Significance of Data Structures and Algorithms in Financial Technology

Ashmitha Nagraj

Software Engineer
nagrajashmitha@gmail.com

Abstract:
Financial Technology (FinTech) has transformed Capital Markets, Payments, Lending and Risk Management through faster processing of financial decisions in real-time and expanding access to financial services digitally. Beneath these user-friendly applications, FinTech platforms rely on foundational Data Structures and Algorithms to provide consistent Latency, Scalable Throughput, Auditing capabilities, and Resilience under adversarial conditions. The purpose of this paper is to review which core Data Structures (Arrays, Linked Lists, Hash Tables, Balanced Trees, Heaps and Graphs) are used to support key FinTech Workload applications (Algorithmic Trading, Fraud Detection, Credit Risk Assessment and Blockchain-based Recordkeeping). In addition, this paper will review the Algorithmic Foundations used to enable common tasks across all these workload applications including Sorting/Searching, Optimization, Statistical Learning and Cryptography, and how Asymptotic Complexity must be evaluated with respect to practical system constraints including Caching Behavior, Concurrency and Failure Modes. There is evidence from the academic literature that Algorithmic Trading can increase liquidity in certain Market Structures while also introducing Systemic Fragility during Stress [1],[2],[3]. And, similarly, there is evidence that Fraud Detection is an inherently adversarial domain where Models and Features must evolve as Attacker Behavior evolves [4],[5]. Lastly, the Paper will discuss several open challenges associated with Scale, Security, Model Governance and Privacy; and evaluate Future-Facing Directions such as Privacy-Preserving Analytics (Federated Learning and Zero-Knowledge Proofs) and Cryptographic Agility to prepare for post-Quantum risk.

Keywords: FinTech, Data Structures and Algorithms, Algorithmic Trading, Fraud Detection, Credit Risk Assessment, Blockchain Systems, Time and Space Complexity, Scalable Systems, Low-Latency Computing, Adversarial Machine Learning, Cryptographic Systems, Model Governance, Privacy-Preserving Analytics, Federated Learning, Post-Quantum Cryptography.

INTRODUCTION

The financial sector has experienced a fundamental shift due to digitization, connectivity, and software defined systems that have enabled the processing of trades, payments, lending, and customer interactions via computerized systems. Competitive advantages in modern FinTech often boil down to the capacity to process vast amounts of different data streams (alternative data, transactional data, identity signal data, etc.) with low and predictable latency, while maintaining accuracy, security and an auditable record of decision making.
A computational view of financial workloads highlights two properties that may be conflicting. The first property relates to latency and throughput; microsecond to millisecond latency can have significant impacts on the quality of execution of orders, and longer tail latencies can be a source of operational risk. The second property relates to control guarantees: access control, data integrity, and evidence gathering to support compliance, incident response and post event reconstruction. The representation of data via a chosen data structure, how it is indexed, updated and shared among threads or nodes directly impact both the theoretical and actual cost of operation in scenarios involving concurrency and bursty traffic [6], [7].

Statistical learning is increasingly being used within FinTech applications to provide classification, ranking and anomaly detection capabilities. Statistical learning improves predictive capability of algorithms but introduces new engineering requirements; including computations of features at scale, monitoring for drift in non-stationarity and providing governance for algorithms that affect customers and the integrity of markets

[4], [8]. Security through cryptography is also critical to enabling secure digital finance; cryptographic primitives such as key exchange, digital signatures, hashing and tamper evident logs [9], [10] must be properly integrated into robust systems that manage keys, provide operational access and facilitate system failures.
This paper will maintain the same structure as the original document and map fundamental data structures and algorithms to their respective major areas of application in the FinTech industry, explain the reasons for the most common choices made by developers and engineers and analyze the problems and opportunities for regulated high-performance financial systems moving forward.

