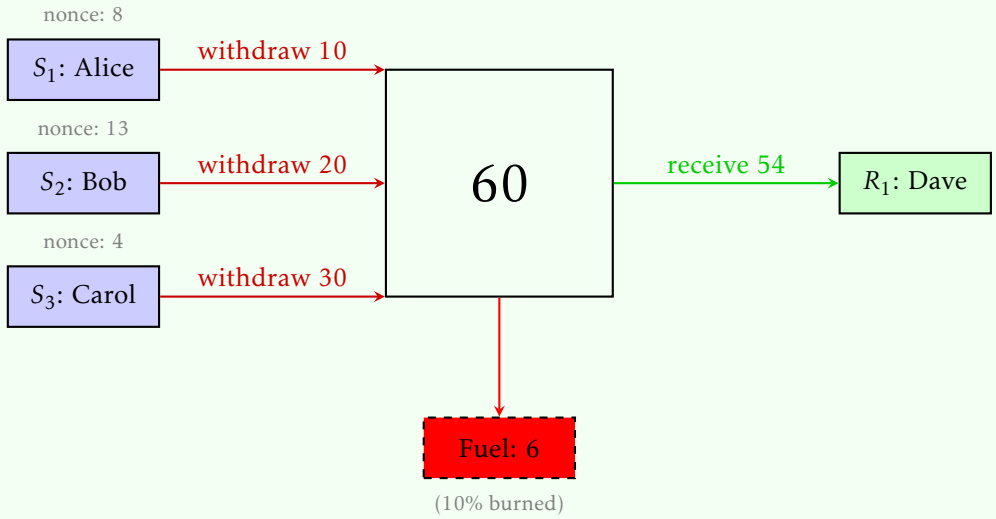


Figure 3.7: Multi-party standard transaction example illustration

The accounts' states after the transaction are shown in Table 3.6.

Table 3.6: State of participant accounts after the transaction

Participant	Role	Balance	Counter
Alice	Sender 1	90	8
Bob	Sender 2	55	13
Carol	Sender 3	120	4
Dave	Recipient 1	554	45

This transaction is atomic: if any sender provides an incorrect counter, the entire transaction fails. For instance, if Bob accidentally used counter 14 instead of 13, the transaction would be rejected even though Alice and Carol provided correct counters.
Dave's counter remains at 45 because receiving funds doesn't increment the counter. When Dave later wants to send a transaction (*e.g.*, to pay for gas), he will need to use counter 46.

The counter mechanism handles several important scenarios:

1. **Transaction Ordering**: If Alice creates two transactions with counters 8 and 9, transaction 9 cannot be processed until transaction 8 is confirmed. This ensures transaction ordering for each account.

2. **Replay Protection**: An attacker cannot replay an old transaction because the counter has already been used. Alice's transaction with counter 8 can only be executed once.

3. **Multi-Sender Atomicity**: In multi-party transactions, all senders must have their counters validated. If any counter is wrong, the entire transaction fails, preventing partial execution.

4. **Independence**: Each account's counter is independent. Alice using counter 8 has no effect on Bob's ability to use counter 13 in the same transaction.

The decision to use individual counters for each sender account rather than a global transaction counter provides several benefits:

1. **Parallel Transaction Creation**: Different accounts can create transactions independently without coordination.
2. **Clear Replay Protection**: Each account's transaction history is independently protected against replay attacks.
3. **Simplified State Management**: The blockchain only needs to track one counter per account, not complex transaction dependencies.
4. **Atomic Multi-Party Transactions**: Multiple senders can collaborate in a single transaction while maintaining their independent transaction sequences.

Redemption Transactions

Redemption transactions convert PoBs into currency rewards, directly crediting the accounts of users who demonstrated desired behaviors. These transactions are streamlined since they only need to specify the recipient account and the reward amount, without tracking UTXOs.

Definition 3.19: Redemption Transaction

A redemption transaction $\mathcal{T}_{\text{redeem}}$ is defined as:

$$\mathcal{T}_{\text{redeem}} = (\pi, R_{\text{producer}}, r)$$

where:

- $\pi = (b \| \sigma_b)$ is the Proof of Behavior being redeemed, with $b = (pk, ts, data)$
- R_{producer} is the account (public key) of the PoB producer, which must match pk in the PoB
- r is the reward amount credited to the producer's account

Note that redemption transactions do not require counters or signatures from the recipient, as they are created by miners on behalf of PoB producers. The PoB itself serves as authorization since it contains the producer's public key and was signed by a valid validator.

Definition 3.20: Redemption Transaction Validity

A redemption transaction $\mathcal{T}_{\text{redeem}}$ with PoB π and recipient R_{producer} is valid if and only if:

$$\begin{aligned}
&\text{(PoB Ownership)} && pk(\pi) = R_{\text{producer}} \\
&\text{(Reward Bounds)} && k' \cdot \text{Quantify}(\pi) - \min_{cur} < r \leq k' \cdot \text{Quantify}(\pi) \\
&\text{(Minimum Value)} && \exists j \in \mathbb{N}^*, r = j \cdot \min_{cur} \\
&\text{(PoB Validity)} && \text{Quantify}(\pi) > 0 \wedge \text{not expired} \wedge \text{not previously redeemed}
\end{aligned}$$

The ownership check ensures rewards go to the correct account. The reward bounds ensure fair compensation while accounting for rounding to the minimum currency unit. The PoB validity check prevents double redemption and expired behaviors.

Example 3.21: Single PoB Redemption

Alice took a metro ride at 9:00 AM, generating a PoB π_{Alice} signed by the metro terminal validator. The PoB contains her public key pk_{Alice} and the behavior data.
The account's state before the redemption is shown in Table 3.7.

Table 3.7: State of participant account before the redemption

Participant	Role	Balance	Counter
Alice	Recipient	90	8

When a miner includes Alice's PoB in block 1000, they create redemption transaction $\mathcal{T}_{\text{redeem}}$:

- **PoB**:
$$\{\pi_{\text{Alice}}\}$$
with $\text{Quantify}(\pi_{\text{Alice}}) = 5$
- **Recipient**:
$$\{R_{\text{Alice}}\}$$
extracted from the PoB
- **Reward**:
$$\{r = 25 \text{ coins}\}$$
with $k' = 5$

The transaction execution does the following (illustrated in Figure 3.8):

- Alice's balance: $90 + 25 = 115$ coins
- Alice's counter: remains 8 (receiving doesn't change counter)
- The PoB is marked as redeemed in the blockchain state
- 25 new coins are created (minted) for this reward

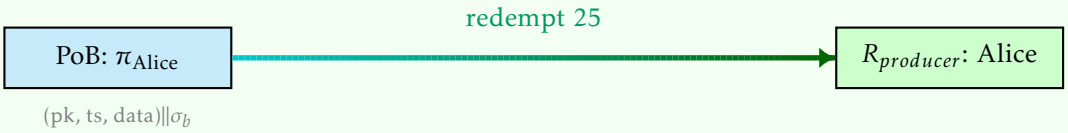


Figure 3.8: Redemption transaction example illustration

Alice doesn't need to sign anything or provide a counter – the miner handles the redemption automatically based on her valid PoB.
The account's state after the redemption is shown in Table 3.8.

Table 3.8: State of participant account before the redemption

Participant	Role	Balance	Counter
Alice	Producer & Recipient	115	8

Coinbase Transactions

Coinbase transactions consolidate all miner rewards into a single transaction per block. This transaction credits the miner's account with rewards for block production, transaction processing, and including others' PoBs.

Definition 3.22: Coinbase Transaction

A coinbase transaction $\mathcal{T}_{\text{coinbase}}$ is defined as:

$$\mathcal{T}_{\text{coinbase}} = (R_{\text{miner}}, r_{\text{total}})$$

where:

- R_{miner} is the miner's account (public key)
- r_{total} is the total reward amount computed as:

$$r_{\text{total}} = k \cdot \text{Quantify}(\pi_{\text{miner}}) + n \cdot b + \sum_{i=1}^{m} k'' \cdot \text{Quantify}(\pi_i)$$

  with:

  - $k \cdot \text{Quantify}(\pi_{\text{miner}})$: reward for miner's own PoB
  - $n \cdot b$: bounties for n standard transactions (b per transaction)
  - $\sum_{i=1}^{m} k'' \cdot \text{Quantify}(\pi_i)$: bonuses for including m other PoBs

The coinbase transaction requires no sender, counter, or fuel since it represents new currency creation rather than a transfer. It must be the first transaction in each block.

Definition 3.23: Coinbase Transaction Validity

A coinbase transaction $\mathcal{T}_{\text{coinbase}}$ in block B is valid if and only if:

$$\begin{aligned}
&\text{(Correct Recipient)} && R_{\text{miner}} = \text{BlockProducer}(B) \\
&\text{(Correct Amount)} && r_{\text{total}} = k \cdot \text{Quantify}(\pi_{\text{miner}}) + n \cdot b + \sum_{i=1}^{m} k'' \cdot \text{Quantify}(\pi_i) \\
&\text{(Minimum Value)} && \exists j \in \mathbb{N}^*, r_{\text{total}} = j \cdot \min_{cur} \\
&\text{(Uniqueness)} && \text{Exactly one coinbase transaction per block}
\end{aligned}$$

Example 3.24: Complete Block Rewards

Miner Mike successfully mines block 1002 using his own PoB from taking the subway. The block contains various transactions and PoBs from other users.
The miner's account's state before the coinbase is shown in Table 3.9.

Table 3.9: State of miner before the redemption

Participant	Role	Balance	Counter
Mike	Miner & Recipient	1155	87

Block Contents:

- Mike's own PoB: Quantify(π_{Mike}) = 3
- 5 standard transactions (ride payments, transfers, *etc.*)
- 3 redemption transactions for other users' PoBs:
  - π_1 with Quantify(π_1) = 1
  - π_2 with Quantify(π_2) = 2
  - π_3 with Quantify(π_3) = 1

Reward Calculation (with $k = 10$, $b = 10$, $k'' = 2$):

- Own PoB reward: $10 \cdot 3 = 30$ coins
- Transaction bounties: $5 \cdot 10 = 50$ coins
- PoB inclusion bonuses: $2 \cdot (1 + 2 + 1) = 8$ coins
- **Total**: $30 + 50 + 8 = 88$ coins

The coinbase transaction $\mathcal{T}_{\text{coinbase}}$:

- **Recipient**:

$$R_{\text{Mike}}$$

- **Amount**:

2540 coins

The coinbase transaction has no specific input and outputs newly minted coins to the miner, as illustrated in Figure 3.9.

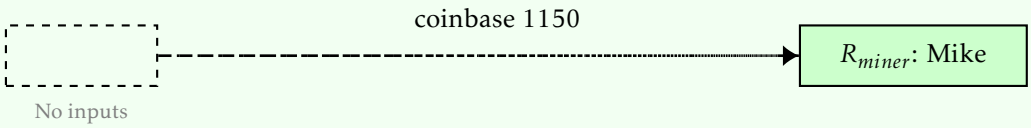


Figure 3.9: Coinbase transaction example illustration

The miner's account's state after the coinbase – and thus the block execution – is shown in Table 3.10.

Table 3.10: State of miner after the redemption

Participant	Role	Balance	Counter
Mike	Miner & Recipient	1243	87

3.3.4 Economic Mechanism

The economic framework of our PoB blockchain serves as the critical bridge between behavioral incentives and system sustainability. Unlike traditional cryptocurrencies that primarily focus on transaction processing and security, our system must carefully balance multiple economic objectives: rewarding desired behaviors, maintaining fair transaction costs, ensuring miner participation, and controlling monetary supply. This section examines three fundamental economic mechanisms that achieve these goals. First, we detail the PoB reward system that creates tiered incentives for miners and behavior producers more generally, establishing a comprehensive ecosystem where all participants contributing valuable behaviors receive appropriate compensation. Second, we analyze the fuel and bounty transaction system, demonstrating how proportional fees and fixed miner rewards eliminate size-based discrimination while maintaining economic security against spam attacks. Finally, we explore the integration of demurrage as a monetary policy tool that counterbalances continuous currency creation, establishing a dynamic equilibrium that maintains stable value while encouraging active participation. Together, these mechanisms create a self-regulating economic environment that aligns individual incentives with collective behavioral goals.

PoB Reward System

In the context of our permissionless blockchain architecture, the reward mechanism is designed to incentivize multiple aspects of network participation. The primary reward is allocated to the miner who successfully creates a new block, compensating both for the block creation itself and for the computational work invested in acquiring mining rights – specifically, the execution of a PoB and the corresponding VDF.

Our system extends beyond rewarding solely the block-producing miner, incorporating a mechanism to incentivize all valid PoB productions. To implement this, miners are permitted to include not only transactions but also PoBs from other network participants in their blocks. The producers of these included PoBs are awarded a secondary reward, albeit at a reduced rate compared to the block-producing miner.

To encourage miners to incorporate these additional PoBs into their blocks, a tertiary reward structure is implemented.

Definition 3.25: PoB Reward System

Let π denote the PoB utilized by the successful miner. Let π' be a PoB produced by a non-miner. The reward structure for PoB participation is defined by:

$$\begin{aligned} r &= k \cdot \mathsf{Quantify}(\pi) && \text{(miner's own PoB)} \\ r' &= k' \cdot \mathsf{Quantify}(\pi') && \text{(non-miner PoB)} \\ r'' &= k'' \cdot \mathsf{Quantify}(\pi') && \text{(miner inclusion bonus)} \end{aligned}$$

where $0 < k'' < k' < k$ determine the reward rates.

This hierarchical reward structure creates a balanced incentive system that promotes both block production and widespread participation in PoB generation, while maintaining appropriate differentiation between the rewards for various levels of contribution to the network's operation and security.

In our blockchain system, the creation of new coins is intrinsically linked to the production of PoBs. Each PoB incorporated into the blockchain necessitates the execution of a specialized

transaction that facilitates the minting of new coins. This mechanism ensures that coin creation is directly tied to the desired behaviors incentivized by the system.

Fuel and Bounty System Analysis. The fuel and bounty system transforms how blockchains price transactions. In traditional systems, users compete for block space by offering higher fees to miners, creating a market where urgency and wealth determine access. Our system replaces this auction mechanism with a radically different approach: every user pays the same percentage of their transaction value as "fuel" that gets permanently destroyed, while miners receive a fixed reward regardless of transaction size. This design prioritizes equal treatment and simplicity over market efficiency, creating profound changes in how users and miners interact with the blockchain.

System Architecture and Fundamental Properties. To understand this system's implications, we must first examine its mathematical structure. The fuel and bounty mechanism operates through four carefully balanced parameters that together ensure both accessibility for users and profitability for miners.

The inequality in Definition 3.14 serves as the system's foundation. It ensures that even the smallest transaction burns enough fuel to cover the miner's bounty. Without this constraint, miners could create minimum-value transactions (spam attack), collect bounties exceeding the fuel cost, and generate infinite currency. This simple inequality thus protects the entire system's integrity.

When a user sends a transaction, they experience a cost structure fundamentally different from traditional blockchains. The cost function $C(v) = f_r \cdot v$ means that transaction costs scale linearly with value. This proportionality changes transaction accessibility: a \$10 transaction pays 10% of what a \$100 transaction pays, whereas in fixed-fee systems, the smaller transaction might pay the same absolute fee, creating a regressive cost structure that discriminates against small transactions.

The burning mechanism deserves emphasis. In traditional systems, fees transfer wealth from users to miners. Here, the fuel simply vanishes, reducing the total currency supply. This creates deflationary pressure proportional to network activity – the more the blockchain is used, the more currency disappears. This elegant mechanism naturally balances currency supply with demand.

Strategic Implications for Users and Miners. The rigid structure of the fuel and bounty system eliminates many strategic complexities that plague traditional blockchains. Users no longer face the anxiety of fee estimation, and miners no longer need sophisticated optimization algorithms. Let's examine how this simplification affects participant behavior.

For users, the question of transaction structuring becomes trivial. In traditional blockchains, users routinely split large transactions to reduce fees, or batch small ones to amortize fixed costs. Our system removes these inefficiencies: splitting a transaction of value v into n smaller transactions yields total cost $\sum_{i=1}^{n} f_r \cdot v_i = f_r \cdot v$, identical to the original transaction cost. This neutrality simplifies wallet software and reduces network congestion from artificial transaction structuring.

For miners, the strategic landscape becomes even simpler. Since all transactions yield identical revenue b, sophisticated algorithms analyzing fee density become pointless. The rational strategy reduces to first-come-first-served (FCFS) processing. This dramatic simplification eliminates entire categories of attacks based on transaction ordering manipulation and removes incentives for miners to collude with users for preferential treatment.

Security Properties and Attack Resistance. The fuel burning mechanism fundamentally alters the economics of blockchain attacks. We now present a rigorous analysis of the system's security properties.

Definition 3.26: Transaction Spam Attack

A *transaction spam attack* is an adversarial strategy where an attacker creates a large number of low-value transactions–possibly sending funds between accounts they control–in an attempt to congest the blockchain or extract mining rewards without meaningful economic transfer.

Theorem 3.27: Spam Attack Immunity

An attacker cannot profit from transaction spam attacks involving self-transactions, regardless of their mining power.

Proof of Theorem 3.27

Consider an attacker with initial balance B creating n self-transactions of value v each, where $v \geq \min_{sent}$.
The attacker incurs costs:

- Fuel burned: $n \cdot f_r \cdot v$ (permanently destroyed)

Even if the attacker controls all mining power (best case), they receive:

- Bounties: $n \cdot b$ (newly created)

The attacker's net profit is:

$$\Pi_{attack} = n \cdot b - n \cdot f_r \cdot v = n(b - f_r \cdot v)$$

By the system constraint $f_r \cdot \min_{sent} \geq b$, and since $v \geq \min_{sent}$:

$$\Pi_{attack} = n(b - f_r \cdot v) \leq n(b - f_r \cdot \min_{sent}) \leq 0$$

Therefore, the attacker loses money proportional to the attack scale. Even controlling all mining power cannot make spam profitable. ■

This impossibility result provides fundamental security. Traditional blockchains rely on probabilistic defenses – spam becomes unprofitable only if attackers control limited mining power. Our system makes spam unconditionally unprofitable through mathematical constraint. This represents a qualitative improvement in security guarantees.

Beyond spam, the system also provides strong resistance against denial-of-service attacks:

Definition 3.28: Denial-of-Service (DoS) Attack

A *denial-of-service (DoS) attack* is an adversarial strategy in which an attacker attempts to prevent honest users from having their transactions confirmed by occupying all available block capacity with attacker-generated transactions. The objective is not to extract direct profit, but to degrade the usability of the blockchain by delaying or excluding legitimate activity.

Proposition 3.29: Denial of Service Resistance

The cost of sustaining a denial-of-service attack scales linearly with duration, with a strict lower bound determined by system parameters.

Proof of Proposition 3.29

To occupy all block space (capacity C transactions) for time period t, an attacker must create $C \cdot t$ transactions. The minimum attack cost is:

$$\text{Cost}_{DoS} = C \cdot t \cdot f_r \cdot \min_{sent} \geq C \cdot t \cdot b$$

This cost is irreversibly burned. Unlike traditional systems where sophisticated attackers might recover fees through mining, every unit of currency spent on the attack is permanently destroyed. The linear scaling ensures that doubling attack duration doubles cost, making sustained attacks economically prohibitive. ■

The combination of unconditional spam resistance and linear DoS costs creates a robust defense against the most common blockchain attacks. Moreover, these defenses emerge naturally from the system's economic structure rather than requiring additional security mechanisms.

Implementation Considerations. Successfully deploying a fuel and bounty system requires careful parameter selection. The key parameters must balance multiple objectives: preventing spam, ensuring miner profitability, maintaining accessibility, and controlling monetary inflation.

Fuel Rate Selection: The fuel rate f_r determines the system's fundamental character. Too high, and transaction costs discourage usage. Too low, and spam becomes affordable while miners lack incentive. We recommend $f_r \in [0.001, 0.05]$ (0.1% to 5%), with the specific value depending on the blockchain's purpose. Payment-focused chains might use lower rates for accessibility, while specialized chains might use higher rates to ensure only serious participants engage.

Bounty Calibration: The bounty b must cover miners' operational costs while remaining sustainable. Setting $b \approx f_r \cdot \text{median}(v)$ ensures typical transactions generate fees comparable to miner rewards, creating economic balance. This calibration prevents long-term inflation or deflation of the currency supply.

Minimum Transaction Threshold: The minimum transaction value $\min_{sent}$ serves dual purposes. It must be large enough to prevent spam (satisfying $\min_{sent} > b/f_r$) yet small enough to maintain accessibility. Setting it around 1% of typical transaction values strikes this balance.

One significant challenge involves adaptation to changing conditions. Unlike traditional fee markets that automatically adjust to demand, our system's fixed parameters cannot respond to network growth or usage changes. This rigidity necessitates governance mechanisms for parameter updates, introducing complexity the system otherwise avoids.

Comparative Analysis with Traditional Fee Markets. To fully appreciate the fuel and bounty system's implications, we must contrast it with conventional blockchain economics. Traditional

fee markets excel at price discovery and efficiency but suffer from accessibility and complexity issues. Our system inverts these trade-offs.

In Bitcoin or Ethereum, fees fluctuate based on network congestion. During high demand, fees can spike dramatically, pricing out small transactions. A $5 transaction might require a $50 fee during congestion, effectively excluding ordinary users. Our system maintains constant percentage fees regardless of network conditions, ensuring predictable costs but potentially causing longer wait times during congestion.

Traditional systems also enable sophisticated strategies like Replace-By-Fee (RBF) and Child-Pays-For-Parent (CPFP), allowing users to adjust fees post-submission. While powerful, these mechanisms add complexity and enable manipulative strategies. Our system's simplicity prevents such strategies entirely – once submitted, a transaction cannot be accelerated regardless of circumstances.

Limitations and Trade-offs. The fuel and bounty system's simplicity comes with significant limitations that must be acknowledged:

1. **No Objective Priority Mechanism:** The inability to express transaction urgency eliminates use cases requiring rapid confirmation. Time-sensitive applications like decentralized exchange arbitrage, auction bidding, or emergency transfers cannot function effectively. Everyone waits in the same queue, whether transferring retirement savings or catching a fleeting market opportunity.

2. **Parameter Rigidity:** Fixed parameters cannot adapt to changing network conditions. A sudden spike in usage cannot be addressed through market mechanisms, potentially leading to extended confirmation times. Similarly, a decrease in usage doesn't automatically reduce fees to attract more transactions.

3. **Value Discrimination:** The proportional fee structure explicitly discriminates against high-value transactions. A $1 million transfer paying 1% ($10,000) in fees might seem excessive compared to traditional systems charging perhaps $10. This may segment the user base and reduce the blockchain's utility for large transfers.

4. **Economic Efficiency Loss:** By preventing price discovery, the system likely operates below optimal efficiency. Some users willing to pay more for inclusion cannot, while others may be priced out despite miners having spare capacity. This deadweight loss is the price of equal treatment.

Theoretical Implications and Future Directions. The fuel and bounty system demonstrates that alternative economic models for blockchains are both possible and potentially superior for specific applications. It challenges the assumption that market mechanisms represent the only viable approach to transaction pricing.

