HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

ANGEWANDTE COMPUTER- UND BIOWISSENSCHAFTEN

LARA BISCHOFF

MIMBLEWIMBLE

18.09.2019

EIGENSCHAFTEN

- Ist ein Blockchain-Protokoll
- Bietet standardmäßige Privatsphäre
- Skaliert hauptsächlich mit der Anzahl der Nutzer und minimal mit der Anzahl an Transaktionen
- Benutzt keine Skriptsprache sondern nur Kryptographie

GLIEDERUNG

1. Entstehung des Protokolls
2. Grin und Beam
3. Pedersen Commitments
4. Schnorr Signaturen
5. Blockchain Aufbau
6. Scriptless Scripts
7. Dandelion
8. Cuckoo Cycle Mining Algorithmus

HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

MIMBLEWIMBLE

ENTSTEHUNG DES PROTOKOLLS

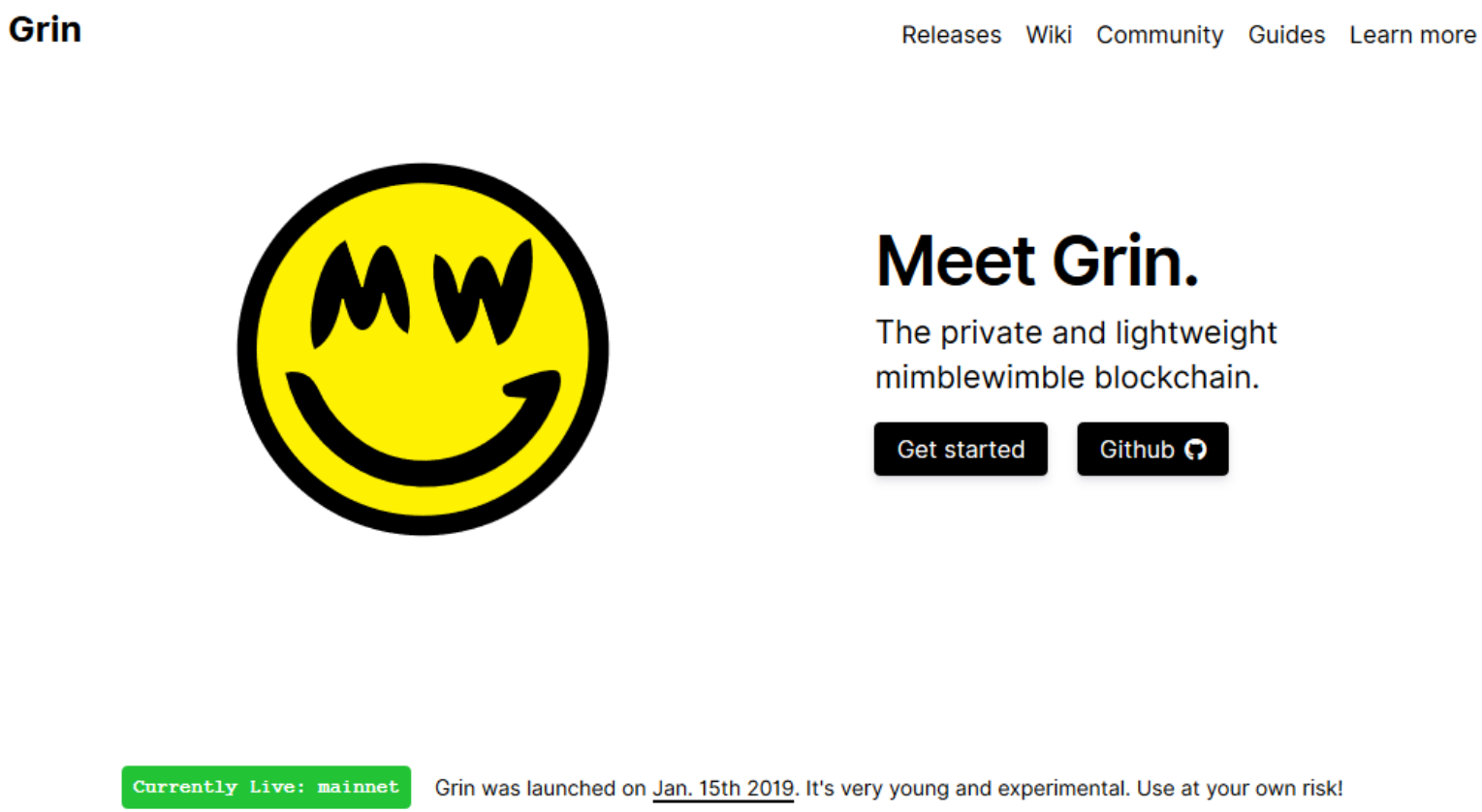
LARA BISCHOFF

ENTSTEHUNG DES PROTOKOLLS

- 2. August 2016 wurde eine Textdatei mit dem Titel MIMBLEWIMBLE über einen .onion-Link zu einer Textdatei von "Tom Elvis Jedusor" veröffentlicht (Name des Voldemord in den französischen Harry Potter-Büchern)
- Mimblewimble ist ein Zauberspruch der Harry Potter Reihe und verhindert, dass das Opfer eine zusammenhängende Sprache sprechen kann
- Tom Elvis Jedusor in seinem Paper: „ I call my creation Mimblewimble because it is used to prevent the blockchain from talking about all user's information”
- Das ursprüngliche Protokoll war nicht sehr detailliert und enthielt sogar einen Fehler, was durch ein neues Paper am 8. Oktober von Andrew Poelstra korrigiert wurde
- Am 20. Oktober erschien "Ignotus Peverell" und kündigte ein Projekt zur Implementierung von Mimblewimble an → das GitHub-Projekt Grin startete

IMPLEMENTIERUNGEN DES PROTOKOLLS

	Grin	Beam
Gestartet	Q4 2016	Q1 2018
Hintergrund	Community Projekt gestartet von Ignotus Peverell	Israelische Firma
Finanzierung	Community	Teil des Mining Rewards
Programmiersprache	Rust	C++
Launch	Anfang 2019	Anfang 2019
Mining	Cuckoo Cycle	Equihash
Cap	Kein Cap	Cap bei 263 Mio. Coins
Ausschüttung	60 Grin/Minute	80 Beam/Minute



https://grin-tech.org/

How to run a Grin node

Michalis Kargakis edited this page on 21 Apr · 17 revisions

This page is intended to provide basic instructions on how to get started running a Grin node.

Prerequisites

- Linux (x86-64 only) or MacOS
- Relatively recent hardware
- Some Linux command line knowledge very helpful
- rust 1.31+ (use rustup- i.e. `curl https://sh.rustup.rs -sSf | sh; source $HOME/.cargo/env`)
  - if rust is already installed, you can simply update version with `rustup update`

Downloading the Software

The latest Grin release can be found on the project Release Page. Binaries are currently provided for Linux and OSX.

