# Saito Consensus: Fixing the Market Failures in Bitcoin

David Lancashire, Richard Parris, edited by Matthew Wilson

January 25, 2023
v. 4.0.2 (this is the Dec 24, 2020 version numner)

**Abstract**

Saito fixes the collective action problems which impede scaling in proof-of-work and proof-of-stake blockchains by coupling a circular (finite) ledger to a consensus mechanism that incentivizes the collection and sharing of transactions and their fees. Saito's measure of 'work' is more granular than traditional blockchain, rewarding the work of distributing transaction data across the network rather than solely paying miners or stakers for the privilege of authoring a block. Blocks in Saito are published as soon as they contain enough 'routing work' and rewards are unlocked via a hash lottery in subsequent blocks - the result is the full elimination of sustainable or profitable majoritarian (such as 51%) attacks in a blockchain which scales according to market demand. In economic terms, Saito is a solution for inducing the free market to deliver a public good. It is the authors belief that the practical limit for a Saito blockchain today is in the order of 100 TB data throughput per day, and that advances in routing technology will push this to the petabyte level within a decade.

## 1. THE PROBLEM

The problem of blockchain scaling is not at the network hardware level: at the time of writing data centers around the world are implementing 400 Gbps network switches while 100 Gbps connections are becoming standard even in lower-tier colocation facilities. There are no technical or physical constraints preventing a blockchain from distributing data between its nodes at arbitrary rates.

What limits blockchain growth is the economic challenge of designing a protocol which provisions existing technology while maintaining the desirable properties of blockchain. In the past, non-economists have waived these considerations under the assumption that as long as money is being paid to the network *at all*, the entire system will enjoy efficient, market driven funding. This is not the case; proof-of-work and proof-of-stake are both afflicted by two market failures described by collective action theory: a *Tragedy of the Commons* problem which leads to blockchain bloat and deferral of cost into the future, and a *Free Rider* problem which leads to an under-provision of user-facing network infrastructure in exchange for an over-provision of paid activities like mining and staking. Either of these problems represent the failure of the free market to efficiently allocate resources towards the blockchain such that the primary beneficiary are the public aspects of the network - rather, the free market without certain incentives will tend to privatize wherever possible. Neither problem is crippling at small scale, but each issue demonstrably forces compromises as bandwidth and storage costs rise with increased use and the infrastructure supporting it is forced to go private.

Faced with the need to pay for *all* network infrastructure, blockchain engineers have effectively conceded the problem to the market. Economists have known since the 1960s how these collective action problems arise when you ask the private sector to fund non-excludable infrastructure (the desirable openness property of blockchain): the economic model will always develop concentrated points of closure where money can be made most efficiently, as Infura does for Ethereum. Businesses which pay for transaction collection must necessarily close access to the fees which subsidize them; the resultant controlled flow of funds into the blockchain undermines the principled openness of the consensus layer.

Because traditional blockchain only subsidizes block production, it is only able to tax and reward block publication - but blockchain performance cannot be quantified and therefore cannot be guided by the incentive to make valid a block via mining or staking. For a consensus mechanism to judge only blocks is to ignore value in the data which makes up those blocks, so Saito's fundamental change is to derive its tax and reward measurements from individual transactions. Equally as important, but simpler to state, Saito's other innovation is to enact a market rate cost for data storage and allow data to be pruned from the chain without breaking consensus. To understand these solutions it is useful to define the economic problems in context.

The tragedy-of-the-commons is an issue created by the existence of a permanent ledger, which encourages nodes to accept payment today for work (in the form of data stores and transmission) which can be offloaded to others tomorrow. This incentive leads to blockchain bloat as well as transaction mis-pricing, since users pay fees

which do not reflect the true cost of their data to the network past the instance those fees are initially processed. The fact that this is a fundamental issue in blockchain is self-evident from the way Satoshi's solution in the Bitcoin White Paper was "not to care;" an approach which stops being viable in networks that operate at scale.

The solution requires all nodes that add transactions to the blockchain to bear the cost of processing and storing those transactions for as long as that data remains on chain. In practice, this requires a market mechanism for accurately determining the price of on-chain data storage. The components described in Section 2 help remedy the tragedy-of-the-commons though the elimination of blockchain storage creep and the inclusion of a meted-out, rather than the upfront, *market driven* payment scheme for nodes which add data onto the chain. This mechanism is described in section 2.