From a mechanism design perspective, the system represents an interesting case study in trading efficiency for other desirable properties. The literature on mechanism design typically focuses on maximizing social welfare or revenue. Our system explicitly sacrifices these goals to achieve equal treatment, simplicity, and accessibility – properties difficult to quantify but crucial for certain applications.

Future research might explore hybrid approaches that retain some benefits of both systems. For instance, a dual-track system could offer both fixed-fee and market-based options, allowing users to choose based on their needs. Alternatively, dynamic parameter adjustment algorithms could provide some adaptability while maintaining the system's core properties.

Conclusion. The fuel and bounty system represents a fundamental reimagining of blockchain economics. By replacing competitive fee markets with proportional burning and fixed rewards, it creates a system that prioritizes equal treatment and simplicity over efficiency and flexibility.

The system excels at democratizing blockchain access. Perfect cost proportionality ensures that wealth doesn't determine transaction priority. Unconditional spam resistance provides robust security without complex mechanisms. These properties make the system ideal for blockchains prioritizing equal participation over maximum efficiency.

However, these benefits come with significant trade-offs. The inability to express urgency limits use cases requiring rapid confirmation. Higher costs for large transactions may drive high-value transfers elsewhere. Fixed parameters reduce the system's ability to adapt to changing conditions. These limitations make the system unsuitable for general-purpose financial infrastructure.

The fuel and bounty system thus occupies a specific niche: blockchains where behavioral incentives and equal access matter more than market efficiency. For systems designed to reward participation rather than maximize throughput – such as the mobility-incentivizing blockchain presented in this work – these trade-offs are not only acceptable but desirable. The system trades away features needed for high finance to gain properties essential for inclusive participation.

Demurrage

The incorporation of demurrage into our account-based blockchain serves a dual purpose: incentivizing currency circulation and establishing monetary equilibrium. Unlike traditional cryptocurrencies where account balances remain static over time, our system implements temporal demurrage whereby account balances automatically decrease at each block according to a predetermined rate.

In an account-based model, demurrage implementation becomes significantly more elegant than in UTXO systems. Rather than tracking the age of individual transaction outputs, we apply demurrage uniformly to all account balances at each block.

Definition 3.30: Demurrage Rate

The demurrage rate applied per block is noted as $\lceil \in [0,1]$ (sometimes as $\lceil_{\text{block}}$).
When block B_i is produced, every account balance x in the global state is reduced multiplicatively by the factor $(1-\lceil)$, *i.e.*,

$$x \longmapsto x \cdot (1-\lceil).$$

The daily demurrage rate $\lceil_{\text{day}}$ is defined in terms of $\lceil_{\text{block}}$ as

$$\lceil_{\text{day}} = 1-(1-\lceil_{\text{block}})^n,$$

where n is the expected number of blocks mined per day. Conversely, given a desired daily rate $\lceil_{\text{day}}$, the equivalent per-block rate is

$$\lceil_{\text{block}} = 1-\sqrt[n]{1-\lceil_{\text{day}}}.$$

When block B_i is produced, every account balance in the global state undergoes automatic reduction according to this factor. The relationship between daily and block demurrage rates follows directly from the above definition.

Example 3.31

Consider a demurrage rate of 10% per day with an average of 144 blocks per day (one block every 10 minutes). To achieve $\lceil_{day} = 0.1$, we require:

$$\lceil_{block} = 1 - \sqrt[144]{1 - 0.1} = 1 - \sqrt[144]{0.9} \approx 0.00073 = 0.073\%$$

Thus, if Alice holds 100 coins at block height h, her balance at block height $h+1$ will be:

$$\text{Balance}_{h+1} = 100 \cdot (1 - 0.00073) = 99.927 \text{ coins}$$

After 144 blocks (approximately one day), her balance becomes:

$$\text{Balance}_{h+144} = 100 \cdot (1 - 0.00073)^{144} \approx 90 \text{ coins}$$

Transaction Handling with Demurrage. When processing transactions in our account-based system with demurrage, the system must account for the temporal depreciation of balances. For a transaction initiated at block height h_{tx} and included in block $h_{inclusion}$, the sender's effective balance is:

$$\text{Balance}_{effective} = \text{Balance}_{h_{tx}} \cdot (1 - \lceil_{block})^{(h_{inclusion} - h_{tx})} \tag{3.2}$$

In other words, the demurrage does not care when a transaction is initiated. Demurrage only cares when the transaction is included in the blockchain. This ensures that users cannot exploit delays between transaction creation and inclusion to avoid demurrage. The transaction validity check becomes:

$$\text{Balance}_{effective} \geq v + f_r \cdot v \tag{3.3}$$

where v is the transaction value and f_r is the fuel rate.

At each new block, the blockchain performs two distinct operations on the global account state. First, it applies demurrage: all account balances are multiplied by $(1 - \lceil_{block})$. Then it processes transactions. Account balances are updated according to the transactions included in the block: standard transactions, redemption transaction and the coinbase transaction.

This ordering ensures consistent state transitions and prevents double-application of demurrage. The state transition function for block B_i becomes:

$$S_{i+1} = \text{ProcessTx}(\text{ApplyDemurrage}(S_i, \lceil_{block}), Tx_i) \tag{3.4}$$

where S_i represents the global state at block i, and Tx_i represents the set of transactions in block i.

3.4 Checkpoint Mechanism

This section introduces the concept of checkpoints as a way to enhance the efficiency and reliability of the consensus process. We begin by outlining the motivation for using checkpoints and how they help reduce redundant computations. Then, we describe the checkpoint creation procedure, followed by a discussion on validator responsibilities in maintaining consistency.

Finally, we examine the security implications and benefits of integrating checkpoints into the protocol.

3.4.1 Motivation and Design Principles

A central design choice in our blockchain architecture is the adoption of an account-based model rather than the traditional UTXO model. This choice has profound implications for checkpointing, since checkpoints serve as cryptographic commitments to the global system state and enable historical pruning. Understanding the relative advantages and disadvantages of these two models provides essential context for our checkpoint design.

In UTXO-based blockchains such as Bitcoin, the global state is implicitly defined by the set of all unspent transaction outputs. A checkpoint must therefore commit to the complete UTXO set, which can reach massive size and exhibits high churn. This leads to large checkpoint snapshots, complex proofs of inclusion or exclusion, and heavy storage requirements. Moreover, since UTXOs encode coin ownership indirectly through transaction graphs, reconstructing balances from a checkpoint often requires additional traversal and verification of prior history. While the UTXO model has the advantage of naturally supporting parallel transaction validation and fine-grained coin traceability, these benefits come at the cost of substantial overhead when compressing state for checkpoints.

In contrast, an account-based model maintains a unified global state consisting of account balances, counters, and associated metadata. A checkpoint in this model reduces to a compact commitment over a key-value map, typically realized as a Merkle root of account balances. This yields several advantages: (i) state proofs are logarithmic in the number of accounts, allowing lightweight verification; (ii) pruning is straightforward, since historical transactions need not be preserved once their effects are absorbed into the checkpointed state; (iii) replay protection through per-account counters ensures determinism without requiring access to old transactions. These properties make account-based checkpoints particularly well-suited for storage-bounded environments and lightweight clients.

An additional conceptual difference arises with respect to fungibility. In UTXO systems, each coin is linked to a specific transaction output, making it possible to trace the complete history of a particular unit of currency. Although this property is sometimes desirable for auditing, it undermines perfect fungibility since coins may be "tainted" by their provenance. By contrast, in account-based systems, coins are merged into account balances and are indistinguishable once deposited. This dilution effect enhances fungibility, as the ledger records only aggregate balances rather than individual coin lineages. Checkpoints in an account-based model thus naturally preserve this stronger notion of fungibility by committing to balance states rather than transaction histories.

The account-based approach is not without trade-offs. One disadvantage is the global sequential dependency introduced by counters: transactions must be validated in strict account order, which reduces the degree of parallelization available to miners and validators. Furthermore, the account model requires careful handling of concurrent updates to prevent double-spending within the same block, whereas the UTXO model enforces this property structurally. Finally, auditing individual coins or reconstructing their provenance is more difficult in the account model, since checkpoints commit only to aggregate balances rather than discrete unspent outputs.

In light of these considerations, our PoB blockchain deliberately adopts the account-based model to optimize checkpoint efficiency. The resulting checkpoints act as compact, verifiable anchors of the entire system state, enabling aggressive pruning of historical Proofs of Behavior and transactions while ensuring that all nodes maintain a consistent and easily verifiable global view. This integration of account-based state management with checkpointing not only reduces

storage growth but also strengthens long-term scalability, accessibility, and fungibility.

System Architecture and Design Principles

The checkpoint mechanism is designed around three fundamental principles that guide its architecture and implementation. First, it must preserve the decentralized nature of the blockchain by avoiding any central authority or privileged participants. Second, it must provide cryptographic guarantees that pruned data was indeed part of the canonical chain, allowing new nodes to bootstrap securely. Third, it must integrate seamlessly with the existing PoB consensus without disrupting normal block production.

To achieve these goals, the mechanism operates through three interconnected components:

1. **Periodic State Commitments**: At designated points in time, the system creates cryptographic snapshots that capture the complete blockchain state. These commitments serve as trusted anchors, allowing nodes to verify the current state without processing the entire history. See Section 3.4.1.

2. **Witness-based Consensus**: Rather than introducing new participants, the system leverages successful miners from each epoch as witnesses for checkpoint creation. These witnesses execute the Practical Byzantine Fault Tolerance (pBFT) protocol to reach agreement on checkpoint contents. See Section 3.4.2.

3. **Safe Historical Pruning**: Once checkpoints are finalized through pBFT consensus, nodes can remove historical blockchain data while maintaining the ability to verify current balances and recent transactions. This pruning is deterministic and verifiable, ensuring all honest nodes maintain consistent views. See Section 3.4.3.

A central challenge in implementing our checkpoint mechanism lies in how to efficiently record collective agreement among many witnesses without overwhelming the blockchain with redundant data. If each witness signature were stored individually, checkpoint size and verification costs would grow linearly with the number of participants, undermining the scalability benefits checkpoints are meant to provide. To overcome this, we require a cryptographic primitive that supports both compactness and efficient aggregation of signatures. This motivates the use of the Boneh–Lynn–Shacham (BLS) signature scheme, whose aggregation property makes it particularly well-suited for checkpoint finality in large, decentralized settings.

Checkpoint Structure and Creation

Having established BLS aggregation as the cryptographic foundation for compressing witness attestations, we now turn to the construction of the checkpoints themselves. A checkpoint must serve as more than a simple block hash: it must provide a compact yet verifiable commitment to the entire blockchain state at a given height, while carrying a proof that a supermajority of witnesses has endorsed it.

To achieve this, we define a checkpoint as a structured tuple combining: (i) a Merkle root summarizing all account balances; (ii) the block hash anchoring it to a precise location in the chain; and (iii) an aggregated BLS signature attesting collective agreement. This structure guarantees that any participant – whether a full node or a lightweight client – can validate the state securely without storing the entire history.

Definition 3.32: Checkpoint Structure

A checkpoint C_h at blockchain height h is a cryptographically secured state commitment consisting of three essential components:

$$C_h = (\text{StateRoot}_h, \text{BlockHash}_h, \Pi_h)$$

where:

- $\text{StateRoot}_h \in \{0,1\}^{256}$: A Merkle root capturing all account balances at height h. This allows any account balance to be verified with a Merkle proof without accessing historical transactions.
- $\text{BlockHash}_h \in \{0,1\}^{256}$: The hash of the specific block at height h that triggered the checkpoint. This anchors the checkpoint to a precise point in the blockchain's history.
- Π_h: An aggregated BLS signature from participating checkpoint witnesses who reached consensus through pBFT. This provides cryptographic proof that a super-majority of witnesses agreed on the state.

The StateRoot component deserves particular attention. By computing a Merkle tree over all account balances, the system creates a compact 256-bit commitment that nonetheless allows any specific balance to be proven with logarithmic-sized proofs. This enables light clients to verify their balances without downloading the entire state, a crucial property for mobile and resource-constrained devices.

Example 3.33

Consider a simplified blockchain state with 8 accounts and their balances of Table 3.11.

Table 3.11: Accounts and Balances

Account	Balance
Alice	1000 coins
Bob	2500 coins
Carol	750 coins
Dave	3200 coins
Eve	1800 coins
Frank	950 coins
Grace	4100 coins
Henry	650 coins

Step 1: Merkle Tree Construction
The system constructs the Merkle tree in Figure 3.10 where each leaf contains an account's balance data.

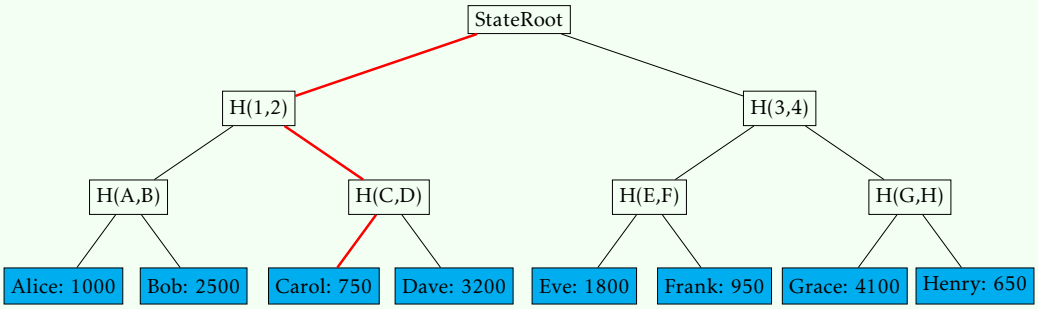


Figure 3.10: Merkle Tree Construction Example

Step 2: Compact Commitment
The StateRoot is a single 256-bit hash that commits to all account balances:

$$\begin{aligned} \text{StateRoot} &= \text{Hash}(\text{H}(1,2)||\text{H}(3,4)) \\ \text{where H}(1,2) &= \text{Hash}(\text{H}(\text{A,B})||\text{H}(\text{C,D})) \\ \text{H}(3,4) &= \text{Hash}(\text{H}(\text{E,F})||\text{H}(\text{G,H})) \end{aligned}$$

Step 3: Logarithmic Proof Size
To prove Carol's balance (750 coins) to a light client, the system only needs to provide:

1. Carol's account data: `Carol: 750 coins`
2. **Merkle proof path** (3 hashes for 8 accounts = $\log_2(8)$ hashes):
   - H(Dave) - sibling hash to compute H(C,D)
   - H(A,B) - sibling hash to compute H(1,2)
   - H(3,4) - sibling hash to compute StateRoot

Step 4: Light Client Verification
The light client verifies Carol's balance without downloading the full state:

$$\begin{aligned} \text{Verify}_1: &\quad \text{H(C,D)} = \text{Hash}(\text{Carol: 750}||\text{H(Dave)}) \\ \text{Verify}_2: &\quad \text{H(1,2)} = \text{Hash}(\text{H(A,B)}||\text{H(C,D)}) \\ \text{Verify}_3: &\quad \text{StateRoot} = \text{Hash}(\text{H(1,2)}||\text{H(3,4)}) \end{aligned}$$

If the computed StateRoot matches the one in the checkpoint, Carol's balance is verified.
Efficiency Analysis:
First, let's specify what constitutes an account entry. Let an account entry be the "round" number 64 bytes total. It comprises of:

- Account address: 20 bytes (160-bit public key hash)
- Balance: 8 bytes (64-bit unsigned integer)
- Transaction counter: 4 bytes (prevents replay attacks)
- Account metadata: 4 bytes (flags, account type, *etc.*)

- Merkle tree overhead: 28 bytes (serialization, padding, tree structure)

The advantage of a Merkle proof approach is shown in Table 3.12.

Table 3.12: Advantage of Merkle Trees

Approach	Data Required	Bandwidth
Download full state	All 8 account entries	$8 \cdot 64$ bytes = 512 bytes
Merkle proof	1 account entry + 3 proof hashes	$64 + 3 \cdot 32$ bytes = 160 bytes

The Merkle proof approach achieves 68% reduction in bandwidth.
For a real blockchain with millions of accounts, the savings become dramatic:

- **Full state**: Millions of account records (gigabytes)
- **Merkle proof**: $\log_2(\text{millions}) \approx 20$ hashes (640 bytes)
- **Savings**: > 99.99% bandwidth reduction

This enables mobile wallets and IoT devices to verify their balances using minimal bandwidth and storage, making the blockchain accessible to resource-constrained devices.

The checkpoint creation process naturally divides the blockchain into epochs – periods between consecutive checkpoints during which normal mining and transaction processing occur. These epochs provide a natural unit for organizing blockchain history and managing witness participation.

Definition 3.34: Checkpoint Epoch

Let $\varepsilon \in \mathbb{N}$ denote the *epoch duration*, defined as the fixed number of blocks between two consecutive checkpoints. A *checkpoint epoch* E_i is the contiguous sequence of ε blocks immediately following a finalized checkpoint C_{i-1} and ending with the block at height $h_{i-1} + \varepsilon$, where h_{i-1} is the height of C_{i-1}. The last block of E_i is called the *checkpoint block* associated with C_i.

Each epoch thus represents a complete period of blockchain activity, bounded by cryptographic state commitments. This epochal structure provides several benefits: it limits the scope of potential reorganizations, creates natural boundaries for data pruning, and establishes clear periods for witness participation in the consensus protocol.

Witness Selection

A critical design decision concerns who should serve as witnesses for checkpoint creation. Rather than introducing a separate validator set, which would complicate the system and potentially centralize control, we leverage the existing mining population. Specifically, any miner who successfully produces a block during an epoch automatically becomes a witness for that epoch's checkpoint consensus.

Definition 3.35: Checkpoint Witnesses

For epoch E_i, the checkpoint witness set W_i comprises all miners who successfully produced at least one block during the previous epoch:

$$W_i = \{m \in \text{Miners} : \exists B \in E_{i-1} \text{ where } \text{miner}(B) = m\}$$

These witnesses must come to an agreement to create a checkpoint C_i

The witness set is deliberately defined as the miners of the *previous* epoch rather than the current one. This choice ensures that the witness committee is already fixed and agreed upon by all honest parties before the epoch in which it operates. With a fixed and globally known committee, we can safely run a Byzantine Fault Tolerant protocol such as pBFT and inherit its agreement, validity and termination guarantees. In contrast, if the witness set were derived from the miners of the current epoch, different forks near the epoch's end could lead to divergent committees. Such inconsistency would undermine the very foundation of pBFT, as its correctness proofs assume a unique, agreed-upon committee. By basing the committee on the finalized set of miners from the previous epoch, we avoid this ambiguity and preserve the integrity of the checkpointing process.

To initialize the system, a dedicated bootstrapping process is required to define the witness set of the first epoch. Since no previous epoch exists, this initial committee must be established exogenously, for example by hard-coding a genesis witness list, by relying on a trusted setup, or by running an open election mechanism before the chain's launch. Once the first epoch is completed and its set of miners is known, the protocol transitions seamlessly to its regular operation, where witness committees are deterministically derived from finalized history rather than external input.

A concrete bootstrapping strategy is to embed an explicit list of initial witnesses directly in the genesis block. For instance, the genesis block may specify a set of 200 public keys that form the first witness committee. These keys could belong to early participants, institutions supporting the launch, or be generated transparently by the system's creators and then distributed. This approach ensures that all nodes start from the same agreed-upon committee, allowing the first checkpoint to be finalized without ambiguity. After this first epoch, the protocol naturally transitions to its standard rule: the set of witnesses is derived from the miners of the previous epoch, removing the need for any further external coordination.

3.4.2 pBFT Consensus Protocol

The Practical Byzantine Fault Tolerance (pBFT) protocol provides deterministic consensus for checkpoint creation. We adapt pBFT to our checkpoint mechanism, where the witnesses from each epoch act as replicas to agree on checkpoint content.

For the pBFT protocol to provide its guaranteed safety and liveness properties, we must bound the fraction of Byzantine (malicious or faulty) witnesses. The standard pBFT requirement is that Byzantine replicas must be fewer than one-third of the total.

Assumption 3.36: Byzantine Bound for pBFT

In any epoch, at most f checkpoint witnesses are Byzantine, where $n = |W_i|$ is the total number of witnesses and $n \geq 3f+1$. This ensures that honest witnesses can reach consensus despite Byzantine behavior, as pBFT requires $2f + 1$ agreeing replicas for progress.

This assumption is reasonable in practice because successful mining already requires significant resource investment, creating a natural barrier to Sybil attacks. Moreover, the random nature of mining success makes it difficult for an adversary to predictably control more than f witnesses in any given epoch.

The notion of a quorum and its associated certificates is central to the checkpoint protocol. By requiring $q = 2f + 1$ signatures, a certificate can only be formed if at least $f + 1$ honest witnesses participate. Since honest witnesses never sign conflicting values at the same height, any two valid certificates must overlap in at least one honest signer. This *quorum intersection* property underlies both checkpoint safety and liveness, and allows us to directly leverage the proven properties of the pBFT protocol.

Definition 3.37: Quorum and Certificates

For a checkpoint instance with witness set W_i and Byzantine bound f (Assumption 3.36), the quorum size is defined as

$$q = 2f + 1.$$

- A *prepared certificate* for a message m at height h is a set of q valid Prepare signatures on $H(m)$ by distinct witnesses in W_i.
- A *commit certificate* for a message m at height h is a set of q valid Commit signatures on $H(m)$ by distinct witnesses in W_i. Its BLS aggregation is denoted Π_h.

Protocol Overview. The pBFT protocol operates through three phases within each view, where a view has a designated primary witness responsible for driving consensus. It is composed of three phases, expanded in Algorithm 3.38:

1. **Pre-Prepare Phase:** The primary computes the state root from its local blockchain and broadcasts a checkpoint proposal to all witnesses.
2. **Prepare Phase:** Upon receiving a valid pre-prepare message, witnesses validate the state root against their local computation. If valid, they broadcast a prepare message. A witness becomes *prepared* after receiving $2f$ matching prepare messages.
3. **Commit Phase:** Prepared witnesses broadcast commit messages. Upon receiving $2f + 1$ matching commits, witnesses finalize the checkpoint by creating an aggregated BLS signature from the commit signatures.

View Change Mechanism. If the primary fails to produce a valid checkpoint within a timeout period T_{view}, witnesses initiate a view change. They broadcast view-change messages for the next view, and upon receiving $2f + 1$ such messages, the protocol moves to the next view with primary $p = v \bmod n$, where v is the view number and n is the total number of witnesses.