Distribution-specific releases will likely appear over time, and will be listed here.

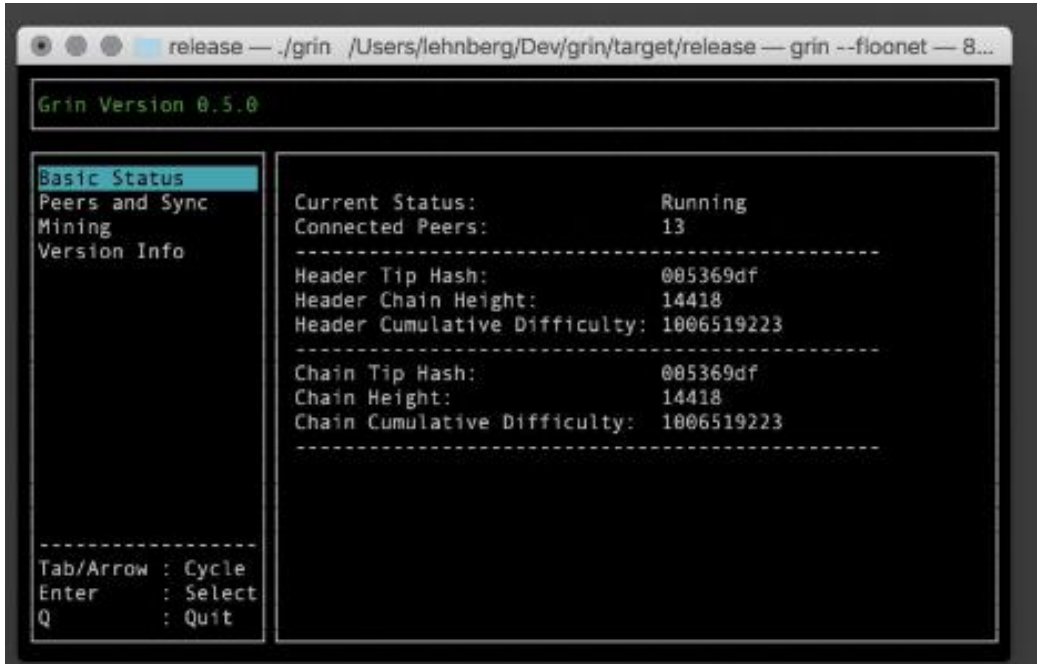
Installation

Linux (all distributions)

Download the zipped binary to your machine, and unzip it using the terminal. This will unzip a single file called `grin`, which contains both the server and wallet software.

https://github.com/mimblewimble/docs/wiki/How-to-run-a-Grin-node

GRIN



D. Lehnberg: „A layperson‘s intro to Grin“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grinconUS0/02-Lehnberg-Layperson_Intro.pdf

GRIN

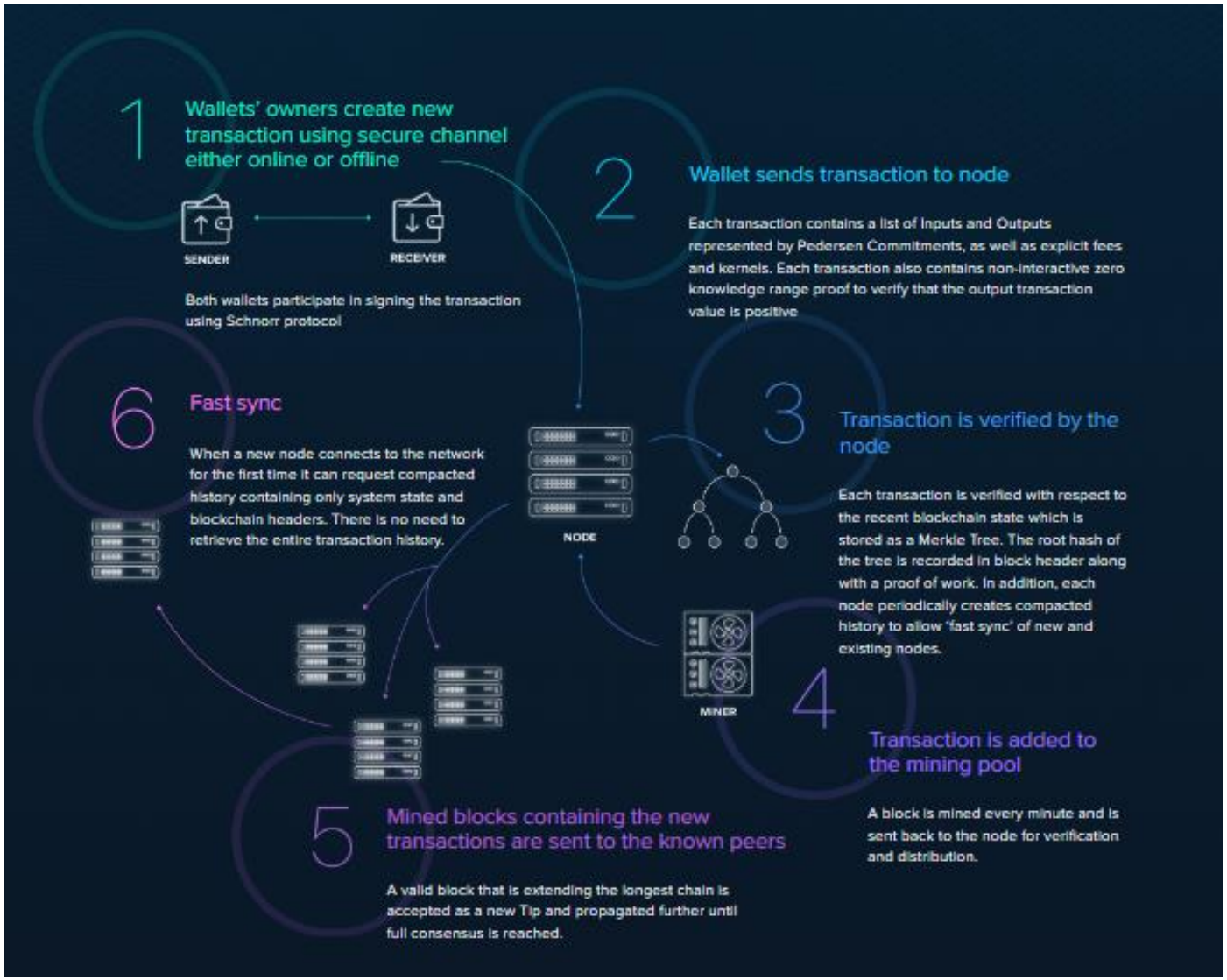
M. Cordner: „The Future of Grin“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grinconUS0/13-Cordner-Future_of_Grin.pdf

BEAM

https://www.beam.mw/

BEAM

https://www.beam.mw/



https://www.beam.mw/

HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

MIMBLEWIMBLE

PEDERSEN COMMITMENTS

CONFIDENTIAL TRANSACTIONS

- Confidential Transactions (CT) erlauben es, die *Values* (Werte) einer Transaktionen unkenntlich zu machen.
- Die expliziten Beträge der Transaktionen wird durch ein Commitment ersetzt:
  - Die in den Outputs enthaltenen Werte werden verborgen
  - Gleichzeitig wird sichergestellt, dass der Ersteller die Werte nachträglich nicht mehr verändern kann.