The free-rider problem is more complicated. It arises in blockchains where payments are made for one type of work (such as mining or staking) at the expense of other vital 'network work' like *networking*. This lopsided incentive structure motivates participants to maximize spending on paid activities at the expense of those actions which consensus does not reward. In the blockchain space, this results in miners and stakers "free-riding" on participants who do the necessary work of serving users through transaction collection, application development and in general running the user-facing network. The problem gets worse as the network scales: a Bitcoin miner that spends a smaller percentage of revenue hashing than its more miserly counterparts will constantly lose market share and eventually be forced to capitulate sans some external advantage to squander. The status-quo solutions to this are either to keep the costs of auxiliary network provisions low, as Bitcoin does, or to embrace the privatization of the user facing network in order to increase throughput, an approach commonly seen in high data throughput chains like Ethereum or BSV, to name a couple.

Likewise, standard economics typically solves the free-rider problem by eliminating the property of "non-excludability" associated with any good or service, restricting its benefits to those who pay the costs of provision. In the blockchain space, this solution fundamentally destroys the openness of the network. Computer scientists often address the problem by adding protective middleware which is itself susceptible to the same economic problems and subsequent attack vectors, such as wrapping consensus payments in closed voting rings. The 'jiggery-pokery' approach can never solve the underlying economic issues - markets are powerful and incentivized enough to undermine these structures and suck value wherever possible.

Without a solution to this problem the choice is between a network which cannot scale because it cannot pay for its own network operations, or a network which does scale at the cost of the core values of blockchain: openness, trustlessness, and economic self-sufficiency. Neither approach is acceptable for building a genuinely open blockchain at massive scale.

The theoretical solution to the free-rider problem requires fixing the underlying incentive structure so that participants are compensated for providing what the network actually needs, rather than primarily rewarding hashing or staking. Because the blockchain requires a quantifiable cost-of-attack, the measure of work in this approach is shifted from 'mining' or 'staking' towards a different form of work which rewards nodes in proportion to the value they provide to users in the network i.e. routing user transactions deep enough into the network for inclusion into the blockchain, with the rate of data-flow dependent on the included transaction fees.

This new measure of work derives itself from the transaction fees that users pay to include their transactions. This has fundamental benefits regarding cost of attack, network scaling, and value provided to users. Securing the rewards from this mechanism from fee-recycling attacks gives the network a quantifiable, negative return on attack in all scenarios even when the attacker has a monopoly on work. These details are fully described in Sections 3 & 4.

## 2. FIXING THE TRAGEDY OF THE COMMONS

Saito solves blockchain storage creep by allowing nodes in the network to delete the oldest block in the ledger as the newest one is added. The interval length of the blocks which must be stored is defined by an "epoch." Epoch length is specified in consensus code. In an extreme case a blockchain designed to handle global traffic for distributed key-exchange applications may have an epoch length corresponding to as short a time as 24 hours.

Saito specifies that once a block falls out of the current epoch, its unspent transaction outputs (UTXO) are no longer spendable, but *must* be rebroadcast. Any UTXO from that set which contains enough tokens to pay the rebroadcasting fee will be included in the very next block as enforced by consensus rules. Block producers do this by creating "automatic transaction rebroadcasting" (ATR) transactions that preserve the original transaction data, but have an entirely fresh and newly-spendable UTXO, and placing these ATR transactions in the newest block. Since these transactions must be re-included, block producers may only produce valid blocks when they possess the a full epoch's worth of his-

tory. After two epochs have elapsed block producers may delete all unpaid block data, though the 32-byte header hash may be retained to prove connection with the original genesis block.

The ATR mechanism fixes the tragedy of the commons problem completely, making it impossible for the blockchain to grow past its ability to fund itself through consensus. They key is ensuring that the 'rebroadcasting fee' paid by ATR transactions is proportional to the average fee paid per byte by all transactions added in the most recent epoch - this gives a market determined price to store data on chain based on what users were most recently willing to pay. As the blockchain expands and saturates the abilities of hardware, the price of transaction fees increases and forces up the fees paid by older transactions, which in turn increases the amount of data pruned. The market reaches equilibrium when old data is removed from the chain at the same pace that new data is added.