Algorithm 3.38: pBFT Checkpoint Consensus

Parameters: witness set W_i, $n \leftarrow |W_i|$, $f \leftarrow \lfloor (n-1)/3 \rfloor$, quorum $q \leftarrow 2f+1$, view timeout T_{view}
Goal: finalize $C_h = (\text{StateRoot}_h, \text{BlockHash}_h, \Pi_h)$ for height h

State (per witness):
$v \leftarrow 0$ ▷ current view

5: $p \leftarrow v \bmod n$ ▷ primary index in a fixed ordering of W_i
6: $locked[h] \leftarrow \perp$ ▷ digest this witness is locked on at height h (or $\perp$)
7: $prepared[h] \leftarrow \perp$ ▷ highest prepared pair $(v^\star, m^\star)$ at height h (or $\perp$)

8: **repeat**
Pre-Prepare (primary p)
9: **procedure** PRE-PREPARE (PRIMARY p)
10: **if** this process is the primary p **then**
11: compute StateRoot_h from local state
12: form $m \leftarrow (v, h, \text{StateRoot}_h, \text{BlockHash}_h)$
13: broadcast $\langle$PRE-PREPARE, $m\rangle$
Prepare (all witnesses)
14: **procedure** PREPARE (ALL)
15: wait up to T_{view} for a valid $\langle$PRE-PREPARE, $m\rangle$
16: **if** m valid and locally recomputed state matches and $(locked[h] = \perp \vee locked[h] = H(m))$ **then**
17: broadcast $\langle$PREPARE, $H(m)\rangle$
18: collect $\geq q$ matching $\langle$PREPARE, $H(m)\rangle$
19: $prepared[h] \leftarrow (v, m)$
20: **else**
21: **goto View-Change**
Commit (all witnesses)
22: **procedure** COMMIT (ALL)
23: broadcast $\langle$COMMIT, $H(m)\rangle$
24: collect $\geq q$ matching $\langle$COMMIT, $H(m)\rangle$
25: $locked[h] \leftarrow H(m)$
26: aggregate the q commit signatures into Π_h
27: **return** $C_h \leftarrow (\text{StateRoot}_h, \text{BlockHash}_h, \Pi_h)$
View-Change (all witnesses)
28: **procedure** VIEW-CHANGE
29: broadcast $\langle$VIEW-CHANGE, v+1, h, $prepared[h]\rangle$
30: collect $\geq q$ such messages $V = \{\langle$VIEW-CHANGE, v+1, h, $prepared_j[h]\rangle\}$
31: $(v^\star, m^\star) \leftarrow \text{HIGHESTPREPARED}(V)$ ▷ highest-view prepared pair among V, if any
32: **if** $m^\star$ exists **then**
33: $m_{\text{new}} \leftarrow m^\star$ ▷ must repropose the highest prepared value
34: **else**
35: compute a fresh valid proposal $m_{\text{new}} \leftarrow (v{+}1, h, \text{StateRoot}_h, \text{BlockHash}_h)$
36: set new primary $p' \leftarrow (v{+}1) \bmod n$
37: **if** this process is p' **then**
38: broadcast $\langle$NEW-VIEW, v+1, V, $m_{\text{new}}\rangle$
39: wait for a valid $\langle$NEW-VIEW, v+1, V, $m_{\text{new}}\rangle$
40: $v \leftarrow v{+}1; \quad p \leftarrow p'$
41: **goto Pre-Prepare** (with $m \leftarrow m_{\text{new}}$)
42: **until** checkpoint finalized

Helper predicates (informal):

43: HighestPrepared(V) returns the $(v^\star, m^\star)$ with maximal $v^\star$ such that at least q messages in V carry matching Prepare evidence for $H(m^\star)$; if none exists, it returns $\perp$.
44: A message $\langle$NEW-VIEW, v+1, V, $m_{\text{new}}\rangle$ is valid iff (i) $|V| \geq q$; (ii) signatures are correct and from distinct witnesses in W_i; and (iii) m_{new} equals HighestPrepared(V) if such exists, else is any valid fresh proposal.

The locking check in Prepare and the carriage of the highest prepared certificate in View-Change/NEW-VIEW are the two levers that give pBFT its cross-view safety.

The pBFT consensus protocol guarantees the following properties for each checkpoint instance (detailed in Section 3.6.3):

- **Agreement.** If two honest witnesses finalize a checkpoint at the same height h, then they finalize the same checkpoint. This follows from quorum intersection: two commit certificates of size $q = 2f+1$ must overlap in at least one honest signer, who never signs conflicting values.
- **Validity.** Any checkpoint that is finalized must contain a state root and block hash consistent with the underlying chain, since honest witnesses only prepare values they can locally recompute and verify.
- **Termination.** Under partial synchrony, every correct checkpoint eventually finalizes. More precisely, once the network is synchronous, any view with an honest primary completes within $O(\Lambda)$ time (with Λ the delay bound), and therefore finalization occurs within $O(f \cdot \Lambda)$ rounds.

Message Complexity and Practical Considerations. The protocol requires $O(n^2)$ message exchanges per checkpoint, as each of n witnesses must broadcast to all others in multiple phases. With 100 witnesses, this means approximately 30,000 messages per checkpoint (100 witnesses · 100 recipients · 3 phases). This quadratic scaling presents significant challenges as the network grows.

The witness set size is inherently tied to the number of successful miners per epoch. With checkpoint interval ε blocks and average block time Δ_t, we expect approximately ε unique miners in the worst case (if no miner produces multiple blocks). In practice, some miners will produce multiple blocks, reducing the witness count. For example, with $\varepsilon = 100$ blocks, we might see 60-80 unique miners.

Traditional broadcasting requires each witness to send messages directly to all others, creating $n(n-1)$ connections. Gossip protocols reduce this by having each node relay messages to only $\log(n)$ randomly selected peers. Messages propagate through the network in $O(\log n)$ rounds with high probability. For 100 witnesses, this reduces each node's sending requirement from 99 messages to approximately 7 messages per phase, a 93% reduction in direct transmissions. However, the total message count remains $O(n^2)$ due to relaying, though network load is better distributed.

Message Aggregation Techniques. Since all witnesses in the prepare and commit phases are signing the same message digest $H(m)$, signatures can be progressively aggregated:

- Witnesses can collect multiple signatures before relaying, reducing message count.

- Using BLS signatures, k individual signatures can be combined into one aggregate signature of the same size.
- A coordinator node (possibly the primary) could collect and distribute aggregated signatures, reducing overhead to $O(n)$ at the cost of introducing a centralization point.

In practice, combining these optimizations – using gossip protocols for message dissemination and implementing signature aggregation – can drastically reduce the actual network overhead compared to naive broadcasting.

The Complete Checkpoint Creation Process

To provide concrete understanding of how the pBFT protocol creates checkpoints in practice, let us walk through a detailed example of the entire process.

Example 3.39

Consider a blockchain with $\varepsilon = 100$ where block B_{1000} triggers a checkpoint. During the current epoch (blocks 901-1000), there were 85 unique miners who successfully produced blocks. These miners automatically become the next checkpoint witnesses. During the previous epoch (blocks 801-900), there were 100 unique miners who successfully produced blocks. These are the current checkpoint witnesses. This example is illustrated in Figure 3.11.

Witness Set Composition. The 100 witnesses (miners from the previous epoch) $W = \{w_0, w_1, \ldots, w_{99}\}$ are ordered by who mined the most cumulative difficulty:

- w_0 (Primary): Alice, who mined block the most cumulative difficulty (and probably the most blocks)
- w_1: Bob
- w_2: Carol
- w_3: Dave
- $\vdots$
- w_{99}: Zara

Byzantine Fault Tolerance Parameters. With $n = 100$ witnesses, the protocol tolerates up to $f = 33$ Byzantine witnesses (since $n = 3f + 1 = 100$). Key thresholds:

- **Prepare threshold**: $2f = 66$ messages required
- **Commit threshold**: $2f + 1 = 67$ messages required

The adapted checkpoint pBFT algorithm runs as follows.

Trigger Detection. When block B_{1000} is mined, all nodes compute check that it is the last of the current epoch, and a checkpoint is triggered.

The 100 witnesses initialize the pBFT protocol:

- *View:* $v = 0$,

- *Primary:* $p = v \bmod n = 0 \bmod 100 = 0$ (Alice is primary),
- *Timer:* start with timeout $T_{\text{view}} = 30$ seconds.

Pre-Prepare. Alice, acting as primary, computes the local state root StateRoot_{1000} and forms the proposal $m = (v = 0, h = 1000, \text{StateRoot}_{1000}, \text{BlockHash}_{1000})$. She signs it and broadcasts

$$\langle \text{PRE-PREPARE}, m, \sigma_{\text{Alice}}(m) \rangle$$

to all other witnesses. Each honest witness w_i receives this message from Alice, verifies her signature, and recomputes StateRoot_{1000} locally. If the values match, they accept Alice's proposal.

Prepare. Each honest witness w_i that accepted Alice's proposal broadcasts a signed prepare message

$$\langle \text{PREPARE}, H(m), \sigma_i(H(m)) \rangle$$

to all other witnesses. Thus Bob, Carol, Dave, ..., Zara each send a PREPARE to all 99 peers. Every honest witness collects these messages; once it has received at least $2f = 66$ distinct matching prepares (including its own), it considers the proposal *prepared*. At this point, each honest witness knows that at least $66 - 33 = 33$ honest peers validated the same state root.

Commit. Every prepared witness w_i now broadcasts a signed commit message

$$\langle \text{COMMIT}, H(m), \sigma_i(H(m)) \rangle$$

to all others. Honest witnesses exchange these commit messages; once any honest witness has collected at least $2f + 1 = 67$ matching commits, it finalizes the checkpoint. At that point, witnesses aggregate the commit signatures into a single BLS signature Π_{1000}, and the finalized checkpoint is

$$C_{1000} = (\text{StateRoot}_{1000}, \text{BlockHash}_{1000}, \Pi_{1000}).$$

Byzantine behavior. Suppose Mallory (w_{13}) equivocates by sending conflicting PREPARE messages to different subsets of witnesses. These inconsistencies are detectable, and honest witnesses ignore her equivocations. Because at most $f = 33$ witnesses can be Byzantine, the honest majority ensures that 67 consistent commits are still reached. If instead Alice (the primary) were faulty and never sent a valid pre-prepare, honest witnesses would wait for the timeout T_{view} to expire, then broadcast VIEW-CHANGE messages. Upon collecting $2f + 1 = 67$ such messages, Bob (w_1) becomes the new primary and restarts the process.

All honest witnesses finalize the same checkpoint C_{1000}. Once Π_{1000} is formed, the checkpoint is deterministic and irreversible: two conflicting checkpoints at height 1000 cannot exist under the $n \geq 3f + 1$ assumption. Thus the state of the blockchain at height 1000 is agreed upon by all.

Finality latency. In practice, the adapted pBFT protocol requires approximately seven rounds of message exchange (Pre-Prepare, Prepare, Commit, plus their acknowledgements and possible retransmissions). Under a synchrony bound Λ in the range of 2–3 seconds,

the total completion time for a checkpoint round is about $7\Lambda \approx 15$–20 seconds. Thus, when block B_{1000} triggers a checkpoint, the witnesses reach deterministic agreement on the state at height 1000 within this short time window. Unlike Nakamoto-style protocols, this finality is absolute: once the aggregated BLS signature Π_{1000} is formed, no conflicting checkpoint at the same height can ever exist.

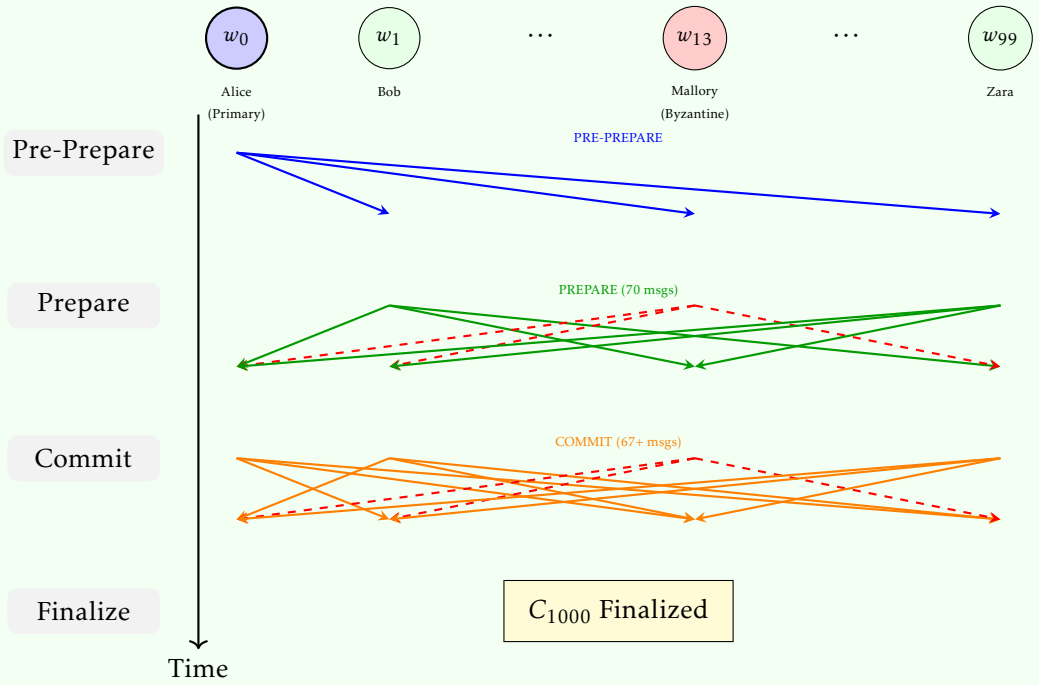


Figure 3.11: pBFT checkpoint creation process showing the three-phase protocol. Alice (w_0) serves as primary in view 0, broadcasting the checkpoint proposal to all witnesses. The protocol progresses through Pre-Prepare, Prepare, and Commit phases, requiring $2f = 66$ prepare messages and $2f + 1 = 67$ commit messages for finalization. Byzantine witnesses like Mallory (w_{13}) may send invalid or no messages (shown with dashed line), but the protocol succeeds with honest majority.

This example illustrates how pBFT's three-phase structure ensures that all honest witnesses agree on the checkpoint content despite Byzantine participants attempting to disrupt the process.

Implementation Considerations for pBFT Checkpoints

When implementing checkpoint mechanisms within the pBFT framework, it is essential to carefully configure the timing parameters to balance safety and liveness. The timeout parameter, denoted as T_{view}, determines how long a witness waits before suspecting a primary failure or network delay. If T_{view} is set too short, honest primaries may be falsely suspected, leading to unnecessary view changes and reduced efficiency. Conversely, if T_{view} is excessively long, recovery from actual faults becomes slower, harming responsiveness.

Another critical parameter is the network delay bound, represented as Λ. This value captures the maximum expected message transmission time under normal operating conditions. For the

system to operate correctly, the timeout must be strictly larger than the delay bound, *i.e.*,

$$T_{\text{view}} > \Lambda$$

This inequality ensures that transient fluctuations in message delays do not prematurely trigger view changes. At the same time, T_{view} should not be chosen excessively larger than Λ, since this would hinder the protocol's responsiveness in the presence of actual faults.

In practice, tuning T_{view} relative to Λ requires empirical evaluation based on observed network conditions. Adaptive strategies may also be employed, where T_{view} is dynamically adjusted in response to measured performance. Such approaches can help maintain an optimal balance between efficiency and robustness, thereby enhancing the reliability of the checkpoint mechanism in pBFT.

3.4.3 Data Pruning and Storage Optimization

The finalization of checkpoints through pBFT consensus enables systematic storage optimization. Once a checkpoint C_h at height h is finalized, the global state of the blockchain is cryptographically fixed: the Merkle root $StateRoot_h$ commits to all account balances and counters, and the aggregated BLS signature Π_h proves agreement by a supermajority of witnesses. This means that most historical data becomes redundant for security purposes.

Pruning is *local and voluntary*: each node decides what to prune depending on its role. We distinguish three classes of nodes, each with different storage strategies.

Light nodes (light clients) minimize storage and bandwidth. After checkpoint C_h, they store only:

- the latest finalized checkpoint $(StateRoot_h, BlockHash_h, \Pi_h)$,
- their own account entry (balance, counter, metadata),
- Merkle proofs linking their account to $StateRoot_h$.

They prune all transactions, PoBs, and historical state transitions. To verify payments or balances, they request Merkle proofs from full nodes. Their storage and bandwidth requirements scale only logarithmically with the number of accounts.

Full nodes validate all new blocks and maintain the blockchain's security guarantees, while pruning unnecessary history. After checkpoint C_h, a full node retains:

- the complete account state consistent with $StateRoot_h$,
- the latest checkpoint $(StateRoot_h, BlockHash_h, \Pi_h)$,
- all blocks produced since C_h (to follow forks and validate chain growth),
- any older blocks still within the PoB expiration window *exp* (to validate unexpired PoBs).

Conversely, they prune:

- expired PoBs (older than *exp* relative to the newest PoB),
- detailed transaction payloads prior to h,
- per-block state transitions before checkpoint h (since the state root certifies the result).

Thus storage grows with the number of accounts N and the checkpoint interval ε, not indefinitely with chain height h.

Archive nodes voluntarily retain the full history from genesis: every PoB, transaction, state transition, and checkpoint. These archive nodes are not required for consensus or security, but they can provide valuable services such as forensic auditing, research, or regulatory compliance. Their existence is optional but beneficial for transparency and long-term archiving.

The three node types are compared in Table 3.13, which summarizes what each keeps and what it prunes.

Table 3.13: Comparison of light, full, and archive nodes: data kept vs. data pruned. Here h denotes the height of the latest finalized checkpoint, and exp is the PoB expiration window (in blocks). Full nodes retain all blocks after h to follow forks, while archive nodes store everything from genesis.

Node type	Data kept	Data pruned
Light node	• Latest finalized checkpoint $(StateRoot_h, BlockHash_h, \Pi_h)$ • Own account entry • Merkle proofs linking account to $StateRoot_h$	• All transaction payloads • All PoBs (expired and unexpired) • All per-block state transitions
Full node	• Latest finalized checkpoint at height h • Complete account state (balances, counters) • All blocks since C_h (to handle forks) • Older blocks within the PoB expiration window exp	• Expired PoBs (older than exp) • Transaction payloads prior to h • Per-block state transitions prior to h
Archive node	• Entire chain from genesis • All PoBs, transactions, and states • All checkpoints (past and present)	

Without pruning, storage grows linearly with the number of blocks, $\mathcal{O}(M \cdot h)$ for M the block size and height h. With checkpoints, full-node storage is bounded by $\mathcal{O}(N + \varepsilon)$ where N is the number of accounts and ε is the checkpoint interval. Light nodes reduce this further to $\mathcal{O}(\log N)$ by relying on Merkle proofs. Archive nodes deliberately accept linear growth in order to maintain full historical traceability.

Implementation considerations. To avoid premature data loss, nodes typically delay pruning until at least one additional checkpoint has been finalized, ensuring redundant availability of state data during propagation. Incentive schemes may encourage some participants to serve checkpoint states and Merkle proofs, while archive nodes can voluntarily or commercially provide historical data. This heterogeneity of node types creates a robust ecosystem: light nodes for accessibility, full nodes for security, and archive nodes for transparency.

3.4.4 Economic Incentives

While the pBFT protocol ensures technical correctness, witnesses also need economic incentives to participate honestly and consistently. Because pBFT requires three distinct rounds of communication (pre-prepare, prepare, commit), witnesses must be motivated to remain active throughout all phases rather than dropping out. This is achieved through a "carrot and stick" mechanism: rewards for cooperation, and penalties for deviation.

Definition 3.40: pBFT Participation Incentives

Let w be a witness in the checkpoint protocol. The incentive scheme is defined by three components:

Collateral (Stick). Before participating, w must lock a collateral stake Ψ_w proportional to its average block reward R_w in the current epoch:

$$\Psi_w \geq \beta \cdot R_w,$$

where $\beta > 1$ is a fixed system parameter. If w equivocates (*e.g.*, sends conflicting messages) or is caught behaving Byzantine, the entire collateral Ψ_w is slashed.

Phase Rewards (Carrot). Witnesses receive explicit rewards for active participation in each protocol phase:

$r_{\text{prepare}} > 0$	reward for broadcasting a valid PREPARE,
$r_{\text{commit}} > 0$	reward for broadcasting a valid COMMIT,
$r_{\text{primary}} > 0$	extra reward to the primary for a valid PRE-PREPARE.

Penalty for Non-Participation (Stick). If w fails to participate in a round, it incurs a monetary penalty $\rho_w > 0$. This penalty is chosen so that

$$\rho_w \geq \max\{c_{\text{rep}}, c_{\text{prim}}\},$$

where c_{rep} and c_{prim} denote the per-round costs of participation for replicas and primaries, respectively. This ensures that remaining silent is never cost-saving. Repeated absences increase ρ_w, so that persistent inactivity becomes economically unsustainable.

Thus, the parameters governing incentives are: Ψ_w (collateral stake), β (collateral multiplier), $r_{\text{prepare}}, r_{\text{commit}}, r_{\text{primary}}$ (phase rewards), and ρ_w (non-participation penalty), calibrated so that honest participation strictly dominates deviation.

This incentive structure ensures that honest participation is always the rational strategy for a witness:

- The *collateral* Ψ_w is scaled to prior block rewards: the more successful a miner has been, the more they must risk. This prevents witnesses from minimizing their stake while still holding significant influence.
- The *phase rewards* $r_{\text{prepare}}, r_{\text{commit}}, r_{\text{primary}}$ are the carrot that motivates active participation in all protocol phases.
- The *penalty* ρ_w for non-participation is the stick that prevents free-riding and gradually removes unreliable witnesses.

The precise values of the parameters β, $r_{\text{prepare}}, r_{\text{commit}}, r_{\text{primary}}$, and the escalation rule for ρ_w are left as system settings to be chosen at deployment. What matters is that (i) collateral scales with potential impact; and (ii) rewards and penalties together make full, honest participation the strictly dominant strategy. This design directly supports the incentive-compatibility analysis of Proposition 3.80 of Section 3.6.3.

3.5 System Analysis

We concentrate the analysis of the blockchain in this section. We discuss expiration of PoBs, the energy consumption of the blockchain, the fuel and bounty structure and its implications as well as the demurrage.

3.5.1 PoB Expiration

The PoB expiration mechanism serves multiple critical functions: preventing replay attacks, limiting the search space for duplicate detection, and ensuring behavioral relevance. Understanding its operation requires examining both its security properties and practical implications.

Example 3.41

Consider a blockchain where exp = 24 hours. Alice takes a metro ride at 9:00 AM on Monday, generating π_A. Bob takes a bus ride at 10:00 AM on Tuesday, generating π_B.
Now suppose a miner on Wednesday wants to create a new block. The most recent PoB already in the blockchain is Carol's bike ride from 2:00 PM on Tuesday (π_C). When the miner considers including Alice's PoB, the system checks: (2:00 PM Tuesday - 9:00 AM Monday) = 29 hours > 24 hours = exp. Therefore, Alice's PoB is rejected as expired. If the miner considers Bob's PoB: (2:00 PM Tuesday - 10:00 AM Tuesday) = 4 hours < 24 hours = exp. Therefore, Bob's PoB is still valid and can be included.
In this example, the validators are the respective transit authorities: the Paris Metro for Alice's π_A and the city bus system for Bob's π_B. When Alice taps her card at 9:00 AM Monday, the metro terminal (acting as a validator on behalf of the Paris Metro authority) generates the timestamp and creates the signed PoB. Similarly, Bob's bus terminal generates his timestamp at 10:00 AM Tuesday. These timestamps are cryptographically signed by the validators at the moment of behavior execution, ensuring they cannot be forged or altered later.
The key insight is that expiration is relative to the newest PoB timestamp already in the blockchain, not the current wall-clock time. This ensures that miners do not need synchronized clocks and that the validation is deterministic based purely on blockchain content.