- Der Betrag v wird durch ein Pedersen-Commitment C ersetzt:

$$\boldsymbol{C = vH + rG}$$

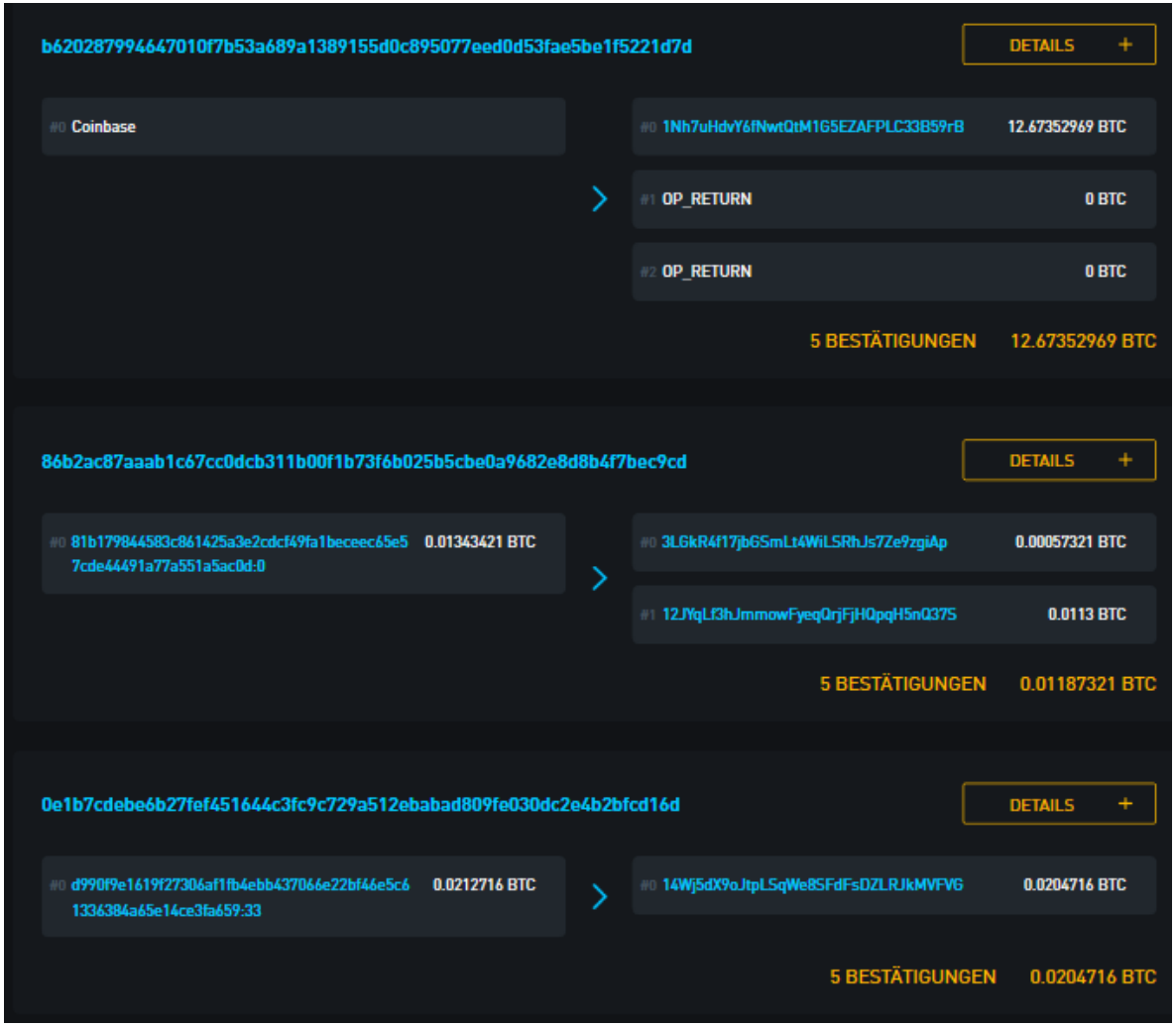
H, G … Generator Punkte auf einer Gruppe mit hartem diskreten Logarithmusproblem

r … (random number) Zufallszahl

v … (value) zu versendender Betrag

G. Fuchsbauer, M. Orrù, Y. Seurin: „Aggregate Cash Systems: A Cryptographic Investigation of Mimblewimble".

CONFIDENTIAL TRANSACTIONS



https://blockstream.info/block/00000000000000000015010a65777cdb05c87b8b11a7249720dc1eb6cc528c2a

CONFIDENTIAL TRANSACTIONS

Overview | **Inputs (5)** | Outputs (11) | Kernels (6) | Comments

#	Commit
1	09846ac0e92f3f1eaeb637c0218a8da995449d50a3c2d4cdbc2b0594df28cf7497
2	080c6825ed6716149c1ff34d3320b4216cf988030508c74d130b2b182a58c7e935
3	095b82543b69356a3d9f7c7083ba2afffb7757ade6d40460125968197012ba129f

Overview | Inputs (5) | **Outputs (11)** | Kernels (6) | Comments

#	Type	Commit	Spent
1	Transaction	09d9838660935a5cf143f0c399af2dcdcb907cfa4e33f923c5b210c4909adc8988	Unspent
2	Transaction	0897d99bc55ad679bb858743ef8fc0aa0cac15cb756ef2fbedbb817d04a4fbefdf	Unspent
3	Transaction	08d45ae3bda8d6f9651f12e4503bc27c3824cf6d72ded2381a1491fbaf3abb5fa9	Unspent
4	Transaction	0886155dd20b3e4302394c9ba54cf2c590a84352560e0c23c30097b3d9fdcfee0d	Unspent

Blockscan: https://grin.blockscan.com/block/216283

PEDERSEN COMMITMENT

- In Mimblewimble gibt es keine Adressen oder Skripte

- Um einen Coin auszugeben, braucht man nur das Wissen von der Eröffnung des Commitments
- $C = vH + rG \rightarrow r$ fungiert als Private Key für dieses Commitment