ROLE OF DATA STRUCTURES IN FINTECH

The way the organization structures the data in modeling for financial services or platform engineering determines how efficiently a system stores, retrieves, and updates information. Since FinTech platforms use OLTP to process transactions, streams to process real-time events, and analytics to analyze performance, all three workload types are often used in the same product, which further emphasizes the need for structures that will minimize the constant factor and algorithmic complexity when using realistic constraints such as cache locality, concurrency control, and network overhead.

Data Structure	Usage in FinTech	Efficiency (Typical Time Complexity)
Arrays	Market data snapshots, historical time-series windows, feature vectors	O(1) access; O(n) search (unsorted)
Linked Lists	Append-only in-memory buffers, queue-like workflows	O(1) insert/delete with pointer; O(n) search
Hash Tables	Idempotency keys, session/auth caches, feature lookups	Average O(1) lookup/insert; worst-case O(n)
Graphs	Entity-link analysis (AML/fraud), dependency modeling	Traversal O(V+E)
Heaps (Priority Queues)	Top-of-book selection, scheduling, top-K alerts	O(log n) insert/delete; O(1) peek
B-Trees / B+-Trees	Database indexing for ledgers, orders, audit logs	O(log n) search/insert/delete

Table 1: Common data structures and their role

Arrays and linked lists:

Arrays provide fast access to elements using indexes and compact storage schemes to support rapid window-based operations on large time series data and rapid vectorized computations of features on large datasets. For example, in a typical financial pipeline, arrays of prices, returns, and volume can be used to create and update common technical indicators such as moving averages and realized volatility. Additionally, arrays allow users to perform many types of batch computations that take advantage of the CPU cache to reduce the actual real-world latency, which may differ from theoretical latency suggested by Big-O notation [6]. Linked lists have lower cache locality than arrays and could potentially be used when the workflow primarily consists of insertions/deletions with no need for random access. Example: some in-memory buffers, event queues, or append-heavy data structures in low level implementations.

Hash tables:

Hash tables enable fast access to values based on keys and are therefore an essential part of idempotent and secure authentication, fraud detection and enrichment services and cache services. Payment APIs often include idempotency keys to ensure that the same request does not get processed multiple times if a client retries; A hash table can rapidly remove duplicates from this expensive business logic prior to its execution. Additionally, hash-based caches in fraud and risk systems quickly join streaming events with recent behavioral information. Example: device fingerprints and velocity counters. The theoretical foundations of

hash tables provide methods to minimize the chance of collisions and increase the robustness against adversarial input [11]

Graphs:
When compared to data about individuals, it may be beneficial to represent relationships of an organization through graphs. AML and Fraud investigations typically involve transactional data; therefore, this data forms a network of all these entities i.e. account holders, device users, merchant providers, beneficiary recipients, and IP addresses. The use of graph-based analytics allows for the detection of possible suspicious activity in clusters of data that would have been difficult to find in record by record-based analysis. Graph traversals and component discovery allow investigators to identify possible relationships or connections and generate graph-based features that will aid in improving model performance [5]. Maintaining efficient connectivity for large-scale modeling of entity linkage is related to classical structures, such as union-find and graph traversal, which continue to be useful when modeling linkage at scale [6].