Efficient Duplicate Detection. The expiration mechanism dramatically reduces the computational overhead of ensuring PoB uniqueness. When validating a new PoB π, miners only need to search backwards through the blockchain until they reach a block where the newest PoB is more than exp older than π's timestamp.

Example 3.42

Suppose the current block contains $\pi_{current}$ with timestamp 2:00 PM, and $exp = 6$ hours. When checking if π_{new} (timestamp 1:00 PM) already exists in the blockchain, miners search backwards until they find a block where the newest PoB has timestamp $\leq$ 7:00 AM (1:00 PM - 6 hours). They can stop searching there because π_{new} cannot possibly exist in earlier blocks – if it did, those earlier blocks would themselves be invalid (containing expired PoBs).
Here, each PoB's timestamp was generated by its respective validator at the time of the behavior. For instance, $\pi_{current}$ (2:00 PM) might have been validated by a bike-sharing station's terminal, while π_{new} (1:00 PM) could be from a subway turnstile. The key point is that these timestamps are validator-generated and signed, not self-reported by users, making them trustworthy for the blockchain's expiration mechanism.

The expiration mechanism prevents several attack vectors:

Replay Attack Prevention: An attacker cannot reuse old PoBs because they become invalid after the expiration window. For instance, if Alice's metro ride from last month was somehow captured by an attacker, they cannot use it to claim rewards today.

Long-Range Attack Mitigation: By limiting PoB validity windows, the system prevents attackers from building alternative blockchain histories using very old behaviors. An attacker cannot suddenly "discover" a large cache of month-old PoBs to create a competing chain.

However the choice of exp value creates important trade-offs. Indeed, a short expiration (*e.g.*, 1 hour) provides stronger security and lower validation overhead but may exclude legitimate behaviors due to network delays or temporary disconnections. A user taking a subway ride in an area with poor connectivity might find their PoB expired before they can submit it. On the other hand, a long expiration (*e.g.*, 1 week) accommodates network issues and allows batch submission of behaviors but increases storage requirements and validation complexity. Users could accumulate behaviors offline and submit them later, but miners must search through more blockchain history.

The expiration mechanism directly affects blockchain efficiency. In a system with exp = 24 hours and an average of 1000 PoBs per hour, miners typically need to check only the last 24,000 PoBs for duplicates rather than the entire blockchain history. This bounded search complexity ensures that validation time remains constant as the blockchain grows, unlike systems where duplicate detection time increases linearly with blockchain age.

Since PoB expiration depends only on timestamps signed by validators (not miner clocks), different miners will always agree on expiration status. This eliminates potential consensus failures that could arise from clock synchronization issues between geographically distributed miners.

3.5.2 Energy Consumption

In our assessment of the system's energy consumption, we focus exclusively on the energy requirements of the consensus mechanism, deliberately excluding two aspects: (i) the verification of valid transactions, which is a universal requirement across all blockchain systems;

and (ii) the optional quantification of third-party PoBs, as this process is not mandatory for system operation. Let us consider a scenario where all miners are utilizing their complete set of non-expired PoBs for mining activities. We do the analysis for honest miners that are using every possible PoB to launch a VDF on the public tip of the blockchain. Under these conditions, each valid PoB corresponds to a single VDF computation. A crucial property of VDFs is their inherent non-parallelizability. Consequently, each non-expired PoB necessitates the dedication of one CPU core for VDF computation. This one-to-one correspondence between PoBs and CPU cores represents a fundamental constraint on the system's energy consumption. This architectural design introduces a significant departure from traditional Nakamoto-style consensus mechanisms. In our system, miners cannot amplify their mining capacity through the mere addition of computational resources. The mining power is intrinsically tied to the quantity and quality of valid PoBs a miner can produce, rather than raw computational capability.

Formally, if we denote the set of all miners as M, and for each miner $m \in M$, we denote their set of non-expired PoBs as Π_m, then the total energy consumption E of the consensus mechanism can be expressed as:

$$E \approx \sum_{m \in M} \sum_{p \in \Pi_m} e_{VDF} \tag{3.5}$$

where e_{VDF} represents the energy consumed by a single CPU core in computing one VDF.

This formulation underscores the system's inherent energy efficiency and scalability characteristics, as the energy consumption scales linearly with the number of valid PoBs rather than with the computational resources that participants might deploy.

It is important to stress that our analysis of energy consumption considers only honest participants executing the protocol as intended. Malicious miners, who might waste computational effort on non-productive VDF executions, are excluded from this assessment, since their actions lie outside the rational operation of the system. From the perspective of honest consensus, the overall energy footprint remains modest: each valid PoB requires at most a single CPU core for the duration of one VDF computation. Thus, although some energy expenditure is unavoidable – and indeed must exist for consensus to be meaningful – it is uniquely justified in our system: for every VDF running on a CPU, there exists a corresponding, verifiably beneficial human behavior that has been carried out. In practice, this translates to negligible consumption when compared to the vast computational races of Proof-of-Work blockchains, whose electricity demand is comparable to that of nation states, or even to many Proof-of-Stake protocols, where validator infrastructures often involve significant server clusters. By design, our PoB system binds energy use directly to meaningful behavioral proofs, ensuring that the cumulative energy cost of maintaining consensus remains very low, even at large scale.

Numerical Example. To provide an order-of-magnitude estimate, consider a representative CPU core consuming $P_{\text{core}} \approx 10$ W when executing cryptographic workloads. If the average VDF evaluation lasts $t_{\text{VDF}} \approx 10$ seconds (in line with our prototype target block time), then the energy per VDF computation is

$$e_{\text{VDF}} \approx P_{\text{core}} \times t_{\text{VDF}} = 10 \times 10\,\text{J} = 100\,\text{J} \approx 2.8 \times 10^{-5}\,\text{kWh}.$$

Suppose the system processes on average 1000 valid PoBs per hour, each requiring one VDF. Then the total hourly consumption amounts to

$$1000 \times 2.8 \times 10^{-5}\,\text{kWh} \approx 0.028\,\text{kWh},$$

which scales to roughly 245 kWh per year. This value is less than the annual electricity consumption of a single European household appliance, such as a refrigerator.

Comparison with Proof of Work. For context, Bitcoin's annual electricity consumption has been estimated using the Cambridge Bitcoin Electricity Consumption Index (CBECI) to range into the order of 100-150 TWh in prior assessments [**cbeci2023**]. More conservative recent estimates (*e.g.*, for 2023) place it around 70 TWh/year [**neumueller2023bitcoin**]. Estimates of Ethereum's pre-Merge electricity use vary, and some secondary sources place it in the tens of TWh per year; however, Ethereum's transition to proof of stake reduced its power demand by over 99 % [**kapengut2023event**]. Thus, our PoB design operates several orders of magnitude below PoW systems even under generous assumptions.

3.5.3 Monetary Policy Analysis

In our PoB blockchain, monetary dynamics per block are driven by two opposing forces: (i) *coin creation* through valid Proofs of Behavior (PoBs); and (ii) *demurrage* that removes a fixed fraction of outstanding balances every block. We model these forces per block to align with the rest of this chapter.

Definition 3.43: Per-block monetary update

Fix a per-block demurrage rate $d \in (0,1)$ and an initial supply $T(0) \geq 0$. For each block index $i \in \mathbb{N} = \{0,1,2,\dots\}$:

- $T(i) \in [0,\infty)$ denotes the total money supply *immediately after* block i;
- $C(i) \in [0,\infty)$ denotes the number of coins *created at* block i (*e.g.*, via valid PoBs).

The per-block evolution of the supply is given by the linear recurrence

$$T(i+1) \;=\; (1-d)\,T(i) \;+\; C(i), \qquad i = 0,1,2,\dots.$$

Intuitively, demurrage leaks out a fraction d of the incumbent stock each block, while $C(i)$ adds fresh issuance. When the inflow stabilizes to a constant, the stock stabilizes at the unique level where "outflow per block" equals "inflow per block."

Lemma 3.44: Stable linear recursion with vanishing input

Let $(e_i)_{i\geq 0}$ satisfy

$$e_{i+1} \;=\; \lambda\, e_i \;+\; u_i, \qquad i = 0,1,2,\dots,$$

with $|\lambda| < 1$ and $u_i \to 0$. Then $e_i \to 0$.

Proof of Lemma 3.44

Fix $\varepsilon > 0$. Since $u_i \to 0$, choose N such that $|u_i| \leq \varepsilon(1-|\lambda|)/2$ for all $i \geq N$. Unfold the recursion from N to $i-1$:

$$e_i \;=\; \lambda^{\,i-N} e_N \;+\; \sum_{k=0}^{i-N-1} \lambda^k\, u_{i-1-k}.$$

Take absolute values and split the sum into a geometric series:

$$|e_i| \le |\lambda|^{i-N}|e_N| + \sum_{k=0}^{i-N-1} |\lambda|^k |u_{i-1-k}| \le |\lambda|^{i-N}|e_N| + \frac{\varepsilon(1-|\lambda|)}{2} \sum_{k=0}^{\infty} |\lambda|^k.$$

Since $\sum_{k\ge 0} |\lambda|^k = 1/(1-|\lambda|)$, we get

$$|e_i| \le |\lambda|^{i-N}|e_N| + \frac{\varepsilon}{2}.$$

Let $i \to \infty$; the first term vanishes, hence $\limsup_{i\to\infty} |e_i| \le \varepsilon/2$. As $\varepsilon > 0$ was arbitrary, $e_i \to 0$. ■

Theorem 3.45: Pointwise convergence of the money supply

Fix a per-block demurrage rate $d \in (0,1)$ and define the total supply sequence $(T(i))_{i\ge 0}$ by

$$T(i+1) = (1-d)\,T(i) + C(i), \qquad i = 0,1,2,\ldots,$$

where $C(i) \ge 0$ is the number of coins created at block i. Assume the creation sequence converges to a finite constant:

$$C(i) \xrightarrow[i\to\infty]{} C_\infty \in [0,\infty).$$

Then the total supply converges to

$$T(i) \xrightarrow[i\to\infty]{} \frac{C_\infty}{d}.$$

Proof of Theorem 3.45

In steady state, "inflow = outflow per block." Outflow is demurrage $d\,T$, inflow is creation C_∞, so the steady-state value must be $T^\star = C_\infty/d$. We prove that $T(i)$ indeed approaches this $T^\star$.

Step 1: Subtract the target. Define the deviation from the candidate limit

$$E(i) = T(i) - \frac{C_\infty}{d}.$$

We rewrite the recursion for $E(i)$ by subtracting C_∞/d from both sides of the update for T:

$$\begin{aligned} E(i+1) &= T(i+1) - \frac{C_\infty}{d} = (1-d)T(i) + C(i) - \frac{C_\infty}{d} \\ &= (1-d)\Big(E(i) + \frac{C_\infty}{d}\Big) + C(i) - \frac{C_\infty}{d} = (1-d)E(i) + \underbrace{\big(C(i) - C_\infty\big)}_{=:u_i}. \end{aligned}$$

Thus E satisfies a linear recursion

$$E(i+1) \;=\; \lambda E(i) \;+\; u_i \quad \text{with} \quad \lambda = 1-d \in (0,1),\;\; u_i = C(i) - C_\infty.$$

Step 2: Apply Lemma 3.44. By assumption $C(i) \to C_\infty$, hence $u_i \to 0$. With $|\lambda| = 1-d < 1$, Lemma 3.44 gives $E(i) \to 0$.

Step 3: Conclude for $T(i)$. Since $E(i) = T(i) - C_\infty/d \to 0$, we have

$$T(i) \;\longrightarrow\; \frac{C_\infty}{d}.$$

Any dependence on $T(0)$ is multiplied by $(1-d)^i$, which decays to 0 as $i \to \infty$. Hence the limit is independent of $T(0)$. ■

The total money supply formula $T(i+1) \;=\; (1-d)\,T(i) \;+\; C(i)$, with $i = 0, 1, 2, \ldots$ (Definition 3.43) is a leaky-tank model: each block pours in $C(i)$ and drains a fraction d. If the inflow settles to a constant C_∞, the tank level settles to the unique height where "drain = inflow," *i.e.*, $d\,T = C_\infty$, hence $T = C_\infty/d$.

Theorem 3.45 makes the money stock *endogenous* to on-chain activity: the steady-state level $T^\star = C_\infty/d$ rises when more behaviors are performed (C_∞ larger due to more valid PoBs) and falls when activity wanes (C_∞ smaller), with demurrage guaranteeing convergence regardless of initial conditions. In contrast to Bitcoin and many other cryptocurrencies – whose issuance is fixed by an exogenous, time-based schedule largely independent of contemporaneous usage – our supply adapts to demand: more usage produces more issuance and a higher $T^\star$, less usage produces less issuance and a lower $T^\star$. Thus the total supply is tied directly to how much the currency is actually used.

Remark 3.46: Oscillations: pointwise vs. time-average

If $C(i)$ or the class mix oscillates persistently (seasonality, protocol cycles), the pointwise limit of $T(i)$ or $P(i)$ may not exist; one obtains a bounded periodic steady state centered near the same level. In that case, the *time average* still centers at $\overline{C}/d$ for T (and the analogous price average), but this is weaker than pointwise convergence.

Economic variables and price level. We now connect the per-block monetary dynamics (Definition 3.43) to two economic quantities that are simple enough for protocol design yet grounded in standard macro ideas. Our modeling is a reduced-form analogue of the classical "quantity theory" (Fisher's equation of exchange), where prices rise when the money stock grows relative to *velocity* (how fast balances turn over) and *real activity* [**Fisher1911**]. Unlike early treatments that sometimes treated velocity as constant, modern macro (from Keynes's liquidity-preference view to contemporary textbooks) emphasizes that velocity is endogenous – shaped by institutions and incentives [**Keynes**, **Mankiw2019**]. Demurrage, long advocated by Gesell, acts exactly on this margin: by imposing a carrying cost on idle balances, it discourages hoarding and tends to stabilize effective turnover [**Gesell**]. For our blockchain setting, we make these ideas operational and auditable on-chain by defining "work" directly from the protocol's PoB quantification and by introducing an explicit velocity term; our price index is therefore a protocol-level analogue of Fisher's relation (not a consumer price index), telling us how much money is chasing how much verified work at the circulation intensity the system sustains [**Fisher1911**, **Keynes**, **Mankiw2019**,

Gesell]. The plan is simple: (i) *measure* real activity directly from PoBs; (ii) define a price index that also includes a standard velocity term; (iii) assume that both activity and velocity stabilize in the long run; and (iv) assume issuance is (asymptotically) proportional to measured work. Under these assumptions, the price level converges to a constant.

We reuse the protocol's own PoB quantification so that consensus and economics share one, auditable measurement.

Definition 3.47: Per-block Work via PoB quantification

Let $\mathrm{PoB}(i)$ be the set of valid Proofs of Behavior included in block i, and let $\mathsf{Quantify} : \mathrm{PoB} \to [0,\infty)$ be the protocol's PoB scoring function. The *per-block work* is

$$W(i) \;=\; \sum_{\pi \in \mathrm{PoB}(i)} \mathsf{Quantify}(\pi).$$

Units. One *work unit* (wu) equals one unit returned by Quantify, so $W(i)$ has units wu/block and inherits the protocol's calibration.

Before defining a price index, we first isolate the role of *velocity*: how quickly outstanding balances turn over relative to the measured work performed in each block. This captures institutional and behavioral frictions (*e.g.*, hoarding lowers velocity) and lets us separate "how much money exists" from "how actively it circulates."

Definition 3.48: Velocity (per block)

Let $V(i) > 0$ denote the *effective velocity multiplier* at block i. It is a dimensionless factor summarizing how frequently the outstanding stock $T(i)$ is mobilized during block i relative to the per-block work $W(i)$. Intuitively, $V(i) = 1$ means each unit of currency turns over once in block i; $V(i) > 1$ indicates multiple effective turnovers; $V(i) < 1$ indicates partial idle balances due to frictions. The precise estimation of $V(i)$ is protocol- and application-dependent (*e.g.*, based on spending/settlement statistics) and is not required for the theoretical results below.

With velocity explicit, we can now define a reduced-form *price level* that rises when there is more money chasing the same effective activity, and falls when the same money chases more effective activity. A scale factor turns the ratio into a reportable index.

Definition 3.49: Price Level

Fix a positive scaling constant $\kappa > 0$ (units). The *price level* at block i is

$$P(i) \;=\; \kappa\,\frac{T(i)}{V(i)\,W(i)} \qquad \text{(defined when } W(i) > 0\text{)}.$$

Here $T(i)$ is the total money stock after block i, $W(i)$ is the per-block work from Definition 3.47, and $V(i)$ is the velocity from Definition 3.48. The ratio $T(i)/(V(i)W(i))$ is dimensionless; κ calibrates units so that $P(i)$ can be reported as an index. In this reduced form, higher T relative to $V \cdot W$ corresponds to a higher price level, while higher V or W (for fixed T) corresponds to a lower price level.

To take limits cleanly, we assume that activity and velocity stabilize to finite, nondegenerate levels.

Assumption 3.50: Work tracks long-run usage/importance

The sequence $\{W(i)\}_{i\geq 0}$ admits a finite limit $W_\infty \geq 0$, with $W_\infty > 0$ under sustained activity:

$$W(i) \xrightarrow[i\to\infty]{} W_\infty.$$

Assumption 3.51: Velocity stabilizes

The velocity factor admits a finite, positive limit:

$$V(i) \xrightarrow[i\to\infty]{} V_\infty \in (0,\infty).$$

To keep the exposition focused, we *abstract away* from PoB subtypes and any fine-grained reward differences. We posit a single, long-run conversion rate from work to coins.

Assumption 3.52: Issuance proportional to work

There exists $\alpha > 0$ (units: coins per wu) such that

$$C(i) = \alpha\, W(i) + r(i), \qquad \text{with} \quad r(i) \xrightarrow[i\to\infty]{} 0.$$

Equivalently (when $W(i) > 0$), the per-work issuance rate satisfies $C(i)/W(i) \to \alpha$.

With these definitions and assumptions, and using the pointwise money-supply limit (Theorem 3.45), we obtain a simple closed-form limit for prices.

Theorem 3.53: Price stability under proportional issuance

Assume the per-block monetary update (Definition 3.43) with demurrage $d \in (0,1)$ and the price index from Definition 3.49. Suppose Assumptions 3.50, 3.51, and 3.52 hold with $W_\infty > 0$ and $V_\infty \in (0,\infty)$. Then

$$P(i) = \kappa \frac{T(i)}{V(i)\,W(i)} \xrightarrow[i\to\infty]{} \kappa \frac{\alpha}{d\,V_\infty}.$$

Proof of Theorem 3.53

By Assumption 3.52 and $W(i) \to W_\infty$, we have $C(i) = \alpha W(i) + r(i) \to \alpha W_\infty =: C_\infty$. The pointwise money-supply limit (Theorem 3.45) gives $T(i) \to C_\infty/d = \alpha W_\infty/d$. Since $W(i) \to W_\infty > 0$ and $V(i) \to V_\infty \in (0,\infty)$, continuity of division yields $\frac{T(i)}{V(i)W(i)} \to \frac{\alpha}{d\,V_\infty}$. Multiplying by κ proves the claim. ■

Theorem 3.53 shows that prices stabilize at a level governed by three transparent determinants: (i) *issuance per unit of verified work* α; (ii) the *demurrage rate* d; and (iii) long-run *velocity* V_∞. Crucially, issuance scales with actual usage via $W(i)$, so the *total money supply and the price level are endogenous to activity*. This is unlike Bitcoin and many cryptocurrencies, where issuance follows an *exogenous* time schedule (*e.g.*, halvings) largely independent of contemporaneous usage; and unlike most fiat regimes, where supply is steered *discretionarily* by policy makers. Here, proportional issuance + demurrage implement an *algorithmic, self-adjusting* policy: more (less) real work eventually raises (lowers) issuance and the stock, while demurrage continuously

offsets idle balances, jointly anchoring prices in the long run [**Fisher1911**, **Keynes**, **Mankiw2019**, **Gesell**].

3.6 Security Analysis

The security of a blockchain system is fundamental to its reliability and long-term adoption. In this section, we analyze the Proof of Behavior (PoB) blockchain through three complementary perspectives. First, we establish a formal correspondence with the Bitcoin Backbone Protocol to demonstrate that PoB inherits key persistence and liveness guarantees from well-studied models. Next, we investigate attack vectors unique to PoB blockchains, outlining potential adversarial strategies such as selfish and stubborn mining and proposing countermeasures. Finally, we evaluate the security of the checkpoint mechanism, which strengthens finality and mitigates long-range attacks. Together, these subsections provide a comprehensive assessment of PoB's resilience, ensuring both theoretical soundness and practical robustness

3.6.1 Bitcoin Backbone Protocol Correspondance

The security of our Proof of Behavior (PoB) blockchain can be understood by establishing a close correspondence with the *Bitcoin Backbone Protocol* of Garay, Kiayias, and Leonardos [**bitcoin_backbone**]. Instead of reproving all properties from scratch, we show how the modeling framework and algorithms of the backbone protocol translate naturally to our PoB construction, which allows us to inherit the same security guarantees.

Backbone Protocol Model. In the Bitcoin Backbone Protocol, the execution of a protocol Π is driven by an environment program $\mathcal{Z}$ that schedules a fixed set of participants $P_1, \ldots, P_n$. Each participant, as well as the adversary $\mathcal{A}$, is modeled as an *Interactive Turing Machine* (ITM) with communication, input, and output tapes. The adversary may corrupt up to t out of the n parties.

Parties interact with two idealized functionalities: (i) a *random oracle* (the hash function used for Proof of Work), with each party limited to q queries per round; (ii) a *diffusion channel*, an eventually reliable broadcast network. This model is deliberately *flat*: every party has equal resources. Real-world differences in hashing power are abstracted by viewing a stronger miner as a cluster of multiple flat-model parties.

On top of this model, the protocol relies on two central ingredients: a method for validating and comparing chains, and a procedure for block generation. The latter is where the correspondence with our PoB system becomes clearest. In the Bitcoin Backbone Protocol, block generation is captured by the Proof of Work algorithm, as transcribed in Algorithm 3.54.

Algorithm 3.54: Proof of Work

```
1: while ctr ≤ q do
2:     if H(ctr, h) < T then
3:         Append block to blockchain
4:         return Blockchain
5:     ctr ← ctr + 1
6: return Blockchain
```

Here success is probabilistic: the counter value ctr satisfying the target condition is *unknown in advance*. In our PoB blockchain, block production via VDFs can be written in an almost identical form in Algorithm 3.55.

Algorithm 3.55: Proof of Behavior – VDF

```
while ctr ≤ q do
    if ctr = τ then
        Append block to blockchain
        return Blockchain
    ctr ← ctr + 1
return Blockchain
```

The only difference is that in PoB, the "winning" counter τ is *fixed* by the quantification of the behavior and therefore known in advance. In PoW it must be discovered probabilistically. Despite this cosmetic difference, both processes share the same structural property: the expected time to success is inversely proportional to the available resource (hashing power in PoW, behavioral quantification in PoB). This makes the two systems equivalent from the perspective of the backbone analysis. See the correspondance in Table 3.14.