- Genau wie in Bitcoin enthält eine Mimblewimble Transaktion eine Liste von Inputs $C = (C_1, \dots, C_n)$ und eine Liste von Outputs $\hat{C} = (\hat{C}_1, \dots, \hat{C}_m)$.
- Ist eine Transaktion ausgeglichen, so gilt (Transaktionsgebühren vernachlässigt):

$$\sum \hat{C} - \sum C = \sum (\hat{v}_i H + \hat{r}_i G) - \sum (v_i H + r_i G) = \left(\sum \hat{r}_i - \sum r_i \right) G$$

- In anderen Worten: das Wissen über r der beteiligten Transaktionen impliziert das Wissen über $E := \sum \hat{C} - \sum C$, dies wird *Excess* einer Transaktion genannt
- Das Wissen über r kann durch das Erstellen einer gültigen Signatur (auf eine leere Nachricht) mit dem öffentlichen Schlüssel E nachgewiesen werden

G. Fuchsbauer, M. Orrù, Y. Seurin: „Aggregate Cash Systems: A Cryptographic Investigation of Mimblewimble".

SCHNORR SIGNATUREN

1. Öffentlicher und privater Schlüssel: $\boldsymbol{P = xG}$
2. Geheime nonce $\boldsymbol{k}$ generieren
   → "Flüchtiges" Schlüsselpaar: $\boldsymbol{R = kG}$
3. Hash-Funktion berechnen: $\boldsymbol{e = H(P, R, m)}$
4. Signatur erstellen: $\boldsymbol{s = k + ex}$

Bob möchte diese Signatur überprüfen

Er kann ebenfalls $\boldsymbol{e}$ berechnen, da $(\boldsymbol{P}, \boldsymbol{R}, \boldsymbol{m})$ öffentlich sind

Aber er kennt die geheime Nonce $\boldsymbol{k}$ und den privaten Schlüssel $\boldsymbol{x}$ nicht

$$\boldsymbol{sG = kG + exG}$$
$$\boldsymbol{sG = (kG) + (xG)e}$$
$$\boldsymbol{sG = R + Pe}$$

→ Bob muss also $(\boldsymbol{sG})$ berechnen und überprüfen, ob dies mit der rechten Seite der Rechnung $(\boldsymbol{R + Pe})$ übereinstimmt

A. Poelstra. „Threshold Signatures and Accountability". [Online]. Available: https://www.youtube.com/watch?v=j9Wvz7zI_Ac&t=1173s

SCHNORR SIGNATUREN

- In der einfachsten Form ist die Signatur wie folgt aufgebaut:
  - Eine Nachricht zum signieren, in diesem Fall die Transaktionsgebühr
  - Ein Private Key x mit seinem korrespondierendem Public Key $P = xG$
  - Eine Nonce k, die nur zum Zweck der Erstellung der Signatur verwendet wird

$$\sum \hat{C} - \sum C = \sum (\hat{v}_i H + \hat{r}_i G) - \sum (v_i H + r_i G) = \left(\sum \hat{r}_i - \sum r_i \right) G$$

- Private Key $\boldsymbol{x} = (\sum \hat{r}_i - \sum r_i)$ → korrespondierender Public Key $\boldsymbol{P = xG} = (\sum \hat{r}_i - \sum r_i)G$

- Dieses Protokoll kann auf eine beliebige Anzahl i von Parteien verallgemeinert werden.

I. Peverell et al. „Contracts“. [Online]. Available: https://github.com/mimblewimble/grin/blob/master/doc/contracts.md

RANGE PROOF

- Bemerkung: Die Berechnungen der *values* werden modulo p ausgeführt (p entspricht der Ordnung der bezüglichen Gruppe)
- Dadurch könnte ein Angreifer eine Transaktion anlegen, die Geld erschafft.

- Um sicherzustellen, dass ein Commitment kein zu großes *value* enthält, wird ein "non-interactive zero-knowledge (NIZK) proof" hinzugefügt.
- Dieser beweist, dass das *value* in dem Intervall $[0, v_{max}]$ liegt (sogenannter *range proof*), wobei v_{max} kleiner als *p* ist.

G. Fuchsbauer, M. Orrù, Y. Seurin: „Aggregate Cash Systems: A Cryptographic Investigation of Mimblewimble".

PEDERSEN COMMITMENT

- Zusammengefasst ist eine Mimblewimble Transaktion ein Tubel $tx = (s, C, \hat{C}, K)$ mit $\mathrm{K} \coloneqq (\pi, E, \sigma)$

s … Anzahl der Coins für die Coinbase Transaktion

C … Input Liste

$\hat{C}$ … Output Liste

K … sogenannter Kernel

π … Liste der *range proofs* für die Outputs

E … Liste der *Excesses* der Transaktion

σ … Signatur

G. Fuchsbauer, M. Orrù, Y. Seurin: „Aggregate Cash Systems: A Cryptographic Investigation of Mimblewimble".

CUT-THROUGH

- Betrachtet man $tx_0 = \left(s_0, C_0, \hat{C}_0, (\pi_0, E_0, \sigma_0)\right)$ und $tx_1 = \left(s_1, C_1, \hat{C}_1, (\pi_1, E_1, \sigma_1)\right)$ dann ist die aggregierte Transaktion tx aus der Zusammenführung von tx_0 und tx_1 einfach:

$$tx := \left(s_0 + s_1, C_0 \| C_1, \hat{C}_0 \| \hat{C}_1, (\pi_0 \| \pi_1, (E_0, E_1), \sigma)\right)$$

σ … aggregierte Signatur für die Public Keys $E := (E_0, E_1)$

- Eine aggregierte Transaktion ist gültig, wenn alle range proofs gültig sind und σ eine gültige aggregierte Signatur für E ist
- Transaktionen können rekursiv aggregiert werden, der Kernel enthält dabei eine Liste von E, eins für jede Transaktion die aggregiert wurde