Market discovery of the true cost of blockchain processing is an important effect of this incentive structure. As storage cost decreases or demand increases, nodes in Saito can accept more transactions and increase the size of blocks without any forks, governance, hard-coded economic variables or manual adjustment at all. Subtler forms of free riding like nodes deleting on-chain data or refusing to store historical blocks are eliminated as well, as nodes which do not store the full epoch cannot determine which transactions to rebroadcast and thus fall out of consensus.

While ATR solves the problem of tragedy of the commons by ousting 'blockchain creep' and giving consensus-driven market prices for space on chain, there still exists the problem of paying for data's path into the network and on-chain in the first place. Solving this problem requires a new consensus mechanism.

## 3. ELIMINATING FREE-RIDING

In Saito, any node is allowed to publish a block once it has collected enough *routing work*. Routing work is a measure derived from the fees embedded in the transactions that a node has either directly received from users or been forwarded from other nodes. Routing work begins equal to the fee included in a transaction, and the work earned is halved every time a transaction is forwarded along its path to a subsequent routing node. One transaction may have multiple paths which each measure distinct sets of routing work, but a transaction's routing work is only counted in the path that led to a canonical block. Routing nodes cryptographically sign transactions as they propagate through the network to mark their claim on the routing work within. Valid blocks must contain a collection of transactions such that their sum of routing work they provide to nodes in the routing paths exceeds a certain threshold, which begins high and decays linearly towards zero over time.

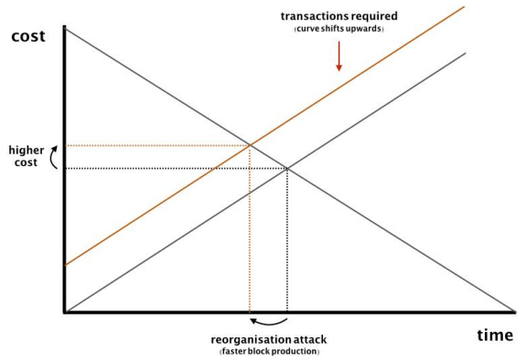


**Figure 1:** The Burn Fee Curve

Since routing work is derived from the fees embedded in transactions, attacking the network by attempting to buy more routing work has a measurable cost: the cost of out-competing all other user transactions. It can be seen from figure 2 that it is impossible for an attacker to produce blocks at a faster pace than the main chain unless they have access to sources of transactions which provides greater routing work than the rest of the network combined. This attack becomes more difficult as an attacker produces consecutive blocks, since the outstanding routing work from the rest of the network builds up the longer the attacker has a monopoly on block production; this cannot be said for Proof of Work or Proof of Stake where all work is orphaned upon block publication; Saito's conservation of useful 'work' outside of the most recently published block represents a significant buff in security over traditional blockchains since the routing work waiting to be included grows at a constant rate which increases the cost of continuing an attack at the same rate. If the attacker does not have exclusive access their source of routing work, other nodes will use it before any transaction hoarding can take place.

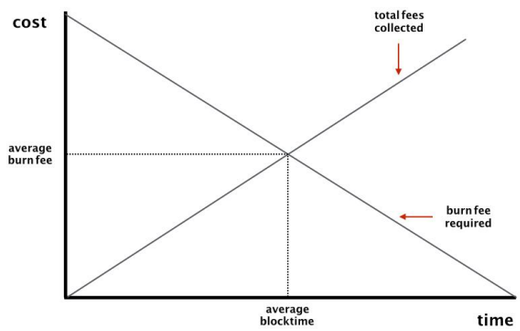


**Figure 2:** Good Actor Burn Fee Costs...

As long as there is no payment for block production (like a mining or staking reward), this system offers comparable security to Bitcoin; cost-of-attack can always be quantified and attackers must spend their own money to attack the chain. Users may wait however many block confirmations are needed to meet their security requirements. Also like Bitcoin, the network can increase the amount of routing work needed for block production to keep block-time constant, so

that as transaction fee volume grows, security scales with it. But payment to nodes must be done somewhere.