Table 3.14: Proof of Work and Proof of Behavior correspondence

Bitcoin Backbone (PoW)	**PoB Blockchain (VDF)**
Hashing power ρ	Behavioral quantification $Q(\pi)$
Random oracle queries	Sequential VDF steps
Success: $H(ctr, h) < T$ (unknown in advance)	Success: $ctr = \tau$ (fixed by quantification)
Expected block time $\propto 1/\rho$	Expected block time $\propto 1/Q(\pi)$
Chain selection: highest cumulative difficulty	Chain selection: highest cumulative difficulty

Definition 3.56: System Correspondence

The Bitcoin Backbone Protocol maps onto our PoB blockchain via the following correspondence (see Table 3.14 for a concise summary):

$$\begin{aligned} \text{Mining power} &\mapsto \text{Aggregate behavioral quantification} \\ \text{Proof of Work} &\mapsto \text{VDF evaluation (steps fixed by quantification)} \\ \text{Block production rate} &\mapsto \text{PoB production rate, weighted by quantification} \\ \text{Chain selection} &\mapsto \text{Selection by cumulative difficulty } \delta \end{aligned}$$

We define the Global Stabilization Time so that it can be used in our synchony assumptions.

Definition 3.57: Global Stabilization Time (GST)

In the *partial synchrony* model, the *Global Stabilization Time (GST)* is an unknown real time after which the network permanently stabilizes: there exists a delay bound $\Lambda > 0$ such that every message sent after GST is delivered within at most Λ time units. Before GST, message delays may be arbitrary and unbounded.

Assumption 3.58: Fairness

Each valid Proof of Behavior (PoB) π entitles its producer to exactly one sequential VDF execution attempt on a given parent block. In particular, a PoB cannot be used to

launch multiple concurrent VDF computations; the expected block production rate of a participant is therefore proportional to the aggregate quantification of their valid PoBs.

Assumption 3.59: Uniqueness

For every parent block B and every valid PoB π, at most one valid block B' can be mined by combining (B, π) with a VDF output. That is, mining rights are uniquely bound to the pair (B, π), and a single PoB cannot be reused to generate multiple distinct sibling blocks at the same height.

Assumption 3.60: Blockchain Correspondance

As in [**bitcoin_backbone**], the guarantees hold under three natural assumptions:

1. *Honest majority:* honest parties collectively hold more quantification than adversaries;
2. *Network diffusion (eventual synchrony):* there exists a delay bound Λ such that after an unknown GST all messages are delivered within Λ; and the difficulty δ is such that during synchronous periods Λ is small relative to the target block interval Δ_t.
3. *Unforgeability:* digital signatures and PoB attestations cannot be forged.

With these assumptions in place, our blockchain inherits the main theorems of the backbone analysis.

Lemma 3.61: Backbone Reduction

Assume synchrony as in Assumption 3.60 with honest power α and adversarial power β (with $\alpha > \beta$), together with Assumptions 3.58 and 3.59. Assume the difficulty controller targets the desired average block time Δt so that the expected number of network-wide successful attempts per round satisfies $\lambda_r \in [\lambda_-, \lambda_+]$ for all rounds r, where $0 < \lambda_- \leq \lambda_+ < 1$ are constants determined by the policy. Then there exist absolute constants $c_H, c_A \in (0, 1)$ (depending only on λ_-, λ_+) such that, in each round r:

$$\Pr[\text{exactly one honest success in } r] \geq c_H\,\alpha$$
$$\Pr[\text{exactly one adversarial success in } r] \leq c_A\,\beta$$

Consequently, over any window of T rounds, except with $\exp(-\Omega(T))$ probability, the number of uniquely successful honest rounds is $\Omega(\alpha T)$ and exceeds the number of uniquely successful adversarial rounds by a linear margin; hence the backbone common-prefix (persistence), chain-growth, and chain-quality properties hold with overwhelming probability.

Proof of Lemma 3.61

Fix a round r. Let n_r^{H} and n_r^{A} be the honest and adversarial numbers of attempts (one per valid PoB by Fairness), with $n_r = n_r^{\mathsf{H}} + n_r^{\mathsf{A}}$. Let p_r be the per-attempt success probability (set by the difficulty); define $\lambda_r^{\mathsf{H}} := n_r^{\mathsf{H}} p_r$, $\lambda_r^{\mathsf{A}} := n_r^{\mathsf{A}} p_r$, and $\lambda_r := \lambda_r^{\mathsf{H}} + \lambda_r^{\mathsf{A}} \in [\lambda_-, \lambda_+]$ by assumption.
Step 1: Lower bound for a unique honest success. By independence of attempts across parties

in the round,

$$\Pr[\text{exactly one honest success}] = n_r^{\mathsf{H}} p_r (1-p_r)^{n_r-1}.$$

Using $(1-p)^m \geq 1-pm$ for $p \in [0,1]$ and $m \in \mathbb{N}$,

$$\Pr[\text{exactly one honest success}] \;\geq\; n_r^{\mathsf{H}} p_r \big(1 - p_r n_r\big) \;=\; \lambda_r^{\mathsf{H}}(1-\lambda_r) \;\geq\; \lambda_-(1-\lambda_+) \cdot \frac{\lambda_r^{\mathsf{H}}}{\lambda_r}.$$

By definition of α and β as fractions of total attempt mass, $\lambda_r^{\mathsf{H}}/\lambda_r = \alpha$. Hence

$$\Pr[\text{exactly one honest success}] \;\geq\; (\lambda_-(1-\lambda_+))\,\alpha.$$

Set $c_H := \lambda_-(1-\lambda_+) \in (0,1)$.
Step 2: Upper bound for a unique adversarial success. Trivially,

$$\Pr[\text{exactly one adversarial success}] \;\leq\; \mathbb{E}[\text{adversarial successes}] \;=\; \lambda_r^{\mathsf{A}} \;=\; \lambda_r \cdot \frac{\lambda_r^{\mathsf{A}}}{\lambda_r} \;\leq\; \lambda_+ \beta.$$

Set $c_A := \lambda_+ \in (0,1)$.
Step 3: Linear gap and concentration. Let X_r be the indicator of "exactly one honest success in r" and Y_r for the adversarial counterpart. By the bounds above, $\mathbb{E}[X_r] \geq c_H \alpha$ and $\mathbb{E}[Y_r] \leq c_A \beta$ for all r. Because the underlying per-attempt trials are independent across rounds, the variables $\{X_r, Y_r\}$ are functions of independent inputs per r, and standard Chernoff (or bounded-differences/Azuma) yields, for all T,

$$\sum_{r=1}^{T} X_r \;=\; \Omega(\alpha T) \quad \text{and} \quad \sum_{r=1}^{T} Y_r \;=\; O(\beta T)$$

with failure probability $\exp(-\Omega(T))$. Since $\alpha > \beta$, the uniquely successful honest rounds dominate by a linear margin. This is exactly the backbone hypothesis from which common-prefix (persistence), chain-growth, and chain-quality follow by the standard arguments. Uniqueness ensures that a success maps to *at most one* valid child per (B, π), so no single attempt can generate siblings and inflate forks. ■

Theorem 3.62: Persistence (Common Prefix)

Under Assumption 3.60 with $\alpha > \beta$, and Assumptions 3.58 and 3.59, there exists a polylogarithmic function $f(\lambda)$ such that, except with negligible probability in the security parameter λ, no two honest parties at any times $t_1 < t_2$ hold ledgers that disagree on the prefix that excludes the last $f(\lambda)$ blocks of the longer ledger. Equivalently, once a block is buried by at least $f(\lambda)$ subsequent blocks, it is never removed from the main chain.

Proof of Theorem 3.62

By Lemma 3.61, the execution is coupled to a backbone execution with a characteristic string in which uniquely successful honest rounds occur with constant density dominating adversarial uniquely successful rounds. The standard common-prefix bound then applies: the probability that later adversarial work overturns a block already buried by a polylogarithmic number of subsequent blocks is negligible in λ. ■

Theorem 3.63: Liveness (Chain Growth)

Under Assumption 3.60 with $\alpha > \beta$, and Assumptions 3.58 and 3.59, there exists a polylogarithmic function $g(\lambda)$ such that, except with negligible probability, any valid transaction that is continuously broadcast by an honest party is included in the main chain within at most $g(\lambda)$ rounds of its first broadcast.

Proof of Theorem 3.63

By Lemma 3.61, chain-growth holds with a positive linear rate governed by the density of uniquely successful honest rounds, and chain-quality guarantees a constant honest fraction of blocks in any sufficiently long window. Hence, in any window of length $g(\lambda)$ (polylogarithmic in λ), the chain necessarily accrues enough honest blocks to include all pending valid transactions with overwhelming probability. ■

While Assumption 3.60 is sufficient to align our blockchain with the Bitcoin Backbone Protocol and to inherit its persistence and liveness properties, this correspondence relies on additional implicit premises. In particular, the analysis treats each valid PoB as granting exactly one sequential VDF execution, uniquely tied to a single block, thereby mirroring the fair lottery abstraction of the backbone model. As we show in the next section, these premises may be violated in adversarial settings, where parallel VDF computations and concurrent sibling block creation undermine the backbone analogy and give rise to PoB-specific attack strategies.

3.6.2 Special Attacks on PoB Blockchains

The backbone correspondence provides valuable insights into the persistence and liveness of the PoB blockchain, but it rests on implicit premises that do not always hold in adversarial environments. Most critically, the assumptions that each PoB grants exactly one sequential VDF execution and that mining rights are uniquely bound to block contents can be violated in practice. Adversaries with significant parallel resources may run multiple VDFs per PoB, while the independence of mining rights from block content allows the creation of concurrent sibling blocks. These deviations weaken the fairness and uniqueness properties assumed in Section 3.6.1 and open the door to novel attack strategies. Table 3.15 summarizes the contrast between the backbone assumptions and the realities of PoB operation, providing the foundation for our subsequent analysis of selfish mining, stubborn mining, and combined attacks.

These violations highlight structural vulnerabilities unique to PoB blockchains; in the following subsections, we analyze how they enable concrete adversarial strategies such as selfish mining, stubborn mining, and concurrent sibling block attacks.

The unique structure of PoB blockchains, where mining power derives from behavioral contributions rather than computational resources, introduces novel attack vectors that extend beyond traditional blockchain vulnerabilities. This section analyzes four critical attack strategies specific to our system and presents corresponding countermeasures.

The first attack is built on the fact that the mining rights for a block are not dependent on the block's contents. This means that a miner who gets mining rights can actually mine multiple sibling blocks instantly. We call this the *Concurrent Sibling Block Attack*. We propose countermeasures for this attack.

The other three attacks resemble each other and are different from the Concurrent Sibling Block Attack. In our security model, we initially assumed that each miner executes exactly one VDF per unexpired PoB. However, a resourceful attacker with sufficient computational

Table 3.15: Comparison of assumptions in the Backbone correspondence (Section 3.6.1) and their validity in PoB under adversarial settings (Section 3.6.2).

Assumption	Backbone correspondence	Special Attacks on PoB
Honest majority	Honest participants collectively produce more quantified behavior than adversaries, ensuring chain quality.	Still assumed, but weakened if adversaries exploit structural vulnerabilities (*e.g.*, parallel VDFs or concurrent blocks).
Network synchrony	Message delays are bounded by Λ, small compared to expected block time.	Still holds; not the main attack surface in PoB.
Unforgeability	Signatures and PoB attestations cannot be forged; validators prevent false behaviors.	Still holds; cryptographic guarantees remain intact.
Unique binding of mining rights to block contents	Mining rights tied uniquely to (parent block + PoB), preventing multiple competing blocks from one PoB.	**Broken:** adversaries can create *concurrent sibling blocks* from the same PoB, enabling forks, double-spends, or selfish strategies.
One sequential VDF per PoB	Each PoB grants exactly one sequential mining attempt, mirroring the fair lottery abstraction of the backbone model.	**Broken:** adversaries can execute multiple VDFs per PoB in parallel, undermining fairness.

capacity (multiple CPUs) could potentially violate this assumption by running multiple VDFs simultaneously. Consider an attacker possessing n unexpired PoBs $\{\pi_1, \pi_2, \ldots, \pi_n\}$ and at least n available CPUs. This attacker can launch parallel VDF computations, fundamentally altering the mining dynamics. The attacks we examine exploit this parallelization capability, combined with strategic withholding of discovered blocks. While these attacks require significant computational resources (linear in the number of PoBs), they pose legitimate threats to system integrity that must be addressed.

Concurrent Sibling Block Attacks

A structural peculiarity of VDF-based consensus in PoB blockchains is that mining rights are determined solely by the parent block B and the miner's PoB π, but not by the candidate block contents. Once a miner obtains the right to extend B using π, the VDF output (y, π_{VDF}) is valid for *any* block header at that height. This independence creates a subtle but powerful vulnerability: a miner can simultaneously construct several sibling blocks at the same height, all carrying the same VDF output and PoB but embedding different transactions. We call this the *Concurrent Sibling Block Attack*.

Attack strategy. An adversarial miner produces one block honestly and broadcasts it, while keeping one or more siblings private. These private forks may later be revealed opportunistically, invalidating honest miners' work. The potential advantage lies in double-spending or selective censorship: if a conflicting transaction is included in the private sibling, the attacker can later reveal it to cancel the honest chain segment.

Detection. In practice, concurrent siblings are almost always detected: once any honest node observes two siblings signed by the same miner on the same parent, it possesses cryptographic evidence of misbehavior. The only case in which detection does not occur is when, at the same moment, a strictly longer chain is propagated: in this case, nodes adopt the longer chain and may not relay the shorter sibling. In such situations, the attack is rendered harmless.

Countermeasure. One might think of preventing concurrent siblings by binding the VDF computation not only to the parent block and the miner's PoB, but also to the *entire contents* of the block. In that case, each possible ordering of transactions would lead to a different hash input, and hence to a different VDF challenge. This would indeed prevent a miner from reusing the same VDF output to generate two distinct blocks. However, this "solution" introduces a much more serious problem: the miner could prepare many different candidate blocks in advance by slightly modifying their contents (for example by reordering the transactions or including different subsets), and then run multiple VDFs in parallel on all of them. The effect would be to reintroduce a *computational race* very similar to Proof of Work, where success depends on who can afford the most parallel processors. This undermines the very purpose of using VDFs to keep mining fair and sequential. For this reason, we deliberately allow the VDF to depend only on the parent block and the PoB, and we address concurrent siblings through a dedicated countermeasure instead.

This countermeasure unfolds in three conceptual phases. Rather than listing protocol steps, we explain here the underlying logic and the reason why each phase is necessary.

Phase 1 – Detection and freezing (Algorithm 3.64): As soon as an honest node encounters two sibling blocks signed by the same miner at the same height, it has undeniable cryptographic proof that misbehavior occurred. At this moment the network must "pause" to prevent the attack from spreading. Freezing block growth at that height ensures that no honest miner wastes effort on building above possibly invalid data, and that everyone has time to examine the evidence. This first reaction isolates the problem: the siblings and their descendants are quarantined, and the miner is flagged as suspected, so the community knows who misbehaved.

Algorithm 3.64: Detection and freezing

Ensure: Block growth is frozen; evidence is shared; attacker marked as suspected.

1: **Trigger**: Node observes two sibling blocks at the same height by the same miner, or receives a valid EVIDENCE alert.
2: Enter *freeze mode*: stop accepting/relaying new blocks at and above that height; continue mempool gossip.
3: Quarantine both siblings and all their descendants.
4: Mark the miner as *suspected*: reject new blocks they mine during freeze.
5: Broadcast an EVIDENCE message containing the two headers and their proofs.

Phase 2 – Agreement on the honest branch (Algorithm 3.65): Once the attack evidence circulates, nodes may disagree on which sibling subchain they extended. The network cannot simply resume and hope for the best: different nodes may have seen different suffixes, and some may have extended one sibling while others extended the other. To restore a single agreed-upon history, the witnesses of the previous epoch run a pBFT round. The goal here is to agree on which branch of the chain will be kept and to record this decision in a *special checkpoint*. This checkpoint includes: (i) the agreed chain tip that is to be continued; (ii)

the headers of the siblings as cryptographic evidence of the attack; and (iii) the permanent exclusion of the attacker's account. This ensures that all honest nodes converge on the same branch, that the attacker is formally banned, and that the decision is irreversibly anchored in the blockchain itself.

To remove any incentive for an attacker to withhold a "stronger/longer" private branch and reveal it later for a double-spend, witnesses do not select the longest suffix. Instead, they apply a *first-seen* rule: among the valid candidate subchains extending the last checkpoint, they keep the subchain that *first appeared publicly* according to a quorum of witnesses' signed arrival attestations. Concretely, each witness provides (for each candidate subchain identifier) a signed record of the earliest local arrival time or sequence number of the subchain's tip. The pBFT round then chooses the subchain whose tip has the best (earliest) quorum-certified first-appearance score and embeds that choice in the special checkpoint. This policy neutralizes the classic "withhold-then-reveal" strategy: even if a secretly built branch is longer, it will lose to the earlier public branch unless the attacker managed to make their private branch publicly visible earlier – which defeats the purpose of withholding in the first place. The checkpoint records (i) the chosen branch/tip; (ii) cryptographic evidence of the sibling attack; and (iii) the permanent ban of the attacker's account, ensuring all honest nodes converge on the same continuation.

Algorithm 3.65: Agreement on the honest branch

Require: Last finalized checkpoint, authenticated channels, and pBFT with view-change.

Ensure: Special Checkpoint finalizing the chain tip, embedding evidence, and excluding the attacker.

1: Exchange views: each witness shares a commitment to its current longest chain since the last finalized checkpoint.

2: Run pBFT:

- *Pre-prepare*: primary proposes a candidate chain extending the last checkpoint and consistent with received views.
- *Prepare*: witnesses validate and echo acceptance; on quorum, continue.
- *Commit*: witnesses confirm; on quorum, the chain is agreed.
- *View change*: if needed, rotate the primary and retry.

3: Issue a Special Checkpoint: sign and publish the agreed tip (height/hash/state), sibling headers as attack proof, attacker's identity with permanent exclusion, and an explicit "concurrent sibling attack" marker.

Phase 3: Recovery through the special checkpoint (Algorithm 3.66): After the witnesses finalize the special checkpoint, all honest nodes adopt it and resume mining strictly on top of the branch it designates. The checkpoint itself decides which suffix of the chain is kept, so that valid honest blocks beyond the attack are preserved automatically. At the same time, all future attacker-produced blocks will be discarded, and the attacker's account is banned from future participation. In this way, the blockchain continues seamlessly, honest rewards and transactions remain valid, and the attack leaves no residual impact. These special checkpoints do not interfere with the epoch duration ε which is the size of an epoch. Even with multiple special checkpoints in an epoch, the size of the epoch remains constant. The

distance between two regular checkpoints is always ε regardless of the presence of special checkpoints within it.

Algorithm 3.66: Recovery through the special checkpoint

Require: Signed Special Checkpoint with valid quorum certificate.
Ensure: Chain resumes safely; attacker is excluded; invalid blocks are rejected.

1: Verify the Special Checkpoint's quorum certificate and roll back to its tip if necessary.
2: Update attacker status: change from *suspected* to **excluded**, and reject any block or transaction involving that account.
3: Resume mining strictly on top of the checkpointed tip.
4: Validator rule for each new block:
   - block must extend the checkpointed (or later) tip,
   - must not contain attacker's account,
   - must satisfy all normal validity checks.
5: Reject any block that violates these conditions.

Together these three phases explain the philosophy of the countermeasure: stop immediately to prevent damage from spreading, agree on a safe recovery point, and resume so that no honest transaction is lost and no attacker benefit survives.

Fork accountability. Formally, if two blocks B_1, B_2 at the same height h satisfy $\mathsf{miner}(B_1) = \mathsf{miner}(B_2)$ with $B_1 \neq B_2$, then the miner's account is invalidated. From this point, any block or transaction involving that account is automatically rejected. This provides strong disincentives: both mining rewards and all future economic activity of the attacker are forfeited.

The countermeasure provides two crucial theorems.

Theorem 3.67: Double-Spend Neutralization

Under this countermeasure, any attempted double-spend via concurrent siblings fails: the special checkpoint finalizes a single honest original branch and discards the attacker's blocks, ensuring that conflicting spends never survive.

Proof of Theorem 3.67

Suppose a transaction T was included on the honest branch but replaced by a conflicting transaction T' on the attacker's sibling. When witnesses finalize the special checkpoint, they agree to keep the original branch and discard the attacker's blocks even if their subchain is longer. As a result, T remains valid while T' is permanently removed. ■

Theorem 3.68: Rational Deterrence

For a rational miner, launching a sibling attack yields negative expected payoff compared to honest mining.

Proof of Theorem 3.68

Sibling publication is detected with overwhelming probability. When detected, the attacker loses all future interactions with the blockchain: from mining rights to payment reception. Thus, even if the attack briefly seemed profitable (which it is not thanks to Theorem 3.67), its outcome is strictly negative once the checkpoint is issued. In contrast, honest mining yields positive expected payoff. Therefore, rational miners are deterred from attempting the attack. ■

Selfish and Stubborn Mining in PoB Blockchains

The breakdown of the one-VDF-per-PoB assumption further enables *selfish mining* strategies. By withholding blocks and selectively releasing them, adversaries can exploit parallel VDF computations to gain a disproportionate share of rewards relative to their quantified behavior. We adapt the selfish mining analysis from Nakamoto-style blockchains to the PoB setting, showing how fairness can be compromised even under honest-majority assumptions.

A refinement of selfish mining, *stubborn mining*, exploits tie-breaking and propagation races to extend adversarial control over the blockchain. In PoB, mining elsewhere than the tip does not impact mining power at the tip. This amplifies the effectiveness of such strategies, since adversaries can force honest miners to waste effort on branches that are later abandoned. We analyze how stubborn mining interacts with PoB's consensus dynamics and checkpoint mechanism.

Finally, adversaries may combine concurrent selfish mining with stubborn mining to maximize disruption. These *combined strategies* are the single most dangerous attack vector on PoB blockchains. They threaten both the persistence and the economic fairness of the system, underscoring the need for checkpointing and other countermeasures introduced later in this section.

Selfish Mining in our context operates as follows: when an attacker successfully mines a block, they withhold it from the network while continuing to mine on their private chain. This strategy aims to waste honest miners' computational efforts by eventually releasing the private chain and invalidating the public chain's recent blocks.

Specifically, consider the blockchain at block B. An attacker with n unexpired PoBs runs n parallel VDFs. Upon successfully mining block B_{A1} (say, using π_1), the attacker:

1. Keeps B_{A1} private rather than broadcasting it
2. Immediately begins mining on top of B_{A1} using the remaining $(n-1)$ PoBs: $\{\pi_2, \ldots, \pi_n\}$
3. Continues this process, building a private chain $B_{A1}, B_{A2}, \ldots, B_{Ak}$
4. Releases the private chain when honest miners approach length $k-1$, invalidating their work

Unlike Proof of Work blockchains where an attacker maintains constant mining power throughout the attack, our PoB system inherently degrades the attacker's capability with each mined block. After mining k blocks privately, the attacker can only launch $(n-k)$ parallel VDFs for the next block. This self-limiting property significantly weakens selfish mining attacks compared to traditional blockchains. A detailed algorithm is given in Algorithm 3.69.