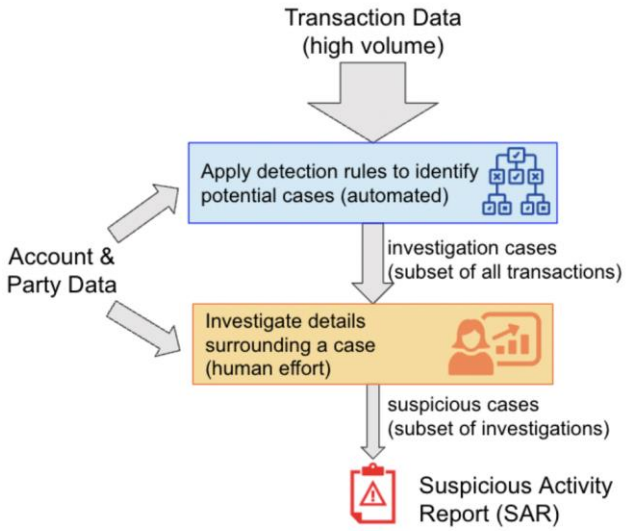

Figure 1: *The AML investigation process*

Heaps and priority queues:
Heaps provide a way for low-latency applications with repeated "best" accesses to efficiently handle priority queue operations. Trading systems can rapidly access the best bid/ask using an ordered data structure and can quickly select the most important priorities for immediate action given potential risks. For example, which order to cancel first when a limit is reached. Priority queues are also used in Risk Monitoring applications to provide the Top K alerts, in these applications, the computational time must be bounded while exposing the highest risk exposure [6].

B-trees and indexing:
The use of persistent data is a fundamental component in all regulated financial systems. Financial ledgers, statements, and audit logs must be both resilient and capable of being queried. B-tree and B+ tree indexes have been implemented into numerous financial database systems as well as ledger implementations due to their ability to maintain a balanced index and minimize disk I/O, as they typically align each node size to each page and block. The analysis of the B-tree provided upper bound estimates of performance on searches and updates; this is one reason that B-tree variants continue to be at the center of financial database and ledger system implementations [12].

Probabilistic and approximate structures:
FinTech systems use probabilistic structures in order to save on both memory and computation, as well as to achieve a better latency budget by employing structures such as Bloom Filters that allow for very efficient membership testing with the potential for false positive matches (i.e., "false alarms") and zero false negative matches (i.e., "misses"), providing a quick pre-filter for de-duplication, caching, and screening known bad or

known good indicators prior to processing much more time consuming scoring logic [13]. This is also a common design pattern where an application has a latency budget and would like to be able to quickly reject obviously either safe or unsafe cases.

SIGNIFICANCE OF ALGORITHMS IN FINTECH

Algorithms orchestrate how financial platforms transform data into decisions. Beyond raw performance, algorithmic choices influence correctness (e.g., consistency of ledgers), risk sensitivity (e.g., tail exposure), and governance (e.g., explainability and reproducibility). The following algorithm families are especially relevant:

Algorithm	Application in FinTech
Sorting and searching	Reconciliation, audit reconstruction, transaction retrieval
Machine learning / statistical learning	Fraud detection, credit scoring, risk classification
Optimization	Portfolio construction, execution scheduling, liquidity/collateral allocation
Cryptographic algorithms	Secure transactions, authentication, integrity, blockchain security

Table 2: Common algorithms and their application in FinTech structures and their role

Sorting and searching:

Sorting and searching form the basis of reconciliation, reporting, and analytics. Sorting allows for ordered events to be reconstructed over time, as well as stable aggregated results. Indexed searches enable rapid access to transactional data and provide auditable evidence for resolving disputes. In a distributed system, the performance limitations and design principles of traditional sorting/searching will also determine the construction of the pipeline for data processing, and the selection of storage [7], [6]. For example, reconciliation pipelines typically sort on timestamp, identifier, or ledger sequence number, followed by reconciling via key-based joins operations where the cost is primarily due to the primitive search and ordering capabilities underlying those operations.

Machine learning and statistical learning:

Predictive models use classification to identify potential frauds, assess a customer's likelihood of defaulting on a loan, and help detect unusual activity patterns (anomalies) by identifying potential frauds using predictive models, estimating a customer’s probability of defaulting on a loan, and detecting anomalies. While logistic regression is often utilized due to its ability to provide probabilities and allow for an explanation of the decision-making process that is necessary in many regulated environments, tree based-methods and SVMs are popular in the fraud-detection arena due to their effectiveness in this area with feature sets that capture the contextual and temporal aspects of a transaction [14]. However, fraud detection is an adversarial and non-stationary problem; therefore, fraud detection systems should continually monitor for "drift" in model performance and retrain the system as the attackers’ strategies evolve [4].