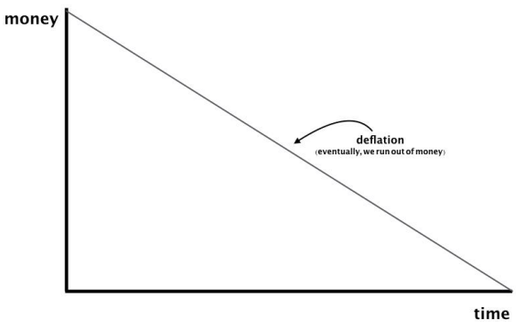


**Figure 3:** Deflation of Burn Fee Over Time

To fund network infrastructure and avoid a deflationary crash, Saito distributes the transaction fees from a completed block to nodes, on average, in proportion their routing work contributed to that block. Fees cannot simply be given directly to block producers as that would allow attackers to use income from one block to generate the routing work needed to produce the next one. Dividing up the rewards between different nodes is preferable, but so long as block producers have any influence over who gets paid, a savvy attacker can Sybil the network or conduct grinding attacks that target the token-issuing mechanism.

Distributing rewards securely is solved in Saito by inverting the classic proof-of-work solution. In Bitcoin, consensus rules make it expensive to produce blocks and fees are then handed to the block producer, along with a block subsidy. This is meant to ensure producing consecutive blocks is expensive, but it also guarantees conditions in which attacks are profitable and sustainable. In Saito the solution is the opposite: first by ensuring that rewards are, on average, proportional to work and not dependent upon who produces the block, and second by using retroactive proof of work to unlock those rewards.

We call this mechanism the "Golden Ticket." This mechanism pays honest nodes for collecting fees regardless of who produces blocks by unlocking payments through a lottery conducted in the *subsequent* block, but only if a proof of work puzzle is solved in time. The system is designed to ensure there is a quantifiable and cost for attacking the system in all scenarios.

## 4. THE GOLDEN TICKET

Whenever a node produces a block, it may collect the marginal surplus of fees tied to that block's routing work past the amount of routing work required by consensus for valid block production. The great majority of fees from this block are thus not distributed upon publishing.

Unlocking that greater remainder of fees requires the network to solve a computational puzzle called 'The Golden Ticket.' This puzzle requires knowledge of the block hash from which the rewards are to be distributed from and thus cannot be calculated in advance. Miners on the network listen for blocks as they are produced and may begin hashing in search of a solution. Should they find a valid solution, they propagate it into the network as a normal fee-paying transaction; multiple successful miners compete for their solution to be included via the transaction fee they include.

Only one solution may included may be included in any block, and that solution must correspond to the proof of work challenge from the block directly before it to be considered valid. If a Golden Ticket goes unsolved, the funds that were not paid out in the previous block are simply not allocated - they remain within their block and eventually fall off the chain, at which point the lost funds are recollected by the consensus layer and eventually redistributed as part of a future block reward.

Should a Golden Ticket solution be found in time, the associated fees are released to the network, split between the miner that found the solution and a random node in the peer-to-peer routing network. The routing node is selected using a random variable sourced from the miner's solution, with each routing node's chance of winning proportional to the overall routing work it contributed to the block being solved.

We call the division of payment between the miners and routers the *paysplit* of the network. It is set to 0.5 by default (half to miners, half to routers) but can be made adjustable as described in the section below. The golden ticket system can be visualized as follows.

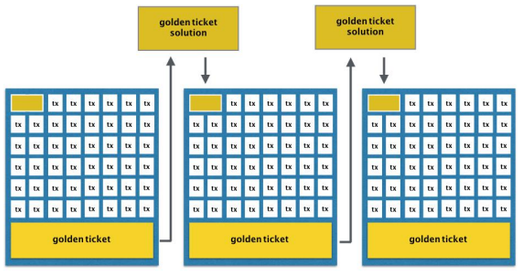


**Figure 4:** The Golden Ticket System

This system has several major advantages over the proof-of-work and proof-of-stake mechanisms, including the elimination majoritarian attacks - that is, the complete amelioration of the threat of a sustainable block re-org attack otherwise known as the 51% attack. Saito explicitly distributes fees to the nodes that service users, collect transactions and produce blocks, doing so in proportion to the amount of value that these actors provide to the overall network - not just to hashers or stakers. Network nodes compete for access to lucrative inbound transaction flow, and will happily fund whatever development activities are needed to get users into the network through their infrastructure, supplying them with primary access to routing work. Of

note, the services provided by the edge nodes to attract Saito usage can include public-facing infrastructure needed by other blockchains.

This is a fundamental shift: where other blockchains explicitly define what activities have value, Saito lets the users signal what services provide value through fee-pricing, while the public network infers who deserves payments rather than having the private market close around where blockchain was once blind and inefficient. Saito incentivizes the efficient delivery of value to users by paying for value rather than an arbitrary subset of network activities; routing work holistically encapsulates what gives a blockchain value and lets network incentives optimize the delivery of that value. Without the need for private closure around vital functions which blockchain used to lack rewards for, the network is guaranteed to remain self-sufficient and open at scale.

The Saito consensus mechanism is also "twice as secure" as its proof-of-work and proof-of-stake counterparts. Honest nodes route transactions to block producers and earn fees in exchange, but attackers are thrown into a catch-22: they must not only spend the same amount of fees as the honest network the produce a competitive chain, but also match 100 percent of the mining output to find enough golden ticket solutions to recapture their attack costs. Even if attackers successfully launch fee-recycling attacks, they must still spend 100 percent of their income on hashing since they do not have the benefit of miners from the honest chain helping unlock fees used in the attack, and can therefore only earn back half of their fees.

The basic version of the Saito system achieves 100 percent fee-security, eliminating the fifty-one percent attack completely. Section 5 describes a modification to this mechanism that pushes security above 100 percent and guarantees that attackers will always lose Saito by attacking the network. Regardless of which implementation is used, the economic problems created by traditional consensus mechanisms which rely on external supply-curves vanish: mining serves as a pure cost function instead of a difficulty function allowing the blockchain to remain secure even as the supply curve for hash becomes perfectly flat.

## 5. ADVANCED SECURITY - POWSPLIT

It is possible to increase attack costs beyond 100 percent of available returns through a "powsplit" mechanism. Note that in the normal Saito implementation with a fixed paysplit of 0.5, the network auto-adjusts mining difficulty so that one solution is found per block, on average. Since miners cannot control the variance at which solutions are found, network difficulty may end up being lower on average than needed for optimal security.

A "powsplit" approach eliminates this problem by adjusting mining difficulty so that one solution is found every N blocks on average. When such a solution is included in the blockchain, if the previous block did not contain a golden ticket, the random variable used to pick the winning routing node is hashed again to select a winner from the unsolved block which preceded it or from a table of stakers as described below. An upper limit to backwards recursion may be applied for practical purposes, as the circular blockchain will recapture any funds that are not paid out.

To become stakers in the network, users broadcast a transaction containing a specially-formatted UTXO. These UTXO are added to a list of "pending stakers" on their inclusion in a block. Once the current staking table has been fully paid-out, all pending staking UTXO are moved into the current staking table. To avoid throttling attacks on the staking mechanism it is wise to not permit stakers to withdraw or spend staked UTXO until they have received payment.

The percentage of network revenue allocated to staking nodes should be proportional to their share of the amount of fees paid into the treasury by the staking mechanism during the previous round. Limits may be put on the size of the staking pool to induce competition between stakers if desirable or prevent users from spamming the staking mechanism in the hope of dissuading honest stakers from participating. In normal situations the looping blockchain and ATR mechanism will prevent stakers from launching spam attacks as multiple UTXO will all pay rebroadcast fees.

To ensure the system works, block producers who rebroadcast UTXO must now indicate in their ATR transactions whether the specific outputs are active in the current or pending staking pool. A hash representation of the state of the staking table may be included in every block in the form of a commitment to allow nodes to verify the accuracy of the staking table, but the ATR rebroadcast mechanism will theoretically allow all nodes to reconstruct the state of the staking pool within one epoch at most.

Mining difficulty can be adjusted upwards if two blocks containing golden tickets are found in a row and downwards if two blocks without golden tickets are found consecutively. A similar punitive cost can throttle the staking payout if two blocks without golden tickets are found in a row (an ever-increasing amount of the staking revenue is withheld). We encourage those interested in the underlying mathematics to consult our papers on the topic. The cost of attacking a Saito network using this mechanism rises significantly above 100 percent.

## 6. ADVANCED SECURITY - PAYSPLIT

There are several modifications to the paysplit mechanism that can be used to increase attack costs. While the version of Saito being launched for production does not include this mechanism, it is possible to add a dynamic voting system to Saito that can allow paysplit to float dynamically. This section describes a theoretical improvement that allows for a floating paysplit that will work under certain assumptions about the rationality of the network.

An implementation of this system modifies blocks so that they include a vote to increase, decrease or hold constant the paysplit of the network. Golden ticket solutions may be then modified so that they contain similar vote on the difficulty of the golden-ticket production function. The consensus variables of the network are updated when and only when golden tickets are solved and included in the blockchain.

Adjusting paysplit like this can change the distribution of fees between routing nodes and miners in real-time. This allows the network to reach an optimal equilibrium rather than an arbitrary one. To prevent this equilibrium from reflecting only the preferences of the routing and mining nodes, we recommend letting the users on the network tag their transactions with an optimal paysplit vote as well: should a user-originated transaction contain such a vote, it may only be included in a block that votes in the same direction. Users who take sides in the ongoing struggle between routers and miners thus sacrifice the reliability and speed of transaction confirmation, but gain marginal influence over how the network allocates fees. Users making votes are also withholding their fees from nodes voting differently to themselves.

Under conditions where network participants exhibit bounded rationality, this mechanism pushes paysplit to the point where the security provided is optimal for all participants given the cost of additional fee collection. De Toqueville compacts secure the equilibrium: any two players in the tripartite network structure (routers, miners, users) may team-up to shift the paysplit back to the desired ideal. While we leave research into this mechanism for the future, a useful thought experiment is exploring how the security of this three-player system degrades to only bitcoin-class security as the paysplit approaches extreme values.

## 7. ADDITIONAL NOTES ON NETWORK SECURITY

Saito's design solves several long-standing problems of note. Hoarding attacks are minimized because nodes that participate in transaction routing maximize revenue by finding the most efficient routing path into the network. Competition encourages the sharing of fees rather than the hoarding of fees. The availability of routing information in blocks also allows participants to check that their peers are faithfully propagating their transactions instead of hoarding them.

Because adding hops to any routing path necessarily reduces the profitability of routing for every node in the path, sybil attacks are also eliminated. Blocks provide the information needed for participants to identify and eliminate sybils in their peer-to-peer networks. And evolutionary pressures ensures that they will purge them: weaker nodes which permit themselves to be sybilled will find themselves driven out of the network by competitive pressures over time.

The routing network also serves a unique defensive mechanism. Routing nodes in Saito can increase the cost of attack in real-time by refusing to route transactions to attackers, forcing an increased reliance by the attacker on their own wallet to fund block production. This mechanism also defends Saito against subtle attacks like monetizing transaction flows and closed-access routing.

As a final observation, we note that the "scalability trilemma" often championed as a fundamental law of blockchain does not exist in the Saito design. There are many obvious configurations of the network in which redirecting fees from miners to routing nodes can simultaneously increase the throughput, decentralization and security of the network simultaneously.

## 8. SUMMARY

The fundamental problems affecting blockchain scaling are economic. Saito fixes these issues, allowing us to build a massively-scalable blockchain which achieves scalability by ensuring that payments flow to the nodes that spend money on network infrastructure.

Those who pour over the technical details of the Saito network will find embedded in it at least seven major innovations in blockchain technology: automatic transaction rebroadcasting, the burn fee, the golden ticket system, paysplit and powsplit, N-block golden tickets, a secure multiparty voting mechanism, and the chain of cryptographic signatures that permits the blockchain to identify and reward productive nodes in the routing network.

Patent protection has been secured on these techniques and we welcome contact from other blockchain projects looking to incorporate one or several of these methods in their own networks. We also encourage readers to visit our website (https://saito.io) which includes an interface for the working network, a roadmap outlining future development plans, and tutorials that can help anyone get started building Saito applications **today**.