Algorithm 3.69: Selfish Mining on PoB Blockchain

Require: last public block B; attacker's unexpired PoBs $\{\pi_1, \ldots, \pi_n\}$
State: private length $k \leftarrow 0$ **(number of unpublished attacker blocks)**
Convention: RunVDFsOn(X, Π) runs one VDF per $\pi \in \Pi$ on parent X

procedure First attacker win(B, π_1)
  Create private block B_{A1} on top of B using π_1
  $k \leftarrow 1$
  RunVDFsOn$(B_{A1}, \{\pi_2, \ldots, \pi_n\})$ ▷ continue privately

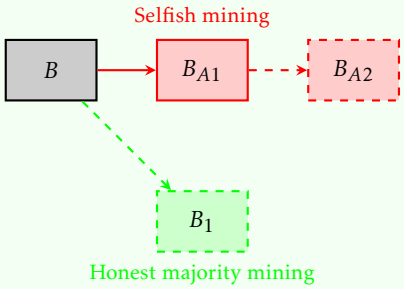
procedure Subsequent attacker win(π)
  Create private block $B_{A(k+1)}$ on top of the current private tip using π
  $k \leftarrow k + 1$
  RunVDFsOn$(B_{A(k+1)}, \{\text{remaining PoBs}\})$ ▷ keep extending

procedure Honest block appears
  if the public chain on top of B has length $k - 1$ **then**
    Publish all private blocks $\{B_{A1}, \ldots, B_{Ak}\}$ ▷ win the race by one
    return ▷ attack round ends
  else
    Continue withholding and keep mining privately ▷ lead ≥ 2

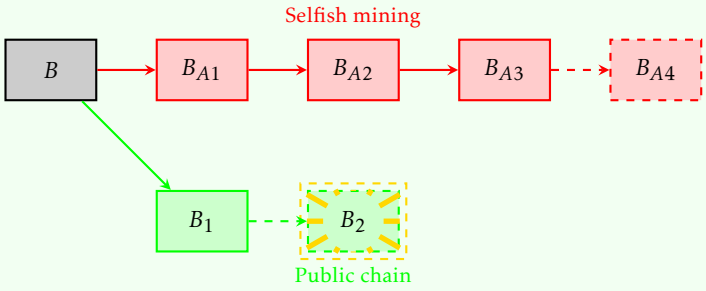
Main loop:
RunVDFsOn$(B, \{\pi_1, \ldots, \pi_n\})$
if attacker finishes first (say with π_1) **then**
  First attacker win(B, π_1)
while attack round not ended **do**
  if attacker finishes first on the private tip with some π **then**
    Subsequent attacker win(π)
  else if an honest block extending B appears **then**
    Honest block appears

Example 3.70

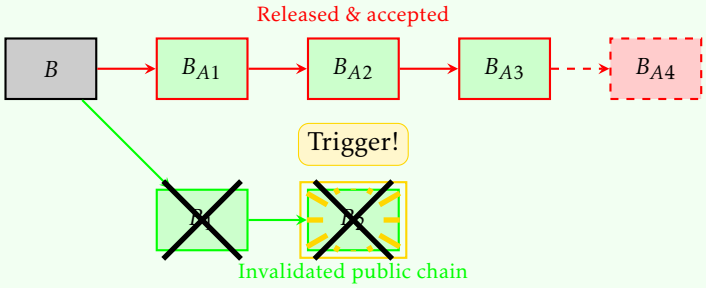
Suppose the block B is the tip of the blockchain. All miners mine atop B. The attacker, Alice, has n available PoBs, and launches n VDFs on B, one for each available PoB. Suppose Alice succeeds in mining the first block B_{A1}. Alice does not release it and continues to mine atop it. To mine B_{A2} Alice launches $n - 1$ VDFs on B_{A1}, one for each PoB available, while the honest majority is still trying to mine B_1.



Later on, Alice has a 2 block lead. Alice has secretely mined B_{A1}, B_{A2} and B_{A3}, and is working on B_{A4}, while the honest majority is catching up and has publicly mined B_1 and is trying to create B_2.



When the honest majority mines and publishes B_2, Alice does not risk the honest majority catching up to her lead which will eventually happen. Alice releases her private chain forcing the honest majority to adopt it since it is the longest (most cumulative difficulty).



The probability of a successful selfish mining attack decreases exponentially with the private chain length. For an attacker to maintain a lead of k blocks, they must win the mining race k times with progressively diminishing resources:

$$P(\text{success}) \leq \prod_{i=0}^{k-1} \frac{n-i}{N}$$

where N represents the total network mining power (total unexpired PoBs).

Stubborn Mining represents a more aggressive strategy where the attacker mines not on the tip, but on a previous block even after the honest chain progresses. The attacker hopes to catch up to the main chain and replace it, rendering all honest work worthless. This attack is specifically dangerous in a PoB blockchain because doing the stubborn mining does affect the attacker's ability to mine on the tip of the blockchain (supposing the attacker has enough CPUs available). The attacker maintains multiple parallel mining efforts:

- Continue extending their private fork
- Simultaneously mine on the public chain (to collect rewards if the attack fails)
- Release the private chain only if it becomes longer than the public chain

In PoW systems, stubborn mining requires sacrificing mining power on the main chain. However, in our PoB system, an attacker with sufficient CPUs can pursue both strategies simultaneously:

- Mining on the main chain ensures regular rewards if the attack fails
- Mining on the private fork offers potential for retroactive rewards if successful
- The only cost is the availability of computational resources (CPUs)

This creates an asymmetric risk profile heavily favoring the attacker, making stubborn mining particularly dangerous in PoB blockchains. We can see an implementation in Algorithm 3.71.

Algorithm 3.71: Stubborn Mining on PoB Blockchain

Require: last public block B
Require: attacker's unexpired PoBs $\{\pi_1, \ldots, \pi_n\}$
Convention: RunVDFsOn(X, Π) runs for each $\pi \in \Pi$ a VDF attempt on X

private length $k \leftarrow 0$
public length $L \leftarrow 0$
PrivTip $\leftarrow B$
PubTip $\leftarrow B$
released $\leftarrow$ **false**

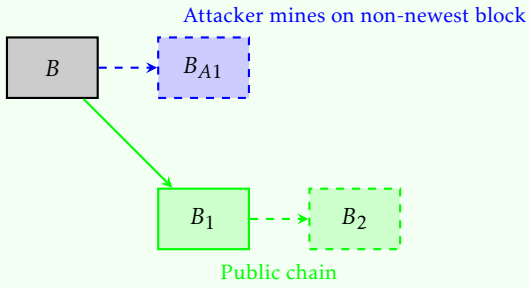
procedure Honest first win(B_1)
    $L \leftarrow 1$
    PubTip $\leftarrow B_1$
    RunVDFsOn(B_1, {remaining PoBs})

procedure Private win(π)
    if $k = 0$ **then**
        Create private block B_{A1} on top of B using π
        PrivTip $\leftarrow B_{A1}$
        $k \leftarrow 1$
    else
        Create private block $B_{A(k+1)}$ on top of PrivTip using π
        PrivTip $\leftarrow B_{A(k+1)}$
        $k \leftarrow k + 1$
    RunVDFsOn(PrivTip, {remaining PoBs})

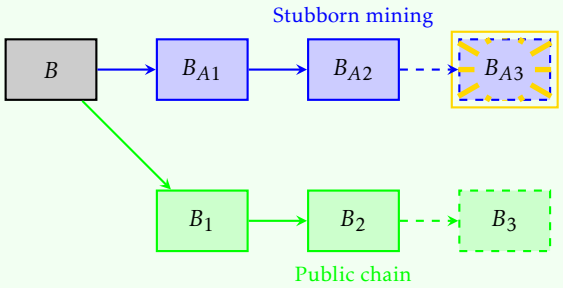
```
20:     if k > L then
21:         Publish {B_{A1}, ..., B_{Ak}}
22:         released ← true

23: procedure Public block appears(B_new)
24:     L ← L + 1
25:     PubTip ← B_new
26:     RunVDFsOn(PubTip, {Π})
27:     if k > L then
28:         Publish {B_{A1}, ..., B_{Ak}}
29:         released ← true

30: procedure Attacker on public tip wins(π)
31:     Create attacker block on PubTip using π
32:     Publish that attacker block
33:     L ← L + 1
34:     PubTip ← the published attacker block
35:     RunVDFsOn(PubTip, {remaining PoBs})
36:     return                ▷ The round ends only when the miner publishes something

37: procedure Release now
38:     if k > 0 then
39:         Publish {B_{A1}, ..., B_{Ak}}
40:     released ← true       ▷ The round ends only when the miner publishes something

41: Main loop:
42: RunVDFsOn(B, {π_1, ..., π_n})
43: if an honest block B_1 on top of B appears first then
44:     Honest first win(B_1)
45: while not released do
46:     if attacker finishes first on the private tip with some π then
47:         Private win(π)
48:     else if attacker finishes first on the public tip with some π then
49:         Attacker on public tip wins(π)
50:     else if an honest block B_new appears on the public chain then
51:         Public block appears(B_new)
52:     else if attacker decides to end the round now then
53:         Release now
```

Example 3.72

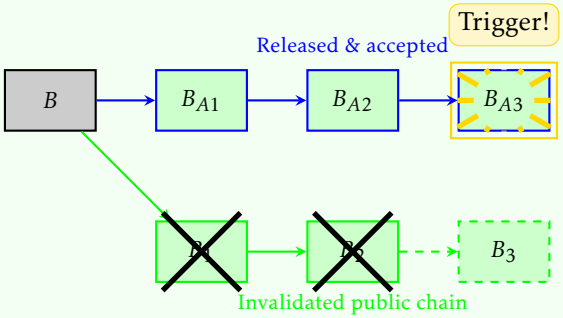
Suppose the block B is the tip of the blockchain. All miners mine atop B. Suppose the honest majority mines first on B and thus creates B_1. The honest majority starts mining on B_1 to create B_2, while the attacker, Alice, still "stubbornly" continues mining on B to create B_{A1}.



With some luck Alice catches up to the honest majority. Alice has secretely mined B_{A1} and B_{A2} and is mining B_{A3}, while the majority has B_1 and B_2 and is mining B_3.



When Alice successfully mines B_{A3} then she has the lead. She can either choose to go on with a selfish mining attack, or to release her secret chain. In this case, Alice chooses to release her chain and the honest majority is forced to accept it since it is the longest (most cumulative difficulty).



Combined Selfish and Stubborn Mining is the most sophisticated attack. It combines selfish and stubborn mining. If the attacker gets ahead, he engages in selfish mining. However he also engages in stubborn mining against himself: even if the longest chain in his view is his own, he still stubbornly continues to mine on other blocks that are not at the tip because these other subchains may grow faster in the future than the current longest one.

More generally, whatever the state of the blockchain and all its subchains, the miner will be mining on top of every single block, whether public or private, using all available PoBs, without publishing his own subchains. Whenever the honest majority catches up to the attacker's longest

chain, the attacker publishes that chain. This is sketched in Algorithm 3.73.

Algorithm 3.73: Combined Mining on PoB Blockchain

Require: last public block B
Require: attacker's unexpired PoBs $\{\pi_1, \ldots, \pi_n\}$
Convention: RunVDFsOn(X, Π) runs, for each $\pi \in \Pi$, a VDF attempt on X

$k \leftarrow 0$
$L \leftarrow 0$
PrivTip $\leftarrow B$
PubTip $\leftarrow B$
released $\leftarrow$ **false**

procedure Private win on parent X with π
    Create a private block B_A^{new} on top of X using π
    $\ell \leftarrow$ length of the private chain ending at B_A^{new}
    if $\ell > k$ **then**
        $k \leftarrow \ell$
        PrivTip $\leftarrow B_A^{\text{new}}$
    RunVDFsOn$(B_A^{\text{new}}$, {remaining PoBs}$)$
    if $L = k - 1$ **then**
        Publish the private chain $\{B_{A1}, \ldots, B_{Ak}\}$ ending at PrivTip
        released $\leftarrow$ **true**

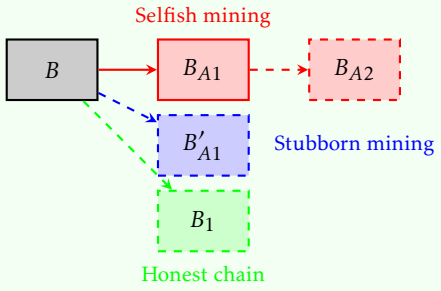
procedure Public block appears B_{new}
    $\ell_{\text{pub}} \leftarrow$ length of the public chain ending at B_{new} (measured from the round's start point)
    if $\ell_{\text{pub}} > L$ **then**
        $L \leftarrow \ell_{\text{pub}}$
        PubTip $\leftarrow B_{\text{new}}$
    RunVDFsOn$(B_A^{\text{new}}$, {remaining PoBs}$)$
    if $L = k - 1$ **then**
        Publish the private chain $\{B_{A1}, \ldots, B_{Ak}\}$ ending at PrivTip
        released $\leftarrow$ **true**

procedure Attacker wins on public tip with π
    Create an attacker block $B_{\text{pub}}^{\text{new}}$ on PubTip using π
    Publish $B_{\text{pub}}^{\text{new}}$
    $\ell_{\text{pub}} \leftarrow$ length of the public chain ending at $B_{\text{pub}}^{\text{new}}$
    if $\ell_{\text{pub}} > L$ **then**
        $L \leftarrow \ell_{\text{pub}}$
        PubTip $\leftarrow B_{\text{pub}}^{\text{new}}$
    RunVDFsOn$(B_A^{\text{new}}$, {remaining PoBs}$)$
    if $L = k - 1$ **then**
        Publish the private chain $\{B_{A1}, \ldots, B_{Ak}\}$ ending at PrivTip
        released $\leftarrow$ **true**

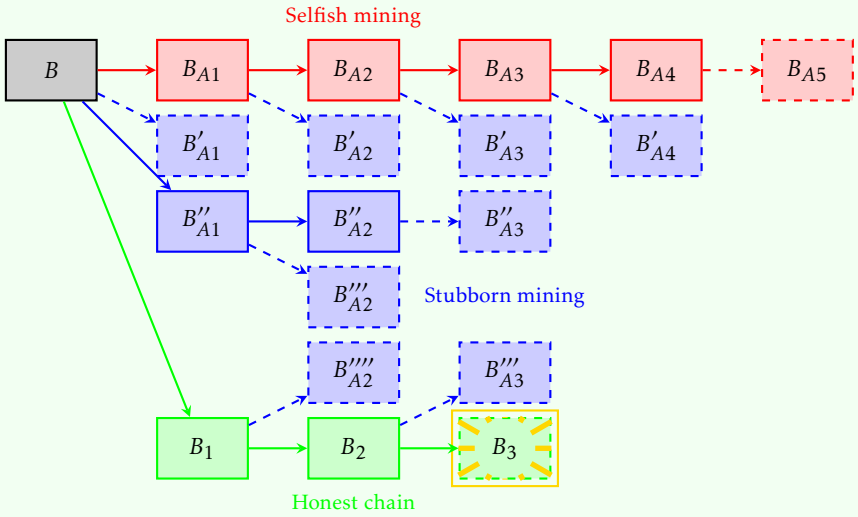
36: **Main loop:**
37: RunVDFsOn(B_A^{new}, $\{\pi_1,\ldots,\pi_n\}$)
38: **while not** released **do**
39: **if** an honest block B_{new} appears **then**
40: Public block appears(B_{new})
41: **else if** the attacker finishes first on some parent X with PoB π and X is private **then**
42: Private win on parent X with π
43: **else if** the attacker finishes first on the public tip with PoB π **then**
44: Attacker wins on public tip with π

Example 3.74

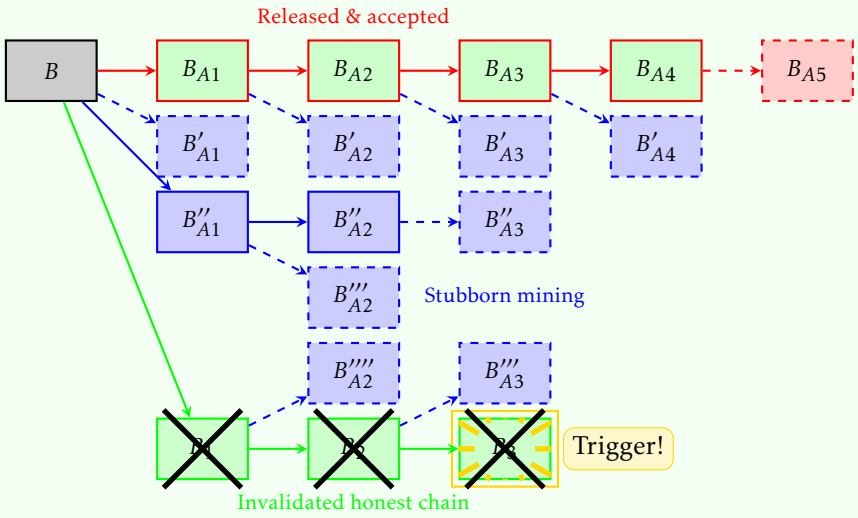
Suppose the block B is the tip of the blockchain. All miners mine atop B. The attacker, Alice, has n available PoBs. She launches n VDFs on B, one for each available PoB. Suppose Alice secretely and successfully mines the first block B_{A1}. The honest majority still tries to mine B_1 on B. Alice starts to mine B_{A2} on B_{A1} using the remaining $n-1$ available PoBs, but stubbornly does not stop the other $n-1$ original VDFs that were mining on B, she is trying to mine B'_{A1} on B.



At some point Alice has multiple subchains. The longest one, if longer than the honest chain, is technically a selfish mining while all the other chains are a result of a stubborn strategy applied against Alice's own longest chain. In this case the longest one is Alice's secret selfish mining subchain (B_{A1}, B_{A2}, B_{A3}) and Alice is trying to extend it with B_{A4}. Eventually the honest majority will catch up to Alice, as here the honest majority published B_1 and B_2 and is working on B_3.



When the honest majority mines and publishes B_3, Alice does not risk the honest majority catching up to her lead which will eventually happen. Alice releases her longest private chain forcing the honest majority to adopt it since it is the longest (most cumulative difficulty).



In summary, the unique structural properties of PoB blockchains expose them to attack vectors that are absent in the backbone model. The ability to run different VDFs (with different PoBs) in parallel on the same block undermines the fairness and uniqueness assumptions underpinning the backbone correspondence, enabling selfish, stubborn, and combined mining strategies. These adversarial behaviors threaten not only chain quality but also the economic soundness of the system. To mitigate these risks, we now turn to the checkpoint mechanism, which strengthens finality and plays a crucial role in restoring security guarantees against such attacks.

Countermeasures

There exist countermeasures for selfish mining and related attacks on regular blockchains. They mainly focus on detecting (via network analysis) and punishing selfish mining [**detect_selfish_mining**, **detect_selfish_mining_2**, **detect_selfish_mining_3**, **counter_selfish_mining**]. We focus our analysis here on special countermeasures on our PoB blockchain.

The resilience of PoB blockchains against selfish, stubborn, and combined attacks depends simultaneously on computational costs, probabilistic bounds, and checkpoint-based finality. Unlike Proof-of-Work, where computational resources can be scaled arbitrarily, in PoB systems every unexpired PoB corresponds to exactly one sequential VDF evaluation. This constraint creates both a natural upper bound on adversarial mining power and a unique structure for analyzing attack feasibility.

CPU Requirements. Suppose an attacker controls n unexpired PoBs. To maximize their chances in a combined strategy, they may attempt to mine on *all possible chain extensions* simultaneously, both public and private. Each extension requires one independent VDF per available PoB, and each VDF requires a dedicated CPU. This creates a rapid blowup in required hardware.

Consider Example 3.74. Alice has n unexpired VDFs and is mining on every possible block with as many VDFs as possible. Here are all the blocks and how many VDFs Alice is running on each of them:

- $B_{A4} : n-3$
- $B'_{A1} : n-2$
- $B'_{A2} : n-2$
- $B'_{A3} : n-3$
- $B''_{A3} : n-2$
- $B'''_{A2} : n-2$
- $B''''_{A2} : n$
- $B'''_{A3} : n$

Which makes a total of $8n-14$.

Assume the attacker controls n unexpired PoBs and wishes to mine on all available tips without pruning *any* possibility. Suppose the attacker wants to mine on k blocks of the blockchain. Call these blocks *anchors*. On each anchor, the attacker can consume any one of the n PoBs first, yielding n concurrent VDFs per anchor. After one PoB is consumed, to preserve every possibility the attacker must be prepared to consume any of the remaining $n-1$ PoBs next, and so on.

Formally, fix a depth $d \in \{1,\dots,h\}$ with $h \le n$. A frontier branch at depth d corresponds to an *ordered* d-tuple of distinct PoBs, *i.e.*, an injective sequence from a size-d index set into the n PoBs. The number of such sequences is the number of length-d permutations

$$P(n,d) \;=\; \frac{n!}{(n-d)!}.$$

Maintaining all orderings at depth d across k anchors therefore requires

$$k \cdot P(n,d) \;=\; k \cdot \frac{n!}{(n-d)!}$$

concurrent VDF evaluations (and hence CPUs). In the worst case, if the attacker insists on preserving all orderings up to depth $h = n$, the instantaneous frontier at depth n alone already demands

$$k \cdot P(n, n) \;=\; k \cdot n!$$

concurrent VDFs. Thus, under order-sensitive exhaustive branching the hardware requirement grows *factorially* in n (*i.e.*, $\Theta(k\,n!)$). Intuitively, each additional PoB multiplies the frontier by a factor that approaches the remaining inventory size, producing the factorial blow-up.

CPUs are necessary but not sufficient in PoB. Each unexpired PoB enables exactly one sequential VDF evaluation; exploiting it in parallel requires a dedicated CPU. Hence CPUs merely *realize* parallelism already latent in the attacker's PoB inventory. Unlike Proof-of-Work – where additional hardware directly increases one's mining share – in a PoB blockchain an attacker must *both* (i) control a sufficiently large pool of unexpired PoBs; and (ii) provision matching CPU capacity. Abundant CPUs without enough PoBs confer no advantage; conversely, a large PoB stock without sufficient CPUs cannot be exploited concurrently.

Checkpoint Role and Detection. The checkpoint mechanism ensures that even short-lived adversarial forks cannot grow without bound. If checkpoints are expected every ε blocks, then any attack must succeed within at most ε rounds, after which finality irreversibly fixes the ledger. Setting ε between a few dozen and a few hundred blocks provides a practical trade-off: frequent checkpoints reduce the attack window to almost nothing, while sparser ones economize on consensus overhead. In all cases, the attacker cannot threaten transactions deeper than one epoch.

Recent advances have also made selfish and stubborn mining increasingly detectable in practice [**detect_selfish_mining**, **detect_selfish_mining_2**, **detect_selfish_mining_3**, **counter_selfish_mining**]. Statistical anomaly detection reveals abnormal orphan rates and propagation delays; network-level monitors can identify withheld-block release patterns; and checkpoint-aware analysis can highlight forks that consistently collapse at epoch boundaries. While these detection methods do not prevent short-term disruption, they provide strong signals for identifying malicious participants, strengthening deterrence through accountability and potential penalties.