Optimization:

Optimization is very important to the field of finance since many goals have the nature of being an optimization problem with constraints. The mean-variance portfolio theory sets out the relationship between expected returns and risk and forms the basis for much of the work on portfolio design and diversification [15]. Optimization can be seen in operational systems when it comes to how to schedule executions (to minimize impact and slippage); managing liquidity; allocating collateral. The use of risk measures matters. Optimizing the conditional value-at-risk (CVaR) gives a tractable method to deal with tail risk and will give a better alignment of extreme loss concerns than variance in most cases [16].

Cryptography:
Cryptographic methods are essential to financial systems to ensure authentication, confidentiality, integrity, and non-repudiation. Public-Key Cryptography supports Key Exchange and Digital Signatures in financial systems; therefore public-key cryptography can be utilized to enable secure communications through digitally signed message and signed instruction of transactions [9],[10]. Hash functions and Merkle tree constructs also provide the foundation for tamper evident logs which are used as part of a broader framework that provides integrity to audit trails, logs which is an important component of blockchain technology. The use of linkable cryptographic methods in conjunction with consensus mechanisms within distributed ledgers collectively provide the basis for appending new data to a ledger while maintaining the integrity of the data in the ledger [17]. However, at the end of the day, regardless of how robust the cryptographic mechanisms may be the system will fail if the software is implemented incorrectly or the keys are poorly managed.

APPLICATIONS IN FINANCIAL TECHNOLOGY

Algorithmic Trading:

Algorithmic Trading encompasses Low-Frequency Order Execution Algorithms as well as High-Frequency Trading (HFT). In HFT, the trader's strategy responds to market micro-structure-based signals in a short horizon; therefore, these systems must consume and process high rate, normalized market data while deciding under very tight latency constraints. Therefore, in this environment, the system data structure must be able to predictably perform accesses to its data, and the algorithm should have bounded per event processing.
The academic community has investigated if algorithmic trading improves market quality. The studies in algorithmic trading and liquidity show that higher frequency algorithmic trading activity can lead to tighter spreads and improved liquidity under certain conditions [1]. Also, low-latency trading studies have shown that speed affects the timing of order placements and cancellations and thus the amount of liquidity and short-term price volatility measured [2]. Systemic events are an example of the risks associated with algorithmic trading. For example, Flash Crash literature shows how interactions among automated trading strategies can increase instability during periods of reduced liquidity and highlight the need for strong controls and monitoring [3].
In terms of implementation, the order handling requirements of trading systems require ordered access to price levels and fast determination of the "best" executable prices. Ordered indexes/priority queues enable top of book operations and arrays enable rolling computations of features; hash tables enable caching of instrument metadata, venue status and pre-trade risk limits. Execution algorithms typically solve constrained optimization problems and choose schedules that balance market impact, deadline completions and risk control limits. The choice matters as much or more than profit as failures can result in regulatory breach, operational incident and market-wide disruption and therefore, it is important to design well-functioning algorithmic trading systems to exhibit deterministic behavior, bounded resource usage and rapid fail-safe mechanisms.

Key Benefits of Algorithmic Trading

- Improved execution consistency and reduced manual error
- Lower transaction cost through automation and smart routing
- Systematic enforcement of pre-trade limits and compliance controls

Challenges of Algorithmic Trading

- Model risk and unexpected market interactions
- Operational failure modes requiring kill switches and surveillance
- Regulatory scrutiny around fairness, stability, and manipulation risk

Fraud Detection and Risk Assessment

Fraud detection and credit risk assessment have been major Fintech applications where algorithms convert raw behavioral signals into actionable decisions. Fraud is adversarial; this means fraudsters are constantly adapting in ways that produce changing or non-stationary data distribution that require more robust analytic techniques. Literature on the topic emphasizes that fraud detection requires a combination of modeling and real-time monitoring since fraudulent patterns evolve continuously [4]. Research studies also identify that reviews of fraud detection emphasize the need to categorize techniques, as well as indicate that evaluation of

such techniques should include consideration of class imbalance and the evolving nature of fraudster behavior [5].

