UMEÅ UNIVERSITY
Department of Computing Science

May 25, 2019

Emergent Routing Strategies in the Lightning Network

Abstract

In payment channel networks, such as the Bitcoin native Lightning Network, the routing nodes receive a fee as compensation for displaced liquidity, time value of money and operational costs. Currently this fee is manually set procuring sub optimal profits to the node operator. The network dynamics may be modeled as a graph and each node as an actor utilizing strategies in fee price setting, preferential attachment, timing, allocation and funding akin to game theoretic models. Further assuming rational actors and strategy propagation are proportional to population suggest similar methodology to evolutionary game theory where a strategy's fitness will emerge as a fraction of population size.

A simulation study was performed where strategies were played against each other to find emergent equilibria under competitive market pressure. Where such equilibrium may lie have further consequences for the network in form of total throughput, routing cost and robustness. This study suggest ... strategies and a robust network topology with short average paths will emerge from free competition.

Name	John-John Markstedt*
Cinnober Supervisor	Oskar Janson
University Supervisor	Jerry Eriksson
Examinor	Henrik Björklund

* whom correspondence should be addressed. E-mail: john-john@markstedts.org

Acknowledgements

I would first like to thank Cinnober for giving me this opportunity and specifically my thesis advisor Oskar Janson for his dedication and vast knowledge in Bitcoin, risk and financial markets. Further I would also like to thank my university supervisor Dr. Jerry Eriksson for his many suggestions and inducing academic rigor into the thesis. Along with the University Examiner Henrik Björklund and the many teachers at Umeå University for these last 5 years.

A special thanks to my parents for always being there and instilling a curiosity in me and an inspiration for science.

Contents

Problem Description

Background

Cinnober is a provider of IT solutions to the infrastructure providers of the global financial industry, including exchanges and clearing houses. Cinnober has solutions from price discovery, trading, clearing to settlement of financial securities.

Bitcoin is a peer-to-peer electronic cash system [66]. The decentralized nature of the system limits the performance to the weakest node - consequently limiting the transaction capacity.

Multiple solutions to reduce the burden on the Bitcoin nodes have been suggested [29**?** , 77**?**]. The solution receiving most attention is the Lightning Network relying on payment channels.

In a payment channel, bitcoin is committed by a Bitcoin transaction and locked up between two parties. Once the transaction is registered, parties in the channel with positive balance may send it to the other party without the need to create a new transaction on the Bitcoin layer. The channel is strictly bidirectional and have a fixed capacity. Each party may settle the channel balance at any point by creating a closing Bitcoin transaction, settling the balance. Usage of payment channels may be very convenient with parties that transact often.

Many payment channels can be aggregated into a network. There is an ongoing effort to build such a network named The Lightning Network. Where a party without a direct channel to another party could route through multiple channels if enough liquidity exists between them. If such a scheme becomes viable in practice; only a small fraction of all transactions would need to be settled on the Bitcoin blockchain. Thus increasing the capacity significantly.

How the routing nodes should operate is still vastly unexplored. The node provides liquidity to allow routing payments and procures fees whenever the channels are used in a payment route. It is likely that a market for liquid channels will emerge, however strategically well allocated liquidity may earn much more than poorly allocated liquidity. The nodes may choose who to open channels with, what fee to set, how much liquidity should be locked up on each channel and when to re-balance channels.

A simulation environment was implemented, along with operational strategies. The strategies were then combined and simulated to see how well they perform against each other.

Aim

This thesis aims to address following three questions:

1. How should a Lightning Network routing node behave to maximize earnings?
2. What network topology will the Lightning Network converge towards as routing nodes becomes increasingly efficient?
3. How will the revenue be distributed between the routing nodes? Will the low barrier to entry reduce the market to low margins?

Related work

Little to nothing in the literature that addresses the Lightning Network Routing Strategies has been found. René Pickardt has made some early attempts to find suitable channel parties with traditional graph heuristics but is yet incomplete and unpublished [75]. Bitmex made an empirical study where different fees were set on Mainnet channels and corresponding returns where retrieved [79][1] . Further conversation with people in the community confirms that this problem set is known yet largely unexplored.

The actual routing in LN has been researched [31] and implementations are currently in use [22, 52, 8, 62]. Further efficient sharing of routing tables have been proposed [41].

A similar problem at first glance, the estimation of the on-chain Bitcoin Network Fee has been covered extensively [63, 48]. However it's neither applicable, nor is it even similar as a problem set. The Bitcoin Fee problem is far simpler as the competing transaction fees(mem-pool) are known at all times and estimating a fee too low can be corrected by

[1] Note that this study was published 27th of Mars, 2019, mid-thesis.

fee bumping[2] or *child-pay-for-parent*[3] implementations.

As the Lightning Network is indeed a network and many problems are yet another incarnation of already solved graph problems. A nodes position in a network has been studied and specifically the measurement of Betweenness Centrality, as introduced by Freeman in 1977 [25], lays a sound basis for fee estimation. Calculating paths in graphs are as old as the computing field itself with Floyd-Warshall, Johnson [49] and Dijkstras algorithms all being appropriate here. The attributes of the network as strategies emerge have consequences on throughput and robustness have also been researched in the field of topology. Especially that of scale free networks by Barabasi and Albert [17] is of concern here.

Emergence of equilibria as the product of strategies of competing actors is neither new or rare in the fields of evolutionary biology nor economics. Ideas as the Standard Price Theory, Evolutionary Stable Strategies and more widely game theory has aided in the formulation of the thesis and construction of the simulations.

Lastly, Bitcoin and off-chain proposals have opened a plethora of possible research topics in which plentiful are ongoing and active.

Structure

This thesis is divided into eight chapters, including this one. The first three chapters give a wide overview of money, Bitcoin and the Lightning Network. Chapters 6 and 7 aim to answer the thesis question by formulating, evaluating and simulating routing strategies. The last chapter 8 discusses the results, draws conclusions and elaborates on future work.

1. **Introduction** introduces the thesis.
2. **Macroeconomics** gives a wide background to currencies and systems built on trust.
3. **The Bitcoin Network** chapter introduces the basic Bitcoin architecture, answers how trust is solved and describes Bitcoin script as it allows further development on top of the base protocol.
4. **Scaling Bitcoin** describes historical attempts to scale Bitcoin and the merit of a wide array of scaling solutions.
5. **The Lightning Network** chapter introduces the most fundamental parts of the Lightning Network protocol.
6. **Evaluation** reduces the Lightning Network to a manageable problem set, formulates strategies and measurements and suggests how these strategies may be evaluated.
7. **Results** show the simulated results of the strategies and their effect on network as a whole in terms of throughput, efficiency and robustness.
8. **Discussion and Conclusions** processes the results, draws conclusions and ties them into the wider ecosystem. The consequences and viability of the Lightning Network are elaborated and future work is discussed.

[2] Broadcasting a new transaction spending the same UTXO with a higher fee.

[3] Using the unconfirmed parent UTXO in a new transaction, a miner must include the parent tx to be able to mine the child tx and receive the child tx fee.

Macroeconomics

Bitcoin was not created in a vacuum and to understand the design of Bitcoin and by extension the Lightning Network one would benefit from understanding money and systems of trust. Nakamoto encoded the string "the times 03/jan/2009 chancellor on brink of second bailout for banks" [67, 5] in the coinbase transaction of the genesis block as an alleged comment on the current banking system as well as a timestamp. He later cemented this view with multiple forum posts while he still was active [70, 68].

Origin of money

On the fringes of our specie's history barter became the first medium in which trade became viable, removing the risk from otherwise delayed reciprocity. It is possible for trade to be mutually beneficial as Nick Szabo so elegantly puts it,

> "individuals, clans, and tribes all vary in their preferences, vary in their ability to satisfy these preferences, and vary in the beliefs they have about these skills and preferences and the objects that are consequent of them, there are always gains to be made from trade." [89]

Barter is by itself quite limited. Consider a system S consisting of n commodities, the amount of possible exchanges between commodities in S is $n * n$. When n eventually increases, the exchange pairs explode exponentially. This makes it difficult to assess fair pricing and decreases the coincidence of a trade[1]. The economist Carl Menger described it as inevitable for money to evolve from a sufficient volume of commodity barter [60]. If the same system S is considered with money - the possible exchange pairs would be reduced to only n pairs.

Characteristics of money

Out of commodity barter; money emerged. History has provided many peculiar forms of money. The Rai stones of Yap islands, the Wampum shells in North America[2] and Aggry beads in Africa to name a few [89]. Although many different types of commodities have acted as money during certain times in history there are some characteristics that seem favorable and recur. The Federal Reserve Bank of Saint Louis lists them as [73][3]:

- **Durability**. The ability to remain intact over time.
- **Portability**. The ability to move across physical distance.
- **Divisibility**. The ability to be divided across scale, to be used in any size of value transaction.
- **Uniformity**. The ability to use one unit interchangeably with another. Also referred to as **fungibility**.
- **Scarcity**. The ability to be scarce over time. Usually quantified by stock-to-flow. The amount of new supply in relation to already existing supply.
- **Acceptability**. The likelihood of being accepted in trade by others.

Many of the previous mentioned monies have expressed many of these characteristics well. Changes in these characteristics have also led to their downfall. Rai stones are carved limestone which are not native to the Yap islands, when outsiders started to bring in these on large ships it quickly changed their stock-to-flow ratio for the worst [82]. Aggry beads met the same fate when Europeans, with efficient means to produce them, started to export them to West Africa. Similarly, modernized shell fishing ruined the Wampum shells function as money [89].

Rare metals have held these attributes since the beginning of history and in many ways still hold them today. Gold especially stands out with it's very low stock-to-flow ratio.

[1] The coincidence of finding another party who is looking to exchange the same commodity pair you are but in reverse order.

[2] The Wampum shells were legal tender as recently as 1710 in North Carolina and long into the 1600s in New England.

[3] Note that only the characteristics are from the Federal Reserve, the descriptions are not.

Bearer promissory notes

Although gold emerged as the commodity best suited as money it still had two major problems,

1. Gold is unsuitable for small transactions due to the limit in divisibility.
2. Gold is expensive to carry around and protect.

Private and central banks began to issue bearer promissory notes to the owners of the underlying asset it stored(e.g. Figure 1). The asset could then be retrieved for the note on demand. The notes themselves could then be traded instead of the underlying gold. This solved both the above problems, notes are easy to carry, conceal and could be minted in very small amounts.

Figure 1: *The private bank 'Stockholms Enskilda Bank'(SEB) issued bearer promissory note. It promises to pay 10 crowns to the bearer upon demand in gold. The exchange rate as per the Scandinavian Convention(Sweden, Denmark 27 may 1873, Norway 1 april 1877)[56] was 1 kilogram gold per 2480 crowns or 1 crown per 0.403g gold [55].*

By the late 19th century almost all major currencies was under the gold standard as seen in Table 1.

Nation	Currency	Period	Years
France	Franc	1814-1914	100
Netherlands	Guilder	1816-1914	98
G.Britain	P. Sterling	1821-1914	93
Switzerland	Franc	1850-1936	86
Belgium	Franc	1832-1914	82
Sweden	Kronor	1873-1931	58
Germany	Mark	1875-1914	39
Italy	Lira	1883-1914	31

Table 1: *Major European nations periods under the gold standard as composed by Ferdinand Lips [57] from 1975 Pickís Currency Yearbook data.*

Bank runs

The introduction of the promissory note solved the two previously mentioned problems but also introduced trust.[4]

Although the gold deposits could be regularly audited by a third party there is little to no guarantee that the bank in question wouldn't issue more notes than gold it holds in reserve. Banks utilizing this sort of practice could go on for a very long time before getting caught. Consider a bank that issue twice as many notes as it holds gold in reserve. It would require 50% of it's notes to be demanded before the bank would default.

There has been many so called bank runs in history. The public gets suspicious about the bank and the trust disappears - leading to massive withdrawals in a short time span. Even if a bank is technically solvent, having assets tied up in different ventures, they could fail to deliver on the notes promises. This can also be viewed as a negative spiral, people start to withdraw, increasing the risk of default, more people start to withdraw due to the new higher risk. R.H. Patterson elaborates in detail the bank run in Great Britain in 1866 leading to the default of Overend, Gurney and Company and the behavior under panic:

> "When a Panic occurs, a much more serious home-drain is produced upon the bank. At such time cheques fall somewhat into disrepute, so that merchants in some cases require payment in cash. The public also, to some extent, take to hoarding[...]. But a very large part of the drain upon the Bank of England in the form of hoarding ,[...],is made by other banks: for these banks being liable to unusual demands on the part of their customers, have to keep in hand a larger stock of money than usual" [74]

Overall most larger institutions stayed solvent and bank runs is more an exception than a rule during long stretches of time. Since all major currencies were denominated in gold the threshold for global trade sank. During the gold standard era the economy boomed with trade and is often known under the French term 'La Belle Epoque' or 'Beautiful era'.

[4] Note that this a much wider problem than for only promissory notes. This trust model is what makes banks in the first place.

Elasticity of money

It all came crashing down with the outbreak of the first world war. Nations central banks began to issue more notes than gold in reserve to fund the war effort. It effectively ended the gold standard and countries not affected by the war followed shortly after.

As response to the Great Depression U.S President Franklin Roosevelt issued Executive order 6102 confiscating all gold coins, bullion and certificates and banned trade in gold [80]. The U.S Dollar was devalued the following year from $20.67 per troy ounce to $35 under the Gold Reserve Act [81] to enable spending their way out of recession.

A new economic era began following ideas of Maynard Keynes of economic interventionism and monetary policies set to aid growth. Keynes rebutted the classical idea that 'supply creates its own demand' and suggested that aggregated demand and aggregated supply may get stuck in an equilibrium with high unemployment [51]. This would motivate government intervention to increase the aggregated demand. While demand may be altered by a government in multiple ways the far most effective one is by altering the money supply[5]. Although the dollar was still technically redeemable for gold the elastic money supply made it possible to steer the economy by means of monetary policy.

The dollar was re-pegged multiple times between 1968-1971 as part of the Nixon chock [23]. In 1971 Nixon ended the Bretton Woods agreement [23], the international agreement of exchange rates between currencies, rendering the dollar and all currencies pegged to the Dollar true fiat currencies. After the initial shock the free floating currencies failed to keep pace with gold due to money supply increases causing inflation(see Figure 2).

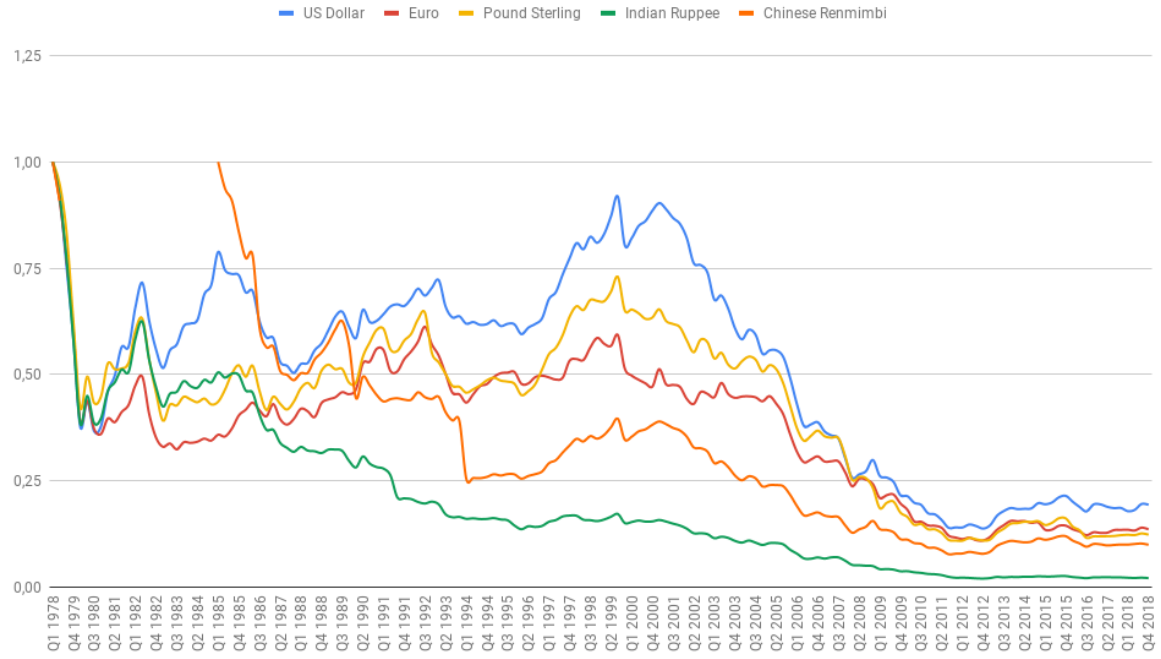


Figure 2: *The relation of major currencies against its value in gold since the end of the Bretton Woods agreement to today. The graph is composed with data from the World Gold Council [7].*

[5]The increased supply can be used directly by government but is usually used by lowering the funds rate, increasing investments and thus demand. Quantitative easing is another term used to describing this method.