3.6.3 Checkpoint Security Analysis

The checkpoint mechanism introduced in Section 3.4 provides a critical layer of security for long-term blockchain operation. Beyond enabling storage optimization, checkpoints create finality guarantees that prevent long-range attacks and provide economic security through validator accountability. This section analyzes the security properties of our checkpoint system, demonstrating its resilience against both safety and liveness attacks.

Security Model and Foundations

Understanding checkpoint security requires examining the interplay between the underlying PoB consensus and the checkpoint layer. While the base blockchain provides probabilistic finality through the persistence property (Theorem 3.62), checkpoints add deterministic finality at designated points in time.

The checkpoint system operates under a Byzantine fault-tolerant model where witnesses – drawn from successful miners during each epoch – must reach consensus on the blockchain state. This creates a two-layer security architecture: the PoB layer ensures ongoing block production

and ordering, while the checkpoint layer periodically crystallizes this order into irreversible commitments.

Definition 3.75: Checkpoint Security Properties

For a checkpoint consensus instance at height h, we distinguish the following blockchain properties with the correspondance to their pBFT nomenclature:

- **Safety** (*pBFT: Agreement*): No two honest witnesses can finalize different checkpoints at the same height.
- **Validity** (*pBFT: Validity*): Any checkpoint that is finalized must contain a state root and block hash consistent with the underlying chain, as honest witnesses only prepare values they can locally verify.
- **Liveness** (*pBFT: Termination*): Every valid checkpoint proposed by an honest primary eventually finalizes, provided the network satisfies partial synchrony (*i.e.*, after some unknown global stabilization time, messages are delivered within a known bound Λ).

These properties complement each other: Safety (Agreement) prevents chain splits at checkpoint boundaries, Validity ensures that only well-formed and locally verifiable checkpoints can be finalized, and Liveness (Termination) guarantees that the protocol continues to make progress so that new checkpoints are eventually added.

Lemma 3.76: pBFT-compatibility of the checkpoint instance

Assume (i) the Byzantine bound $n \geq 3f+1$ for the epoch witness set W_i (Assumption 3.36), (ii) quorum size $q = 2f+1$ and the certificate definitions (Definition 3.37), (iii) the checkpoint structure $C_h = (\text{StateRoot}_h, \text{BlockHash}_h, \Pi_h)$ (Definition 3.32), and (iv) Algorithm 3.38 implements pBFT with the standard *locking* rule and view-change that reproposes the *highest prepared* value. Under partial synchrony (after GST, all messages arrive within Λ), the checkpoint consensus instance at height h is a standard pBFT instance (same quorum thresholds, same phase logic, same view-change invariant).

Proof of Lemma 3.76

By (i)–(ii), the replica set W_i and thresholds match pBFT's $n, f, q = 2f+1$ regime; by (iii), the decided value is a well-formed application message; by (iv), the Pre-Prepare/Prepare/Commit phases and the View-Change/NEW-VIEW rule *(repropose highest prepared)* coincide with pBFT's safety-critical invariants. Partial synchrony supplies the liveness precondition. Therefore, the checkpoint instance is behaviorally equivalent to a textbook pBFT instance, so any pBFT meta-theorem applicable under these assumptions applies here verbatim. ■

Corollary 3.77: Checkpoint Safety (Agreement)

Under the hypotheses of Lemma 3.76, no two honest witnesses finalize different checkpoints at the same height h.

Proof of Corollary 3.77

Immediate from pBFT Agreement (quorum-intersection + locking + highest-prepared) via Lemma 3.76. ■

Corollary 3.78: Checkpoint Validity

Under the hypotheses of Lemma 3.76, any finalized checkpoint $C_h = (\text{StateRoot}_h, \text{BlockHash}_h, \Pi_h)$ is valid (the components match values locally verified by honest witnesses).

Proof of Corollary 3.78

Immediate from pBFT Validity (only locally verified proposals can be prepared/committed) via Lemma 3.76. ■

Corollary 3.79: Checkpoint Liveness (Termination)

Assuming partial synchrony and at most f Byzantine witnesses with $n \geq 3f + 1$, every valid checkpoint eventually finalizes; after GST, finalization occurs within $O(f \cdot \Lambda)$.

Proof of Corollary 3.79

Immediate from pBFT Termination under partial synchrony and bounded view changes, via Lemma 3.76. ■

Economic Security Analysis

Beyond cryptographic security, the checkpoint mechanism incorporates economic deterrents that make attacks financially irrational. This economic layer provides additional security by aligning validator incentives with honest behavior.

The checkpoint system requires witnesses to maintain collateral proportional to their mining rewards. This stake serves as a bond that can be slashed for protocol violations, creating immediate economic consequences for misbehavior. The elegance of this design lies in its self-scaling nature: more successful miners have more at stake, precisely when their potential impact on the system is greatest.

Proposition 3.80: Incentive Compatibility of pBFT Participation

Under the participation incentive scheme of Definition 3.40, full honest participation in all phases of the checkpoint protocol (PRE-PREPARE, PREPARE, COMMIT) is a strictly dominant strategy for any rational witness.
Formally, let $c_{\text{participate}}$ denote the cost of participating in all phases, $r_{\text{prepare}}, r_{\text{commit}}, r_{\text{primary}}$ the corresponding rewards, ρ_w the penalty for non-participation, and Ψ_w the collateral stake of witness w. Then we have:

$$U_{\text{honest}} > U_{\text{silent}} \quad \text{and} \quad U_{\text{honest}} > U_{\text{byz}},$$

where $U_{\text{honest}}, U_{\text{silent}}, U_{\text{byz}}$ denote the utilities of honest, silent, and Byzantine strategies respectively. Hence, for every rational witness, honest participation strictly dominates

deviation.

Proof of Proposition 3.80

Setup. Fix a round of the checkpoint protocol and a rational witness w. Let $r_{\text{prepare}}, r_{\text{commit}}, r_{\text{primary}} > 0$ be the phase rewards, $\rho_w > 0$ the non-participation penalty, and Ψ_w the full-collateral at risk, as specified in Definition 3.40. Let c_{rep} (resp., c_{prim}) denote the per-round operational cost of participating as replica (resp., primary). By Definition 3.40, the penalty is calibrated so that $\rho_w \geq \max\{c_{\text{rep}}, c_{\text{prim}}\}$, and Byzantine equivocation triggers full slashing of Ψ_w. By pBFT liveness under partial synchrony and $n \geq 3f+1$ (see Corollary 3.79), a valid round completes and honest participants deterministically obtain their phase rewards.

Case A: w is the primary. The primary also acts as a replica (it must send PREPARE and COMMIT). Consider three strategies:

1. Honest participation: Sending valid PRE-PREPARE, PREPARE, and COMMIT yields:
$$U_{\text{hon}}^{\text{prim}} = r_{\text{primary}} + r_{\text{prepare}} + r_{\text{commit}} - c_{\text{prim}}.$$
2. Silence: Remaining inactive yields only the penalty:
$$U_{\text{sil}}^{\text{prim}} = -\rho_w.$$
3. Byzantine deviation: Equivocation (or any slashable deviation) loses the full collateral:
$$U_{\text{byz}}^{\text{prim}} = -\Psi_w.$$

Comparison of honest with silence. Using $\rho_w \geq c_{\text{prim}}$ and the positivity of phase rewards,

$$U_{\text{hon}}^{\text{prim}} - U_{\text{sil}}^{\text{prim}} = \left(r_{\text{primary}} + r_{\text{prepare}} + r_{\text{commit}}\right) + \rho_w - c_{\text{prim}} \geq r_{\text{primary}} + r_{\text{prepare}} + r_{\text{commit}} > 0,$$

so $U_{\text{hon}}^{\text{prim}} > U_{\text{sil}}^{\text{prim}}$.

Comparison of honest with Byzantine deviation. We have

$$U_{\text{hon}}^{\text{prim}} - U_{\text{byz}}^{\text{prim}} = \left(r_{\text{primary}} + r_{\text{prepare}} + r_{\text{commit}} - c_{\text{prim}}\right) + \Psi_w \geq \Psi_w - c_{\text{prim}} \geq \Psi_w - \rho_w.$$

Choosing the (already free) collateral multiplier $\beta > 1$ in Definition 3.40 large enough so that $\Psi_w > \rho_w$ (a standard security margin) gives $U_{\text{hon}}^{\text{prim}} > U_{\text{byz}}^{\text{prim}}$.

Case B: w is a replica (non-primary). Again consider three strategies:

1. Honest participation. Sending valid PREPARE and COMMIT yields
$$U_{\text{hon}}^{\text{rep}} = r_{\text{prepare}} + r_{\text{commit}} - c_{\text{rep}}.$$
2. Silence: Remaining inactive yields only the penalty:
$$U_{\text{sil}}^{\text{rep}} = -\rho_w.$$

3. Byzantine deviation: Equivocation (or any slashable deviation) loses the full collateral:
$$U_{\text{byz}}^{\text{rep}} = -\Psi_w.$$

Comparison honest with silence. By $\rho_w \geq c_{\text{rep}}$ and $r_{\text{prepare}}, r_{\text{commit}} > 0$,

$$U_{\text{hon}}^{\text{rep}} - U_{\text{sil}}^{\text{rep}} = \left(r_{\text{prepare}} + r_{\text{commit}}\right) + \rho_w - c_{\text{rep}} \geq r_{\text{prepare}} + r_{\text{commit}} > 0,$$

hence $U_{\text{hon}}^{\text{rep}} > U_{\text{sil}}^{\text{rep}}$.

Comparison honest with Byzantine deviation. Similarly,

$$U_{\text{hon}}^{\text{rep}} - U_{\text{byz}}^{\text{rep}} = \left(r_{\text{prepare}} + r_{\text{commit}} - c_{\text{rep}}\right) + \Psi_w \geq \Psi_w - c_{\text{rep}} \geq \Psi_w - \rho_w,$$

which is strictly positive under the same security margin $\Psi_w > \rho_w$.

In both roles (primary and replica) honest participation strictly dominates both silence and Byzantine deviation. Since honesty strictly dominates deviation in both the primary and replica roles, dominance holds for every witness regardless of whether the primary role rotates. Rotation merely ensures fairness of rewards across witnesses, but is not needed for the dominance conclusion itself. Therefore, under Definition 3.40 (with penalty calibration $\rho_w \geq \max\{c_{\text{rep}}, c_{\text{prim}}\}$ and full slashing), full honest participation is a strictly dominant strategy for any rational witness. ■

This economic security complements the cryptographic guarantees, creating multiple layers of defense against attacks. Even if an adversary could theoretically execute an attack, the economic cost makes it irrational.

Integration with PoB Consensus

The checkpoint mechanism seamlessly integrates with the underlying PoB consensus, creating a unified security model. Checkpoints do not interfere with normal block production but instead provide periodic anchors that strengthen the overall system.

Furthermore, the checkpoint system respects the behavioral incentives at the core of our blockchain. Successful miners who demonstrated valuable behaviors become checkpoint witnesses, extending their contribution to system security beyond block production. This creates a virtuous cycle where positive behaviors lead to greater system influence, which in turn requires responsible participation to maintain.

3.7 Application and Implementation

The versatility of Proof of Behavior blockchains allows them to support a wide range of applications that couple economic incentives with verifiable human actions. We distinguish two complementary aspects: (i) general application domains where PoB provides new possibilities beyond existing behavior-based tokens; and (ii) our current prototype implementation, EcoMobiCoin, which serves as a proof of concept.

3.7.1 Applications of PoB Blockchains

The generality of PoB stems from its behavioral abstraction: any action that can be cryptographically attested may serve as mining input. This flexibility enables numerous application domains:

Sustainable Mobility. The most direct application is ecological mobility, where public transportation, cycling, carpooling, or walking can be rewarded. Our flagship EcoMobiCoin exemplifies this case: ticket validation or trusted sensor data generates PoBs, which in turn drive mining rights and user rewards.

Health and Well-Being. Fitness activities (walking, running, cycling, swimming) can be validated via smartphones or wearables equipped with secure enclaves or trusted sensors. Unlike existing centralized platforms such as Sweatcoin, a PoB blockchain offers decentralized validation and currency issuance. Trusted hardware and virtualization have already been proposed to serve this role [**sensor**].

Recycling and Circular Economy. Recycling centers can issue PoBs when users return plastic, glass, or electronic waste. These proofs provide tamper-resistant evidence of ecological contribution and allow for transparent, incentive-compatible recycling ecosystems.

Renewable Energy. Producers of solar, wind, or hydro power can receive PoBs linked to certified energy generation. Unlike projects such as SolarCoin, where validation is centralized, decentralized validator networks could confirm production events, ensuring permissionlessness.

Carbon and Environmental Accounting. PoBs can represent avoided emissions (*e.g.*, commuting by bicycle instead of by car) or carbon sequestration (*e.g.*, afforestation). Integrating these behaviors directly into blockchain consensus creates new possibilities for decentralized climate finance.

The common thread across these domains is that PoB transforms verified human actions into scarce digital assets. This enables incentive structures that are simultaneously decentralized, energy-efficient, and aligned with societal goals.

3.7.2 Prototype Implementation: EcoMobiCoin

To validate the feasibility of our design, we implemented a PoB-based blockchain prototype, *EcoMobiCoin*, focused on ecological mobility. The work is available online [**EMC_LIMOS**] and the code is open source [**EMC_LIMOS_code**].

Data Source. In collaboration with the French company *MyBus* [**MyBus**], we obtained two anonymized datasets of ticket validation data over a few months. One dataset contained the data of five cities and the other contained the data of Mulhouse. Each timestamped validation was treated as a PoB. To diversify behaviors, we also incorporated our own mobility traces including walking, bicycling, and carpooling. This combined dataset allowed us to test the system with both institutional and self-collected ecological actions.

Behavior Types. The prototype supported four mobility modes: walking, bicycling, carpooling, and public transportation. Public transportation behaviors were drawn from the MyBus dataset, while walking, bicycling, and carpooling traces were gathered directly by us. To classify transportation modes from GPS traces, we developed a classifier. However, its performance proved unreliable in practice, limiting the robustness of automated mode detection.

Quantify Function. The Quantify function was defined as a linear mapping of distance to value, weighted by mobility-specific coefficients. For a behavior π corresponding to mode m and distance d_m, we set

$$\mathsf{Quantify}(\pi) = \kappa_m \cdot d_m, \tag{3.6}$$

where κ_m is a coefficient reflecting the ecological benefit of mode m. In this formulation, more sustainable modes and longer trips yielded higher mining potential. This simple structure was intended as a proof of concept, with future refinements to include dynamic coefficients or more nuanced ecological scoring.

VDF Delay Calibration. In the theoretical design, the VDF delay is defined in Definition 3.12. However, that formulation proved impractical in our prototype. We were unable to tune the difficulty parameter δ to achieve mining times in the intended range of approximately ten seconds: due to floating-point granularity, the smallest representable value of δ was still too large, which in turn caused the computed delay to collapse either to 0 or to values far exceeding the target. To remedy this, the implementation *replaced* the original rule with

$$t \;=\; \frac{\delta \cdot H(B \,\|\, \pi_{miner})}{\mathsf{Quantify}(\pi_{miner})}, \tag{3.7}$$

where H is a hash of the block header B concatenated with the miner's proof π_{miner}. Dividing by $\mathsf{Quantify}(\pi_{miner})$ makes higher-valued behaviors reduce the expected delay, while the multiplicative hash factor introduces dispersion that mitigates the discretization effects of δ. This is the delay rule actually used in the prototype implementation.

Implementation Architecture. The architecture of the prototype differs in several important respects from the specification presented in this chapter. These differences should not be interpreted as a departure from the PoB design, but rather as a reflection of the fact that the prototype was developed during the earliest stages of the project, when many of the mechanisms now presented in full detail had not yet been conceived. Most notably, the prototype did not implement the checkpointing mechanism of Section 3.4, which today is central for providing pruning and deterministic finality.

Beyond these structural aspects, other divergences concern practical design decisions. The consensus layer relied on Wesolowski's construction of verifiable delay functions [**VDF2**], chosen for its efficiency, although this made calibration of mining times particularly delicate. No explicit block size limit was defined: behaviors were capped at 100,000 per block, while transactions had no enforced ceiling, as the intended ratio-based limiting system remained disabled in code. Furthermore, compatibility constraints prevented the system from running on smartphones, since the GMP library required for the VDF was not functional on Android platforms. Finally, the prototype was heavily modeled on Ethereum's codebase which provided a practical foundation and future potential support for smartcontracts. Taken together, these differences highlight the provisional character of the prototype and motivate the challenges and insights discussed in the following paragraph. For ease of comparison with the theoretical parameters listed in Table 3.2, the main parameters of the implementation are summarized in Table 3.16.

Table 3.16: Blockchain parameters (implementation prototype).

Symbol	Value	Description
exp	24 hours	PoB expiration window
δ	variable	Mining difficulty parameter
P	10 blocks	Difficulty adjustment period
Δt	10 seconds	Desired average block time
f_r	10%	Transaction fee (fuel) rate
b	21,000	Fixed miner bounty per transaction
k	2×10^{18}	Miner's reward rate for own PoB
k'	1×10^{18}	Non-miner's reward rate for own PoB
k''	1,000	Miner's reward rate for including others' PoBs
min_{cur}	1	Minimal subdivision of currency (prototype unit)
d_{block}	0.073%	Demurrage rate per block
d_{day}	10%	Demurrage rate per day
ε	—	Epoch length, *i.e.*, blocks between checkpoints
Λ	—	Network synchrony bound, *i.e.*, message delay upper bound
T_{view}	—	pBFT view-change timeout deadline
θ	—	Checkpoint witness collateral

Challenges and Insights. The prototype surfaced several technical challenges: unstable VDF calibration, unreliable mobility classification, the absence of checkpoints, and overly permissive transaction inclusion. These issues should be understood not as flaws of the PoB design itself, but as limitations of an early exploratory implementation. Indeed, the prototype was built at a time when many of the mechanisms presented in this chapter – notably checkpoints, refined Quantify functions, and calibrated VDF delay rules – had not yet been fully developed. Nevertheless, the exercise was invaluable: it demonstrated the feasibility of grounding blockchain consensus in real-world ecological behaviors, and it provided crucial feedback that directly informed the subsequent evolution of the PoB blockchain into its present form.

3.8 Conclusion

The chapter began with two central problems in contemporary blockchain systems: (i) *excessive energy consumption*, especially in Proof-of-Work protocols; and (ii) *the centralization paradox* of behavior-incentive cryptocurrencies, where verification of human actions often relies on trusted third parties, undermining permissionlessness. Our design sought to overcome both challenges simultaneously by introducing a generic *Proof of Behavior* (PoB) consensus mechanism, coupled with *Verifiable Delay Functions* (VDFs), thereby ensuring that (i) mining rights are earned by performing verifiable behaviors rather than by consuming electricity; and (ii) behavioral validation can be incorporated into a decentralized consensus process.

Achievements. We developed a blockchain architecture that formalizes and integrates PoB into a complete system. Energy efficiency is achieved by tying mining capacity to sequential VDF executions bound to PoBs, reducing energy usage to at most one CPU per valid behavior. Permissionlessness is preserved since anyone able to produce a valid PoB may participate. Economically, the introduction of the *fuel and bounty* fee model prevents miner bias towards transaction size and mathematically rules out spam attacks. Monetary stability is achieved

through proportional issuance combined with demurrage, ensuring that supply dynamically adjusts to actual usage. Finally, we demonstrated the framework's versatility through the *EcoMobiCoin* application, rewarding eco-responsible mobility behaviors.

Security Analysis. A central contribution of this chapter is the rigorous security evaluation of PoB blockchains. By establishing a correspondence with the Bitcoin Backbone Protocol, we showed that our system inherits the main backbone theorems: *Persistence* (Theorem 3.62), *Liveness* (Theorem 3.63). These results ensure that transactions once buried sufficiently deep remain irreversible, that every valid transaction is eventually included, and that honest miners receive their fair share of blocks.

At the same time, we identified attack vectors unique to PoB systems. In particular, the independence of mining rights from block contents enables the *Concurrent Sibling Block Attack*, while the ability to run multiple VDFs in parallel per PoB enables *selfish mining*, *stubborn mining*, and their combinations. We proposed dedicated countermeasures, most notably a checkpoint mechanism based on pBFT and BLS aggregation, which provides deterministic finality. In this layer, we proved *Checkpoint Safety* (Corollary 3.77), *Checkpoint Validity* (Corollary 3.78), and *Checkpoint Liveness* (Corollary 3.79). Combined with the incentive-compatibility result of Proposition 3.80, these guarantees ensure that even in the presence of adversaries, forks cannot persist across epochs, finalized states remain valid, and the protocol continues to make progress.

Limitations and Future Work. Several challenges remain unresolved. Behavior verification continues to face the tension between decentralization, tamper-resistance, and privacy. The communication overhead and parameter calibration of the checkpoint mechanism require further study. Our prototype currently omits the checkpoint layer, and the quantification of PoBs was simplified, limiting the scope of empirical evaluation. On the adversarial side, while we proposed countermeasures to selfish and stubborn mining, their effectiveness warrants further theoretical refinement and practical testing. Future work will therefore focus on: (i) designing decentralized and privacy-preserving PoB validation mechanisms; (ii) developing post-quantum secure primitives for long-term resilience; (iii) optimizing checkpoint efficiency; and (iv) further researching detection and punishment strategies for selfish mining.