Figure 2: Fraud Detection and Risk Assessment

The data structures used by the model enable it to operate at scale and with sufficient latency to make operational decisions in near real-time. The computational requirements of features are typically met through the need for rapid joining of streaming events with recent event history, leading to the use of hash-based caches and windowed arrays. Bloom Filters provide memory-efficient membership checking that enables rapid identification of known-bad indicator flags [13]. Graph structures allow for the representation of the relationship among accounts, devices, merchants and beneficiaries etc. Therefore, facilitate the use of link analysis to support both investigation and feature generation [5].

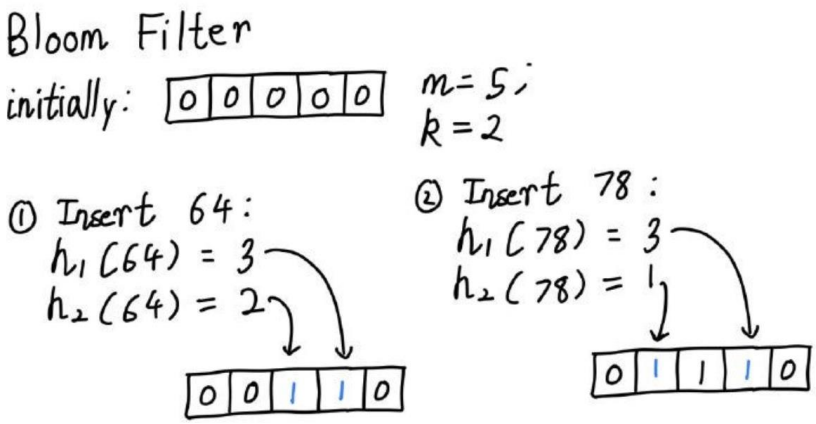

Figure 3: Bloom Filter

Credit risk assessment applies similar concepts to estimating the probability of default and expected loss severity for given exposures. Logistic Regression has been widely adopted for traditional credit scorecard development due to its interpretability, stability and ease of validation. Credit scoring literature highlights the trade-offs between prediction accuracy and explainability that is important for regulatory review and ensuring fairness to customers [8]. Credit risk engines must be capable of providing stable and repeatable decisions in very short timeframes, further emphasizing the importance of efficient data structures and consistent pipelines for analytical processing.

FRAUD DETECTION TECHNIQUES

- Statistical anomaly detection (outlier scoring, velocity rules)
- Supervised learning (logistic regression, trees, SVMs)
- Graph-based link analysis for networks of suspicious entities

RISK ASSESSMENT TECHNIQUES

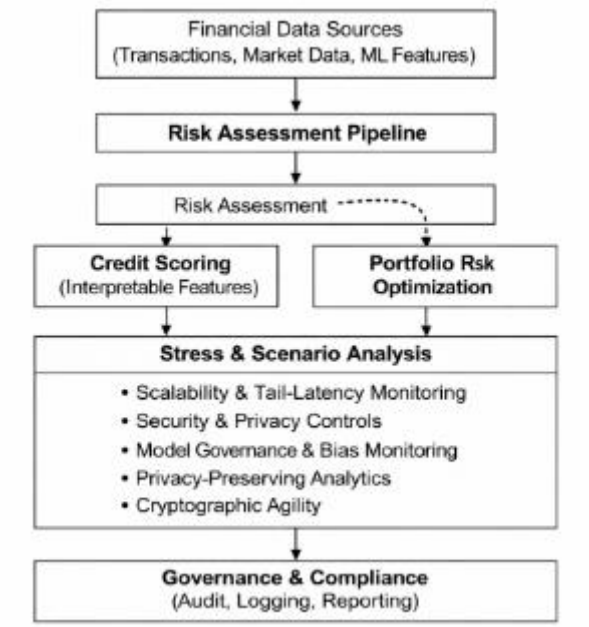

Figure 4: Risk Assessment pipeline integration.