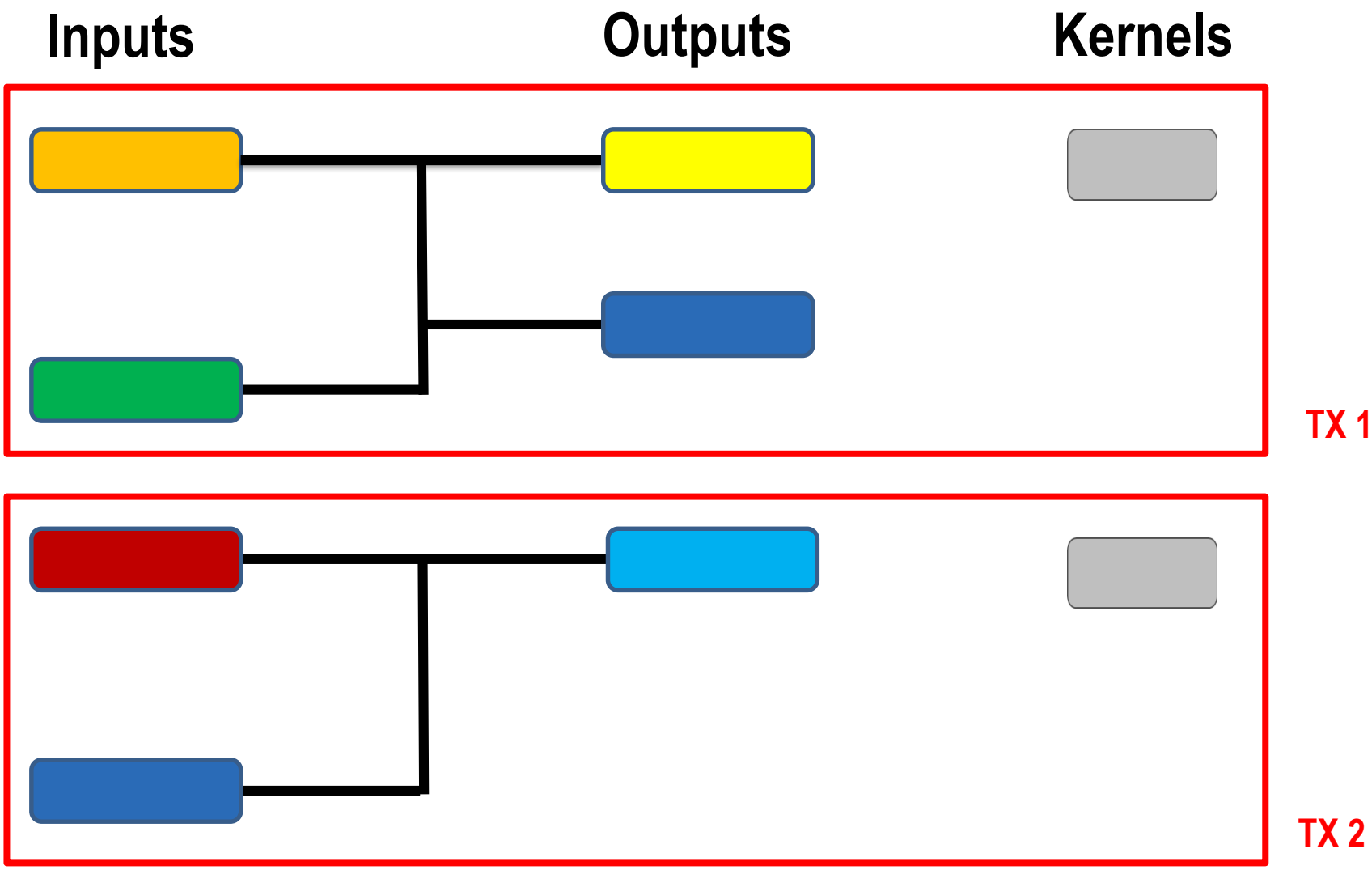
- **Cut-through:**
  - Angenommen wir haben einen Coin, der in zwei Transaktionen vorkommt
  - Er taucht in Output tx_0 auf und wird durch den Input in tx_1 referenziert
  - Dadurch kann man den Coin aus den Ein- und Ausgabelisten der aggregierten Transaktion tx löschen und tx ist weiterhin gültig.

G. Fuchsbauer, M. Orrù, Y. Seurin: „Aggregate Cash Systems: A Cryptographic Investigation of Mimblewimble".

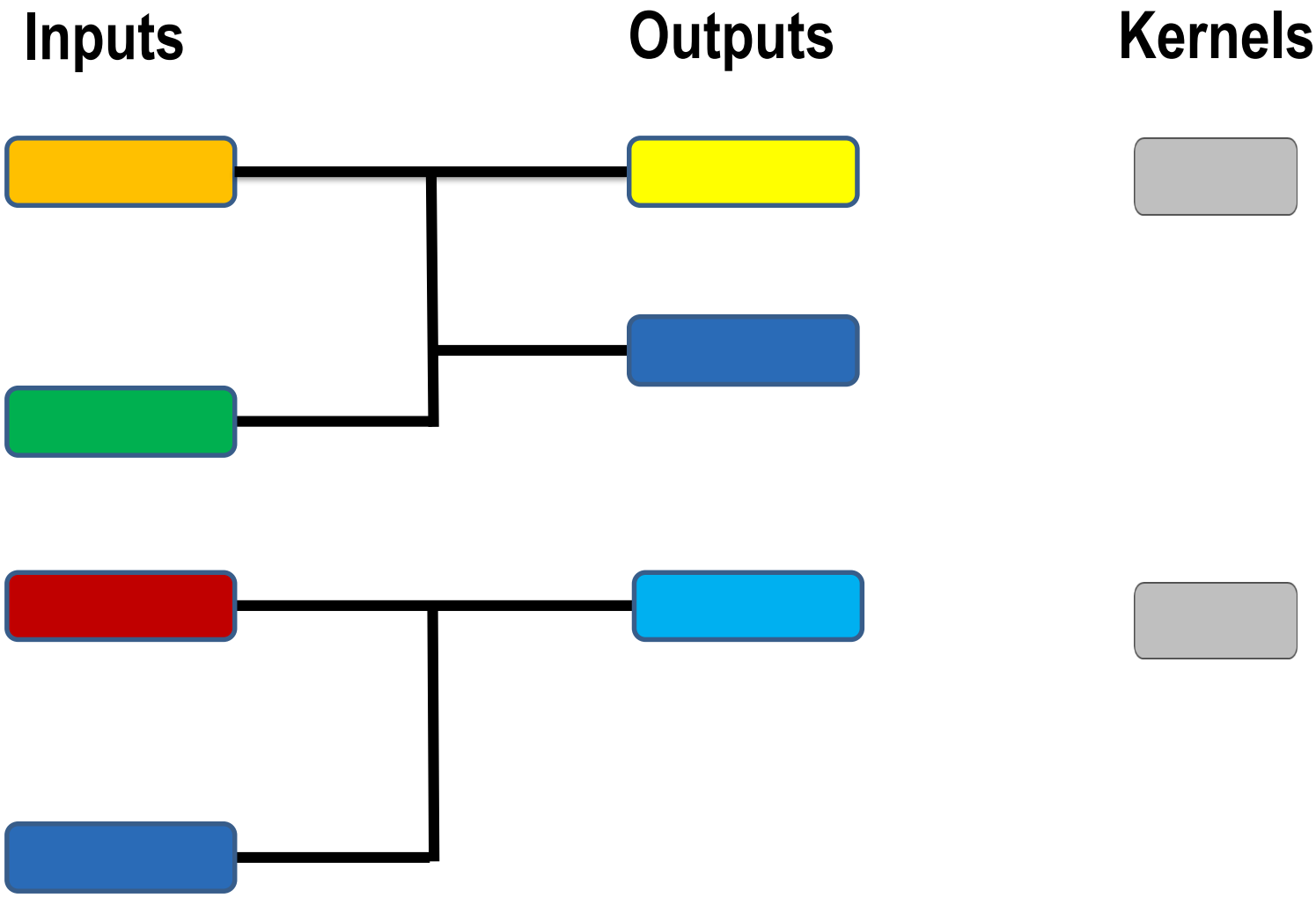
HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