Nation	Currency	Period	Highest monthly Inf.	Avg. daily Inf.	Type of index
Hungary	Pengő	Aug. 1945 - Jul. 1946	4.19×10^{16}%	207%	Consumer
Zimbabwe	Dollar	Mar.2007 - Mid-Nov.2008	7.96×10^{10}%	98.0%	Implied Exchange
Yugoslavia	Dinar	3 Apr.1992 - Jan.1994	313,000,000%	64.6%	Consumer
Germany	Papiermark	Aug. 1922 - Dec.1923	29,500%	20.9%	Wholesale
China	Yuan	Oct.1947 - Mid-May.1949	5,070%	14.1%	Wholesale
Venezuela(§)	Bolívar	Jan. 2018 - ongoing	-	4.1%	-

Table 2: *Hyperinflation of selected fiat currencies. From the Hanke-Krus Hyperinflation Table [44]. §The Venezuela numbers [43] are only a projection by the International Monetary Fund and the dates and rate will most likely change with more accurate data and when the inflationary period is over.*

Hyperinflation

The major currencies seen in Figure 2 are far from worst in their ability to hold value. Fiat currencies are highly dependent on its authority and their willingness to increase the money supply and history provides us with some catastrophes as seen in Table 2.

During the last 250 years hyperinflation[6] has occurred at least 54 times [44, 43]. Many countries going through multiple inflationary periods, e.g. China 43-45, 47-49 and Georgia 92, 93-94. Hyperinflation devastate all savings and effectually letting savings subsidize the newly printed currency. The inflation narrows the time horizon of trade, since capital will go worthless in a short while, making it very difficult to formalize any long term investment or running a normal economy.

Trust

The economic system built upon fiat currencies is based on the trust in the issuing body not to increase the money supply excessively. It's almost been common knowledge that value is lost in currencies, moving money away from currency and bonds into stock and real estate. Modern currencies have been decoupled from the underlying reasons of its nascence, leading up to the failures of 08 coinciding with Nakamoto's publication completing full circle.

Relevancy

In many ways it helps to think of bitcoin as both a bearer instrument and as a monetary policy and how the system is designed as such. For further reading these recommendations are good starting points:

- Nick Szabo's Enumerated and above cited Shelling out [89, 87].
- Saifedean Ammous's book The bitcoin standard. [82]
- Waren Weber's paper A bitcoin standard. [91]

[6] There is no official definition of what constitutes hyper inflation but a rule of thumb seem to be over 50% annual inflation.

The Bitcoin Network

Bitcoin is a peer-to-peer electronic cash system without any trusted third party and was described by Satoshi Nakamoto in October 2008 [66]. The name Bitcoin refers to both the operating network of nodes and the system's native currency. The network and protocol are usually refereed to with an upper case B, Bitcoin, and the currency with a lower case b, bitcoin.[1]

Digital Signatures

Part of the solution is provided by one way asymmetric encryption utilizing the `secp256k1` elliptic curve. Asymmetric encryption or public-key encryption allows the generation of key pairs with the ability to derive a public key from a private key without the ability to derive the private key from the public key and the ability to prove ownership of the private key through signatures without disclosing the private key. Ownership and expenditure of bitcoin are proved by signing a previous transaction output payable to the public key derived bitcoin address[2] with the accompanying private key.[3]

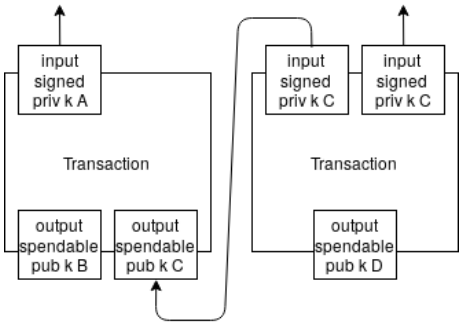


Figure 1: *A simplified illustration of two transactions.*

Bitcoin is essentially a long ledger of inputs and outputs. Note that the whole output must be spent in the same transaction and usually there is an output leading back to an address the spender controls with the 'change'. The sum of all the inputs and the sum of all the outputs usually do not line up, the differential is the implicit miners fee the miner receives if the transaction is included in a block.

The Double-Spending problem

While digital signatures are able prove ownership of bitcoin, there is nothing to stop the owner of an amount of bitcoin to sign two different transactions spending the same transaction output and broadcasting them to different parts of the network. This is called the Double-Spending problem[4] which is a variation of the more general Byzantine Generals Problem [**?**].

The network reaches a coherent global view over the transaction history by a method known as *Proof-of-work*. The Bitcoin *Proof-of-work* algorithm is very similar to that of hashcash as it uses a hash-based cost function which requires a probabilistic amount of work and that the proof of the work has a fixed cost to verify [15][5].

The *Proof-of-work* algorithm is directly tied into the bitcoin transaction history data structure composite of a chain of blocks as seen in Figure 2.

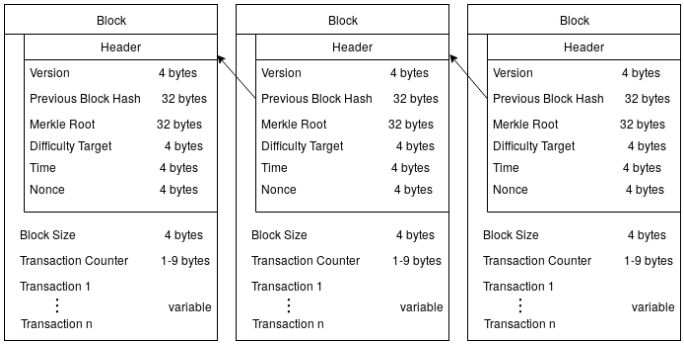


Figure 2: *Three blocks linked by the hash of the previous block's header(blockchain). The content in the header is immutable and can't be replaced without redoing the Proof-of-work.*

[1] While it's convention to not capitalize currencies in the English language, it's not widely followed in mainstream or semi-mainstream outlets. Also the plural form of currencies is equivalent to the singular. The convention will be followed in this report.

[2] A bitcoin address is derived from the public key by hashing the public key with both `SHA256` and `RIPEMD160`

[3] This part of the solution has been suggested many times before, earliest 1983 [**?**]

[4] At least known before 1993, mentioned in multiple articles in Advances in Cryptology - CRYPTO '93 e.g brands [**?**].

[5] It's similar to hashcash in both being hash-based with a cost function that is non-interactive, publicly auditable, trapdoor-free and have an unbounded probabilistic cost [15]. Bitcoin uses the double `SHA256` hash function.

The 'work' part of the algorithm is to find a sufficiently small hash over the block's header. A hash is valid if it's smaller than a number imposed by a *difficulty target*. There is no way to reverse engineer a sufficient hash and the algorithm repeatedly hashes the block header, changing the nonce, until a hash is found. Consensus is reached by regarding the longest chain of work to be the valid chain. The blocks are in a sense chained through these hashed headers and each additional block adds cumulative work on the blocks below reducing the chance of ever being reverted. The difficulty target adjusts every 2015th[6] block [4] as part of the consensus algorithm and forces the average time between blocks to always be ten minutes. Running the bitcoin *Proof-of-work* algorithm is known as *mining* as per the similarities with the finding and minting of gold coins.

Merkle Tree

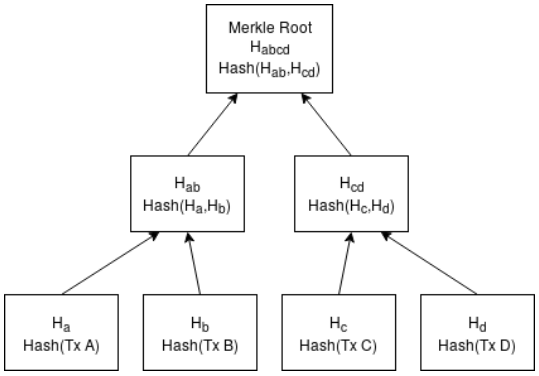


Figure 3: *A merkle tree resulting from four transactions A, B, C, D.*

The content in the block header is practically immutable since a change would make the hash invalid. The transactions are not in the header but are still immutable by the use of a Merkle tree(see Figure 3). A Merkle tree is a binary tree in which the parent is the hash of its two children. The root hash of the tree is included in the header and if any transaction were to change it would also cause the root hash to change rendering each transaction immutable by implication. Bitcoin uses the double `SHA256` as the Merkle tree hash function.

Incentive

The miners are incentivized by receiving the transaction fees from the transactions included in the mined block as well as a subsidy. The subsidy is on the form of a coinbase transaction which has only outputs, this is how new bitcoin is minted. Every fourth year the subsidy halves and by 2140 it will be zero. The miner must follow the consensus rules and mined blocks not following the rules will not be propagated through the network causing the miner to loose out on the block reward.

A miner, or a miner cartel, controlling more than 50% of the hashing power could technically double-spend transactions by mining blocks on a secret chain A, instead of the public chain B, without broadcasting them to the greater network and simultaneously spending outputs OB on chain B. When the chain A is broadcast and if it contains more work than B the outputs OB would be invalidated and never have transpired per the consensus rules. A huge problem with such an attack is that the confidence in the network would decrease causing the value of the 'stolen' bitcoin outputs OA to be close to worthless. The attacker would still own a lot of specialized hardware, only capable of running the `SHA256` hashing algorithm, that is worthless without utility outside the network. In a sense such an attack would cause mutual destruction for both the network and the attacker.

Script

Transactions utilize a forth-like polish-notation stack based execution scripting language [12]. Each transaction output consist of a locking script[7]. In order to spend an output a valid unlocking script must be provided in the input to the spending transaction that solves the locking script[8]. Validation is performed by executing the unlocking script and locking script sequentially as seen in Figure 4.

```
 OP_1    OP_2 OP_ADD OP_3 OP_EQUAL
unlock             lock
```

Figure 4: *A simple predicate. Evaluated from left to right (1 2 +) -> 3 and (3 3 =) -> true. This lock script can obviously be solved by anyone and should not be used with real bitcoin.*

The two scripts added together forms a predicate. If the predicate is true the transaction is

[6] This is off by one bug, 2016 blocks is two weeks

[7] Also refereed to as scriptPubKey. Locking script is a broader term and technically scriptPubKey scripts $\subseteq$ locking scripts.

[8] scriptSig, witness $\subseteq$ unlocking scripts.

valid. Nakamoto considered naming the language predicate but went with script to be more inclusive to a broader audience [69].

Pay to Public Key Hash

The Pay-to-Public-key-hash or P2PKH for short was the standard script for a long time. It allows only the owner of the private key to the public key to spend the output.

```
<Sig><Pub_Key>
    unlock

OP_DUP OP_HASH160 <Pub_Key_Hash>
OP_EQUALVERIFY OP_CHECKSIG
    lock
```

Figure 5: *The P2PKH locking and unlocking scripts.*

The P2PKH pushes the signature produced by the private key and the public key to the stack. The public key gets duplicated and the top one gets hashed. The `OP_HASH160` is a double hash and first hashes the key with `SHA256` and then `RIPEMD160`. This is done to hide the public key until expenditure which may be useful if, for example, the ECDSA would break and private keys could be reversed. In the early days of the network this obfuscation technique wasn't used, for example the first transaction between Nakamoto and Finney did not hide the public key [35]. The `OP_EQUALVERIFY` pops the two public key hashes and terminates the script with false if they are not equal. The `OP_CHECKSIG` verifies that the signature is indeed a match with the public key and returns true.

Pay to Script Hash

The Pay-to-Script-Hash or P2SH is a more flexible transaction than the P2PKH and was introduced with BIP016[9] [10]. It allows the locking script to only be the hash of the redeemable script. If a long script with many public keys is considered as seen in Figure 6.

```
0 <Sig A> <Sig B>
    unlock

2 <Pk_A><Pk_B><Pk_C> 3 OP_CHECKMULTISIG
    lock
```

Figure 6: *Showing a multisig transaction which require to two signatures out of three to be spent. Note the `0` in the unlock script, it's there due to a bug with `OP_CHECKMULTISIG` which pops an extra item from the stack. If not for the `0` it would try to pop an empty stack. The dummy value must be a `0` with BIP 147 compliant node implementations [53].*

This multisig transaction(in Figure 6) could be rewritten as a P2SH by hashing the locking script with `SHA256` and `RIPEMD160` and utilizing it as seen in Figure 7.

```
0 <Sig A> <Sig B> <redeem script>
    unlock

OP_HASH160 <redeem script hash>
OP_CHECKVERIFY
    lock
```

Figure 7: *The P2SH transaction which is essentially equivalent to the multisig in Figure 6. Note that 'redeem script' corresponds to the lock script in Figure 6.*

Converting to a P2SH transaction does two things:

- It shifts the fee burden from the sender to the recipient. The locking script is shorter, the unlocking script is longer.
- All unspent UTXOs must be kept in the RAM of the node, shifting the burden, thus reducing the node RAM load.

P2SH also allows the script to be used as an address as per BIP013 [9]. By simply sending to the script address one could potentially fund any type of complex transaction with no added complexity to the sender.

Even though Script is not Turing complete it allows many quite advanced systems on top of it, transaction output scripts are sometimes referred to as 'smart contracts'[10].

[9] Bitcoin Improvement Proposal

[10] First formally defined by Nick Szabo in 1994 [88]

```
IF
  IF
    2
  ELSE
    <30 days> CHECKSEQUENCEVERIFY DROP
    <Pubkey D> CHECKSIGVERIFY
    1
  ENDIF
  <Pk A><Pk B><Pk C> 3 CHECKMULTISIG
ELSE
  <90 days> CHECKSEQUENCEVERIFY DROP
  <Pubkey D> CHECKSIG
ENDIF
```

Figure 8: *A more complex multisignature locking script involving timelocks.*

One example of a more advanced script is this multisignature locking script with timelocks seen in Figure 8. The script consists of three clauses, the clause executed is determined by the unlocking script since the condition to `IF` is the top item on the stack, not the following item as in most languages. The output could be spent by either clause being triggered:

- 2 out 3 of the A, B, C signatures.
- 30 days passed, D signature and 1 out of A, B, C signatures.
- 90 days passed and D signature.

Relative lock time

The script(Figure 8) uses a construction known as a relative time lock and uses the `OP_CHECKSEQUENCEVERIFY` opcode in the bitcoin scripting language [37]. The relative lock time will only become true after the defined time since the parent transaction was included in a block has passed. To avoid the block miners to become oracles of time, the median time of the eleven last blocks are used as the lock time(BIP113 [50]). The `OP_CHECKSEQUENCEVERIFY` was activated by BIP112 [26]. The block time is therefor approximately an hour behind real time.

The Lightning Network utilize many complex transactions and more will be seen in chapter 5.

Network

The Bitcoin network consists of nodes complying with the consensus protocol. In the beginning only the reference implementation available but with the growth of popularity others have implemented independent implementations [4, 27, 54]. For the most part, they validate and propagate transactions and blocks.

The mining component of nodes has mostly been decoupled from ordinary nodes with the advent of ASIC[11]. Mining nodes still do everything as an ordinary node does as it needs the transaction history to construct valid blocks. Non mining nodes validate and propagate blocks and transactions across the network. It allows some benefits in constructing transactions but mainly allows the operator to validate that the network operates as expected instead of trusting that it does.

Construction of transactions does not require running a node and many bitcoin wallets do not operate as a node.

[11] Application Specific Integrated Circuit, in this case hardware that is built only to run `SHA256`.

Scaling bitcoin

There is no secret that the Bitcoin blockchain will not be able to scale on its own. When Nakamoto first published the whitepaper on the cryptographic mailing list his first response was in fact that such a solution won't scale very well [32].

Andreas Antonopoulos has drawn similarities between the scaling of bitcoin with the scaling of the internet:

> "Bitcoin is failing to scale. If we're really lucky, bitcoin will continue to fail to scale gracefully for 25 years, just like the internet." [13]

This quote is accurate as Usenet used a *Store-and-forward* method for content to reach every server in the network. *Store-and-forward* does not scale very well and is somewhat similar to the base Bitcoin protocol. The internet moved away from *Store-and-forward* to more direct routing and Bitcoin is too, allowing transactions to happen off-chain.

In this chapter the various scaling alternatives will be discussed.

Block Size limit

In 2010 Nakamoto reduced the block size limit from 32 to 1 Megabyte with two commits. The first commit introduces the new `BLOCK_SIZE_LIMIT` constant [64] and the other commit enforcing the blocks to be under the limit [65]. Nakamoto did not give any explanation as to the reason of this limit. As there was only a few transactions per block at this time a limit reduced the worst case scenario for a *Denial-of-Service* attack where blocks are filled with dummy transactions.

Increasing the amount of transactions would also increase the imposing cost for the ecosystem by forms of bandwidth, processing and storage - reducing the speed by which the blocks propagate through the network. By imposing a block limit the cost of operating a node would be kept low allowing more peers to be able to participate in the network.

With a limit on 1 Megabyte; Bitcoin is capable of 10 transactions per second or 350.000 per day. This in contrast with a traditional payment processor such as visa who have capacity 65.000+ transactions per second[**?**]. In it's current form Bitcoin does not have close to enough capacity as a payment processor for a significant fraction of the global economy.

Increasing block size limit

There has been many proposals to increase the block size. Including a handful of BIPs:

- **BIP 100** allows the miner to vote on the block limit by encoding a proposed value in the coinbase unlocking script. A 75% supermajority may increase the block size limit every 2016 blocks. Each period change is limited to only increase by 5% from the previous period. [40]
- **BIP 101** proposes an immediate increase to 8 megabyte blocks and continue to double every second year. In the two year periods the size would increase at a linear pace based on the block timestamp. [11]
- **BIP 102** is a simple one time increase to 2 megabytes. The BIP would be triggered if 95% of the latest blocks signaled support for the upgrade. [39]
- **BIP 103** aims to increase the block size in accordance with the technology. In practice however it increases the block size with 4.4% every 97 day, 17.7% annually. [93]

Most of these BIPs are implemented as a hard forks. Older nodes would regard the bigger blocks as being over the limit and thus invalid. If poorly executed, without a great majority upgrading the software in time, the network would be forked into two networks with diverging transaction histories.

Contentious Hard Forks

No proposal to increase the block size has yet received enough support to be activated. This has caused some controversy among community members and multiple have attempted to increase the limit by force.

One of the earliest attempts to increase the block size was when node Bitcoin XT implemented BIP

101 mentioned above. It failed to reach the miner consensus it required to activate. The BIP 101 was then removed and replaced by a 2 megabyte block limit which forked bitcoin into an additional network - Bitcoin Classic. Mike Hearn, co-creator of Bitcoin XT along with Gavin Andersen, has made his intention clear arguing that the block size must increase for bitcoin to reach any adoption beyond a fringe niche [47].

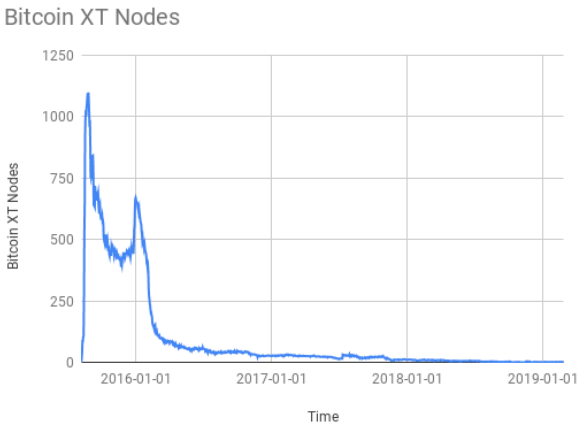


Figure 1: *Historic Bitcoin XT node count. As of 2019-02-25 only three nodes are still active. Data retrieved from coin.dance [6].*

As figure 1 attests, bitcoin XT and classic failed to receive support and is now dead.

The Bitcoin Cash saga

The failures of Bitcoin Classic were just the beginning of the scaling debate. The proposal of Segregated Witness [58] was met with strong opposition as it was considered an overtly complicated and risky upgrade when the block limit could simply be increased.

Segregated Witness is in a way an increase in block space as it moves the unlocking script(witness) from the transaction to its own structure. Since each transaction takes up less space, more transactions fit in each block. The upgrade also addresses malleability, which many off-chain solutions require, and aligns economics incentives with resource costs [14]. Although its possible to implement the Lightning Network without a malleability fix, it increases its viability [86].

The malleability fix also addresses covert `ASICBOOST` or `ANTBLEED`. Covert `ASICBOOST` utilizes that part of `SHA256` may be pre-calculated if the last 16 bits remain the same. The last 16 bytes in the bitcoin header include the last 4 bytes of the Merkle root. By finding two Merkle Roots with the same last 4 bytes, allowing for less computation in finding hashes. To find a matching pair of roots, many roots must on average be tried. The space of viable roots is reduces if the transaction witness isn't malleable [85] as the witness can't be manipulated to change the transaction and thus the Merkle root. Gregory Maxwell alluded that `ASICBOOST` had in fact been implemented in ASIC chips and proposed a fix [59]. Bitmain, the largest chip manufacture, made a statement two days later addressing Maxwell's accusation [21]. There is no way of ultimately proving if covert `ASICBOOST` was ever used[1].

When BIP148 [38], a *user-activated-soft-fork* scheduled for 22 August 2017 to activate Segregated Witness, was getting traction among nodes, the opposing party(including aforementioned Bitmain) decided to hard fork bitcoin into two by raising the block size limit. The hard fork took place on the first day of August 2017 and created Bitcoin Cash.

Bitcoin Cash received significant economic support initially, although significantly below the original chain.

Figure 2: *Historic Bitcoin Cash/Bitcoin price ratio. Data retrieved from Yahoo Finance.*

[1] In further conversation with Bitmain representative Nishant Sharma at their headquarters in Beijing in December 2017 he stated that they would never use `ASICBOOST` of moral reasons. `ASICBOOST` was not longer viable on Bitcoin at that time, but was on Bitcoin cash and other currencies. He also stated that Bitmain had been in negotiation to buy the patent from Timo Hanke and Sergio Demian Lerner and had thus filed a patent for `ASICBOOST` with the Chinese government. The deal ultimately didn't fell through, and while this information is hearsay at best, the patent sparked many rumors at the time and drove opposition and is one reason for the eventual split.

Although support for Bitcoin Cash has dwindled over the following years, it set a precedence of how forks may be utilized. It also showed that support from miners is not necessary. During the second half of 2017 many more forks happened similarly to bitcoin cash.

The SegWit2x Compromise

Most forks are insignificant. However SegWit2x carry some historical weight. SegWit2x was originally a compromise known as the Hong Kong Agreement reached in Hong Kong in February 2016 between many industry representatives and members of the development community [**?**]. The agreement was somewhat ambiguous leading to fall out and a subsequent New York Agreement were reached at the Consensus conference in May 2017 [**?**]. Notable absent from the latter agreement were the whole Core development team leading to the `UASF` and Bitcoin cash fork. Many actors still viewed the New York Agreement as valid after the split and many supported a hard fork to 2MB block limit on the main Bitcoin chain in late 2017. The fork was ultimately called off and only a few rogue actors followed through with it. A critical bug in the software led to the forked chain to halt in it's infancy.

Removing or reducing decentralization

Many other crytocurrency systems, which are seen as competing, have reduced decentralization in order to increase throughput. Some may not even be considered *peer-to-peer* but instead rely on a traditional *client-server* architecture which current financial systems are already using. These system easily scale. Most however use some sort of combination of increased block limit and/or some level of trusted nodes akin to *oracles*.

Sidechains

One proposal involve multiple chains where one could peg bitcoin to an alternate, less secure, chain. Multiple transactions could then take place on the alternate chain and then released back on the main chain. The alternate chains could regularly be thrown away and replaced, removing much of the storage burden of the network. [**?**]

Off-chain scaling

Off-chain solutions, like the Lightning Network, utilize the contractual nature of transaction. Schemes of transactions can be implemented such as that an obligation, in form of a transaction, may be held privately without broadcasting it. The obligation may then be changed multiple times without the rest of the network knowing and only in worst case, require to broadcast the transaction.

Most schemes involve payment channels, where two parties locks in funds between each other and then the balance between them can be updated without using the blockchain. Multiple channels may be aggregated together into a network. Consider a channel between Alice and Bob and a channel between Bob and Carol. If Alice wish to pay Carol, she may update her channel with Bob if and only if Bob updates his channel with Carol.

Payment channel networks may be constructed in many different ways, a few distinct proposals has been suggested. Decker and Wattenhofer suggested a scheme with Duplex micropayment channels [29]. The Lightning Network is further discussed in the following chapter. Recently a new scheme was published, the eltoo protocol, which may be integrated into the Lightning Network stack [**?**].

Rationale for small blocks

Bitcoins advantage does not lie in its ability to scale. In fact, centralized solutions can trivially outperform Bitcoin many times over with regard to cost, speed and reliability. Its strength lie in transparency, immutability and that it require no trust in any other party at all. These characteristics are both unique and valuable. Solutions to scale the network ought to not compromise the fundamental properties making it worth to scale in the first place. Therefor the cost of participation must remain low, which require that the cost of validating blocks must remain low and thus require small blocks.

The Lightning Network

The Lightning Network is a network of micro-payment channels on top of the Bitcoin Network. The network was first conceived and defined in 2014 [77], became viable with the segregated witness upgrade [58] and was activated through a *user-activated-soft-fork* in mid 2017 [38]. The upgrade moves(segregates) the unlocking script(witness) from the transaction which fixes transaction malleability since multiple unlocking scripts may be valid for a single locking script.

The network protocol has been defined [2] with four independent actors implementing protocol compliant nodes [52, 8, 22, 62].

Payment channels

The Lightning Network consists of many payment channels between both routing and non-routing nodes. A payment channel is essentially a transaction containing funds locked up on the bitcoin network. It requires signatures from both parties to be able to spend the transaction and cease the channel.

Initially each party has a signed transaction splitting the funds to their original committed amount[1]. The parties can then agree on a different balance signing a new transaction invalidating the previous spending transaction essentially allowing for infinite transactions inside the channel without any burden on the bitcoin network.

Funding Transaction

A payment channel is funded by a funding transaction defined in the lightning paper by following steps [77],

1. Create the parent (Funding Transaction)
2. Create the children (Commitment Transactions and all spends from the commitment transactions)
3. Sign the children
4. Exchange the signatures for the children
5. Sign the parent
6. Exchange the signatures for the parent
7. Broadcast the parent on the blockchain

It is critical that the signatures of the spending transaction are exchanged before the funding transaction otherwise the funds could be held hostage by an uncooperative partner[2]. By exchanging the spending transaction first either party could broadcast it to retrieve the initial funds.

Commitment Transaction

In order for a channel to be useful the creation of a new Commitment Transaction[3], updating the balance, would need to invalidate the previous one. The previous transaction is after all still a valid bitcoin transaction. This creates a problem since each party would benefit from different commitment transactions being broadcast to the network.

The solution is to construct the Commitment Transaction similar to that of a fidelity bond where one party would be penalized if violating the agreement. In this case violation is broadcasting any but the latest commitment transaction. To be able to ascribe blame to the violating party the commitment transactions for the same channel state must be unique as otherwise one would not know which party broadcast the transaction to the network. Only uniqueness is however not enough since there is no way to enforce the penalty for violation.

The enforcement is possible if revocation is built into the commitment transaction, where the other party could revoke a faulty broadcast. This became possible with the addition of relative block maturity introduced with BIP68(see section 3.3.3). The commitment transaction has two outputs, one is the other party's funds spendable immediately. The other output has two redeemable paths. Either,

[1] In the RFC each channel is funded by only one party so the initial split is always all to one party.

[2] This is possible with the `SIGHASH_NOINPUT` transaction defined in BIP118 [30] which decouples the signature from the specific transaction hash of the output. This is not yet activated and only single source funding is currently available

[3] It may be easier to think of these types of transaction outputs as smart contracts, as in this context it's the same thing.

1. The broadcasting party can spend the funds after a defined relative block time has passed. Essentially leaving time window for dispute for the violated party to enforce the penalty(Revocable delivery).

2. The non broadcasting party can refute that the commitment transaction is not last signed spending spending the output as penalty.

Remember that the commitment transactions is reversed for the other party. This type of output which enable revocation is called a Revocable Sequence Maturity Contract(RSMC).

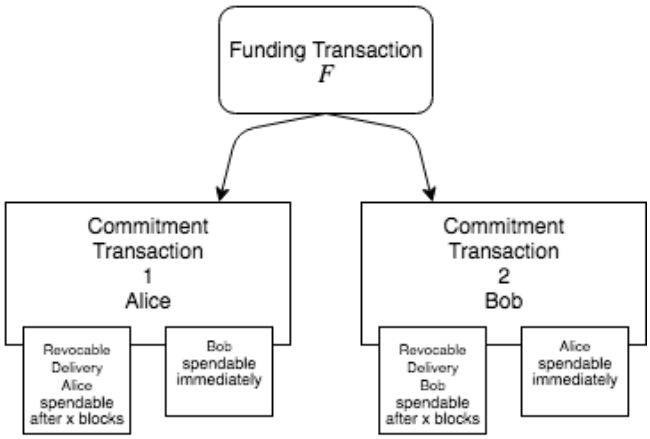


Figure 1: *Commitment transactions.*

Revocation of old commitment transaction

In order to update the channel balance a new pair of commitment transactions is created. The previous commitment transactions is invalidated by both signing a Breach Remedy Transaction(BRT) spending from their own previous Commitment Transactions RSMC output and handing it over to the other party. If a previous commitment transaction were to be broadcast the BRT would take precedence since the revocable delivery takes a certain amount of time before it is valid. The Breach Remedy Transaction will send all funds to the non faulty party, this is the penalty for violating the contract.

It is necessary for each party to monitor the blockchain for old commitment transactions and to broadcast the BRT in case it happens. If the BRT is not broadcast in the designated time the revocable delivery output of the previous commitment transaction will become valid and may be spent, settling the incorrect channel balance is settled and valid on the blockchain.

Alice and Bob opens a channel

Alice and Bob wants to open a channel. They agree on funding the channel with **0.5฿** each. They construct a funding transaction F which require both Alice and Bobs signatures to be spent. Alice constructs a CT_{A1} with two outputs $RSMC_{A1}$ and SO_{B1} and similarly Bob constructs CT_{B1} with outputs $RSMC_{B1}$ and SO_{A1}. Alice signs CT_{B1} and Bob signs CT_{A1}[4]. Alice and Bob exchange signatures for F and either of them broadcast F.

Alice wants to buy a cookie from Bob for **0.2฿** and Bob agrees to sell. They each constructs new Commitment Transactions CT_{A2}, CT_{B2}, each with new $RSMC_{A2}$, $RSMC_{B2}$ and SO_{B2}, SO_{A2} that reflects the new balance **0.7฿** to Bob and **0.3฿** to Alice without signing them. Before Bob can hand Alice the cookie the old CT_{A1}, CT_{B1} is invalidated by Alice signing a BRT_{B1} spending from $RSMC_{A1}$. Although Bob has nothing to gain by broadcasting CT_{B1} as he would net **0.2฿** less - Bob signs BRT_{A1} spending from $RSMC_{B1}$. Alice signs CT_{B2} and Bob signs CT_{A2}. The channel balance is updated.

If Alice tries to undo the cookie purchase by broadcasting CT_{A1} giving **0.5฿** each. Bob simply broadcasts the BRT_{B1} spending from Alice's share of the CT_{A1} and Bob receives **1฿** and Alice **0฿**.

As long as the channel is open Alice and Bob may settle any imbalances between them with a new commitment transaction.

Contracts and implementation

Signing and sending BRTs is unnecessarily inefficient if a HD wallet is used to generate keys[92]. HD wallets have a master key which can derive new keys in a tree structure. Each key in the tree can derive its children and no child can derive its parent. This works both for the public and the private keys independent of each other. A new key is used for each CT and if that is deep in the tree, say a millionth layer down, and can simply be disclosed to the other party whenever the CT is invalidated. The public master key may be shared with

[4] A note on notation - The lowered A,B indicates whether the output is spendable by Alice or Bob. The number denote the channel state the output corresponds to. SO stands for Spendable Output.

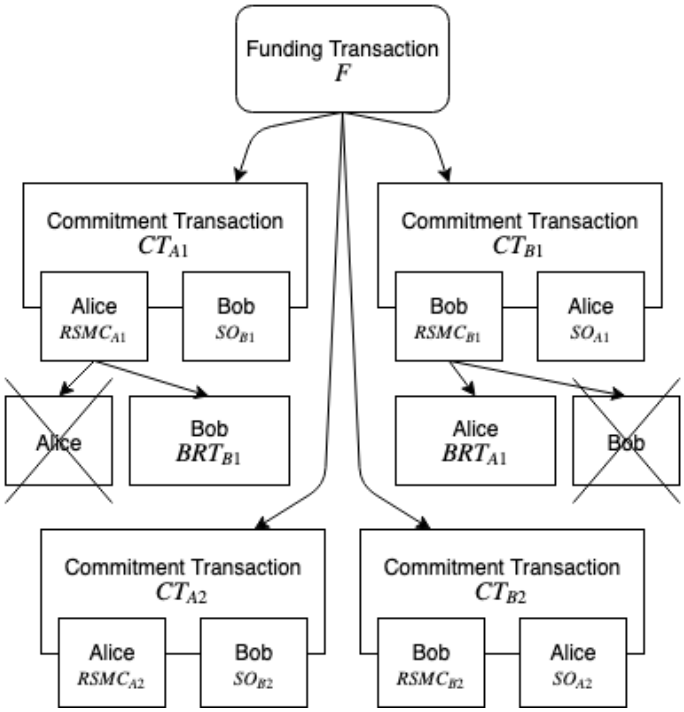


Figure 2: *Showing the channel balance state between Alice and Bob after the second Commitment transaction has been made.*

the other party so all public keys can be derived for transaction construction. With each new invalidated CT the previous CT's parent key may be used. So each party only needs to store the latest disclosed key as the other child keys can be derived from it.

```
OP_IF
  <time> OP_CHECKSEQUENCEVERIFY OP_DROP
  OP_HASH160 <A pk> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
  2 <A r pk><B r pk> 2 OP_CHECKMULTISIGVERIFY
OP_ENDIF
```

Figure 3: *A Revocable Sequence Maturity Contract.*

Consider a Revocable Sequence Maturity Contract seen in Figure 3. It is constructed by Alice and she used Bob's master public key to derive Bob's revocation public key in some prearranged manner. To invalidate the CT, Alice need only to disclose the private key corresponding to her revocation public key for Bob to be able to revoke a violation.

Derived revocation address

The 2 of 2 multi-signature clause paying out to the violated party may be implemented in a denser form by paying to a blinded key neither party control. The private key is then derived from knowledge split between the two parties. When a new CT is created the secret is shared so only the violated party could spend from the revocation clause in case the old CT is broadcast. So hence forth only a revocation pubkey is used to denote a revocation output.

Closing channels

It's possible to close a channel by broadcasting either of the latest CTs. However the broadcasting needs to wait the designated time period, losing out on the time value of money. Instead both parties may agree to construct a new Exercise Settlement Transaction(EST) with direct spendable outputs instead of a $RSMC$ output. Signing the EST would close the channel and allowing both parities immediate access to the funds.

Network wide payments

By combining channels into a network, a payment could potentially ripple through multiple channels and may allow parties without a direct channel to transact. Such a multi-hop payment would need the same trust-less custodial properties for the routing middle nodes as between direct channel parties shown above. This is solved by utilizing a Hashed Timelock Contract(HTLC) for each of the channels in the route.

Hashed Timelock Contract

A $HTLC$ is an extra output added to a CT which enforces a payment from Alice to party Bob if and only if Bob can produce and disclose a preimage R of a hash in a designated time period.

```
OP_IF
  OP_HASH160 <H_R> OP_EQUALVERIFY
  2 <A PK_2><B PK_2> OP_CHECKMULTISIGVERIFY
OP_ELSE
  <time> OP_CHECKSEQUENCEVERIFY OP_DROP
  2 <A PK_1><B PK_1> 2 OP_CHECKMULTISIGVERIFY
OP_ENDIF
```

Figure 4: *A Hashed Timelock Contract. It's not yet viable for payments in this state.*

A $HTLC$ can be seen in Figure 4 and contains two validation paths.

1. If Bob has the knowledge of R that produces H_R he can validate the first path. (Execution)
2. If Bob fails to prove knowledge of preimage R, Alice may validate the second path after the designated time period has passed. (Timeout)

Revocation of HTLCs

$HTLC$s have a similar problem as with direct payment channels, if an old CT is broadcast both validation paths may be valid. In the same manner as before, the $HTLC$ is made revocable by the construction of two different contracts for each party $HTLC_{A1}$ and $HTLC_{B1}$. So here again the possibility to ascribe blame exists. Adding revocable paths to $HTLC_{A1}$ and $HTCL_{B1}$ is necessary[5] but not enough since the execution path is also valid. A period for dispute must still be had as the disclosure may be correct but may be from an old commitment transaction. The same is true for the timeout path as Bob may not have knowledge of R in an old commitment transaction.

Instead the timeout / execution path to oneself is spent into new transactions with a dispute period and ability for the other party to revoke an incorrect broadcast.

```
OP_DUP OP_HASH160 <H_RPK> OP_EQUAL
OP_IF
  OP_CHECKSIG
OP_ELSE
  <PK_B> OP_SWAP OP_SIZE 32 OP_EQUAL
  OP_NOTIF
    OP_DROP 2 OP_SWAP <PK_A> 2 OP_CHECKMULTISIG
  OP_ELSE
    OP_HASH160 <R_preimage> OP_EQUALVERIFY
  OP_ENDIF
OP_ENDIF
```

Figure 5: *The $HTLC_{A1}$ as offered by Alice*

The $HTLC_{A1}$ seen in Figure 5 can only be broadcast by Alice and contains three validation paths:

1. Bob can revoke if it's not the last CT constructing a BRT spending the output.
2. The output is spent by a pre-signed transaction to a new transaction ST with a dispute period.
3. Bob can disclose knowledge of R and spend the output.

The order of the two last clauses is reversed for Bob's $HTLC_{B1}$ as if Bob can prove knowledge of R the output is spent to a new transaction ST for disputation.

```
OP_DUP OP_HASH160 <H_RPK> OP_EQUAL
OP_IF
  OP_CHECKSIG
OP_ELSE
  <PK_B> OP_SWAP OP_SIZE 32 OP_EQUAL
  OP_IF
    OP_HASH160 <R_preimage> OP_EQUALVERIFY
    OP_DROP 2 OP_SWAP <PK_A> 2 OP_CHECKMULTISIG
  OP_ELSE
    OP_DROP <time> OP_CHECKLOCKTIMEVERIFY OP_DROP
  OP_ENDIF
OP_ENDIF
```

Figure 6: *The $HTLC_{B1}$ as offered by Bob*

The $HTLC_{B1}$ seen in Figure 6 thus contain:

1. Alice can revoke if it's not the last CT constructing a BRT spending the output.
2. Bob can prove knowledge of R the output is spent to a new transaction ST with a dispute period.
3. After the designated time Alice can spend the output.

The new transactions by which both the second clauses spends to is constructed and pre-signed before the $HTLC$s is exchanged simply adds time for revocation as seen for ST_{B1} in Figure 7.

```
OP_IF
  <PK_RA>
OP_ELSE
  <time> OP_CHECKSEQUENCEVERIFY OP_DROP
  <PK_B>
OP_ENDIF
OP_CHECKSIG
```

Figure 7: *Showing the transaction ST_B spends to, the keys are reversed for ST_A.*

[5] It's necessary to include this path as an uncooperative party may never broadcast the execution/timeout transaction in which revocation is possible.

A $HTLC$ is terminated off-chain by creating a new commitment transaction, without $HTLC$ outputs, reflecting new the balance state as both parties knows what would transpire if the transaction were broadcast. In the creation of the new CT both parties disclose both private keys corresponding to the two revocation outputs as a penalty if an old CT were broadcast. Hence most transactions will be kept off chain and only held in case of an uncooperative party.

In this manner Hashed Timelock Contracts may be used without the need for either of the channel parties to trust each other.

Note that key generation and disclosure may be used for $HTLC$ in the same manner previously disused in section 5.1.5.

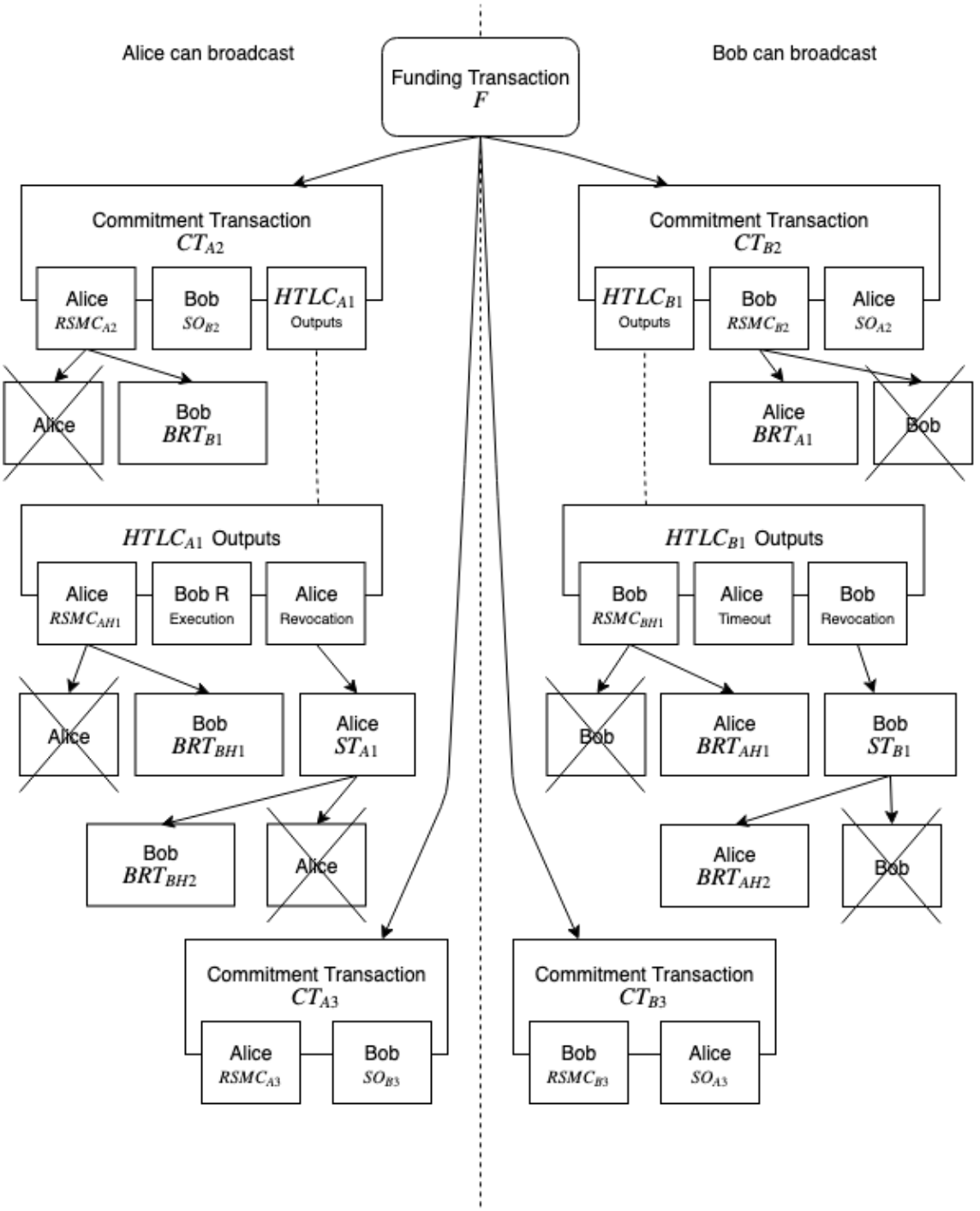


Figure 8: *Shows the state of transactions after Alice and Bob have agreed to a new Commitment Transaction CT_{A3}, CT_{B3}. The transactions to the left side can be broadcast by Alice and the ones to the right by Bob. With the signing of the new CTs Bob and Alice disclose keys for BRT_{AH1}, BRT_{AH2}, BRT_{BH1}, BRT_{BH2} as penalty if an old commitment transaction is broadcast.*

Multi-hop payments

With properly funded channels and Hashed Timelock Contracts in place network wide payments utilizing multiple channels is possible. The network uses a Gossip protocol[6] where public channels are broadcast. Each node may then find a path through multiple channels and construct a chain of $HTLC$s with decreasing time locks.

Alice wants to pay David but have no direct path to Carol. She finds a route via Bob and Carol to David as seen in Figure 9.

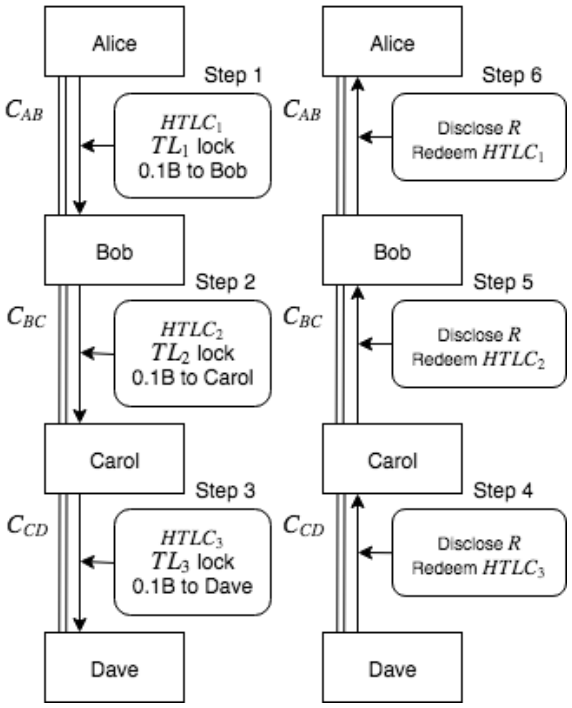


Figure 9: *Shows a multihop payment from Alice to Dave across three channels C_{AB}, C_{BC}, C_{CD}. It's critical that the HTLCs is constructed in the correct order and with decreasing timesteps $TL_1 > TL_2 > TL_3$.*

Alice and David agree on size of the payment and David constructs R and shares H_R with Alice. Alice then uses H_R to construct a $HTLC_1$. After Alice has created $HTLC_1$ with TL_1, Bob is comfortable creating $HTLC_2$ with a $TL_2 < TL_1$ as the only way Carol could redeem $HTLC_2$ is by disclosing R to Bob. Bob in turn may then disclose R to Alice and redeem $HTLC_1$. If the timelocks were improperly set as $TL_2 > TL_1$ the $HTLC_1$ may be invalid by the time Carol redeem $HTLC_2$ leaving Bob paying Carol but without getting payed by Alice.

[6] as per BOLT#7 routing-gossip in Lightning RFC[2]

Routing fee

The routing nodes receive a fee as compensation for displaced channel liquidity, the time value of money and operational cost of bandwidth, computation and electricity. The fee is structured with a base fee F_B and a fee F_R proportional to satoshis forwarded AF. The base fee is always paid independent of payment size while the proportional part is the product of size and fee. The base fee is denominated in microsatoshi (**µS**) and the proportional in microsatoshi per satoshi forwarded (**µS** / **S**). The fee for a payment P is thus:

$$F_P = F_B + (AF * F_R/1000000)$$

This fee structured does not discriminate for channel balance and it is difficult to 'discount' transactions that 'balance' the network. Reasons for this is given in Chapter 8.

Evaluation

With the eventual growth of the Lightning Network a fee market for channel liquidity will emerge. In this chapter the dynamics by which this fee market operates are sought. Strategies and routing node behavior are suggested in an effort to create a simulation where strategies can be tested against each other.

Lightning network as a graph

The Lightning Network may be modeled as a directional Graph G with vertices representing nodes and edges representing channels. As each channel may have different fees and liquidity in each direction it's useful to represent each channel with two edges. One edge in each direction.

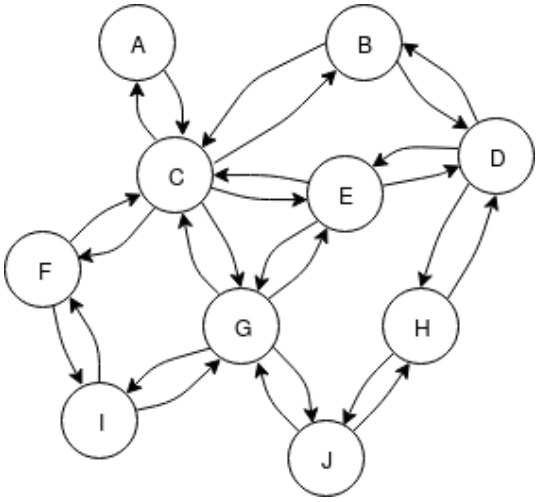


Figure 1: *A representation of LN as a graph. Each channel is symbolized by two edges.*

Modeling the network as a graph allows the use of graph theory. Thus, routing concepts of centrality and analytical frameworks may be borrowed from the extensive literature on graphs. The current topology of both Testnet and Mainnet are visualized in Figure 3.

Fee as a competitive equilibrium

It may be useful to view the fee market as existing under normal free market pressures and apply the conventional long run demand and supply curve [24] to the fee market.

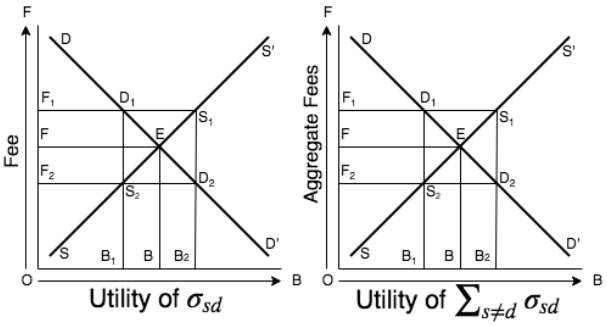


Figure 2: The fee market as viewed according to standard price theory. Both by single pair and all pairs in aggregate.

Consider Figure 2. The demand curve DD' signifies users demand to utilize a shortest path in the lightning network and the supply curve the existing competing paths between s and d. Say there exists an equilibrium point E by with the market price will move towards. Suppose the average fee price is greater than equilibrium at F_1 with demand F_1D_1 and supply F_1S_2, and more operators would open a path between s and d driving the curve fee towards E. If it instead lay under the equilibrium at F_2 their would be an excess supply F_2S_1 and low demand F_2D_2, making operators loose money by having liquidity locked up on a path not used. Thus leading operators to close channels between s and d, again driving the fee towards E.

The fee market may also be seen in its aggregate, where the axis represents the whole network. Here the drivers might be seen as new routing nodes joining if its a profitable venture or leaving if its a unprofitable one. Here again driving prices towards equilibrium.

It may be unusual to view demand through the usage of its medium, their is of course a dimension here between payment systems. But there's also a the possibility of streamable payments where the say one's rent might be paid hourly or a documentary per second dependent on the fee price. With a

decreased fee these payments becomes more viable driving demand.

The cost of procuring a shortest path between s and d might not be equal for two different routing nodes dependent on already existing paths and set strategies. The viability of a strategy is in essence determined in its ability to be included in shortest paths between nodes and still procuring sufficient high fees.

Fee floor

The fee price, assuming rational actors, can only fall to a level where the operators are able to sustain their operation. The fees received must compensate for the *time-value-of-money*, the security risk of having liquidity on a connected public node and the operational cost of running and managing the software. Bhatia has proposed a standardized reference rate as Lightning Network Reference Rate(LNRR) similar to LIBOR[1], OIS[2] and SOFR[3] [19]. Operational cost isn't included in LNRR as with managed funds a operational fee is removed after the returns are calculated. The competition of fees drives down the price until a certain point, a floor, where there is better returns elsewhere. Therefor a the fee has a floor at

$$F_{floor} = C_{op} + C_{risk-prem} + C_{risk-free-rent}$$

Note that the operational costs are independent of liquidity while the security risk premium and risk-free-returns are dependent.

Security Risk premium

Operating a routing node requires the keys to be "hot", i.e. to be active and used on a live system. This carries a risk, especially considering that the capacity is willingly broadcast, creating a honey pot for attackers. The pseudo-anonymous architecture leaves little to no legal reproach and the bitcoin is most likely gone forever.

As theft seldom happens it's not obvious how it should be account for. It is essentially a Credit Event, e.g Bankruptcy. Credit Default Swaps (CDS) is type of insurance where the seller pays the buyer in case the Credit Event takes place. The buyer in turn pays a premium, usually monthly. If such an instrument were created, the routing node could buy protection and account the monthly premium as the risk cost.

Time value of money

The routing nodes channels require funds to be locked up to be able to route. As such there is an opportunity cost as the funds could be invested elsewhere. In most cases this is equivalent to the Risk Free Interest Rate.

Operational cost

The operational cost, i.e. electricity and the human labor needed to operate the node. Note that with this definition operational security is excluded as it should be reflected in the Security Risk Premium.

Routing strategies under Game Theory

There are clearly marked dynamics at play here. Node operator have many different options at their disposal and although all actors may not be rational their behavior should approximate the behavior of maximizing profit. This phenomena has many names, e.g. in economic theory optional behaviors is sometimes refereed to exist along an *agenda field* and the iterable process of finding the maxima as climbing the *maximand hill* where an equilibrium lay at the top [24]. Further each actor's profit depends on every other actor's behavior reasonable to regard it as a game theoretic problem. More exactly it resembles a subset of game theory similar to that of biological ecosystems first studied by W. D. Hamilton [42] and later expanded to a rigorous framework by J.M Smith et al. [84, 83] as Evolutionary Game Theory(EGT).

In an ecosystem, species survival and propagation in the gene-pool depends on its behavior relative to the population. An aggressive animal may do very well in competition for feed in a population of mostly coy animals. However against a population of mostly aggressive beings it may have to fight most encounters and end up wounded leading to a coy behavior to be preferred. Unsuccessful individuals die off and are removed from the population and subsequently successful individuals prosper and it's genes increases in the population. In game theory a Nash equilibrium is a state in which no actor can change strategy and receive

[1] London Interbank Offered Rate.
[2] Overnight indexed swap.
[3] Secured Overnight Financing Rate.

a strictly higher payoff [71]. In EGT this notion is slightly modified and named Evolutionary Stably Strategy(ESS) which disallows for an actor to change strategy and receive an equal payoff as that would allow for the population to constantly drift between those two strategies.

Although Lightning nodes do not pass on genes, successful strategies are more likely to be copied than unsuccessful ones. Especially if strategies and tool are built as extensions/plugins and made available as open source and allowed to freely propagate. Similarly stable points where no node can change strategy and increase profit suggests the viability of the played strategies and hints at the emerging topology.

To evaluate strategies, the game is divided into six categories.

1. **Fee**, Determine the fee on the outgoing edge of the channel.
2. **Preferential Attachment**, Determine which nodes to open channels with.
3. **Timing**, When to open and close channels.
4. **Allocation**, How much capital should be allocated for each channel.
5. **Funding**, How much capital is available to the node.
6. **Re-balance**, When and how aggressively should a node re-balance its channels.

Most categories are quite distinct and natural to separate but there is some ambiguity here. Preferential Attachment could very well be considered as the same category but are kept apart to keep the complexity down. Each routing node may then choose a strategy per category, it is sufficient to only consider pure strategies.

Mixing strategies

Mixing of strategies, e.g behaving 'aggressive' 70% of the time and 'coy' 30% of the time, is perfectly reasonable to assume. However convergence to a mixture is equivalent to the population mixture of pure strategies. Any equilibrium with only pure strategies could thus be transposed in to a mixed strategy [33]. This conveniently reduces the complexity of the simulation without compromising the results.

However, the strategies success are dependent on the graph state and thus previously played strategies, this is not the case here.

Non-advertised nodes

The network is not only made up of actors, or players, trying to maximize profit. But also, of course, of nodes wanting to utilize the network as a payment medium. Since routing requires a bit of commitment, regular users can open private channels and not broadcast these to the rest of the network. These non-advertised nodes have other priorities than profit, as it doesn't receive any fees, like having highly liquid paths to many nodes. It may also be worth deviating from the shortest payment paths to increase privacy and paying a higher fee than necessary.

Although this thesis concerns the routing nodes, the behavior of non-advertised affect the routing nodes. If non-advertised nodes prefer to open channels with high capacity nodes, in attempt to receive high liquid paths, it would be preferred to have high capacity for a routing node.

Emerging strategies

The Lightning Network consists of nodes, each utilizing different strategies whose success depends on strategies by other nodes. Over time successful strategies will emerge. Which strategies is stable, how should a routing node behave? Will certain biases be preferred, essentially forcing the network into a certain structure, e.g. scale free hub and spoke or random mesh network? Will only highly liquid nodes be profitable?

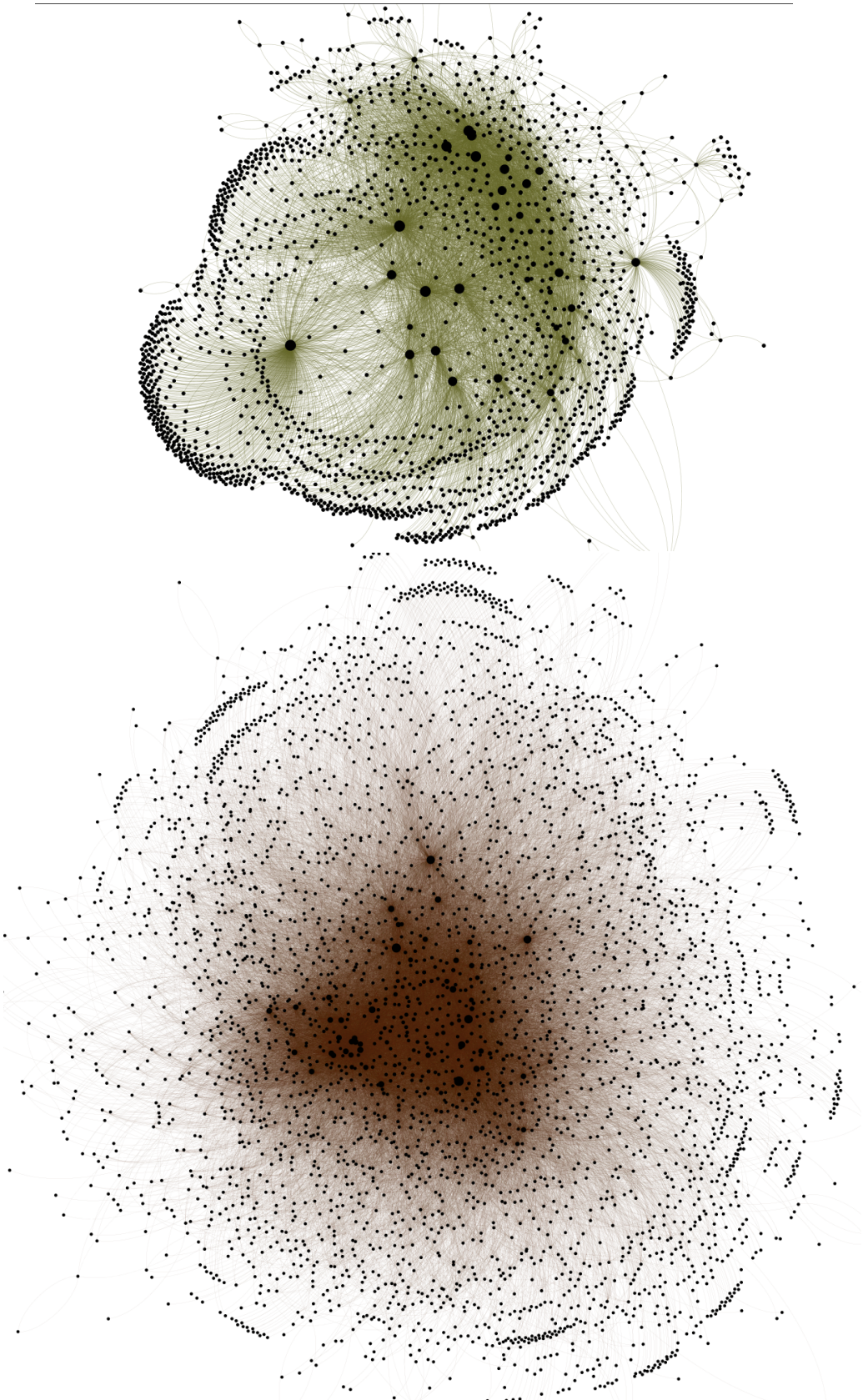


Figure 3: *Shows testnet above and mainnet below as of 2019-02-25. Node size corresponds to degree, larger degrees to larger size. Positioning is a mix between Force Atlas and Fruchterman Raingold. Network data retrieved by running a c-lightning node and visualized with Gephi [1].*

Optimal fee price

To determine the fee price, one could intuitively imagine that an increased fee price would yield higher fees but for less pairs as other paths may become shorter for pairs to route through. So the amount of shortest paths passing through a vertex must be calculated. In 1977 Freeman introduced a measurement named *Betweenness Centrality* [36] and defined it as

$$g(v) = \sum_{s \neq v \neq d} \frac{\sigma_{sd}(v)}{\sigma_{sd}}$$

where σ_{sd} are all shortest paths from s to d and v is the vertex. As the fee is determined on an edge basis the definition is altered to regard edges instead of vertices.

It is possible to determine the optimal fee for an edge if some assumption, of the probability of all pairs transacting and the probability of the size of the transactions, are made. If a uniform probability over all pairs and that the shortest paths are used are assumed the optimal price for one payment size is given by

$$x_{opt} \text{ is the optimal price } iff$$

$$f : X \to \mathbb{Z} \text{ and } (\forall x \subseteq X) f(x_{opt}) \geqslant f(x) \text{ where}$$

$$f(x) = x \sum_{s \neq e \neq d} \frac{\sigma_{sd}(e(x))}{\sigma_{sd}}$$

The σ_{sd} represents the total amount of shortest paths between s and d, while $\sigma_{sd}(e(x))$ is the sum of shortest path specifically through edge e with weight x. The function $f(x)$ behaves as seen in figure 4 for a fictitious generated graph.

Freeman also defined a centrality measurement for whole graphs, it utilizes the betweenness centrality of every vertex and is thus brutal to calculate in practice for larger graphs.

Efficient algorithms for edge betweenness centrality have been developed [25]. Although it is possible to run such an algorithm for each potential fee price to find the optima it would be very slow. Instead the optimal fee price an for outgoing channel e, still holding the assumptions as before, may be calculated by following steps.

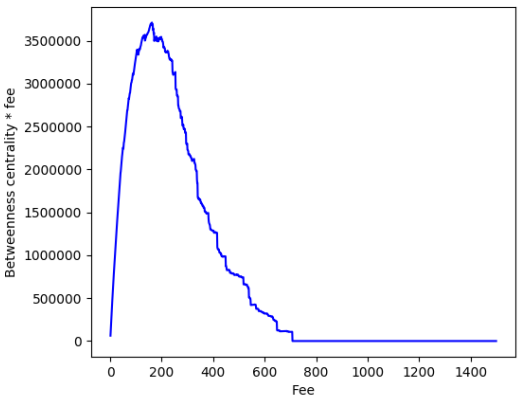


Figure 4: Shows the betweenness centrality * fee / fee for an edge in a generated graph with 1000 vertices, 5000 edges, uniform weight distribution over 1-1500 and random attachment policy. It holds intuitively as for low fees, many shortest paths passes but procure small overall profits as the fee is small. The profits increases with the increase in fees until a point where the decrease in paths overtakes the increase in fees.

1. Calculate the shortest path from all-to-all vertices without going through e with either Floyd-Warshall [45] or Johnson [49] obtaining table A.
2. Calculate shortest path from the source of e, V-to-all explicitly going through e with Dijkstra obtaining table D where e weight set to 0.
3. Retrieve the difference for all pairs between the all-to-all distance with the all-to-V-to-all distance, A[s][d] - (A[s][V] + D[d]). Throw away the negative values and produce a cumulative summation over the differences in a table H.
4. Select x_{opt} such that

$$\forall x(xH(x)) \leq x_{opt}H(x_{opt})$$

Whenever Floyd-Warshall or Johnson should be used depends on the sparseness of G as Floyd-Warshall have a time complexity of

$$O(V^3)$$

only dependent on vertices whereas Johnson

$$O(V^2 log(V) + VE)$$

depends on both edges and vertices.

Cost function

As the previous solution naively assumes a fixed payment size, it's altered to encompass a payment size distribution. The fee is proportional to the size of the payment, $F_B + F_P S$, thus the shortest path may be different for two different payment sizes. The problem is thus a combinatorial optimization problem,

$$\max_{x,y\in\mathbb{Z}} \sum_{\phi\in\rho} \rho(\phi) \sum_{a\neq b} \begin{cases} \Omega_{abe}(x,y,\phi) & \Omega_{abe}(x,y,\phi) < \sigma_{ab}(\phi) \\ 0 & \Omega_{abe}(x,y,\phi) > \sigma_{ab}(\phi) \\ \dfrac{\Omega_{abe}(x,y,\phi)}{\overline{\overline{\sigma_{ab}}}+1} & \Omega_{abe}(x,y,\phi) = \sigma_{ab}(\phi) \end{cases}$$

where $\Omega_{abe}(x,y,\phi)$, and $\sigma_{ab}(\phi)$ is given by

$$\Omega_{abe}(x,y,\phi) = \left(\sum_{c\in\sigma_{ae_s},\sigma_{e_db}} c_b + c_p\phi \right) + x + y\phi$$

$$\sigma_{ab}(\phi) = \left(\sum_{c\in\sigma_{ab}} c_b + c_p\phi \right)$$

where $\rho(\phi)$ is the fraction of payments size ϕ. e_s, e_d is edge source and destination nodes. σ_{ab} is the edge set of the shortest path without going through e. $\overline{\overline{\sigma_{ab}}}$ is the number of paths equally short with the shortest path.

The function over F_B, F_P in $f(x,y)$ follows the same pattern as over F and is shown in Figure 5.

The combinatorial optimization problem may be solved similarly as before but with a modified Floyd-Warshall. Instead of the weights being integers, the weights are linear cost functions. For each new path the cumulative cost functions of edges in the paths are compared. If both m and k are greater than any previous found path it can be thrown away. If either m or k is greater their intersection can be stored as an index. If the intersections are stored in order only a logarithm of previously found paths need to be compared in the worst case. This finds all the shortest paths for each node pair over all transaction sizes. The paths may then be adjusted over intersection distance and payment size frequency, giving the optimal price.

This is however still a NP-hard problem and network growth will eventually make an exact solution infeasible.

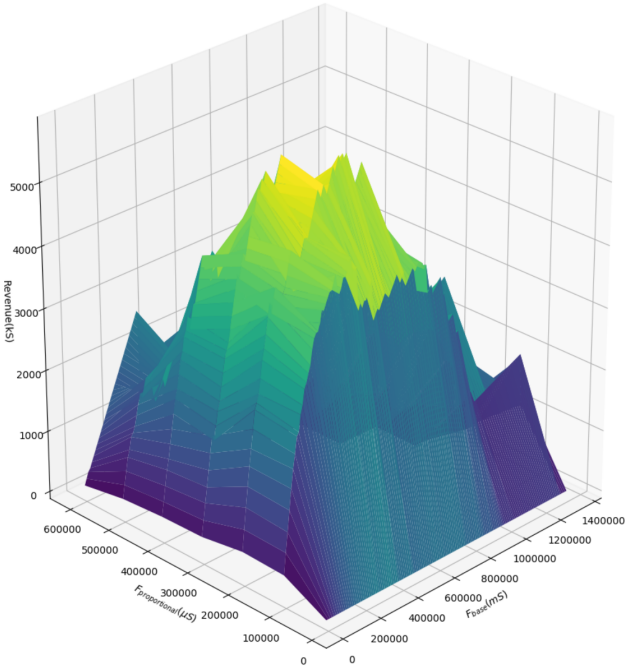


Figure 5: Shows the relative revenue for a well connected edge over base and proportional fee when all other for a well connected edge in a generated graph. Assuming a uniform payment size distribution over 2,000 Satoshi - 2,000,000 Satoshi.

Convert to probabilistic model

It is cumbersome to calculate the optimal price for all edges; for simulation purposes it is completely infeasible.

Instead a probabilistic fee model may be created. With a defined simulation environment, including actors and their strategies, after simulation a well connected edge is selected and the relative revenue field is calculated as in Figure 5. The field is then normalized and converted to a probability. The probability may then be used as a fee strategy where the probability of choosing a fee corresponds with the relative revenue. This process can be repeated multiple times, the next simulation uses the previous fee model.

Topology and Preferential Attachment

Setting fee price is only one aspect of a routing node, maybe more importantly is that of opening

and closing channels. As the protocol now stands[4] it only allows for single source funding of channels. There is a non negligible cost to opening channels and it would be very beneficial for a node if other nodes opened channels with it instead of the other way around. This is also a prerequisite to earn profit as otherwise the node may only earn back its own funding.

Therefor the policies determining by which preferences each node takes into account to open channels is of interest. Further these policies also affects the topology of the network, the average shortest path and robustness in case of node failures.

Traditional random models

Random graphs have been studied extensively, specifically the Erdős–Rényi model [34]. An ER graph is created by denoting a graph $G_{n,N}$ with n vertices and N edges. The edges is then selected randomly from set of $\binom{n}{2}$ possible edges. A common approach today is to write $G_{n,p}$ with p being the probability of two vertices having an edge. It's difficult to justify LN behaving as an Erdős–Rényi graph as it allows for isolated points and there is no inherent age bias present.

Callaway et al. [28] added a phase transition to random graphs as to adjust for a supposed age bias. Consider $G_{n,\delta}$ where at first the graph is empty and for each time step a vertex v is added to G and an edge is selected over $\binom{G_v}{2}$ with a probability δ until G has n vertices. $G_{n,\delta}$ would have on average $n\delta$ edges and since $n\delta < n$ it only produces sparse graphs. As before this may be difficult to apply to the Lightning Network as a poorly connected graph would have high average paths and surely many payments would fail. It is however interesting to regard the degree distribution compared to that of Erdős–Rényi as it's exponential given as

$$p(k) = \frac{(2\delta)^k}{(1+2\delta)^{k+1}} \tag{6.1}$$

compared to Erdős–Rényi which is binomial

$$p(k) = \binom{n-1}{k} p^k (1-p)^{n-1-k} \tag{6.2}$$

as follows from definition as $\binom{n-1}{k}$ are all possible combinations of k edges, p^k the probability of any combination taking place and $p^k(1-p)^{n-1-k}$ to exclude combinations of larger degree k.

[4]Is very plausible that that BIP66 may be activated allowing multi-sourced channels [30]

Thus adding an age bias causes the characteristics of the degree distribution to change as seen in Figure 6.

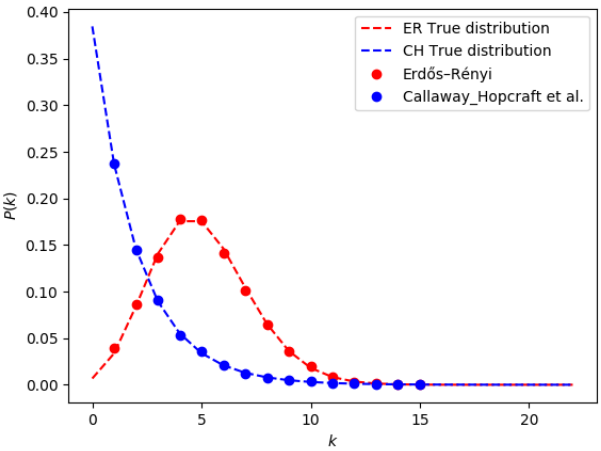


Figure 6: Shows the degree distribution for an Erdős–Rényi graph(●) with n = 10,000, p = 0.0005 and its true distribution given by Equation 6.2(|) compared to the degree distribution for a Callaway et al. graph(●) with n = 10,000, $\delta = 0.8$ and its true distribution given by equation6.1(|).

A more appropriate random model may be to consider $G_{n,m}$ where each time step t a vertex is added to the graph and is connected with m previous vertices and creates and edges when t < n. The probability of any previous vertex i being selected is thus $p_i = \frac{m}{t_i}$. As a node may not be viewed as part of LN before edges are made, it's reasonable to assume a node creates edges as part of joining the network. $G_{n,m}$ would also have an exponential degree distribution.

Scale free models

In 1999 Barabási and Albert showed that many networks degree distribution follows a power law and that the distribution was independent from scale, thus scale free [17]. Scale free networks follows a degree distribution of

$$p(k) \propto k^{-\gamma} \tag{6.3}$$

and large network has been shown to follow this model. With notable networks of web hyperlinks having $\gamma_{www} = 2.1 \pm 0.1$ and an actor collaboration graph having $\gamma_{actor} = 2.3 \pm 0.1$. Further suggestions have been made that network have an

innate bias, as what has many links is well known and what is well known will receive many links - akin on the *Pareto distribution*("80-20 rule") or the *Matthew Effect*[5]. A model was suggested to generate a scale free graph $G_{n,m}$ by starting with a graph with m fully connected vertices, and for each time step adding a new vertex and creating m edges with a bias towards vertices with high degree. The vertex have a preferential attachment and the probability of a vertex receiving an edge from the new vertex is,

$$\Pi(i) = \frac{k_i}{\sum_j k_j}$$

Such a graph follows the scale free property with a $\gamma_{ba} = 2.9$ and can be seen in Figure 7

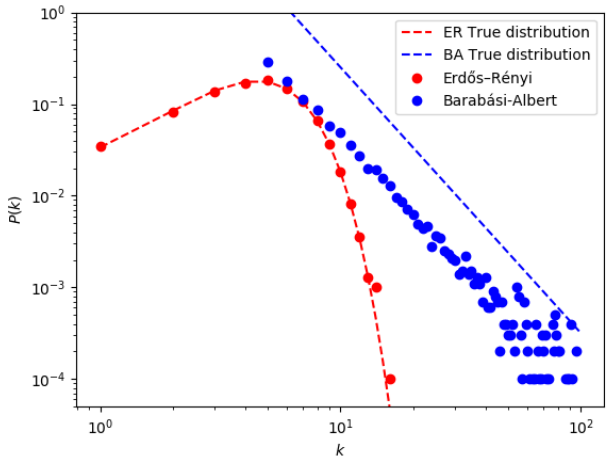


Figure 7: Shows the degree distribution for an Barabási–Albert graph(●) with n = 10,000, m = 5 and the scale free6.3(|) with γ = 2.9 compared Erdős–Rényi graph(●) with n = 10,000, p = 0.0005 and its true distribution given by Equation 6.2(|).

The actual degree distribution of the lightning network can be seen in Figure 8.

It should again be stated that the Lightning Network is still in a very early state of its development and most of the nodes and transactions over the network are made by enthusiasts trying it out. So the current state of the degree distribution may indicate very little of future states.

[5]As per KJB `Matthew 13:12 - For whosoever hath, to him shall be given, and he shall have more abundance: but whosoever hath not, from him shall be taken away even that he hath.` [3], after remark by Robert K. Merton [61].

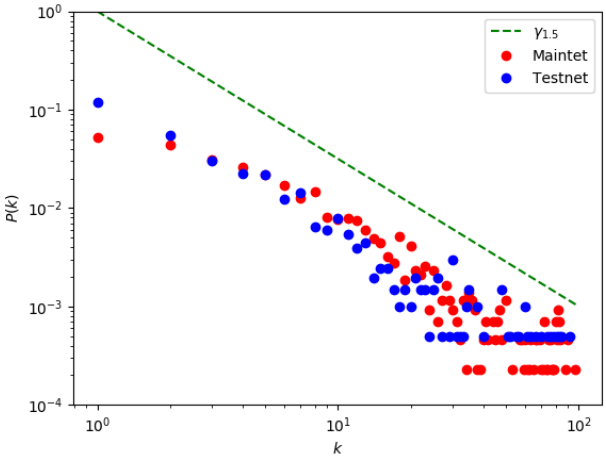


Figure 8: Shows the degree distribution for both the Mainnet (●) and Testnet (●) compared to the scale free function 6.3(|) with $\gamma = 1.5$. The data is retrieved running a c-lightning node [22] on 2019-03-21.

Mediation-driven attachment

Rather than explicitly attaching with a degree bias as with the Barabási-Albert model, Hassan et al. [46] have suggested a more subtle approach by attaching to a random neighbor of a randomly selected mediator node. Consider a node i which degree k_i and neighbors $1, 2, ..., k_i$ each having a degree of $k_1, k_2, ...k_{k_i}$ would thus have a probability $\Pi(i) \propto k_i$ and more explicitly,

$$\Pi(i) = \frac{1}{N}\left[\frac{1}{k_1} + \frac{1}{k_2} + ... + \frac{1}{k_{k_1}}\right] = \frac{\sum_{j=1}^{k_i} \frac{1}{k_j}}{N}$$

They also show that it will lead to a scale free network, however the degree distribution γ is dependent on m which differ from Barabási-Albert.

Inverse Barabási-Albert

It may seem ridiculous that selecting nodes with low degree would be successful, however doing the opposite to the majority may be very promising. As these niche routes have considerably less competition, especially considering a node created at a large t. If differentiation is a good strategy, this may capture it. The selection probability follows as,

$$\Pi(i) = \frac{\frac{1}{k_i}}{\sum_j^n \frac{1}{k_j}}$$

where $\Pi(i)$ is the probability that node i is selected with degree k_i

Fitness models

Node attachment may be influenced by fitness heuristics not intrinsic to the topology. E.g. Google overtook incumbent search engines having larger degree. The Bianconi-Barabasi model [20] was created to account for external heuristics as,

$$\Pi(i) = \frac{\eta_i k_i}{\sum_{j=1}^n \eta_j k_j}$$

By regarding the attachment probability growth over time,

$$\frac{\partial k_i}{\partial t} = m \frac{\eta_i k_i}{\sum_{j=1}^n \eta_j k_j}$$

they are able to show that not all fitness distributions $\rho(\eta)$ results in a scale free network.

As there are many possible node heuristics beyond degree, e.g. capacity, availability and fee, these heuristics could be blend into the attachment models.

Although Bianconi and Barabasi only studied fitness with the Barabási-Albert preferential attachment, it can however be added to the other models as well.

Time biased random model,

$$\Pi(i) = \frac{\eta_i}{\sum_{j=1}^n \eta_j}$$

Inverse Barabasi-Albert model,

$$\Pi(i) = \frac{\eta_i \frac{1}{k_i}}{\sum_j^n \eta_j \frac{1}{k_j}}$$

Mediation driven attachment model,

$$\Pi(i) = \frac{\eta_i \sum_{j=1}^{k_i} \frac{1}{k_j}}{N \sum_{j=1}^n \eta_j}$$

The η_i can of course be a combination of heuristics $\eta_i = \phi_{i1} * \phi_{i2} * ... * \phi_{ik}$, scaled by constants in relation to each other and/or feature scaling normalized $\eta_i = c_1\rho(\phi_{i1}) * c_2\rho(\phi_{i2}) * ... * c_k\rho(\phi_{ik})$.

Non degree fitness models

All previous models have assumed that degree is relevant[6]. However it doesn't need to be expressed so explicitly, following from the Lognormal Fitness Attachment (LNFA) [18],

$$\Pi(i) = \frac{\prod_{k=1} \phi_{ik}}{\sum_{j=1}^n \prod_{k=1} \phi_{jk}} = \frac{\eta_i}{\sum_{j=1}^n \eta_j}$$

which is evidently the same as the time biased random model.

Robustness

Robustness is mainly regarded in terms of connectivity where reduction into sub components are measured when a percentage of either edges or nodes are removed. Usually the giant component, the largest component, as a percentage of the total network is indicative of partitions. As LN channels are virtual, edge connectivity doesn't make sense, which leaves node connectivity. Scale free networks are more robust than random networks[7] when nodes fail at random, but scale free networks are highly vulnerable if hubs are targeted [**?**].

The current robustness of Mainnet is seen in Figure 9 and 10 for accidental/random failures and targeted attacks respectively.

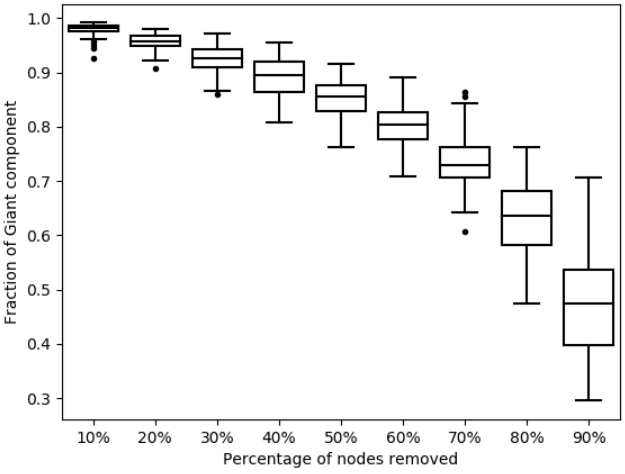


Figure 9: Shows the Giant Component as a fraction of network when a percentage is removed at random, repeated 100 times per percentage. Retrieved from Mainnet running a c-lightning node [22] on 2019-04-17..

[6] Which is probable, it may though be useful to express it as any other heuristic.

[7] Note that time biased random networks is not considered here.

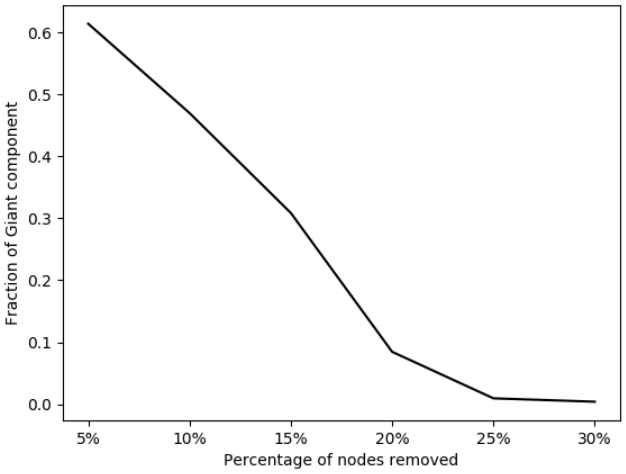


Figure 10: Shows the Giant Component as a fraction of network when a percentage of the nodes with highest degree are removed. The data is retrieved from Mainnet running a c-lightning node [22] on 2019-04-17.

Average shortest path

Each additional hop required in a payment path decreases the chance of having enough liquidity. It also increases failures in other dimensions, e.g. node compatibility and protocols levels below to fail. The average path growths at the same rate as the diameter, the longest shortest path between two nodes. For Erdős–Rényi the average path thus follows [**?**],

$$\ell \sim d \approx \frac{ln\ N}{ln\ pN}$$

For Barabási-Albert it is even shorter, with a double logarithmic correction it's considered 'ultrasmall' [**?**],

$$\ell \sim \frac{ln\ N}{ln\ ln\ N}$$

The average path can be gathered empirically with Floyd-Warshall or Johnson,

$$\ell = \frac{\sum_{i \neq j} \sigma_{ij}}{N(N-1)}$$

where σ_{ij} is the length of the shortest path between i and j.

Re-balancing channels

The ability to route payments is wholly dependent on having liquidity available. Constructing a route and pay to oneself is possible with the current protocol which changes the balance of two edges for a fee. By doing a regular *depth-first-search* cycles can be identified. For a route to be useful it must leave the two channels in a more balanced state than before. Here two simple heuristics are suggested.

Linear displacement

If all increments towards the middle of the channel is considered equally valuable, then a the utility of a path is the combined liquidity towards balance per fee cost. Thus giving utility function,

$$U(c) = \begin{cases} \dfrac{2|(B_{1S} - B_{1M})|}{f_c(|B_{1S} - B_{1M}|)} & if\ B_{1S} - B_{1M} < B_{2M} - B_{2S} \\ \dfrac{2|(B_{2M} - B_{2S})|}{f_c(|B_{2M} - B_{2S}|)} & otherwise \end{cases}$$

where B_{1S}, B_{2S} are the states of the out and in channels and B_{1M}, B_{2M} their middles and f is the total fee for cycle c.

$$f_c(L) = \sum_{i=1}^{n} B_i + P_i L$$

with B_i is the base fee, P_i is the proportional fee for node i in path c.

Edge biased displacement

As a channel with most liquidity at one edge of the channel would only be able to route in one direction. Thus channels at one edge may be regarded with a bias in such that those channels would be preferred to re-balance. The bias can be stated explicitly to modify the linear displacement,

$$U'(c) = U(c)\left(1,5-\left(1-\frac{B_{1S}}{2B_{1M}}\right)\right)^s \left(1,5-\left(\frac{B_{2S}}{2B_{2M}}\right)\right)^s$$

given a scalar s by which the bias can by adjusted.

Utility strategy scalar

Each node can define how high the utility value must be to consider to re-balancing the channel by routing a payment to oneself.

Funding and allocation strategies

The funding available to a node and the amount allocated to each channel strategies are easily defined by an integer. Funding may to some extent rely on external factors, but most likely a node reaches a point were it may be profitable to start a second node instead of increasing the funding on the first one. Allocation clearly depends on the average size of payments. The two categories together determines the amount of open channels.

Simulation

The aforementioned strategies and a simulation environment were implemented in python 3.6 with extensive use of the Networkx library. Environment variables and strategies are detailed in A.

Actors were given a set of strategies, one per category. For each time step a number of transactions were routed through the network, the fees were distributed to the nodes in the route and liquidity displaced. If a node received to little fees in relation to the liquidity it provided, it would go 'bankrupt' and be removed from the pool before the next time step. Each time step a number of new nodes would join. The strategies the new node utilize would be proportional to the actors already present in the network. Non-advertised nodes may also be in the network, they never route or go bankrupt, but may receive or send the initial payment. Otherwise the utilize the same strategies as the routing nodes. The simulation does not require non-advertised nodes and are not used in all simulations in the next Chapter.

Results

The results are retrieved running simulations based on the previous chapters. The different parameters are defined in detail in Appendix B.

Every simulation is run ten times and the mean and standard deviation(σ) are calculated. The source code for the simulation is available online [**?**] . The settings used to generate these particular results is also available in the preset directory.

Fee Price

A probabilistic fee model, as suggested in Section 6.5.2, was generated by running three sequential simulation cycles. A simulation was then set up with half the initial nodes using the generated fee model, while the other half used an uniform random model. The fee model clearly outperforms the random model as seen in Figure 1. The results also shows that the fee is more crucial than the attachment strategy.

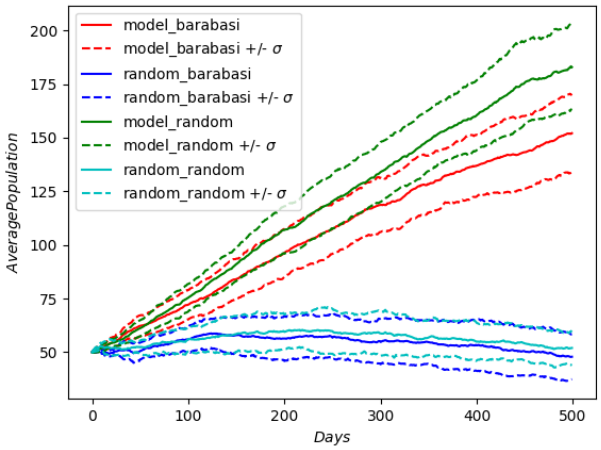


Figure 1: The history of the node count averages with one standard deviation, for 10 runs. Initially half of the nodes utilizing the generated fee model and the other half a random model. Similarly the attachment strategies are split between Random and Barabasi-Albert. All other strategies are kept equal.

Attachment

The different attachment strategies, excluding biases, were simulated against each other. The results is seen in Figure 2

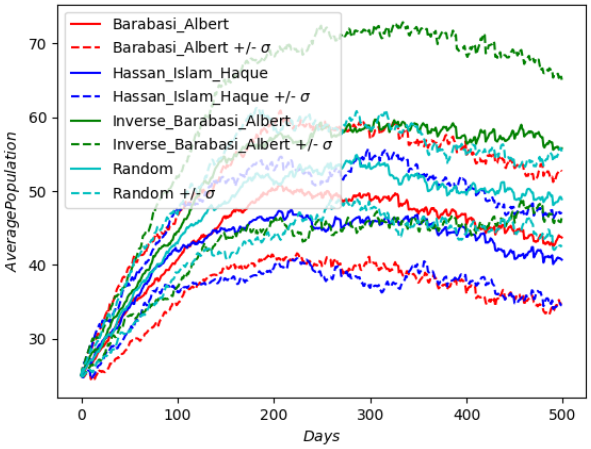


Figure 2: Shows the node count average with one standard deviation for four attachment strategies, Random, Barabasi-Albert, Hassan-Islam-Haque and Inverse-Barabasi for. All else being equal.

Funding

A simulation with four different level of funding was simulated. Each increasing step had twice the funding. The results seen in Figure 5 suggests that it may be advantageous to be an outlier. Since allocation per channel is equal, this also hints at the optimal amounts of open channels.

However, if the non-advertised nodes would have a fitness attachment selection favoring nodes with high capacity - the well-funded nodes would be highly advantageous as seen in Figure [].

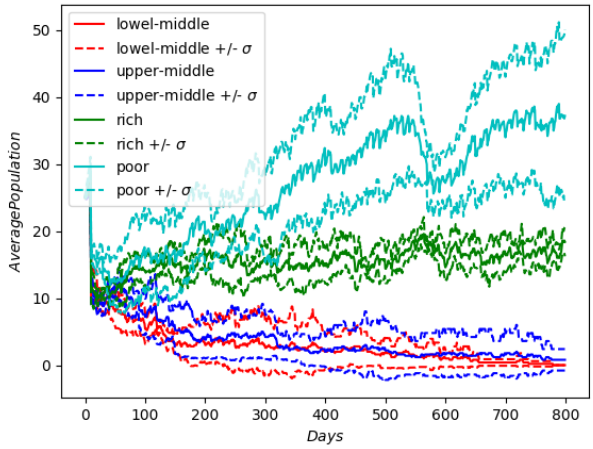


Figure 3: Displays the node count and node count average with one standard deviation for four different levels of funding. All other strategies being equal.

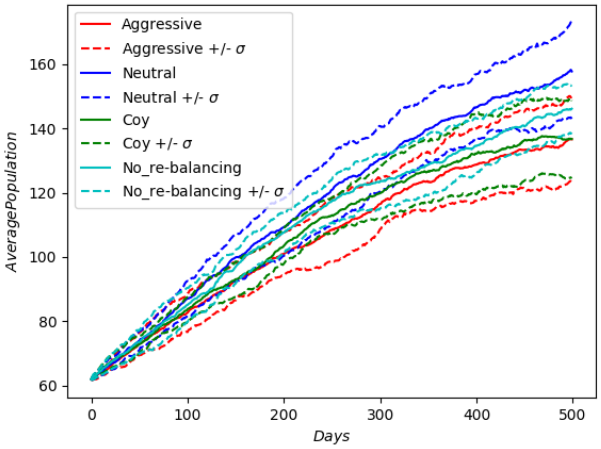
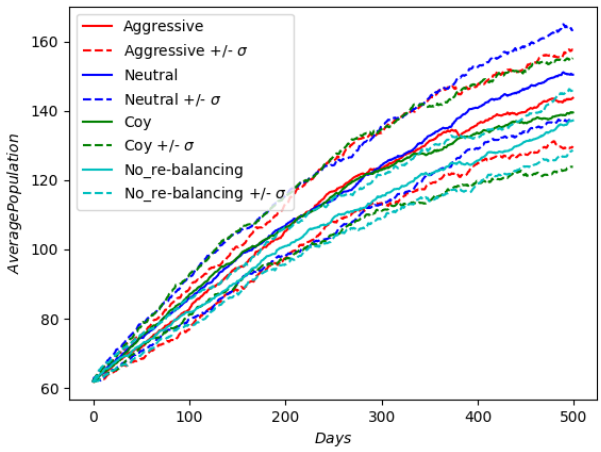


Figure 4: The above Figure shows the linear displacement strategy with different aggressiveness. While the Figure below shows the edge displacement strategy with different level of bias towards the edge. They look rather similar. They also have a little overall importance, as they all do quite well.

Re-balancing channels

The two different re-balancing strategies were trie with various level of aggressiveness. The strategie were first simulated separately and then the bes level of aggressiveness of each strategy were simu lated again against each other.



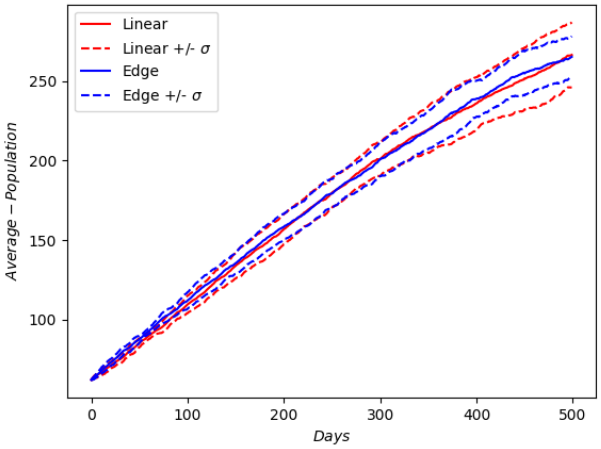


Figure 5: The above Figure shows the best linear displacement strategy against the best strategy from Figure 4.

Revenue distribution

To explore the innate distribution of the network, only one set of strategies was simulated. Every routing node behaved exactly the same, as it would be the case were the distribution would be expected to be the least extreme. Hence, the extreme tendency here in Figures 6, 7 suggest that the real

distribution would be at least this extreme.

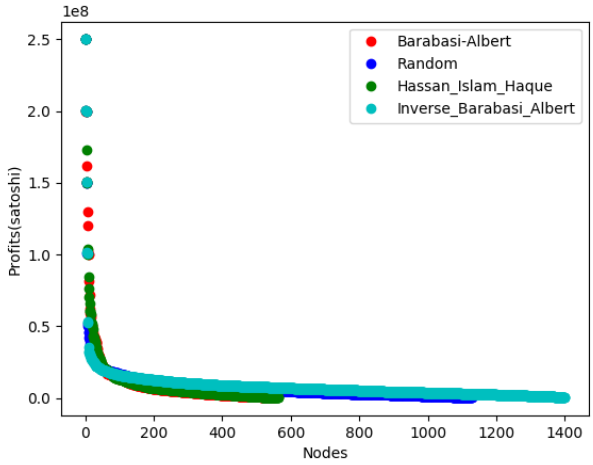


Figure 6: Shows the four different simulations separate. Aggregated over two simulations each. Note that environment variables are equal and shows that a broader selection bias supports more nodes. Barabasi-Albert

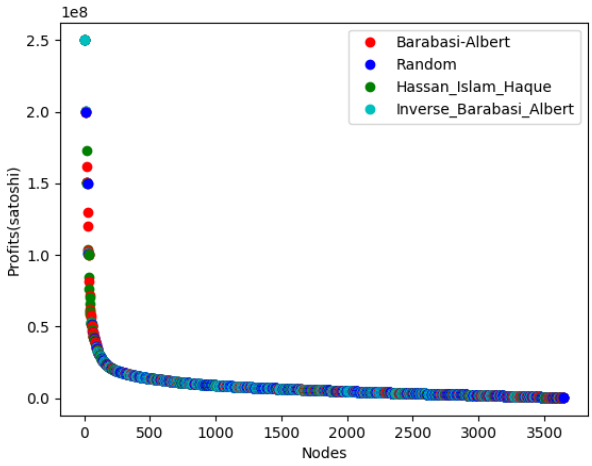


Figure 7: Shows

The exponential distribution is akin to previously mentioned Pareto Principle, although in Figures 6, 7 20% of the network account for 60% of the gross revenue. Note that if profit / net revenue where considered the quotient would be larger. The distribution shown here is in line with the wealth distribution in countries [**?**].

Discussion & Conclusions

First, there is three points that should be remarked.

1) The topology and efficiency of the Lightning Network will form as a product of profitable behavior, bottom up. It will not form from intervention, where a random mesh network is seen as preferable and the actors aim to fulfill such a model.

2) The behavior regarded in this thesis leads to a sufficiently robust network with short average paths. Strategies that fulfill valuable niches in the network will be profitable and the emerging real network will most likely be more efficient than any conceived model as nodes struggle in the most Darwinian sense to provide liquid paths between paying nodes.

3) The revenue distribution is exponential above the 'bankruptcy' level due to inherent properties of networks. This holds true even if all nodes utilize the same strategies and are equally funded. It has been the narrative that the market dynamics would force down the fees towards the operational costs[1] - leading to low margins for node operators with little overall upside. This is only partially true in that routing fees will be competitive but not that it will lead to a low margin business.

Limitations with simulation

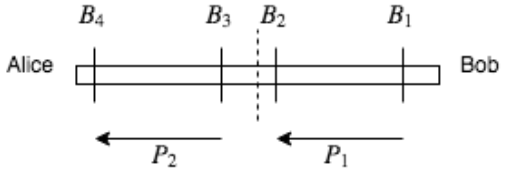
Simulating many nodes behavior when each depend on everyone else is not computationally easy. Even without calculating the best price for each edge and relying on the suggested probabilistic model. This reduces the amount of nodes that can be simulated, especially considering the scope of this thesis. Although a few aspects of smaller networks probably holds true in larger networks, such as the revenue distribution; it most likely does not for attachment. In the comparison of attachment models, the Barabasi-Albert and Hassan-Islam-Haque models comes up short - this is most likely due to the networks size. As hubs becomes more useful the larger the network.

[1] Including *time-value-of-money* and the *security-risk-premium*.

Balanced Channels

At first sight it may seem obvious that the fee price structure is flawed as it cannot incentive, e.g give a discount, for a route in such a direction so it leaves channels in a more balanced state. A change in fee structure obviously change the validity of each strategy.

Suppose there exists a channel between Alice and Bob with 10 million satoshis in it. Two payments, P_1 and P_2 are routed through the channel from Bob to Alice. Both use the same amount of liquidity, 3,500,000 satoshis each, but they start from two different initial states B_1 and B_3.



The two payments would incur the same fee,

$$F_{P_1 P_2} = F_B + (3,500,000 * F_R/1,000,000)$$

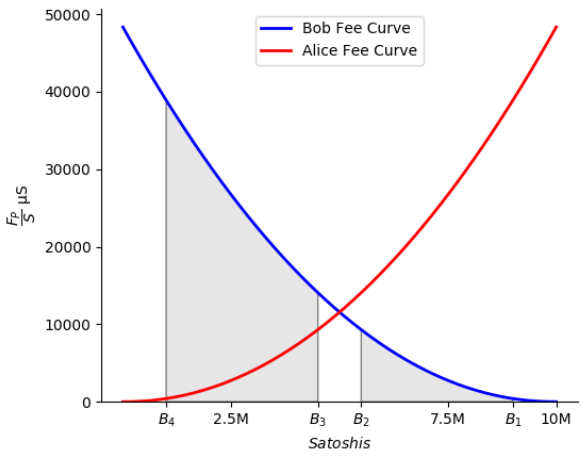
but leave the channels in completely different states. P_1 at B_2 and P_2 at B_4. It is of course possible to bump the channel price after P_1 to be more expensive, but suppose a third larger payment P_3 going from B_1 to B_4 would still run into this problem.

If a naive structure, that takes the state, is considered so as the issues of such a solution arise.

Each satoshi in the channel could be seen as a different bracket, with a different price for each satoshi moved. The channels would then broadcast a cost function instead of the fixed fee.

From the cost function the fee may be retrieved by calculating the area under the graphs. Some deterministic way to round the fee to whole satoshis and some way to verify the function is formulated correctly must be defined.

The fees for P_1, P_2 would be very different with this brackets method.



$$F_{P_1} = F_B + \int_{B_1}^{B_2} f(x)dx = F_B + 13,502,153,930\mu S$$

$$F_{P_2} = F_B + \int_{B_3}^{B_4} f(x)dx = F_B + 88,984,850,361\mu S$$

Here the fees are much larger for the payment that unbalances the channel than the payment that balances it.

However there are some clear problems with this approach. Every new payment would require each node on the path to rebroadcast the new balance to all other nodes[2]. Which would scale even worse than the Bitcoin bottom layer in message passing. There would also be privacy issues as any observant node could track the payments through balance updates[3].

There is thus no clear way to 'discount' routing in a balancing direction.

It is further unclear how much immediate balance will matter for routing success and other solutions have been proposed to levitate this pressure point. E.g René Pickhardt JIT-routing [76] and branching - utilizing multiple paths to fulfill the liquidity needed.

Future work

Future work might be interesting in many different directions.

[2]As brought up by Andrea Rapitzu. [78]
[3]As brought up by ZmnSCPxj. [94]

Protocol changes and growth

As the lightning network protocol is still under development and small changes in the Bitcoin protocol may change the dynamics of which the nodes operate. E.g. It seem probably that funding channels by both parties will be available in the near future.

It is difficult to say how universal these results are although surely many strategies suggested herein will still be successful with protocol changes. Each strategy is also dependent on what every other node in the network does. As the network grow - strategies and behavior will change. Thus the practicality of the results of these simulations are very context dependent, with introduction of new strategies, protocol changes etc. will require this model to be gradually updated to remain relevant.

Network theory

Albert-László Barabási recently remarked that the bottleneck in network theory is mainly data-driven [16]. The Lightning Network provides accurate network data with clear economic incentives for each actor. This may, in many ways, are unprecedented. There might be worth to study the network, in addition to its ability to scale Bitcoin, for what it suggest for networks in general. Many studies, cited herein, have been done on attachment heuristics in relation to fitness - it has however, been difficult to apply in practice to real networks. The Lightning Network may be a better empirical test-bed for models than any preceding network.

Game Theory

Much similar to Network Theory, Game theory have been difficult to use in empirical context. Especially in EGT as nature seldom have clear pay-off matrices. Even very clear examples, such as with the Bacteriophage shown by Turner and Chao [90, 72], still contains significant noise. Market data on the other hand, especially prices, is available in droves and it is arguable easier to apply game theory to economics. It usually still require effort to retrieve data, e.g. geo-locations of competing businesses. The Lightning Network might give us further insight into regional nuances and due to completeness bridge the gap between individual strategy with full market Marshallian dynamics.

Fee complexity

The optimal fee price for each edge will most likely be to expensive to calculate in the near future as the network grows. It is not uncertain how much low capacity nodes affect the network and larger operators might be able to simply prune away these nodes before calculation. Another approach is to approximate, calculating the upper and lower bounds with adjustable granularity, akin to tolerance [**?**].

Empirical gathering

There is still many assumptions in this the

(genetic algorithm)

(lightning factories)

References

[1] Gephi. https://github.com/gephi/gephi.

[2] Lightning, request for comments. https://github.com/lightningnetwork/lightning-rfc.

[3] *King James Bible*. 1611.

[4] Bitcoin core. https://github.com/bitcoin/bitcoin, 2009-.

[5] The coinbase transaction of the genesis block.
TX 4a5e1e4baab89f3a32518a88c31bc87f618f76673e2cc77ab2127b7afdeda33b, 2009.

[6] coin.dance. https://coin.dance/, 2019.

[7] The world gold council. https://gold.org, 2019.

[8] Acinq. eclair. https://github.com/ACINQ/eclair.

[9] Gavin Andersen. Bip 0013 address format for pay-to-script-hash, January 2012.

[10] Gavin Andersen. Bip 0016 pay to script hash, January 2012.

[11] Gavin Andersen. Bip 0101 increase maximum block size, June 2015.

[12] Andreas Antonopoulos. *Mastering Bitcoin*. O'Reilly Media, first edition, July 2014.

[13] Andreas Antonopoulos. *The Internet of Money*. Merkle Bloom LLC, first edition, August 2016.

[14] Andreas Antonopoulos. Segregated witness and aligning economic incentives with resource costs. August 2016.

[15] Adam Back. Hashcash - a denial of service counter-measure. 2002.

[16] Albert-László Barabási. Scale-free networks: A decade and beyond. *Science*, 325:412–413, July 2009.

[17] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, October 1999.

[18] Michael Bell, Supun Perera, Mahendrarajah Piraveenan, Michiel Bliemer, Tanya Latty, and Chris Reid. Network growth models: A behavioural basis for attachment proportional to fitness. 2008.

[19] Nik Bhatia. The time value of bitcoin. June 2018.

[20] Ginestra Bianconi and Albert-László Barabási. Competition and multiscaling in evolving networks. *Europhysics letters*, 54:436–442, March 2001.

[21] Bitmain. Regarding recent allegations and smear campaigns. April 2017.

[22] blockstream. c-lightning. https://github.com/ElementsProject/lightning.

[23] Michael D. Bordo. The imbalances of the bretton woods system 1965 to 1973: U.s. inflation, the elephants in the room.

[24] Kenneth E. Boulding. *Evolutionary Economics*. SAGE publications Inc, first edition, 1981.

[25] Ulrik Brandes. On variants of shortest-path betweenness centrality and their generic computation. November 2007.

[26] BtcDrak, Mark Freienbach, and Eric Lombrozo. Bip 0112 checksequenceverify, august 2015.

[27] btcsuite. btcd. `https://github.com/btcsuite/btcd`.

[28] Duncan S. Callaway, John E. Hopcroft, Jon M. Kleinberg, M. E. J. Newman, and Steven H. Strogatz. Are randomly grown graphs really random?

[29] Christan Decker and Roger Wattenhofer. A fast and scalable payment network with bitcoin duplex micropayment channels.

[30] Christian Decker. Bip 0118 sighash noinput, February 2017.

[31] Giovanni Di Stasi, Stefano Avallone, Roberto Canonico, and Giorgio Ventre. Routing payments on the lightning network. 2018.

[32] James A. Donald. Re: Bitcoin p2p e-cash paper. Cyberpunk mailing list, 2008.

[33] David Easley and Jon Kleinberg. *Networks, Crowds and Markets*. Cambridge University Press, 2010.

[34] Paul Erdős and Albért Rényi. On random graphs i.

[35] TX f4184fc596403b9d638783cf57adfe4c75c605f6356fbc91338530e9831e9e16. block 170.

[36] Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40:35–41, Mars 1977.

[37] Mark Freienbach, BtcDrak, Nicolas Dorier, and kinoshitajona. Bip 0068 relative lock-time using consensus-enforced sequence numbers, May 2015.

[38] Shaolin Fry. Bip 0148 mandatory activation of segwit deployment, 2017.

[39] Jeff Garzik. Bip 0102 block size increase to 2mb, June 2015.

[40] Jeff Garzik, Tom Harding, and Dagur Valberg Johannsson. Bip 0100 dynamic maximum block size by miner vote, august 2015.

[41] Cyril Grunspan and Ricardo Perez-Marco. Ant routing algorithm for the lightning network.

[42] William D. Hamilton. The genetical evolution of social behaviour. i. *Journal of Theoretical Biology*, 7:1–16, February 1964.

[43] Steve H. Hanke. Venezuela hyperinflation, October 2018.

[44] Steve H. Hanke and Nicholas Krus. *World Hyperinflations*. August 2012.

[45] Bakhtiar Hasan. *Algorithms Notes for Professionals*.

[46] Kamrul Hassan, Liana Islam, and Syed Arefinul Haque. Degree distribution, rank-size distribution, and leadership persistence in mediation-driven attachment networks. *Physica A*, 469:23–30, 2017.

[47] Mike Hearn. Why is bitcoin forking, august 2015.

[48] Nicolas Houy. The economics of bitcoin transaction fees. February 2014.

[49] Donald B. Johnson. Efficient algorithms for shortest paths in sparse networks. *Journal of the Association for Computing Machinery*, 24:1–13, January 1977.

[50] Thomas Kerin and Mark Freienbach. Bip 0113 median time-past as endpoint for lock-time calculations, august 2015.

[51] John Maynard Keynes. *The General Theory of Employment, Interest, and Money*. 1936.

[52] Lightning labs. Lightning node daemon. `https://github.com/lightningnetwork/lnd`.

[53] Johnson Lau. Bip 0147 dealing with dummy stack element malleability.

[54] lightninglabs. Neutrino. `https://github.com/lightninglabs/neutrino`.

[55] Nils Linder. *Nordisk Familjebok*. C. E. Gernandt, second edition, 1911.

[56] Nils Linder. *Nordisk Familjebok*. C. E. Gernandt, second edition, 1913.

[57] Ferdinand Lips. *Gold Wars The Battle Against Sound Money As Seen From A Swiss Perspective*. The Foundation for the Advancement of Monetary Education, 2001.

[58] Eric Lombrozo, Johnson Lau, and Pieter Wuille. Bip 0141 segregated witness, December 2015.

[59] Gregory Maxwell. Inhibiting a covert attack on the bitcoin pow function. Bitcoin-dev mailing list, Linuxfoundation, April 2017.

[60] Carl Menger. On the origins of money. *Economic Journal*, pages 239–55, 1892.

[61] Robert K. Merton. The matthew effect in science. *Science*, 186(3810):56–63, January 1968.

[62] MITDCI. lit. `https://github.com/mit-dci/lit`.

[63] Malte Möser1and and Rainer Böhme. Trends, tips, tolls: A longitudinal studyof bitcoin transaction fees. *Lecture Notes in Computer Science*, January 2015.

[64] Satoshi Nakamoto. fix openssl linkage problems,. bitcoin commit a30b56ebe76ffff9f9cc8a6667186179413c6349.

[65] Satoshi Nakamoto. only accept transactions sent by ip address if -allowreceivebyip is s.... bitcoin commit 172f006020965ae8763a0610845c051ed1e3b522.

[66] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. October 2008.

[67] Satoshi Nakamoto. main.cpp, bitcoin line #1673, 2009.

[68] Satoshi Nakamoto. bitcointalk.org topic 13 #46, 2010.

[69] Satoshi Nakamoto. bitcointalk.org topic 195 #1611, 2010.

[70] Satoshi Nakamoto. bitcointalk.org topic 57 #415, 2010.

[71] John F. Nash JR. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36:48–49, 1950.

[72] Martin A. Nowak and Karl Sigmund. Phage-lift for game theory. *Nature*, 398:441–443, April 1999.

[73] Federal Reserve Bank of Saint Louis. Functions of money. economic-lowdown.

[74] R. H. Patterson. On our home monetary drains, and the crisis of 1866. *Journal of the Statistical Society of London*, 33(2):216–242, June 1870.

[75] René Pickhardt. Automatically-generating-a-robust-topology-for-the-lightning-network-on-top-of-bitcoin, 2019.

[76] René Pickhardt. Just in time routing (jit-routing) and a channel rebalancing heuristic. Lightning-dev mailing list, March 2019.

[77] Joseph Poon and Thaddeus Dryja. The bitcoin lightning network: Scalable off-chain instant payments. 2016.

[78] Andrea Raspitzu. re: Fee structure. Lightning-dev mailing list, March 2019.

[79] BitMEX Research. Rouing fee economics. March 2019.

[80] Franklin D. Roosevelt. Executive order 6102.

[81] Franklin D. Roosevelt. The gold reserve act, January 1934.

[82] Ammous. Saifeadean. *The Bitcoin Standard*. Wiley, mars 2018.

[83] John M. Smith. *Evolution and the Theory of Games*. Cambridge University Press, 1982.

[84] John M. Smith and G.R. Price. The logic of animal conflict. *Nature*, 246:15–18, November 1973.

[85] Jimmy Song. Examining bitmain's claims about asicboost. April 2017.

[86] Jimmy Song. Transaction malleability explained. August 2017.

[87] Nick Szabo. Unenumerated. `http://unenumerated.blogspot.com/`.

[88] Nick Szabo. Smart contracts. 1994.

[89] Nick Szabo. Shelling out. 2002.

[90] Paul E. Turner and Lin Chao. Prisoner's dilemmain an rna virus. *Nature*, 398:441–443, April 1999.

[91] Warren E. Weber. A bitcoin standard: Lessons from the gold standard. October 2015.

[92] Pieter Wuille. Bip 0032 hierarchical deterministic wallets, February 2012.

[93] Pieter Wuille. Bip 0103 block size following technological growth, June 2015.

[94] ZmnSCPxj. re: Fee structure. Lightning-dev mailing list, March 2019.

Appendices

Strategy Guide

This appendix contains a short summary of each implemented strategy and tunable environment variable. For examples of how simulations may be created a set of configured presets, used to generate the results, are available at `https://github.com/Johnstedt/lightning-strategy-game-simulation/tree/master/src/presets`.

Simulation Environment

initial_nodes `[amount]`

The amount of nodes in the beginning of the simulation.

payment_distribution `[json]`

The distribution of the size of the payments.

payments_per_step `[amount]`

The amount of payments each step.

risk_free_rent `[annual_percent]`

The risk free interest rate. In annual percentage of liquidity.

risk_premium `[annual_percent]`

The security risk premium. In annual percentage of liquidity.

operational_cost `[annual_cost(satoshis)]`

The annual operational cost in satoshi.

new_nodes_per_step `[amount]`

The amount of new nodes each simulation step.

new_nodes_policy `[option]`

The policy by which the strategy of the new node is selected. Either

new_nodes_quantity `[json]`

time_steps `[steps]`

fee `[amount(satoshis)]`

initial_mode `[option]`

"payment$_distribution$" : "$type$" : "$random$","$interval$" : "low" : 2000,"$high$" : 2000000,"$initial_private_nodes$" : 80,"$payments_per_step$" : 600,"$risk_free_rent$" : 0.30,"$risk_premium$" : 0.17,"$operational_cost$" : 200000,"$immunity_period$" : 5,"$new_nodes_per_step$" : 2,"$changed_private_per_step$" : 0,"new_nodes_policy" :

"$mirror$","$new_node_quantity$" : "$percentage$","$time_steps$" : 500,"fee" : 10000,"$initial_mode$" : "$exact$"

Strategies

Following is a list of all implemented strategies and their arguments,

Fee Strategies

The Fee Strategy determine which fee(F_B, F_P) to set to each channel.

random_fee `interval_base interval_proportional`

Sets the base- and proportional fee randomly given intervals. This is used as a sanity check benchmark for the other strategies.

probability_model `path-to-json`

Each edge price is set following a json with four attributes. An array of the range of base fees accompanied another array with the probability for each price. Similarly an array of the range of proportional fees and an array with their probability. These models may be generated with `price_strategies.py`, with simulates a simulation and selects a central edge, calculates the optimal fee curve from chapter 6, and creates a model with probability correlating with expected relative profit. The price model may be used in a new simulation generating a new model. This process can be repeated many times.

Preferential Attachment Strategies

The Preferential attachment determine the bias a node use in selecting a node to open a channel with.

random_attachment

Selects a random node in the network and opens a channel. It must not be confused with the Erdős–Rényi model as it has an innate age bias.

barabasi-albert

Selects a node in the network with a probability dependent on degree according to the Barabási-Albert model. The probability is given by:

$$\Pi(i) = \frac{k_i}{\sum_j k_j}$$

where $\Pi(i)$ is the probability to select i with degree k_i.

hassan_islam_haque

Randomly chooses a node and then selects one of its neighbors randomly according to the Mediation-driven attachment model. The probability is given by:

$$\Pi(i) = \frac{1}{N}\left[\frac{1}{k_1} + \frac{1}{k_2} + ... + \frac{1}{k_{k_1}}\right] = \frac{\sum_{j=1}^{k_i} \frac{1}{k_j}}{N}$$

where $\Pi(i)$ is the probability to select i with degree k_i.

inverse_barabasi_albert

Selects a node in the network with a probability inversely dependent on degree. The probability is given by:

$$\Pi(i) = \frac{\frac{1}{k_i}}{\sum_j^n \frac{1}{k_j}}$$

where $\Pi(i)$ is the probability to select i with degree k_i.

Timing Strategies

The Timing strategies determine when a node should close a channel.

sanity_check

Does not close any channel whatsoever. It is used as a sanity check benchmark for the other strategies.

bankruptcy_avg_usage

Does close a channel if the usage, on average, leads to bankruptcy. A channel is closed if and only if:

$$\frac{recent\_profit}{channel\_liquidity} < \frac{floor\_bankrupcy}{total\_funding}$$

bankruptcy_avg_usage_aggressiveness `scalar`

Similar to **bankruptcy_avg_usage** but scaled with an aggressiveness scalar:

$$\frac{recent\_profit}{channel\_liquidity} < \frac{floor\_bankrupcy}{total\_funding}\texttt{scalar}$$

$$\texttt{scalar} \in \mathbb{R} \text{ and } 0 < \texttt{scalar}$$

A `scalar` greater than 1 may be called a *hawkish* strategy and one lesser than 1 may be called *dovish*.

Funding strategies

The funding strategy determine how much the node has available to fund channels.

funding `[satoshis]`

Funds the node with `satoshis`. Although relative a node with large quantity may be called a *rich* strategy and one with small quantity a *poor* strategy. It may seem dissonant to reefer to wealth as a strategy but remember that profit is relative to funding and one operator may have many nodes instead of one large if a *poor* strategy has higher *fitness*. Further beetle size is similarly referred to as a strategy in EGT.

Re-balance

The re-balancing strategy determine if it is worth to re-balance a pair of channels by routing a payment to oneself.

sanity_check

Does not re-balance channels whatsoever. It is used as a sanity check benchmark for the other strategy.

linear_displacement `[depth]` `[ratio]`

Does a *breath-first-search* for cycles with `[depth]`. If the `[ratio]` given by an Utility(6.7.1) as,

$$U(c) = \begin{cases} \dfrac{2|(B_{1S} - B_{1M})|}{f_c(|B_{1S} - B_{1M}|)} & if\ B_{1S} - B_{1M} < B_{2M} - B_{2S} \\ \\ \dfrac{2|(B_{2M} - B_{2S})|}{f_c(|B_{2M} - B_{2S}|)} & otherwise \end{cases}$$

where B_{1S}, B_{2S} are the states of the out and in channels and B_{1M}, B_{2M} their middles and f is the total fee for the cycle c.

$$f_c(L) = \sum_{i=1}^{n} B_i + P_i L$$

with B_i is the base fee, P_i is the proportional fee for node i in path c.

edge_biased_displacement `[depth]` `[ratio]` `[scalar]`

Does a *breath-first-search* for cycles with `[depth]`. If the `[ratio]` given by a biased to aforementioned Utility function as,

$$U'(c) = U(c)\left(1,5 - \left(1 - \frac{B_{1S}}{2B_{1M}}\right)\right)^s \left(1,5 - \left(\frac{B_{2S}}{2B_{2M}}\right)\right)^s$$

where s is a scalar given as argument `[scalar]`