- Credit scoring models with interpretable features [8]
- Portfolio risk optimization [15], [16]
- Stress and scenario analysis embedded in risk pipelines
- Challenges and future prospects

Data structures and algorithms enable speed and scale, but they also introduce new categories of risk.

Scalability and tail behavior:
Scalability is an issue of performance; however, scalability also has significant impacts on a company's ability to fulfill its market obligations and operational goals. Thus, systems which are unable to predict when they will degrade under heavy loads pose potential compliance and reputational risks; this leads to preferences for predictable data structures, example: balanced binary trees for ordered access, bounded queues, etc. and architectures that isolate critical latency pathways from less critical workloads.

Security and privacy:
Financial information has high economic and regulatory value. Cryptographic protection of financial data is both necessary to protect confidentiality and insufficient because the confidentiality of such data also requires secure management of keys used for cryptographic operations, secure control of who can access that data, and secure auditing and logging to detect misuse of that data. This is evident from how distributed ledger technology has been developed. While the integrity model of distributed ledgers is strong, application-specific vulnerabilities to attack and key compromises continue to represent significant potential failure modes for these systems; further, research studies indicate that there are still many security-related challenges related to cryptocurrencies [17], and thus future financial technologies platforms must consider security as a system property and not simply as an additional library function.

Model governance and fairness:
Machine learning models can encode historical bias, degrade under distributional shift, or fail in edge cases that matter disproportionately in finance therefore financial applications require rigorous validation, monitoring for drift, and escalation paths for ambiguous cases [4] explanations are not only ethical they also have operational necessity for audits, customer disputes, regulator inquiries, especially credit decisioning [8].

Privacy-preserving analytics:
The greater use of federated learning in distributed environments will help support these types of privacy and governance objectives because the model is developed at the edge of a network and there is no central repository that stores all the raw data. The ability to prove a statement about compliance or correctness to an auditor or regulator without having to reveal the underlying data is one area where zero-knowledge proof systems provide some direction in areas of research discussed in the security literature for anonymous payments and cryptographic proofs. Like other privacy-enhancing technologies, these methods also do not constitute "plug-and-play" solutions. They increase the complexity of how data is analyzed by adding additional layers of abstraction that require detailed threat modeling. However, they do provide the opportunity to find a balance between analytic capabilities and confidentiality.

Cryptographic agility and post-quantum preparation:
Although before 2020 large scale fault-tolerant quantum computers were not operational as a viable commercial or enterprise product base, the long-term threat model has relevance to financial institutions holding long lived records of this nature. A practical engineering implication is that systems should be designed with cryptographic agility, the ability to rotate an algorithm or key, or add new ones in mind, to reduce the risk of needing to migrate the entire system when a particular platform cannot support newer versions of the algorithm or key; thus, providing some form of future proofing.

CONCLUSION

Data Structures and Algorithms form the foundation of FinTech technology. The data structures and algorithms used in FinTech determine how quickly a system can respond- latency, how reliable it can be - reliability and how well it can govern itself -governance. Building on the core data structures such as arrays, hash tables, graphs, heaps and balanced trees allows developers to create systems to store, index and relate large amounts of financial information. As well as creating systems that store and relate financial information, sorting and searching allow developers to create systems that support reconciliations and audits, optimizing creates systems that support risk aware decision making, statistical learning enables developers to create systems that detect and score suspicious activity in real time and finally cryptography provides the means by which secure transactions can occur and ensure data integrity.

The fundamental challenge facing FinTech as it evolves further than simply using advanced algorithms will be creating systems that will continue to be correct, secure and auditable even when subject to an adversarial environment or changing market conditions. In order to address these challenges, privacy preserving analytics and proof systems represent possible solutions to balance automation and confidentiality obligations, however regardless of the solution implemented, robust monitoring and governance processes will remain required to manage the risks associated with models developed and deployed by organizations.