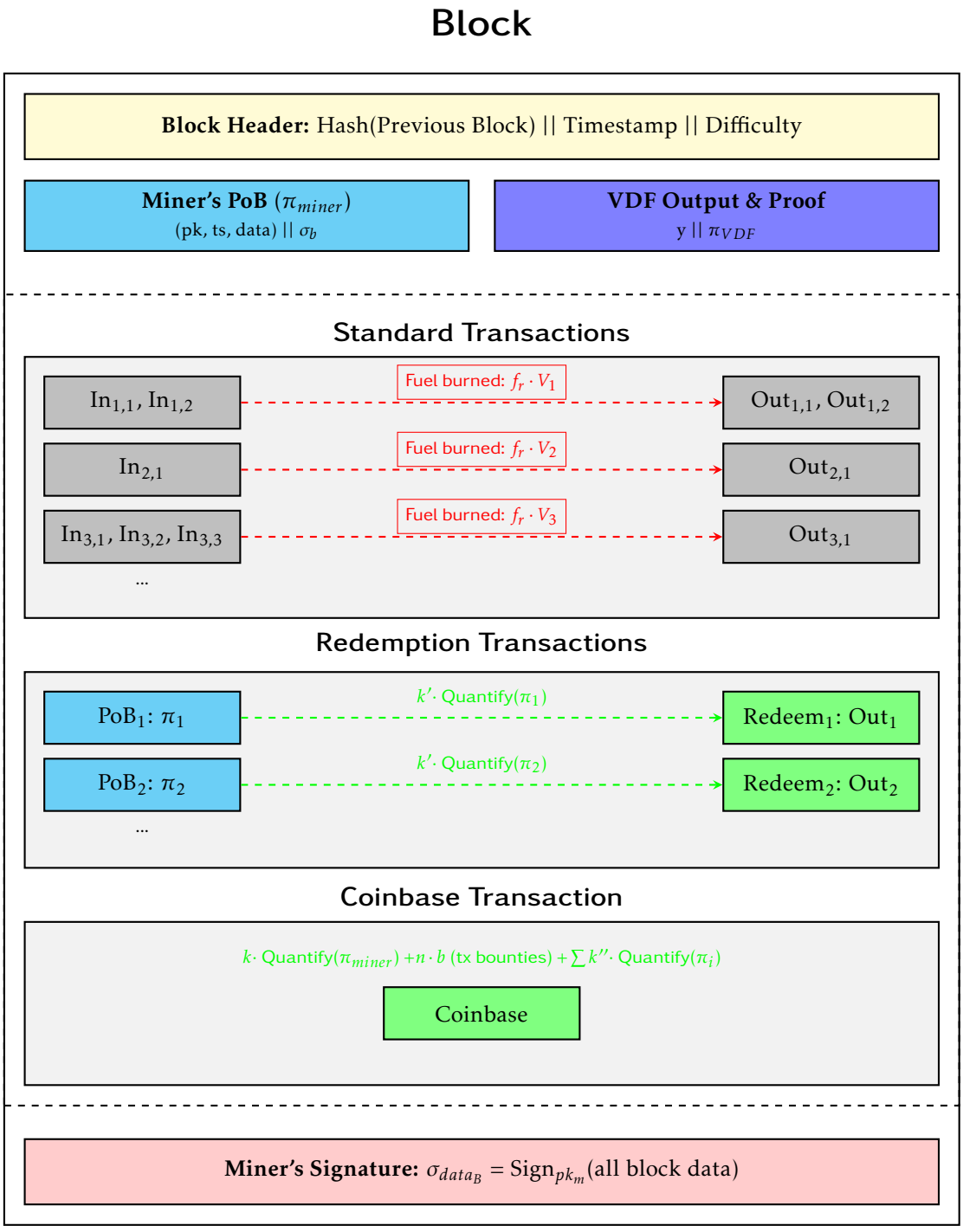


Figure 3.4: Full block structure

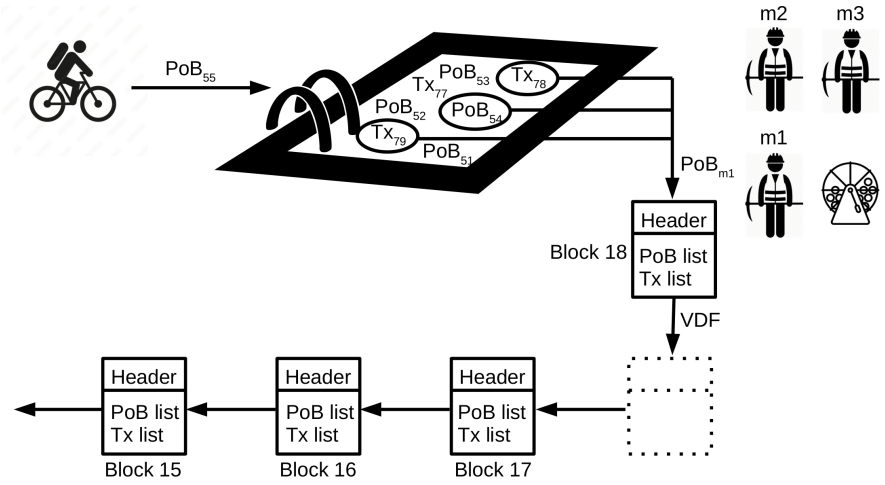


Figure 3.5: Mining workflow in the PoB blockchain. Users submit behaviors and transactions to a shared pool, while miners compute VDFs using their own PoBs, simulating a lottery process. A miner is selected for the next block.

4

General Conclusion

The final chapter of this manuscript is devoted to synthesizing the main results obtained throughout our study and situating them within the broader landscape of cryptography, distributed systems, and monetary theory. Whereas the preceding chapters each focused on specific constructions and analyses, this conclusion takes a step back to highlight the overarching contributions, methodological lessons, and conceptual insights that have emerged. In addition, it reflects on the limitations of our approaches and identifies promising directions for further research, thereby providing both closure to the manuscript and an opening toward future work.

Outline of the current chapter

4.1 Conclusion **145**
4.1.1 Local Cryptocurrencies 145
4.1.2 Proof of Behavior 146
4.2 Perspectives **147**
4.2.1 Perspectives on Local Cryptocurrencies 147
4.2.2 Perspectives on Proof of Behavior 148

4.1 Conclusion

Communication has always been the foundation of human civilization. Its history is marked by transformative milestones: the origin of speech, the invention of writing enabling transmission of ideas across time and space, the printing press revolutionizing information dissemination, the telegraph enabling rapid long-distance contact, and the internet fostering global peer-to-peer communication. Each advance has not only expanded human connectivity but also introduced new threats, necessitating the development of protective mechanisms. Cryptology, the study of secure communication in the presence of adversaries, emerged as a response. An unexpected consequence of peer-to-peer networks has been the rise of distributed ledgers, which allow mutually distrustful participants to reach consensus on a shared state of data. When this shared data concerns financial value, we obtain cryptocurrencies.

This thesis has traced the conceptual and technical precursors to blockchain systems, critically analyzed their evolution, and examined their limitations. In particular, we devoted attention to

the interplay between cryptographic protocols, incentive structures, and governance models.

4.1.1 Local Cryptocurrencies

Chapter 2 was devoted to the problem of digitalizing local currencies, a challenge that has proven persistent ever since community-driven money first began to migrate into the digital realm. Historically, local currencies sought to strengthen community ties, increase monetary velocity, and reduce economic leakage. Yet their transition into digital formats often relied on centralized architectures, which introduced multiple vulnerabilities. Centralization created single points of failure, exposed users to surveillance risks, and concentrated decision-making power in the hands of a few administrators, undermining the democratic ethos of local money. Moreover, most existing digital implementations lacked rigorous mechanisms to ensure geographical confinement or to incentivize genuinely local spending. The paradox was evident: local currencies were designed to resist monetary outflows, yet once digital, they risked being borderless and fungible across domains.

This chapter directly addressed these difficulties by proposing four blockchain-based constructions of increasing sophistication. The *Base-Local* model served as a foundational framework, emulating paper-based scrip currencies within a transparent and secure blockchain environment. It overcame the reliance on centralized servers by distributing trust among local merchants, who assumed the role of blockchain miners in a permissioned setting. The *Geo-Limited* model advanced this design by enforcing geographical restrictions through cryptographic distance-bounding protocols, with local shops acting as certificate authorities. This construction demonstrated how blockchain could achieve a property long sought by local money designers: the ability to confine transactions within a delimited space. The third construction, *Geo-Demurrage*, introduced the novel concept of geographical demurrage, whereby the value of money decays as a function of distance traveled. This represented a conceptual breakthrough: instead of absolute borders, one can design currencies whose "effective range" is defined dynamically by a distance-sensitive incentive function. Finally, the *Uni-Local* model showed how these concepts could be scaled into a universal local cryptocurrency (LCoin), in which demurrage alone preserves local spending incentives even in the absence of hard boundaries. This progression illustrated how the limitations of earlier digital local currencies–centralization, lack of enforcement of locality, and static design–could be systematically overcome through the combination of blockchain technology, cryptographic primitives, and innovative monetary theory.

The chapter's contributions can thus be seen on three levels. First, it provided a rigorous economic formalization of geographical demurrage, extending Silvio Gesell's ideas on temporal demurrage into the spatial domain. Second, it demonstrated the practical applicability of distance-bounding protocols, previously studied mostly in security contexts, to the field of monetary design. Third, it proposed a suite of adaptable blockchain frameworks capable of being tailored to diverse community needs, thereby significantly advancing the state of the art in local currency digitalization. In sum, this chapter bridged theoretical monetary economics with cryptographic innovation, producing results that both solve longstanding design problems and open new conceptual horizons.

4.1.2 Proof of Behavior

The second major contribution of this dissertation has been the introduction and detailed study of a novel consensus paradigm, the *Proof of Behavior* (PoB). The central idea was to replace energy-intensive proof-of-work computations with proofs derived from verifiable human actions, thereby creating a blockchain that is simultaneously *permissionless*, *energy-efficient*, and

behaviorally incentivizing.

This chapter began by identifying a fundamental dichotomy in existing behavior-incentive cryptocurrencies: systems that are truly decentralized often fail to verify behaviors, while systems that effectively incentivize behaviors typically rely on centralized validation. To resolve this "verification paradox," we proposed a generic PoB framework integrating Verifiable Delay Functions (VDFs) and aggregate BLS signatures, thus ensuring that behaviors themselves can become the mining input without reverting to centralized authorities.

Our contribution is multifaceted. First, we provided a formal definition of PoB, together with a Quantify function serving both as validator and impact assessor of behaviors. Second, we proposed a complete blockchain architecture, including a novel transaction system based on the *fuel and bounty* model, which ensures neutrality of fees across transaction sizes, and the introduction of temporal demurrage, balancing money creation by systematically discouraging hoarding. Third, we incorporated a checkpoint mechanism leveraging pBFT and BLS aggregation, which achieves storage pruning, scalability, and finality. Finally, we validated the feasibility of our design through a prototype, EcoMobiCoin, applied to ecological mobility incentives, demonstrating how public transportation, cycling, walking, and carpooling can be integrated into a consensus protocol.

The chapter also contained a comprehensive security analysis. We showed the correspondence of PoB with the Bitcoin backbone protocol, guaranteeing persistence and liveness under honest-majority assumptions. At the same time, we highlighted new attack vectors specific to PoB, such as the concurrent sibling block attack or selfish and stubborn mining enabled by parallel VDF execution. Countermeasures were proposed, especially through the checkpoint layer, thereby reinforcing both technical and economic security.

Altogether, the results achieved in this chapter demonstrate that PoB can reconcile decentralization with behavioral incentivization, laying down the first generic framework for human-action-based consensus protocols. This constitutes a substantive step towards sustainable, socially aligned, and technically sound blockchains.

4.2 Perspectives

The perspectives outlined below illustrate that the future of blockchain technology is neither purely technical nor purely economic, but a synthesis of the two. Blockchains are *technological artifacts* that simultaneously serve as *socio-economic instruments*. By extending our investigation into local economies, incentive design, and novel consensus protocols, this thesis has laid groundwork for reimagining blockchains as tools that are not only secure and efficient, but also aligned with the broader objectives of human society. The challenge ahead is to translate these insights into systems that can be deployed in practice, evaluated in real-world conditions, and refined through interdisciplinary collaboration. Only then will we be able to determine whether blockchains can truly evolve from financial innovations into instruments for sustainable and equitable social organization.

4.2.1 Perspectives on Local Cryptocurrencies

Although the blockchain-based frameworks proposed for local currencies address longstanding challenges, several avenues remain open for future exploration. These directions can be grouped into four main lines of research.

Privacy-Preserving Locality Proofs. One of the foremost challenges is the tension between enforcing locality and protecting user privacy. Current distance-bounding and certificate-based location proofs inevitably reveal aspects of a user's position, which could raise surveillance concerns in real-world deployments. Future work should investigate privacy-preserving primitives–such as zero-knowledge proofs, anonymous credentials, or homomorphic encryption–to reconcile geographical enforcement with anonymity. Achieving robust guarantees of both proximity and privacy would constitute a major step toward secure and socially acceptable deployments.

Calibration of Geographical Demurrage. The economic consequences of different demurrage functions remain largely unexplored. While the chapter introduced candidate models (linear, quadratic, exponential, stepwise), their effects on monetary velocity, adoption, and community cohesion require systematic study. Adaptive mechanisms, possibly informed by machine learning or reinforcement learning, could dynamically adjust demurrage rates in response to evolving economic conditions. The key challenge is to balance flexibility with simplicity so as not to discourage user participation.

Governance and Redistribution Mechanisms. Demurrage proceeds present open questions of allocation. Should the collected value be burned, redistributed to local merchants (acting as miners), or channeled toward community projects? Each option has implications for redistribution, trust, and sustainability. Research is needed on governance models that distribute demurrage benefits in a transparent and legitimate way. Hybrid approaches, possibly inspired by decentralized autonomous organizations (DAOs), may strike a balance between efficiency and community legitimacy.

Scalability and Socio-Economic Impact. The Uni-Local model demonstrates the possibility of universal local currencies, yet such systems may raise regulatory challenges and compete with sovereign money. A critical question is whether such models naturally cluster into regional subnetworks, forming sustainable ecosystems rather than global competitors. Empirical validation through pilot programs will be indispensable for measuring real impacts on employment, velocity of money, and social cohesion. Only through such deployments can theoretical claims about local cryptocurrencies be rigorously tested.

In summary, the perspectives on local cryptocurrencies extend across technical, economic, and sociopolitical dimensions. Addressing the challenges of privacy, demurrage calibration, governance, and scalability will require interdisciplinary efforts, but the potential payoff is considerable: a new generation of monetary systems that are both technologically resilient and socially transformative.

4.2.2 Perspectives on Proof of Behavior

Although the PoB paradigm represents a significant conceptual and technical advance, several open problems and promising research avenues remain. They can be grouped into four main directions.

Security and Adversarial Modeling. The security analysis has shown that Proof of Behavior blockchains introduce novel attack surfaces compared to traditional consensus systems. In particular, selfish and stubborn mining strategies pose a concrete threat in the PoB context, where parallel VDF executions can be exploited to create multiple candidate blocks in advance. Future work should therefore focus on designing mechanisms to detect such adversarial behavior

and enforce appropriate countermeasures. This includes the development of slashing protocols, adaptive difficulty adjustments, and reputation-based penalties that discourage miners from deviating from honest strategies. A deeper game-theoretic modeling of PoB adversarial incentives will be necessary to ensure robustness under strategic manipulation.

Economic Design and Cross-Domain Applications. The integration of temporal demurrage and the fuel-and-bounty system opens several unresolved economic questions. For instance, how do these mechanisms impact monetary stability in the long run? How should governance of parameters (demurrage rate, difficulty adjustment, reward coefficients) be organized in a decentralized yet efficient manner? Game-theoretic modeling and empirical simulations are needed to study incentive compatibility at scale. Furthermore, while EcoMobiCoin exemplifies the mobility sector, the PoB framework is inherently generic: recycling, renewable energy production, carbon sequestration, and health-related behaviors are natural domains for extension. A long-term perspective is to design a cross-domain PoB blockchain, capable of integrating heterogeneous human behaviors into a single incentive-compatible ledger.

Implementation and Empirical Validation. A crucial direction is the practical implementation and experimental deployment of PoB blockchains. While EcoMobiCoin provided a proof-of-concept prototype, future work should aim at building full-scale testbeds to evaluate performance, usability, and resilience under real-world conditions. Pilot programs in domains such as urban mobility, waste management, or energy conservation could provide valuable empirical data on incentive effectiveness, security overhead, and user adoption. Such experiments would not only validate the theoretical assumptions underlying PoB but also guide refinements in protocol design, economic calibration, and governance mechanisms.

Post-Quantum and Long-Term Robustness. Finally, many cryptographic primitives used in our design, notably VDFs and BLS signatures, are not post-quantum secure. Investigating post-quantum alternatives, and analyzing their integration with PoB consensus, represents an important direction to ensure long-term viability.

In summary, future research on PoB should simultaneously strengthen its cryptographic foundations, refine its behavioral verification models, expand its economic toolkit, and test its applicability to broader societal contexts. Addressing these questions would not only consolidate PoB as a viable consensus mechanism but also advance the broader vision of *blockchain for good*.

Contents

Abstract **xxiii**

Acknowledgments **xxv**

Contents at a Glance **xxvii**

Contents **xxix**

List of Tables **xxxi**

List of Figures **xxxiii**

A Game of Trust **1**
In the beginning was the Word . . . 1
Put not your trust in princes . . . 3
David and Goliath . . . 4
Test all things; hold fast what is good . . . 5

1 General Introduction **9**
1.1 Introduction . . . 9
1.2 Cryptography . . . 10
1.2.1 Public Key Cryptography . . . 10
1.2.2 Hash Functions . . . 12
1.3 Money . . . 13
1.3.1 Money . . . 13
1.3.2 Demurrage . . . 16
1.3.3 Digital Currency . . . 17
1.4 Consensus . . . 19
1.4.1 Consensus . . . 19
1.4.2 Practical Byzantine Fault Tolerance (pBFT) . . . 20
1.5 Blockchains . . . 21
1.5.1 (De)centralization . . . 23
1.5.2 Governance . . . 25
1.6 Manuscript Outline and Contributions . . . 27

2 Local Cryptocurrency **29**
2.1 Introduction . . . 29
2.1.1 Challenges in Digitalization of Local Currencies . . . 31
2.1.2 Contributions . . . 31
2.1.3 Related Work . . . 33
2.1.4 Outline . . . 35
2.2 Economics . . . 36
2.2.1 Demurrage . . . 36
2.2.2 Foundations of Local Currencies . . . 39
2.2.3 Geographical Demurrage: A Spatial Approach to Local Currencies . . . 41
2.3 Distance Bounding . . . 45
2.4 Local Cryptocurrency . . . 47
2.4.1 Base-Local . . . 49
2.4.2 Geo-Limited . . . 50
2.4.3 Geo-Demurrage . . . 51
2.4.4 Uni-Local . . . 53
2.5 Conclusion . . . 54

3 Generic Blockchain on Generic Human Behavior **57**
3.1 Introduction . . . 57
3.1.1 Contributions . . . 59
3.1.2 Related Work . . . 60
3.1.3 Outline . . . 62
3.2 Foundational Concepts . . . 62
3.2.1 Verifiable Delay Functions . . . 62
3.2.2 BLS Signature Scheme . . . 63
3.2.3 Proof of Behavior . . . 65
3.3 System Architecture . . . 70
3.3.1 Block Structure . . . 70
3.3.2 Mining and Consensus Mechanism . . . 72
3.3.3 Transaction System . . . 75
3.3.4 Economic Mechanism . . . 85
3.4 Checkpoint Mechanism . . . 91
3.4.1 Motivation and Design Principles . . . 92
3.4.2 pBFT Consensus Protocol . . . 97
3.4.3 Data Pruning and Storage Optimization . . . 104
3.4.4 Economic Incentives . . . 106
3.5 System Analysis . . . 107
3.5.1 PoB Expiration . . . 107
3.5.2 Energy Consumption . . . 108
3.5.3 Monetary Policy Analysis . . . 110
3.6 Security Analysis . . . 115
3.6.1 Bitcoin Backbone Protocol Correspondance . . . 115
3.6.2 Special Attacks on PoB Blockchains . . . 119
3.6.3 Checkpoint Security Analysis . . . 134
3.7 Application and Implementation . . . 138
3.7.1 Applications of PoB Blockchains . . . 139
3.7.2 Prototype Implementation: EcoMobiCoin . . . 139
3.8 Conclusion . . . 141

4 General Conclusion **145**
4.1 Conclusion 145
4.1.1 Local Cryptocurrencies 145
4.1.2 Proof of Behavior 146
4.2 Perspectives 147
4.2.1 Perspectives on Local Cryptocurrencies 147
4.2.2 Perspectives on Proof of Behavior 148

Contents **151**

Security Analysis of Distributed Ledgers

Abstract

Cryptography emerged from a fundamental recognition: secure communication cannot rely on trusted intermediaries or centralized authorities. This skepticism toward concentrated power finds parallel expression in the Austrian school of economics, particularly Friedrich Hayek's critique of government monetary monopolies and advocacy for competing currencies. At the intersection of cryptographic distrust and economic liberalism, Bitcoin introduced blockchain technology as a foundation for competing digital currencies secured by cryptography rather than central authorities, enabling decentralized consensus where trust emerges paradoxically from mutual distrust among network participants.

This thesis advances distributed ledger theory through two contributions. First, we address the digitalization of local currencies through four blockchain-based constructions: *Base-Local* establishes a foundational framework; *Geo-Limited* enforces geographical constraints via distance-bounding protocols; *Geo-Demurrage* introduces geographical demurrage where currency value decreases with distance; and *Uni-Local* demonstrates universal scalability while maintaining local spending incentives.

Second, we introduce *Proof of Behavior* (PoB), a consensus mechanism replacing energy-intensive computations with verifiable human actions. We resolve the dichotomy between decentralization and behavioral verification through a framework integrating Verifiable Delay Functions for sequential mining. The architecture includes a fuel-and-bounty transaction system ensuring size-neutral fees, temporal demurrage balancing money creation, and a checkpoint mechanism leveraging pBFT consensus with BLS signature aggregation to provide deterministic finality and enable storage pruning. We establish security guarantees through correspondence with the Bitcoin Backbone Protocol while identifying and mitigating PoB-specific attacks including selfish mining and concurrent sibling blocks. The prototype EcoMobiCoin validates feasibility by rewarding ecological mobility.

These contributions demonstrate that blockchain technology can serve socially aligned objectives while maintaining rigorous security and decentralization principles essential to trustless systems.

Keywords: distributed ledgers, blockchains, consensus, cryptocurrency

---

Analyse de Sécurité des Registres Distribués

Résumé

La cryptographie est née d'une constatation fondamentale : la sécurité des communications ne peut reposer sur des intermédiaires de confiance ou des autorités centralisées. Ce scepticisme à l'égard du pouvoir concentré trouve un écho dans l'école autrichienne d'économie, en particulier dans la critique de Friedrich Hayek à l'égard des monopoles monétaires gouvernementaux et son plaidoyer en faveur de la concurrence entre les monnaies. À la croisée de la méfiance cryptographique et du libéralisme économique, Bitcoin a introduit la blockchain comme fondement de monnaies numériques concurrentes sécurisées par la cryptographie plutôt que par des autorités centrales. La blockchain a permis un consensus où la confiance émerge paradoxalement de la méfiance mutuelle entre les participants au réseau.

Cette thèse fait progresser les registres distribués grâce à deux contributions. Premièrement, nous abordons la digitalisation des monnaies locales à travers quatre constructions basées sur la blockchain : *Base-Local* établit un cadre fondamental ; *Geo-Limited* impose des contraintes géographiques via des protocoles de limitation de distance ; *Geo-Demurrage* introduit une surestarie géographique ; et *Uni-Local* démontre une évolutivité universelle tout en maintenant les incitations à dépenser localement.

Deuxièmement, nous introduisons la preuve de comportement (*Proof of Behavior*, PoB), un mécanisme de consensus qui remplace les calculs énergivores par des actions humaines vérifiables. Nous résolvons la dichotomie entre décentralisation et vérification comportementale en intégrant des fonctions de délai vérifiables pour la preuve de travail séquentielle. L'architecture comprend un mécanisme de checkpoint tirant parti du consensus pBFT avec des agrégations de signatures. Nous établissons des garanties de sécurité grâce à la correspondance avec le protocole Bitcoin Backbone tout en identifiant et en atténuant les attaques spécifiques à PoB, notamment le minage égoïste et les blocs frères concurrents. Le prototype EcoMobiCoin valide la faisabilité en récompensant la mobilité écologique.

Ces contributions démontrent que la technologie blockchain peut servir des objectifs sociaux tout en maintenant les principes rigoureux de sécurité et de décentralisation essentiels aux systèmes sans confiance.

Mots clés : registres distribués, blockchains, consensus, cryptomonnaies

---

Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes