UNCHAINED RESEARCH

# How did Satoshi think of bitcoin?

SOUND MONETARY POLICY SOLVED THE DOUBLE SPEND PROBLEM

DHRUV BANSAL

12 DECEMBER 2023

# What hath Satoshi wrought?

## Preface

Bitcoin is often compared to the internet in the 1990s, but I believe the better analogy is to the telegraph in the 1840s.[1]

The telegraph was the first technology to transmit encoded data at near-light speed over long distances. It marked the birth of the telecommunications industry. The internet, though it is bigger in scale, richer in content, and many-to-many instead of one-to-one, is fundamentally still a telecommunications technology.

Both the telegraph and the internet rely upon business models in which companies deploy capital to build a physical network and then charge users to send messages through this network. AT&T's network has historically transmitted telegrams, telephone calls, TCP/IP packets, text messages, and now TikToks.

The transformation of society through telecom has led to greater freedoms but also greater centralization. The internet has increased the reach of millions of content creators and small businesses, but has also strengthened the grasp of companies, governments and other institutions well-positioned enough to monitor and manipulate online activity.

But bitcoin is not the end of any transformation—it's the beginning of one. Like telecommunications, bitcoin will change both human society and daily life. Predicting the full scope of this change today is akin to imagining the internet while living in the era of the telegraph.

This series attempts to imagine this future by starting with the past. This initial article traces the history of digital currencies before bitcoin. Only by understanding where prior projects fell short can we perceive what makes bitcoin succeed—and how it suggests a methodology for building the decentralized systems of the future.

# Contact

Follow Dhruv Bansal on twitter: @dhruvbansal

Send Dhruv Bansal an email: dhruv@unchained.com

Follow Unchained on twitter: @unchainedcom

Send Unchained an email: hello@unchained.com

# Outline

I. Decentralized systems are markets

II. Decentralized markets require decentralized goods

III. How can decentralized systems price computations?

IV. Satoshi’s monetary policy goals led to bitcoin

V. Conclusion

# How did Satoshi think of bitcoin?

Satoshi was brilliant, but bitcoin didn't come out of nowhere.

Bitcoin iterated on existing work in cryptography, distributed systems, economics, and political philosophy. The concept of proof-of-work existed long before its use in money and prior cypherpunks such as Nick Szabo, Wei Dai, & Hal Finney anticipated and influenced the design of bitcoin with projects such as bit gold, b-money, and RPOW.

Consider that, by 2008, when Satoshi wrote the bitcoin white paper,[2] many of the ideas important to bitcoin had already been proposed and/or implemented:

- Digital currencies should be P2P networks
- Proof-of-work is the basis of money creation
- Money is created through an auction
- Public key cryptography is used to define ownership & transfer of coins
- Transactions are batched into blocks
- Blocks are chained together through proof-of-work
- All blocks are stored by all participants

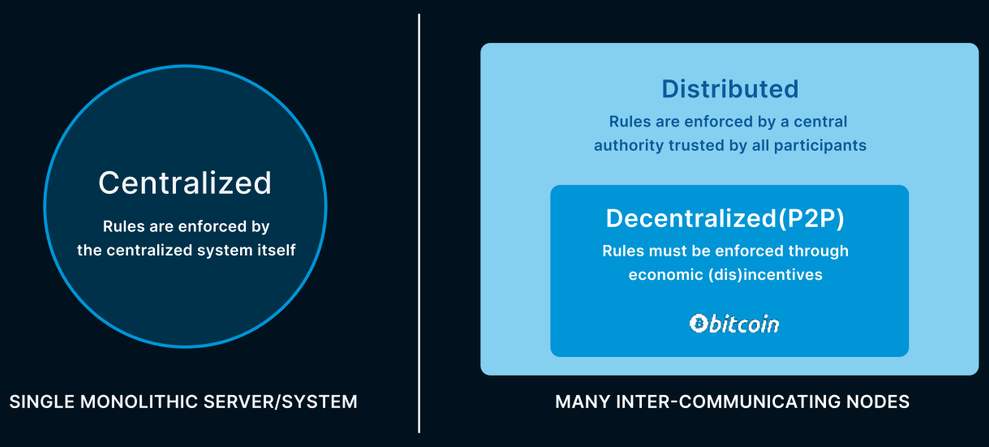
Bitcoin leverages all these concepts, but Satoshi didn't originate any of them. To better understand Satoshi's contribution, we should determine which principles of bitcoin are missing from the list below.

Some obvious candidates are the finite supply of bitcoin, Nakamoto consensus and the difficulty adjustment algorithm. But what led Satoshi to these ideas in the first place?

This article explores the history of digital currencies and makes the case that Satoshi's focus on sound monetary policy is what led bitcoin to surmount challenges that defeated prior projects such as bit gold and b-money.

# I. Decentralized systems are markets

Bitcoin is often described as a decentralized or distributed system. Unfortunately, the words “decentralized” and “distributed” are frequently confused. When applied to digital systems, both terms refer to ways a monolithic application can be decomposed into a network of communicating pieces.




It’s common to illustrate the terms “centralized”, “distributed”, and “decentralized” using pictorial network diagrams.

For our purposes, the major difference between decentralized and distributed systems is not the topology of their network diagrams, but the way they enforce rules. We take some time in the following section to compare distributed and decentralized systems and motivate the idea that robust decentralized systems are markets.

## Distributed systems rely upon central authorities

In this work, we take “distributed” to mean any system that has been broken up into many parts (often referred to as “nodes”) which must communicate, typically over a network.

Software engineers have grown adept at building globally distributed systems. The internet is composed of distributed systems collectively containing billions of nodes. We each have a node in our pocket that both participates in and relies upon these systems.

But almost all the distributed systems we use today are governed by some central authority, typically a system administrator, company, or government that is mutually trusted by all nodes in the system.

Central authorities ensure all nodes adhere to the system’s rules and remove, repair, or punish nodes that fail to do so. They are trusted to provide coordination, resolve conflicts, and allocate shared resources. Over time, central authorities manage changes to the system, upgrading it or adding features, and ensuring that participating nodes comply with the changes.

The benefits a distributed system gains from relying upon a central authority come with costs. While the system is robust against failures of its nodes, a failure of its central authority may cause it to stop functioning overall. The ability for the central authority to unilaterally make decisions means that subverting or eliminating the central authority is sufficient to control or destroy the entire system.

Despite these trade-offs, if there is a requirement that a single party or coalition must retain central authority, or if the participants within the system are content with relying upon a central authority, then a traditional distributed system is the best solution. No blockchain, token, or similar decentralized dressing is required.

In particular, the case of a VC- or government-backed cryptocurrency, with requirements that a single party can monitor or restrict payments and freeze accounts, is the perfect use case for a traditional distributed system.

## Decentralized systems have no central authorities

We take “decentralized” to have a stronger meaning than “distributed”: decentralized systems are a subset of distributed systems that lack any central authority. A close synonym for “decentralized” is “peer-to-peer” (P2P).

Removing central authority confers several advantages. Decentralized systems:

- Grow quickly because they lack barriers to entry—anyone can grow the system by simply running a new node, and there is no requirement for registration or approval from the central authority.
- Are robust because there is no central authority whose failure can compromise the functioning of the system. All nodes are the same, so failures are local and the network routes around damage.
- Are difficult to capture, regulate, tax, or surveil because they lack centralized points of control for governments to subvert.

These strengths are why Satoshi chose a decentralized, peer-to-peer design for bitcoin:

> *Governments are good at cutting off the heads of... centrally controlled networks like Napster, but pure P2P networks like Gnutella and Tor seem to be holding their own.*
>
> Nakamoto, 2008

But these strengths come with corresponding weaknesses. Decentralized systems can be less efficient as each node must additionally bear responsibilities for coordination previously assumed by the central authority.

Decentralized systems are also plagued by scammy, adversarial behavior. Despite Satoshi’s nod to Gnutella, anyone who’s used a P2P file sharing program to download a file that turned out to be something gross or malicious understands the reasons that P2P file sharing never became the mainstream model for data transfer online.

Satoshi didn't name it explicitly, but email is another decentralized system that has evaded government controls. And email is similarly notorious for spam.

## Decentralized systems are governed through incentives

The root problem, in all of these cases, is that adversarial behavior (seeding bad files, sending spam emails) is not punished, and cooperative behavior (seeding good files, only sending useful emails) is not rewarded. Decentralized systems that rely upon their participants to be good actors fail to scale because they cannot prevent bad actors from also participating.

Without imposing a central authority, the only way to solve this problem is to use economic incentives. Good actors, by definition, play by the rules because they're inherently motivated to do so. Bad actors are, by definition, selfish and adversarial, but proper economic incentives can redirect their bad behavior towards the common good. Decentralized systems that scale do so by ensuring that cooperative behavior is profitable and adversarial behavior is costly.

The best way to implement robust decentralized services is to create markets where all actors, both good and bad, are paid to provide that service. The lack of barriers to entry for buyers and sellers in a decentralized market encourages scale and efficiency. If the market's protocols can protect participants from fraud, theft, and abuse, then bad actors will find it more profitable to either play by the rules or go attack a different system.

## II. Decentralized markets require decentralized goods

But markets are complex. They must provide buyers and sellers the ability to post bids & asks as well as discover, match and settle orders. They must be fair, provide strong consistency, and maintain availability despite periods of volatility.

Global markets today are extremely capable and sophisticated, but using traditional goods and payment networks to implement incentives in a decentralized market is a nonstarter. Any coupling between a decentralized system and fiat money, traditional assets, or physical commodities would reintroduce dependencies on the central authorities that control payment processors, banks, & exchanges.


Decentralized systems cannot transfer cash, look up the balance of a brokerage account, or determine the ownership of property. Traditional goods are completely illegible from within a decentralized system. The inverse is not true – traditional systems can interact with bitcoin as easily as any other actor (once they decide they want to). The boundary between traditional and decentralized systems is not an impassable wall, but a semi-permeable membrane.

This means that decentralized systems cannot execute payments denominated in any traditional good. They cannot even determine the balances of fiat-dominated accounts or the ownership of real estate or physical goods. The entire traditional economy is completely illegible from within decentralized systems.

Creating decentralized markets requires trading new kinds of decentralized goods which are legible and transferable within decentralized systems.

## Computation is the first decentralized good

The first example of a “decentralized good” is a special class of computations first proposed in 1993 by Cynthia Dwork and Moni Naor.[3]

Because of deep connections between mathematics, physics, and computer science, these computations cost real-world energy and hardware resources—they cannot be faked. Since real-world resources are scarce, these computations are also scarce.

The input for these computations can be any kind of data. The resulting output is a digital “proof” that the computations were performed on the given input data. Proofs contain a given “difficulty” which is (statistical) evidence of a given amount of computational work. Most importantly, the relationship between the input data, the proof, and the original computational work performed can be independently verified without appeal to any central authority.

The idea of passing around some input data along with a digital proof as evidence of real-world computational work performed on that input is now called “proof-of-work”.[4] Proofs-of-work are, to use Nick Szabo’s phrase, “unforgeable costliness”. Because proofs-of-work are verifiable by anyone, they are economic resources that are legible to all participants in a decentralized system. Proofs-of-work turn computations on data into decentralized goods. Dwork & Naor proposed using computations to limit the abuse of a shared resource by forcing participants to provide proofs-of-work with a certain minimum difficulty before they can access the resource:

*In this paper we suggest a computational approach to combatting the proliferation of electronic mail. More generally, we have designed an access control mechanism that can be used whenever it is desirable to restrain, but not prohibit, access to a resource.*

## Dwork & Naor, 1993

In Dwork & Naor's proposal, an email system administrator would set a minimum proof-of-work difficulty for delivering email. Users wanting to send email would need to perform a corresponding number of computations with that email as the input data. The resulting proof would be submitted to the server alongside any request to deliver the email.

Dwork & Naor referred to the difficulty of a proof-of-work as a "pricing function" because, by adjusting the difficulty, a "pricing authority" could ensure that the shared resource remained cheap to use for honest, average users but expensive for users seeking to exploit it. In the email delivery market, server administrators are the pricing authorities; they must choose a "price" for email delivery which is low enough for normal usage but too high for spam.

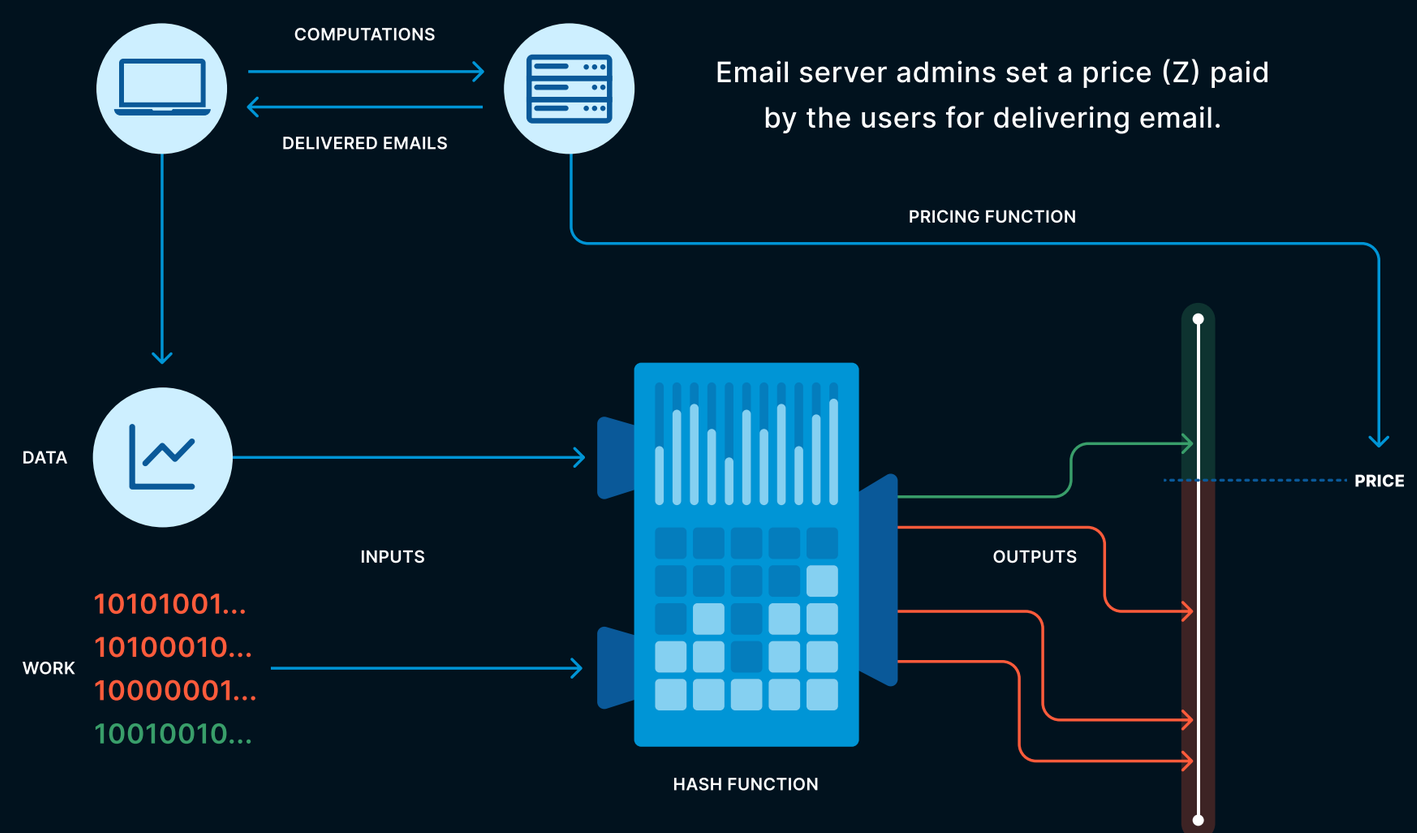
Though Dwork & Naor framed proofs-of-work as an economic disincentive to combat resource abuse, the nomenclature "pricing function" and "pricing authority" supports a different, market-based interpretation: users are purchasing access to a resource in exchange for computations at a price set by the resource's controller.

In this interpretation, an email delivery network is really a decentralized market trading email delivery for computations. The minimum difficulty of a proof-of-work is the asking price for email delivery denominated in the currency of computations.

## Currency is the second decentralized good

But computations aren't a good currency.

The proofs used to "trade" computations are only valid for the input used in those computations. This unbreakable lilnk between a specific proof and a specific input means that the proof-of-work for one input can't be reused for a different input.




Proof-of-work was originally proposed as an access control mechanism for limiting spam emails. Users would be expected to provide proofs-of-work alongside any emails they wanted to send. This mechanism can also be thought of as a market where users are purchasing email deliveries with computations at a price chosen by the email service provider.

This constraint is useful – it can be used to prevent the work done by one buyer in the market from being re-spent by another. For example, HashCash, the first real implementation of the market for email delivery, included metadata such as the current timestamp and the sender's email address in the input data to its proof-of-work computations. Proofs produced by a given user for a given email can't be respent for sending a different email.

But this also means that proof-of-work computations are bespoke goods. They aren't fungible, they can't be re-spent,[5] and they don't solve the coincidence-of-wants problem. These missing monetary properties prevent computations from being currency. Despite the name, there is no incentive for an email delivery provider to want to accumulate HashCash, as there would be for actual cash.

Adam Back, inventor of HashCash, understood these problems:

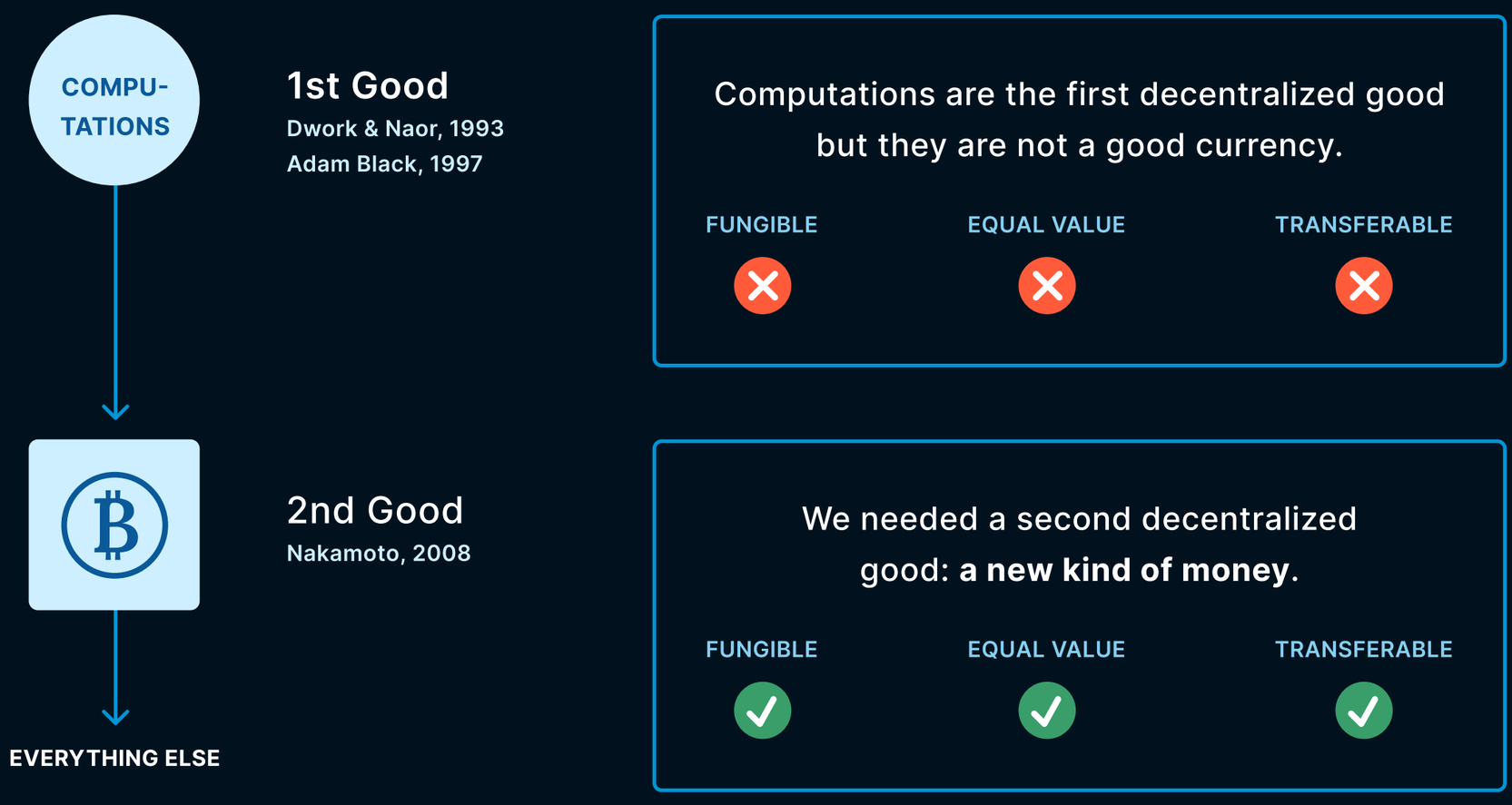
> *hashcash is not directly transferable*
> *because to make it distributed,*
> *each service provider accepts payment*
> *only in cash created for them.*
>
> *You could perhaps setup a digicash style*
> *mint (with chaumian ecash)*
> *and have the bank only mint cash on*
> *receipt of hash collisions*
> *addressed to it.*
>
> *However this means you've got to trust*
> *the bank not to mint unlimited*
> *amounts of money for it's own use.*
>
> Adam Back, 1997

We don’t want to exchange bespoke computations for every individual good or service sold in a decentralized economy. We want a general purpose digital currency that can directly be used to coordinate exchanges of value in any market.

Building a functioning digital currency while remaining decentralized is a significant challenge. A currency requires fungible units of equal value that can be transferred among users. This requires issuance models, cryptographic definitions of ownership and transfer, a discovery and settlement process for transactions, and a historical ledger. None of this infrastructure is required when proof-of-work is thought of as a mere “access control mechanism”.

Moreover, decentralized systems are markets, so all these basic functions of a currency must somehow be provided through paying service providers...in the units of the currency that’s being created!

Like compiling the first compiler, a black start of the electrical grid, or the evolution of life itself, the creators of digital currencies were confronted with a bootstrapping problem: how to define the economic incentives that underlie a functioning currency without having a functioning currency in which to denominate or pay those incentives.




Computations and currency are the first and second goods in decentralized markets. Proof-of-work alone allows for the exchange of computations but a functioning currency requires more infrastructure. It took 15 years for the cypherpunk community to develop that infrastructure.

# The first decentralized market must trade computations for currency

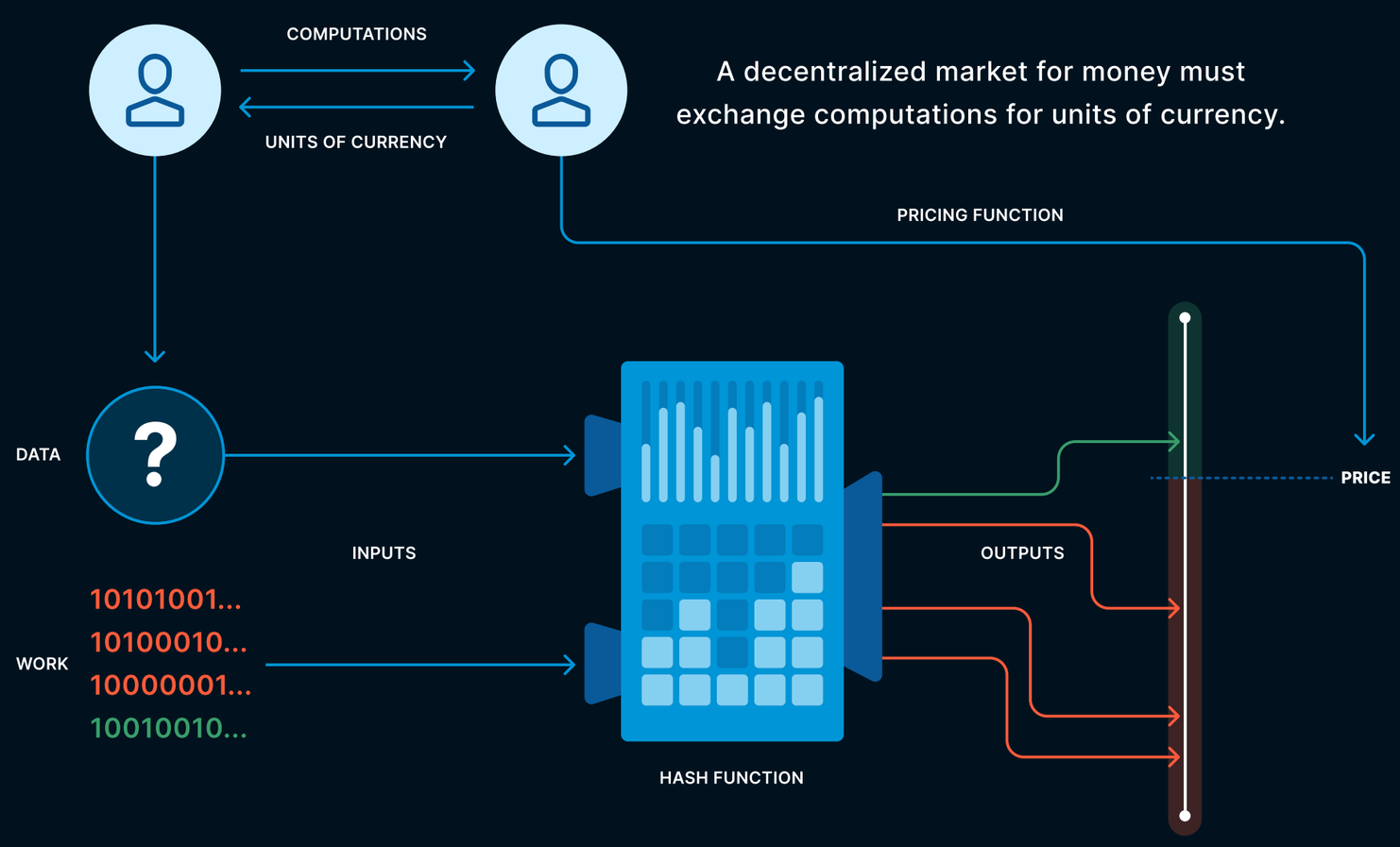
Progress on this bootstrapping problem comes from properly framing its constraints.

Decentralized systems must be markets. Markets consist of buyers and sellers exchanging goods. The decentralized market for a digital currency only has two goods that are legible within it:

1. Computations, through proof-of-work
2. Units of the currency we're trying to build

The only market trade possible must therefore be between these two goods. Computations must be sold for units of currency or, equivalently, units of currency must be sold for computations. Stating this is easy—the hard part is structuring this market so that simply exchanging currency for computation bootstraps all the capabilities of the currency itself!

The entire history of digital currencies, culminating in Satoshi's 2008 white paper, was a series of increasingly sophisticated attempts at structuring this market. The following section reviews projects such as Nick Szabo's bit gold and Wei Dai's b-money. Understanding how these projects structured their markets, and why they failed, will help us frame why Satoshi and bitcoin succeeded.




Decentralized systems are markets and markets require the exchange of goods. The first two decentralized goods are computations and currency, so the fundamental decentralized system must be a market exchanging computations for currency. Somehow engaging in this trade repeatedly must bootstrap the entire infrastructure of the currency itself.

# III. How can decentralized systems price computations?

A major function of markets is price discovery. A market trading computations for currency must therefore discover the price of computation itself, as denominated in units of that currency.

We don’t typically assign monetary value to computations. We typically value the capacity to perform computations because we value the output of computations, not the computations themselves. If the same output can be performed more efficiently, with fewer computations, that is usually called “progress”.

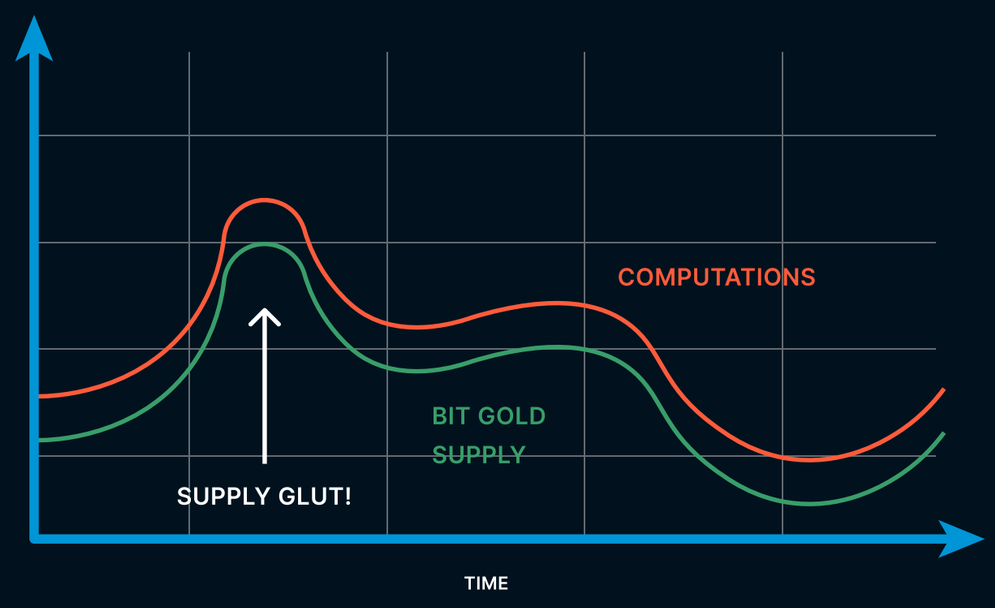
Proofs-of-work represent specific computations whose only output is proof that they were performed. Producing the same proof by performing fewer computations and less work wouldn’t be progress—it would be a bug. The computations associated with proofs-of-work are thus a strange and novel good to attempt to value.

When proofs-of-work are thought of as disincentives against resource abuse, it is not necessary to value them precisely or consistently. All that matters is that the email service provider sets difficulties low enough to be unnoticeable for legitimate users yet high enough to be prohibitive for spammers. There is thus a broad range of acceptable “prices” and each participant acts as their own pricing authority, applying a local pricing function.

But units of a currency are meant to be fungible, each having the same value. Due to changes in technology over time, two units of currency created with the same proof-of-work difficulty—as measured by the number of corresponding computations—may have radically different real-world costs of production, as measured by the time, energy, and/or capital to perform those computations . When computations are sold for currency, and the underlying cost of production is variable, how can the market ensure a consistent price?

Nick Szabo clearly identified this pricing problem when describing bit gold:

> *The main problem...is that proof of work schemes depend on computer architecture, not just an abstract mathematics based on an abstract "compute cycle." ...Thus, it might be possible to be a very low cost producer (by several orders of magnitude) and swamp the market with bit gold.*
>
> Szabo, 2005




A decentralized currency created through proof-of-work will experience supply gluts and crashes as the supply of computations changes over time. To accommodate this volatility, the network must learn to dynamically price computations.

Early digital currencies attempted to price computations by attempting to collectively measure the "cost of computing". Wei Dai, for example, proposes the following hand-wavy solution in b-money:

> *The number of monetary units created is equal to the cost of the computing effort in terms of a standard basket of commodities. For example if a problem takes 100 hours to solve on the computer that solves it most economically, and it takes 3 standard baskets to purchase 100 hours of computing time on that computer on the open market, then upon the broadcast of the solution to that problem everyone credits the broadcaster's account by 3 units.*

Dai, 1998

Unfortunately, Dai does not explain how users in a supposedly decentralized system are supposed to agree upon the definition of a "standard basket", which computer solves a given problem "most economically", or the cost of computation on the "open market". Achieving consensus among all users about a time-varying shared dataset is the essential problem of decentralized systems!

To be fair to Dai, he realized this:

> *One of the more problematic parts in the b-money protocol is money creation. This part of the protocol requires that all [users] decide and agree on the cost of particular computations. Unfortunately because computing technology tends to advance rapidly and not always publicly, this information may be unavailable, inaccurate, or outdated, all of which would cause serious problems for the protocol.*

Dai, 1998

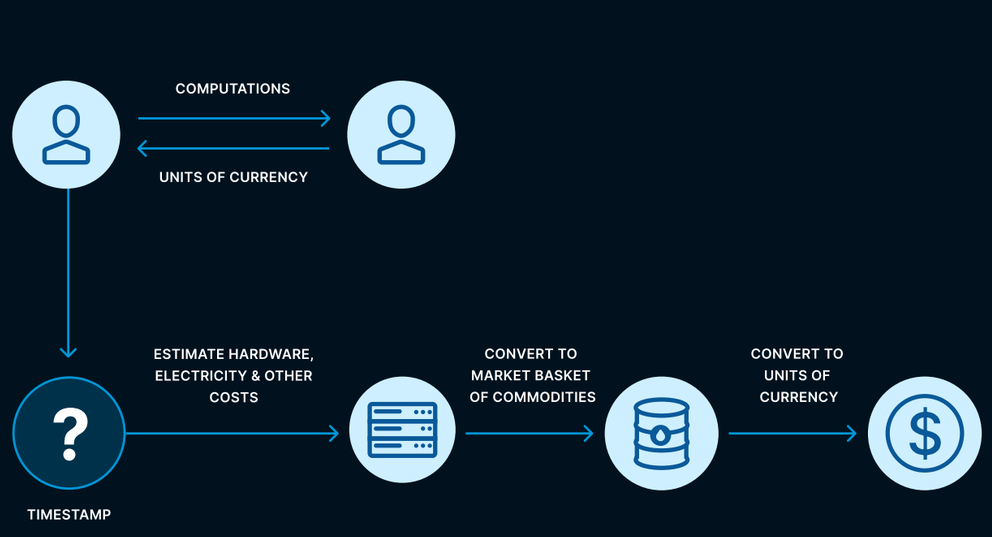
Dai would go on to propose a more sophisticated auction-based pricing mechanism, one that was a direct inspiration for Satoshi's design in bitcoin. We will return to this auction scheme below, but first let's turn to bit gold, and consider Szabo's insights into the problem.

## Use external markets

Szabo claims that proofs-of-work should be "securely timestamped":

> *The proof of work is securely timestamped. This should work in a distributed fashion, with several different timestamp services so that no particular timestamp service need be substantially relied on.*
>
> Szabo, 2005

Szabo links to a page of resources on secure timestamping protocols but does not describe any specific algorithm for secure timestamping. The phrases "securely" and "distributed fashion" are carrying a lot of weight here, hand-waving through the complexities of relying upon one (or many) "outside the system" services for timestamping.[6]




The time a unit of digital currency was created is important because it links the computations performed to real-world production cost.

Regardless of implementation fuzziness, Szabo was right—the time a proof-of-work was created is an important factor in pricing it because it is related to the cost of computation:

> *...However, since bit gold is timestamped, the time created as well as the mathematical difficulty of the work can be automatically proven. From this, it can usually be inferred what the cost of producing during that time period was...*
>
> Szabo, 2005

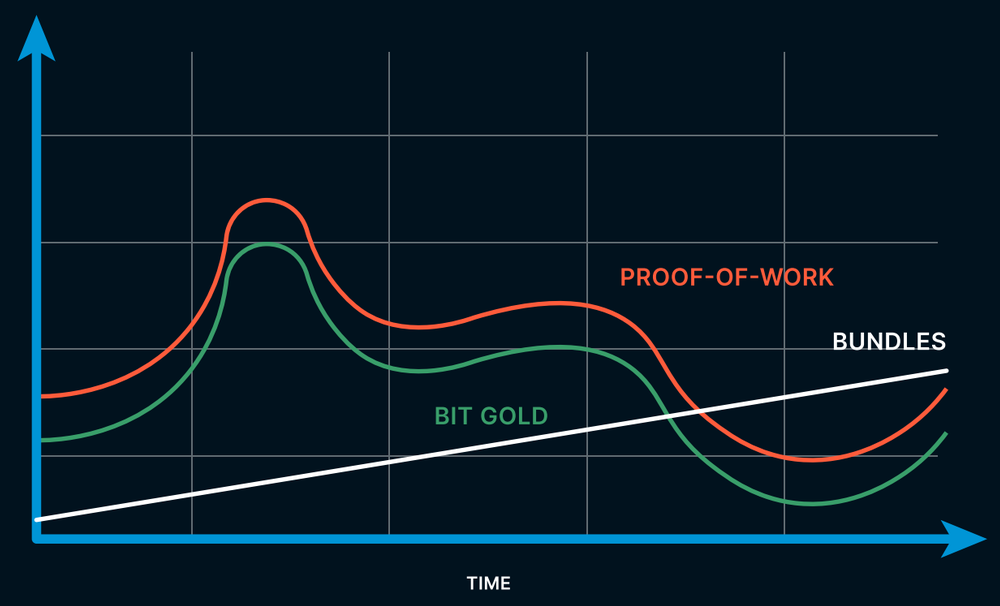
"Inferring" the cost of production is important because bit gold has no mechanism to limit the creation of money. Anyone can create bit gold by performing the appropriate computations. Without the ability to regulate issuance, bit gold is akin to a collectible:

> *...Unlike fungible atoms of gold, but as with collector's items, a large supply during a given time period will drive down the value of those particular items. In this respect "bit gold" acts more like collector's items than like gold...*
>
> Szabo, 2005

Bit gold requires an additional, external process to create fungible units of currency:

> *...[B]it gold will not be fungible based on a simple function of, for example, the length of the string. Instead, to create fungible units dealers will have to combine different-valued pieces of bit gold into larger units of approximately equal value. This is analogous to what many commodity dealers do today to make commodity markets possible. Trust is still distributed because the estimated values of such bundles can be independently verified by many other parties in a largely or entirely automated fashion.*
>
> Szabo, 2005

To paraphrase Szabo, “to assay the value of... bit gold, a dealer checks and verifies the difficulty, the input, and the timestamp”. The dealers defining “larger units of approximately equal value” are providing a similar pricing function as Dai’s “standard basket of commodities”. Fungible units are not created in bit gold when proofs-of-work are produced, only later when those proofs are combined into larger “units of approximately equal value” by dealers in markets outside the network.




Bit gold has no way to limit the supply of money. Because the supply of computations changes, individual tokens of bit gold must be assayed and bundled into larger, fungible units. This post-hoc, external process is how money is created in bit gold.

To his credit, Szabo recognizes this flaw:

> *...The potential for initially hidden supply gluts due to hidden innovations in machine architecture is a potential flaw in bit gold, or at least an imperfection which the initial auctions and ex post exchanges of bit gold will have to address.*
>
> Szabo, 2005

Again, despite not having arrived at (what we now know as) the solution, Szabo was pointing us at it: because the cost of computation changes over time, the network must respond to changes in the supply of computation by adjusting the price of money.

# Use internal markets

Szabo's dealers would have been an external market that defined the price of (bundles of) bit gold after its creation. Is it possible to implement this market within the system instead of outside it?

Let's return to Wei Dai and b-money. As mentioned earlier, Dai proposed an alternative auction-based model for the creation of bmoney. Satoshi's design for bitcoin directly evolved from b-money's auction model[7] so we quote it at length here:

> *So I propose an alternative money creation subprotocol, in which [users]... instead decide and agree on the amount of b-money to be created each period, with the cost of creating that money determined by an auction. Each money creation period is divided up into four phases, as follows:*
>
> - ***Planning.*** *The [users] compute and negotiate with each other to determine an optimal increase in the money supply for the next period. Whether or not the [network] can reach a consensus, they each broadcast their money creation quota and any macroeconomic calculations done to support the figures.*
> - ***Bidding****. Anyone who wants to create b-money broadcasts a bid in the form of where x is the amount of b-money he wants to create, and y is an unsolved problem from a predetermined problem class. Each problem in this class should have a nominal cost (in MIPS-years say) which is publicly agreed on.*
> - ***Computation.*** *After seeing the bids, the ones who placed bids in the bidding phase may now solve the problems in their bids and broadcast the solutions. Money creation. E*
> - ***Money creation.*** *Each [user] accepts the highest bids (among those who actually broadcasted solutions) in terms of nominal cost per unit of b-money created and credits the bidders' accounts accordingly.*
>
> Dai, 1998

B-money makes significant strides towards the correct market structure for a digital currency. It attempts to eliminate Szabo's external dealers and allow users to engage in price discovery by directly bidding against each other.

But implementing Dai's proposal as written would be challenging:

- In the "Planning" phase, users bear the burden of negotiating the "optimal increase in the money supply for the next period". How "optimal" should be defined, how users should negotiate with each other, and how the results of such negotiations are shared is not described.

- Regardless of what was planned, the "Bidding" phase allows anyone to submit a "bid" to create b-money. The bids include both an amount of b-money to be created as well as a corresponding amount of proofof-work so each bid is a price, the number of computations for which a given bidder is willing to perform in order to buy a given amount of b-money.

- Once bids are submitted, the "Computation" phase consists of bidders performing the proof-of-work they bid and broadcasting solutions. No mechanisms for matching bidders to solutions is provided. More problematically, it's not clear how users should know that all bids have been submitted – when does the "Bidding" phase end and the "Computation" phase begin?

- These problems recur in the "Money Creation" phase. Because of the nature of proof-of-work, users can verify the proofs they receive in solutions are real. But how can users collectively agree on the set of "highest bids"? What if different users pick different such sets, either due to preference or network latency?

Decentralized systems struggle to track data and make choices consistently yet b-money requires tracking bids from many users and making consensus choices among them. This complexity prevented b-money from ever being implemented.

The root of this complexity is Dai's belief that the "optimal" rate at which b-money is created should fluctuate over time based on the "macroeconomic calculations" of its users. Like bit gold, b-money has no mechanism to limit the creation of money. Anyone can create units of b-money by broadcasting a bid and then doing the corresponding proof-of-work.

Both Szabo and Dai proposed using a market exchanging digital currency for computations yet neither bit gold nor b-money defined a monetary policy to regulate the supply of currency within this market.

## IV. Satoshi's monetary policy goals led to bitcoin

In contrast, a sound monetary policy was one of Satoshi's primary goals for the bitcoin project. In the very first mailing list post where bitcoin was announced, Satoshi wrote:

*The root problem with conventional currency is all the trust that's required to make it work. The central bank must be trusted not to debase the currency, but the history of fiat currencies is full of breaches of that trust.*

Satoshi, 2009

Satoshi would go on to describe other problems with fiat currencies such as risky fractional reserve banking, a lack of privacy, rampant theft & fraud, and the inability to make micropayments. But Satoshi started with the issue of debasement by central banks – with a concern about monetary policy.

Satoshi wanted bitcoin to ultimately reach a finite circulating supply that cannot be diluted over time. The "optimal" rate of bitcoin creation, for Satoshi, should thus eventually be zero.

This monetary policy goal, more than any other characteristic they personally (or collectively!) possessed, was the reason Satoshi "discovered" bitcoin, the blockchain, Nakamoto consensus, etc. —and not someone else. It's the short answer to the question posed in the title of this article: Satoshi thought of bitcoin because they were focused on creating a digital currency with a finite supply.

A finite supply of bitcoin is not only a monetary policy goal or a meme for bitcoiners to rally around. It's the essential technical simplification that allowed Satoshi to build a functional digital currency while Dai's b-money remained just a fascinating web post.

Bitcoin is b-money with an additional requirement of a predetermined monetary policy. Like many technical simplifications, constraining monetary policy enables progress by reducing scope. Let's see how each of the phases of b-money creation is simplified by imposing this constraint.

## All 21M bitcoin already exist

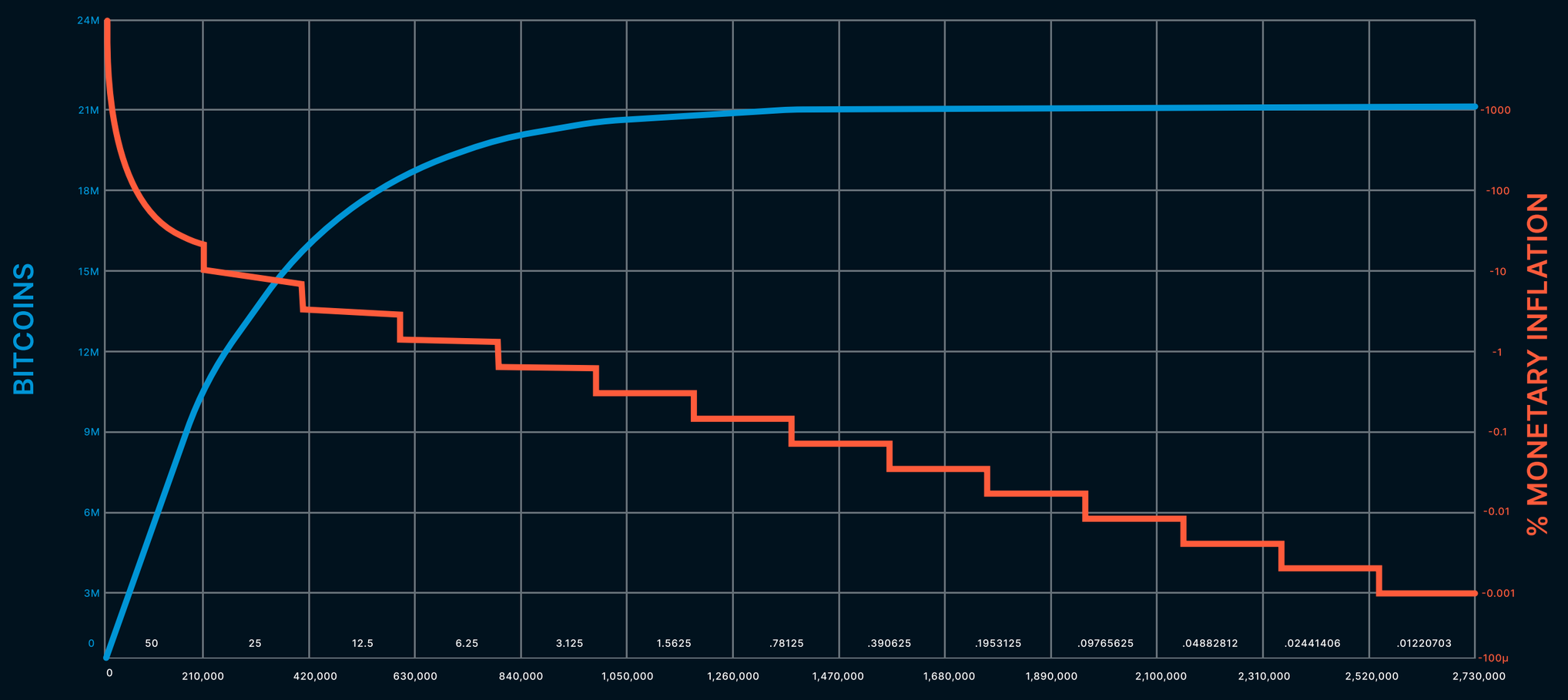
In b-money, each "money creation period" included a "Planning" phase, in which users were expected to share their "macroeconomic calculations" justifying the amount of b-money they wanted to create at that time. Satoshi's monetary policy goals of a finite supply and zero tail emission were incompatible with the freedom granted by b-money to individual users to create money. The first step on Satoshi's journey from b-money to bitcoin was therefore to eliminate this freedom. Individual bitcoin users cannot create bitcoin. Only the bitcoin network can create bitcoin, and it did so exactly once, in 2009 when Satoshi launched the bitcoin project.

Satoshi was able to replace the recurring "Planning" phases of b-money into a single, predetermined schedule on which the 21M bitcoin created in 2009 would be released into circulation. Users voluntarily endorse Satoshi's monetary policy by downloading and running the Bitcoin Core software in which this monetary policy is hard-coded.

This changes the semantics of bitcoin's market for computations. The bitcoin being paid to miners is not newly issued; it's newly released into circulation from an existing supply.

This framing is crucially different from the naive claim that "bitcoin miners create bitcoin". Bitcoin miners are not creating bitcoin, they're buying it. Bitcoin isn't valuable because "bitcoin are made from energy"—bitcoin's value is demonstrated by being sold for energy.

Let's repeat it one more time: bitcoin isn't created through proof-of-work, bitcoin is created through consensus.




Satoshi eliminated the requirement for ongoing "Planning" phases from b-money by doing all the planning up front. This allowed Satoshi to hard-code a sound monetary policy but also simplified the implementation of bitcoin.

## Bitcoin is priced through consensus

This freedom granted to users to create money results in a corresponding burden for the bmoney network. During the “Bidding” phase the b-money network must collect and share money creation “bids” from many different users.

Eliminating the freedom to create money relieves the bitcoin network of this burden. Since all 21M bitcoin already exist, the network doesn’t need to collect bids from users to create money, it merely has to sell bitcoin on Satoshi’s predetermined schedule.

The bitcoin network thus offers a consensus asking price for the bitcoin it is selling in each block. This single price is calculated by each node independently using its copy of the blockchain. If nodes have consensus on the same blockchain (a point we will return to later) they will all offer an identical asking price at each block.[8]

The first half of the consensus price calculation determines how many bitcoin to sell. This is fixed by Satoshi’s predetermined release schedule. All bitcoin nodes in the network calculate the same amount for a given block:

```
$ bitcoin-cli getblockstats <block_height> { ... “subsidy”: 6250000000, ... } # 6.25 BTC
```

The second half of the consensus asking price is the number of computations the current subsidy is being sold for. Again, all bitcoin nodes in the network calculate the same value (we will revisit this difficulty calculation in the next section):

```
$ bitcoin-cli getdifficulty { "result": 55621444139429.57, ... }
```

Together, the network subsidy and difficulty define the current asking of bitcoin as denominated in computations. Because the blockchain is in consensus, this price is a consensus price.

Users in b-money also were presumed to have a consensus “blockchain” containing the history of all transactions. But Dai never thought of the simple solution of a single consensus asking price for the creation of new b-money, determined solely by the data in that blockchain.

Instead, Dai assumed that money creation must go on forever. Individual users would therefore need to be empowered to affect monetary policy – just as in fiat currencies. This perceived requirement led Dai to design a bidding system which prevented b-money from being implemented.

This added complexity was removed by Satoshi’s requirement of a predetermined monetary policy.

## Time closes all spreads

In the “Computation” phase of b-money, individual users would perform the computations they’d committed to in their prior bids. In bitcoin, the entire network is the seller – but who is the buyer?

In the email delivery market, the buyers were individuals wanting to send emails. The pricing authority, the email service provider, would set a price that was considered cheap for individuals but expensive for spammers. But if the number of legitimate users increased, the price could still remain the same because the computing power of individual users would have remained the same.

In b-money, each user who contributed a bid for money creation was supposed to subsequently perform the corresponding number of computations themselves. Each user was acting as their own pricing authority based on their knowledge of their own computing capabilities.

The bitcoin network offers a single asking price in computations for the current bitcoin subsidy. But no individual miner who finds a block has performed this number of computations.[9] The individual miner's winning block is proof that all miners collectively performed the required number of computations. The buyer of bitcoin is thus the global bitcoin mining industry.

Having arrived at a consensus asking price, the bitcoin network will not change that price until more blocks are produced. These blocks must contain proofs-of-work at the current asking price. The mining industry therefore has no choice if it wants to "execute a trade" but to pay the current asking price in computations.

The only variable the mining industry can control is how long it will take to produce the next block. Just as the bitcoin network offers a single asking price, the mining industry thus offers a single bid – the time it takes to produce the next block meeting the network's current asking price.

> *To compensate for increasing hardware speed and varying interest in running nodes over time, the proof-of-work difficulty is determined by a moving average targeting an average number of blocks per hour. If they're generated too fast, the difficulty increases.*
>
> Nakamoto, 2008

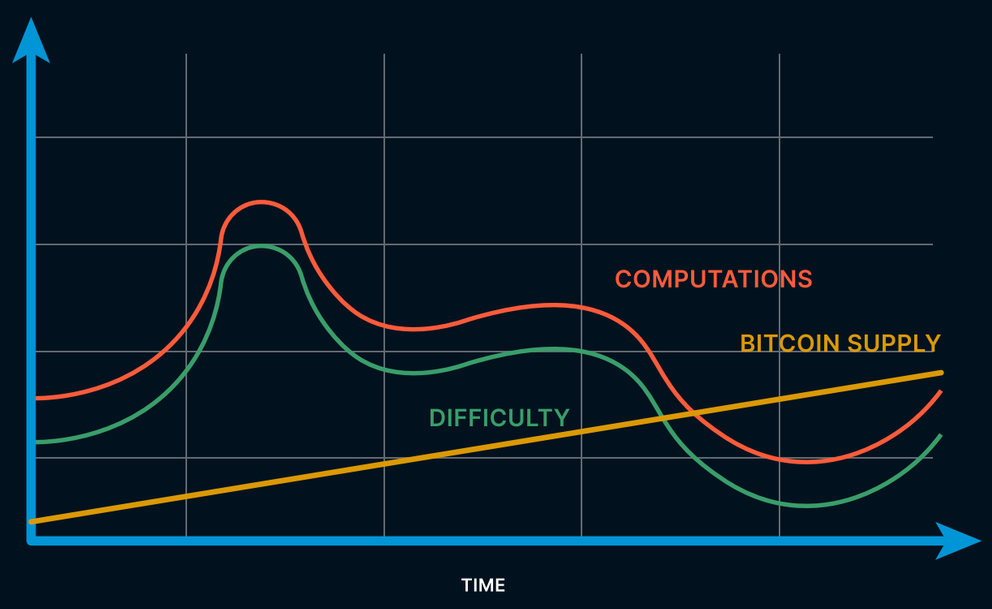
Satoshi is modestly describing the difficulty adjustment algorithm, often cited as one of the most original ideas in bitcoin's implementation. This is true, but instead of focusing on the inventiveness of the solution, let's instead focus on why solving the problem was so important to Satoshi in the first place.

Projects such as bit gold and b-money didn't need to constrain the rate in time of money creation because they didn't have a fixed supply or a predetermined monetary policy. Periods of faster or slower money creation could be compensated for through other means, e.g. external dealers putting bit gold tokens into larger or smaller bundlers or bmoney users changing their bids.

But Satoshi's monetary policy goals required bitcoin to have a predetermined rate at which bitcoin was to be released for circulation. Constraining the (statistical) rate at which blocks are produced over time is natural in bitcoin because the rate of block production is the rate at

which the initial supply of bitcoin is being sold. Selling 21M bitcoin over 140 years is a different proposition than allowing it to be sold in 3 months.

Moreover, bitcoin can actually implement this constraint because the blockchain is Szabo's "secure timestamping protocol." Satoshi describes bitcoin as first and foremost a "distributed timestamp server on a peer-to-peer basis," and early implementations of the bitcoin source code use the world "timechain" rather than "blockchain" to describe the shared data structure that implements bitcoin's proof-of-work market.[10]




Unlike bit gold or b-money, tokens in bitcoin do not experience supply gluts. The bitcoin network uses the difficulty adjustment to change the price of money in response to changes in the supply of computations.

Bitcoin's difficulty readjustment algorithm leverages this capability. The consensus blockchain is used by participants to enumerate the historical bids made by the mining industry and readjust the difficulty in order to move closer to the target block time.

## A standing order creates consensus

The chain of simplifications caused by demanding strong monetary policy extends to the "Money creation" phase of b-money.

User-submitted bids in b-money suffer from "nothing at stake" problem. There is no mechanism to prevent users from submitting bids with a huge amount of b-money for very little work. This requires the network to both track which bids have been completed and only accept

the “highest bids…in terms of nominal cost per unit of b-money created” in order to avoid such nuisance bids. Each b-money participant must track an entire order book worth of bids, match bids with their subsequent computations, and only settle such completed orders with the highest prices.

This problem is an instance of the more general problem of consensus in decentralized systems, also known as the “Byzantine generals” or sometimes the “double-spend” problem in the context of digital currencies. Sharing an identical sequence of data among all participants is challenging inside an adversarial, decentralized network. Existing solutions to this problem – so-called “Byzantine-fault tolerant (BFT) consensus algorithms” – require previous coordination among participants or a supermajority (>67%) of participants to not behave adversarially.

Bitcoin doesn’t have to manage a large order book of bids because the bitcoin network offers a single consensus asking price. This means bitcoin nodes can accept the first (valid) block they see that meets the network’s current asking price – nuisance bids can easily be ignored and are a waste of a miner’s resources.

Consensus pricing of computations allows the matching of buy/sell orders in bitcoin to be done eagerly, on a first-come, first-served basis. Unlike b-money, this eager order matching means that bitcoin’s market has no phases – it operates continuously, with a new consensus price being calculated after each individual order is matched (block is found). To avoid forks caused by network latency or adversarial behavior, nodes must also follow the heaviest chain rule. This greedy order settling rule ensures that only the highest bids are accepted by the network.

This combination eager-greedy algorithm, where nodes accept the first valid block they see and also follow the heaviest chain, is a novel BFT algorithm which rapidly converges on consensus about the sequence of blocks. Satoshi spends 25% of the bitcoin white paper demonstrating this claim.[11]

We established in previous sections that bitcoin’s consensus asking price itself depends on the blockchain being in consensus. But it turns out that the existence of a single consensus asking price is what allows the market for computations to eagerly match orders, which is what leads to consensus in the first place!

Moreover, this new “Nakamoto consensus” only requires 50% of participants to not be adversarial, a significant improvement on the prior state of the art. A cypherpunk like Satoshi made this theoretical computer science breakthrough, instead of a traditional academic or industry researcher, because of their narrow focus on implementing sound money, rather than a generic consensus algorithm for distributed computing.

# IV. Conclusion

Satoshi saw b-money as a powerful framework for building a digital currency but one that was incomplete because it lacked a monetary policy. Constraining b-money with a predetermined release schedule for bitcoins reduced scope and simplified implementation by eliminating the requirement to track and choose among user-submitted money creation bids. Preserving the temporal pace of Satoshi's release schedule led to the difficulty adjustment algorithm and enabled Nakamoto consensus, widely recognized as one of the most innovative aspects of bitcoin's implementation.

There is a lot more to bitcoin's design than the aspects discussed so far. We have focused this article on the "primary" market within bitcoin, the market which distributes the initial bitcoin supply into circulation.

The next article in this series will explore the market for bitcoin transaction settlement and how it relates to the market for distributing the bitcoin supply. This relationship will suggest a methodology for how to build future markets for decentralized services on top of bitcoin.

# Acknowledgements

I've been ranting about bitcoin and markets for years now and must thank the many people who listened and helped me sharpen my thinking.  In particular, Ryan Gentry, Will Cole and Stephen Hall met with me weekly to debate these ideas. I would not have been able to overcome countless false starts without their contributions and their support. Ryan also helped me begin talking about these ideas publicly in our Bitcoin 2021 talk. Afsheen Bigdeli, Allen Farrington, Joe Kelly, Gigi, Tuur Demeester, and Marty Bent, have all encouraged me over the years and provided valuable feedback. I must also apologize to Allen for turning out to be such a lousy collaborator. Finally, Michael Goldstein may be better known for his writing & memes, but I'd like to thank him for the archival work he does at the Nakamoto Institute to keep safe the history of digital currencies.

# Footnotes

*[1] The title of this series is taken from the first telegraph message in history, sent by Samuel Morse in 1844: "What hath God wrought?".*

*[2] Bitcoin: A Peer-to-Peer Electronic Cash System, available: https://bitcoin.org/bitcoin.pdf*

*[3] Pricing via Processing or Combatting Junk Mail by Dwork and Naor available: https://www.wisdom.weizmann.ac.il/~naor/PAPERS/pvp.pdf*

*[4] Despite originating the idea, Dwork & Naor did not invent "proof-of-work"—that moniker was provided later in 1999 by Markus Jakobsson and Ari Juels.*

*[5] Hal Finney's RPOW project was an attempt at creating transferable proofs-of-work but bitcoin doesn't use this concept because it doesn't treat computations as currency. As we'll see later when we examine bit gold and b-money, computations cannot be currency because the value of computations changes over time while units of currency must have equal value. Bitcoin is not computations, bitcoin is currency that is sold for computations.*

*[6] At this juncture, some readers may believe me dismissive of the contributions of Dai or Szabo because they were inarticulate or hand-wavy on some points. My feelings are the exact opposite: Dai and Szabo were essentially right and the fact that they did not articulate every detail the way Satoshi subsequently did does not detract from their contributions. Rather, it should heighten our appreciation of them, as it reveals how challenging the advent of digital currency was, even for its best practitioners.*

*[7] Dai's b-money post is the very first reference in Satoshi's white paper, available: http://www.weidai.com/bmoney.txt*

*[8] There are two simplifications being made here:*

*a. The number of bitcoin being sold in each block is also affected by the transaction fee market, which is out of scope for this article, though lookout for subsequent work.*

*b. The difficulty as reported by bitcoin is not exactly the number of expected computations; one must multiply by a proportionality factor.*

*[9] At least not since the bad old days when Satoshi was the only miner on the network.*

*[10] Gigi's classic Bitcoin is Time is a great introduction to the deep connections between bitcoin and time. https://dergigi.com/2021/01/14/bitcoin-is-time/*

*[11] Satoshi blundered both in their analysis in the white paper and their subsequent initial implementation of bitcoin by using the"longest chain" rule instead of the "heaviest chain" rule.*