Therefore, continued collaboration between software engineers, security experts and financial domain experts will be required to develop FinTech platforms that are trustworthy, scalable and compliant.

REFERENCES:

[1] Hendershott, T., Jones, C. M., & Menkveld, A. J. (2011). Does algorithmic trading improve liquidity? *The Journal of Finance, 66*(1), 1–33. https://doi.org/10.1111/j.1540-6261.2010.01624.x
[2] Hasbrouck, J., & Saar, G. (2013). Low-latency trading. *Journal of Financial Markets, 16*(4), 646–679. https://doi.org/10.1016/j.finmar.2013.05.003
[3] Kirilenko, A. A., Kyle, A. S., Samadi, M., & Tuzun, T. (2017). The Flash Crash: High-frequency trading in an electronic market. *The Journal of Finance, 72*(3), 967–998. https://doi.org/10.1111/jofi.12498
[4] Bolton, R. J., & Hand, D. J. (2002). Statistical fraud detection: A review. *Statistical Science, 17*(3), 235–255. https://doi.org/10.1214/ss/1042727940
[5] Ngai, E. W. T., Hu, Y., Wong, Y. H., Chen, Y., & Sun, X. (2011). The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision Support Systems, 50*(3), 559–569. https://doi.org/10.1016/j.dss.2010.08.006

[6] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms* (3rd ed.). MIT Press.
[7] Knuth, D. E. (1998). *The art of computer programming, Volume 3: Sorting and searching* (2nd ed.). Addison-Wesley.
[8] Hand, D. J., & Henley, W. E. (1997). Statistical classification methods in consumer credit scoring: A review. *Journal of the Royal Statistical Society: Series A (Statistics in Society), 160*(3), 523–541. https://doi.org/10.1111/j.1467-985X.1997.00078.x
[9] Diffie, W., & Hellman, M. (1976). New directions in cryptography. *IEEE Transactions on Information Theory, 22*(6), 644–654. https://doi.org/10.1109/TIT.1976.1055638
[10] Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM, 21*(2), 120–126. https://doi.org/10.1145/359340.359342
[11] Carter, J. L., & Wegman, M. N. (1979). Universal classes of hash functions. *Journal of Computer and System Sciences, 18*(2), 143–154. https://doi.org/10.1016/0022-0000(79)90044-8
[12] Bayer, R., & McCreight, E. (1972). Organization and maintenance of large ordered indexes. *Acta Informatica, 1*(3), 173–189. https://doi.org/10.1007/BF00288683
[13] Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM, 13*(7), 422–426. https://doi.org/10.1145/362686.362692
[14] Bhattacharyya, S., Jha, S., Tharakunnel, K., & Westland, J. C. (2011). Data mining for credit card fraud: A comparative study. *Decision Support Systems, 50*(3), 602–613. https://doi.org/10.1016/j.dss.2010.08.008
[15] Markowitz, H. (1952). Portfolio selection. *The Journal of Finance, 7*(1), 77–91. https://doi.org/10.1111/j.1540-6261.1952.tb01525.x
[16] Rockafellar, R. T., & Uryasev, S. (2000). Optimization of conditional value-at-risk. *Journal of Risk, 2*(3), 21–41.
[17] Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J. A., & Felten, E. W. (2015). SoK: Research perspectives and challenges for Bitcoin and cryptocurrencies. In *2015 IEEE Symposium on Security and Privacy* (pp. 104–121). IEEE. https://doi.org/10.1109/SP.2015.14
[18] McMahan, H. B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)* (pp. 1273–1282).
[19] Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., & Virza, M. (2014). Zerocash: Decentralized anonymous payments from Bitcoin. In *2014 IEEE Symposium on Security and Privacy* (pp. 459–474). IEEE. https://doi.org/10.1109/SP.2014.36