MIMBLEWIMBLE

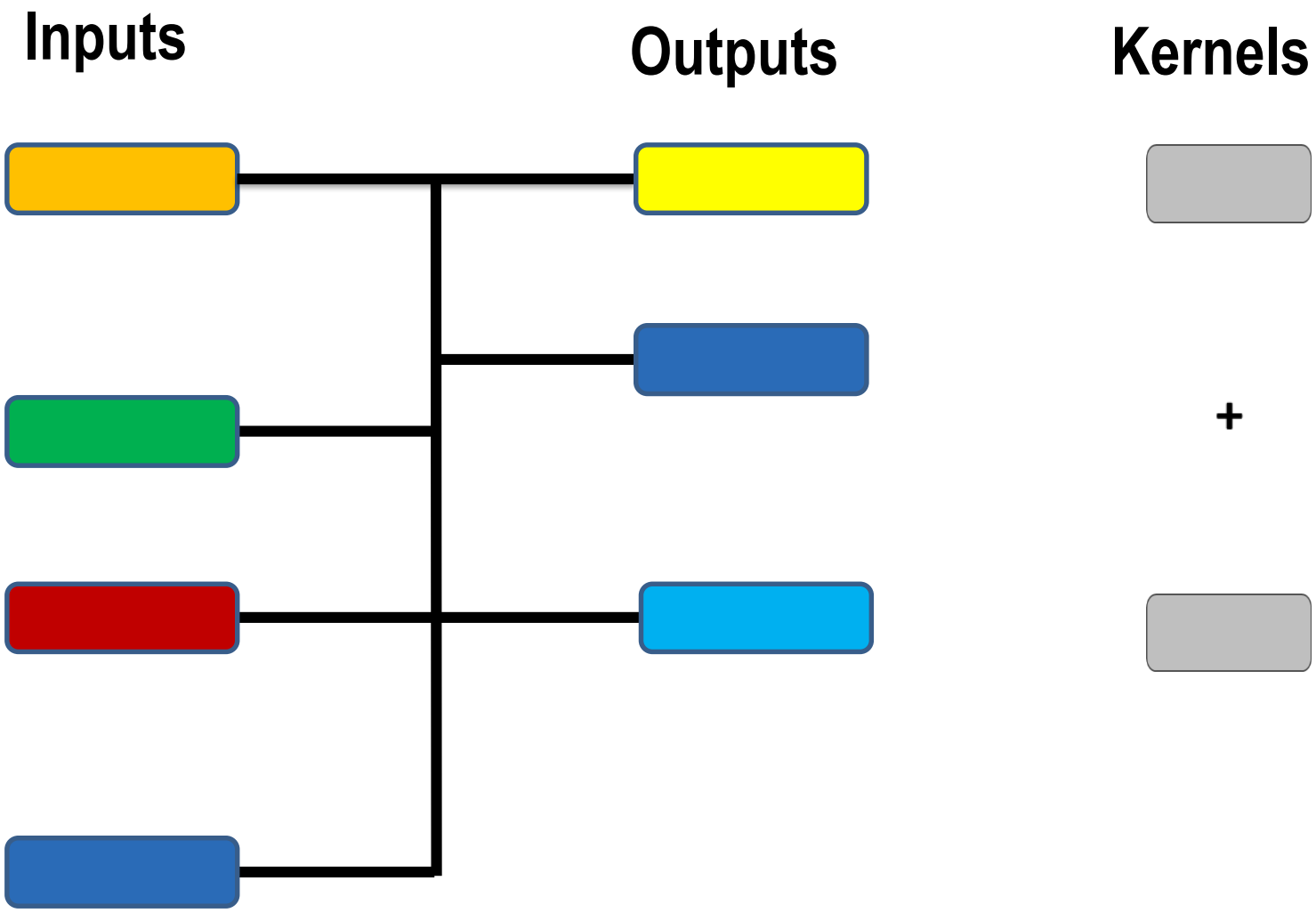
AUFBAU DER BLOCKCHAIN



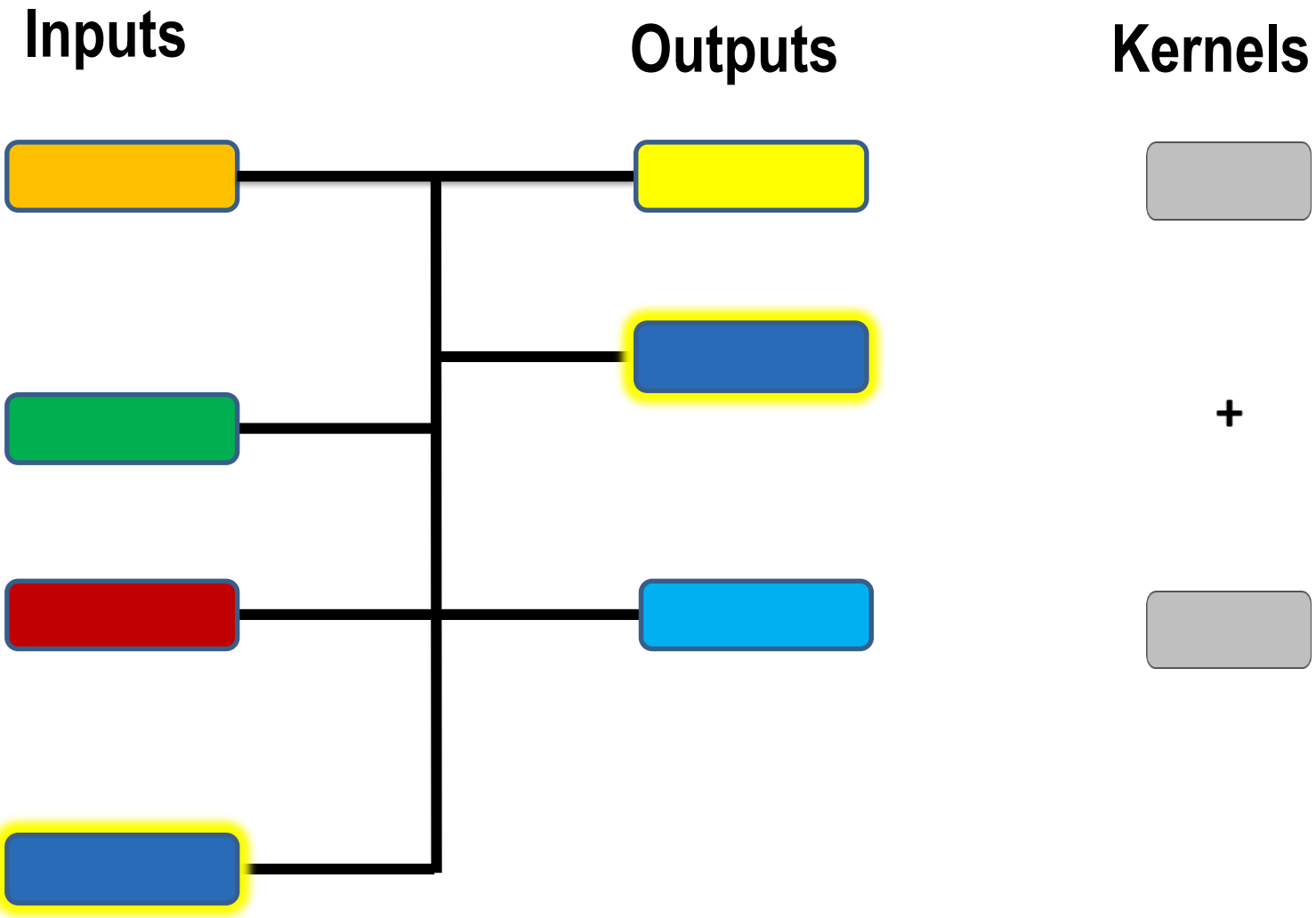
A. Poelstra



A. Poelstra

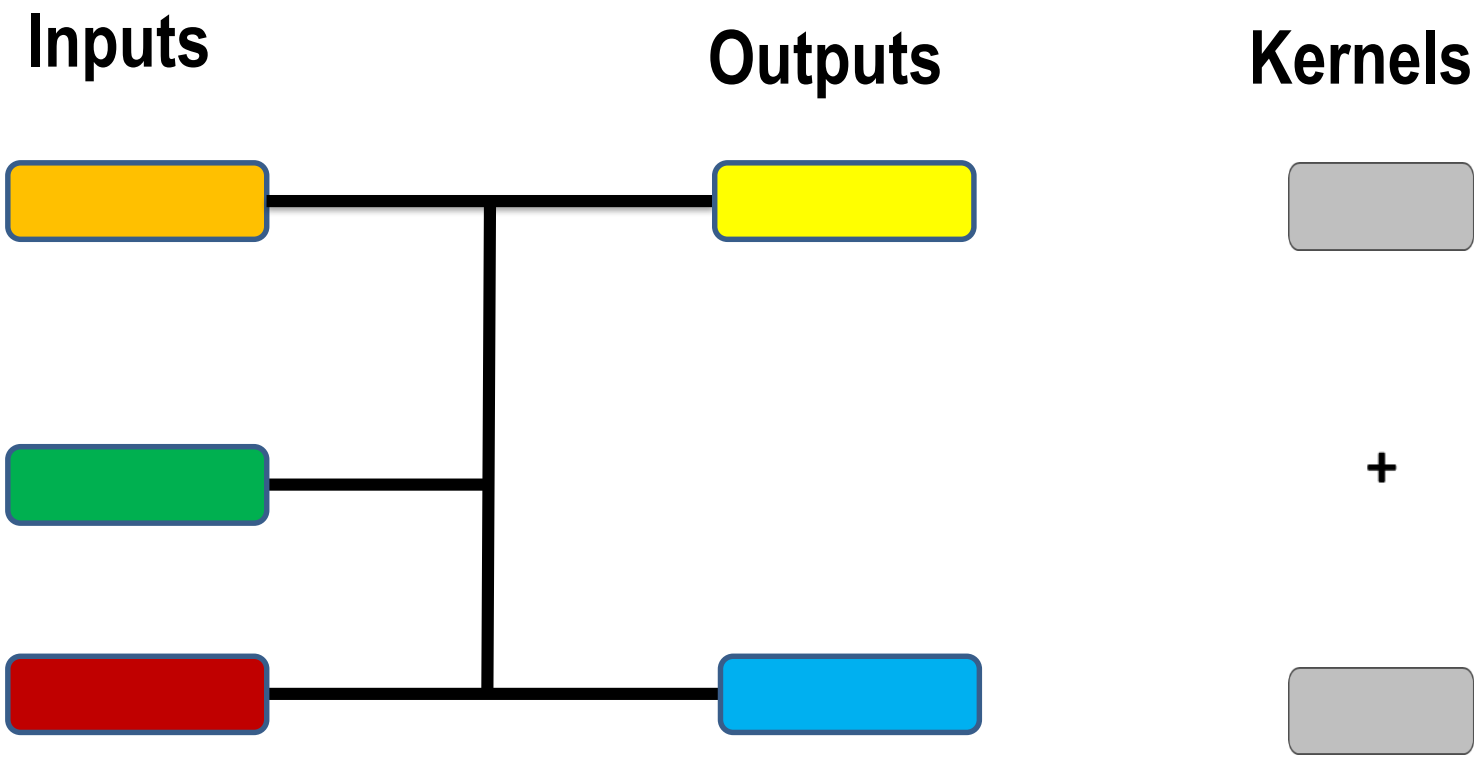


A. Poelstra



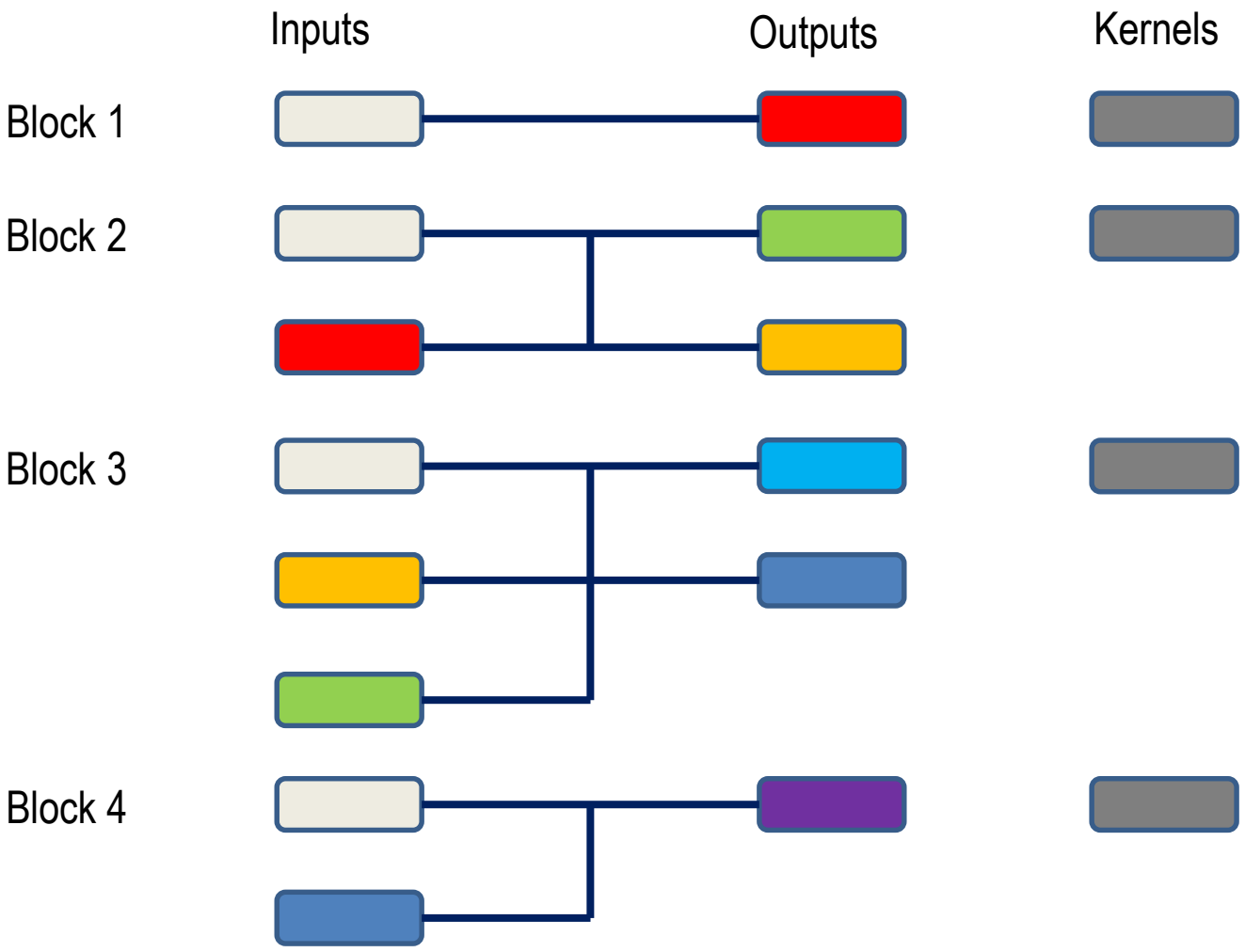
A. Poelstra

TRANSAKTIONEN



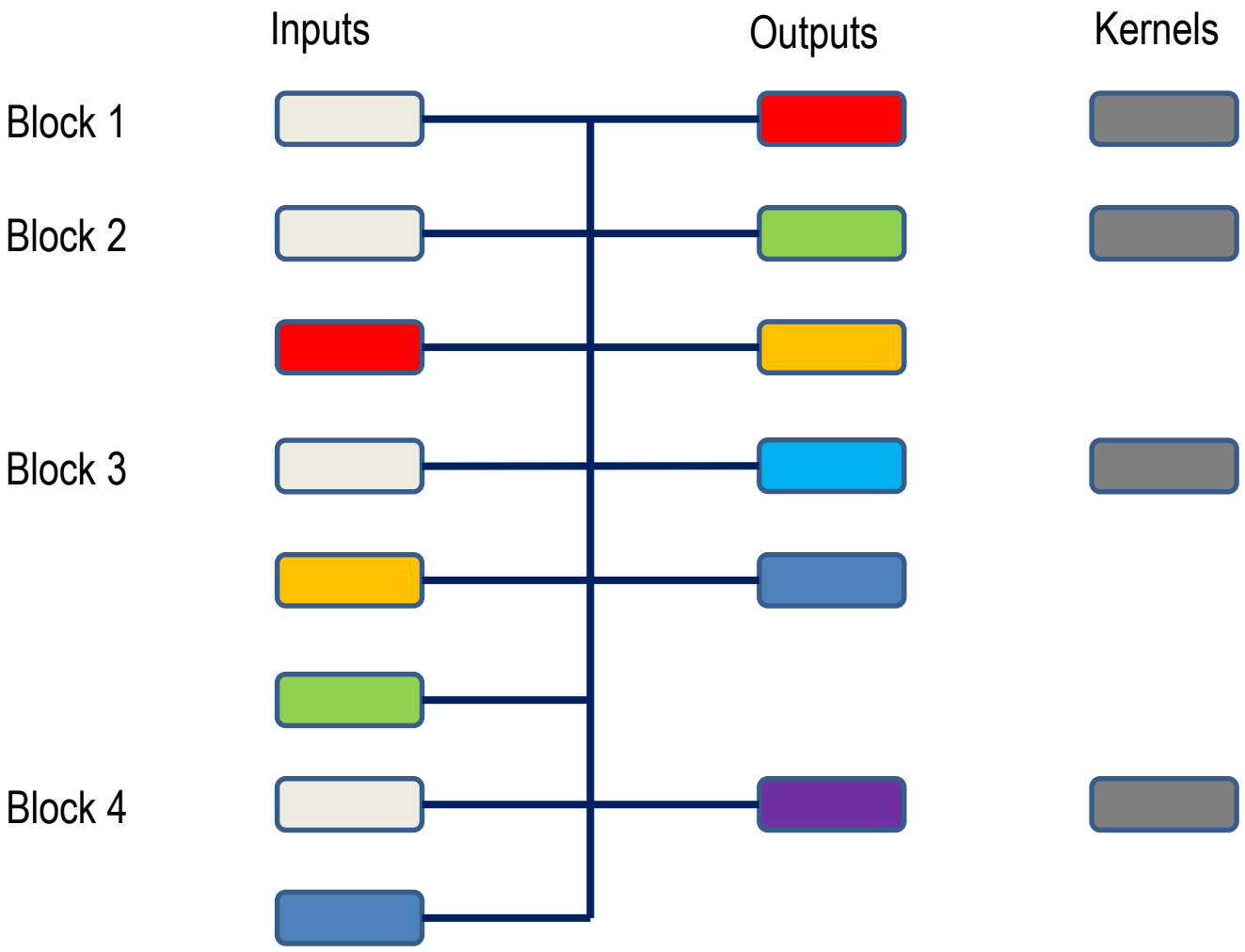
A. Poelstra

AUFBAU DER BLOCKCHAIN



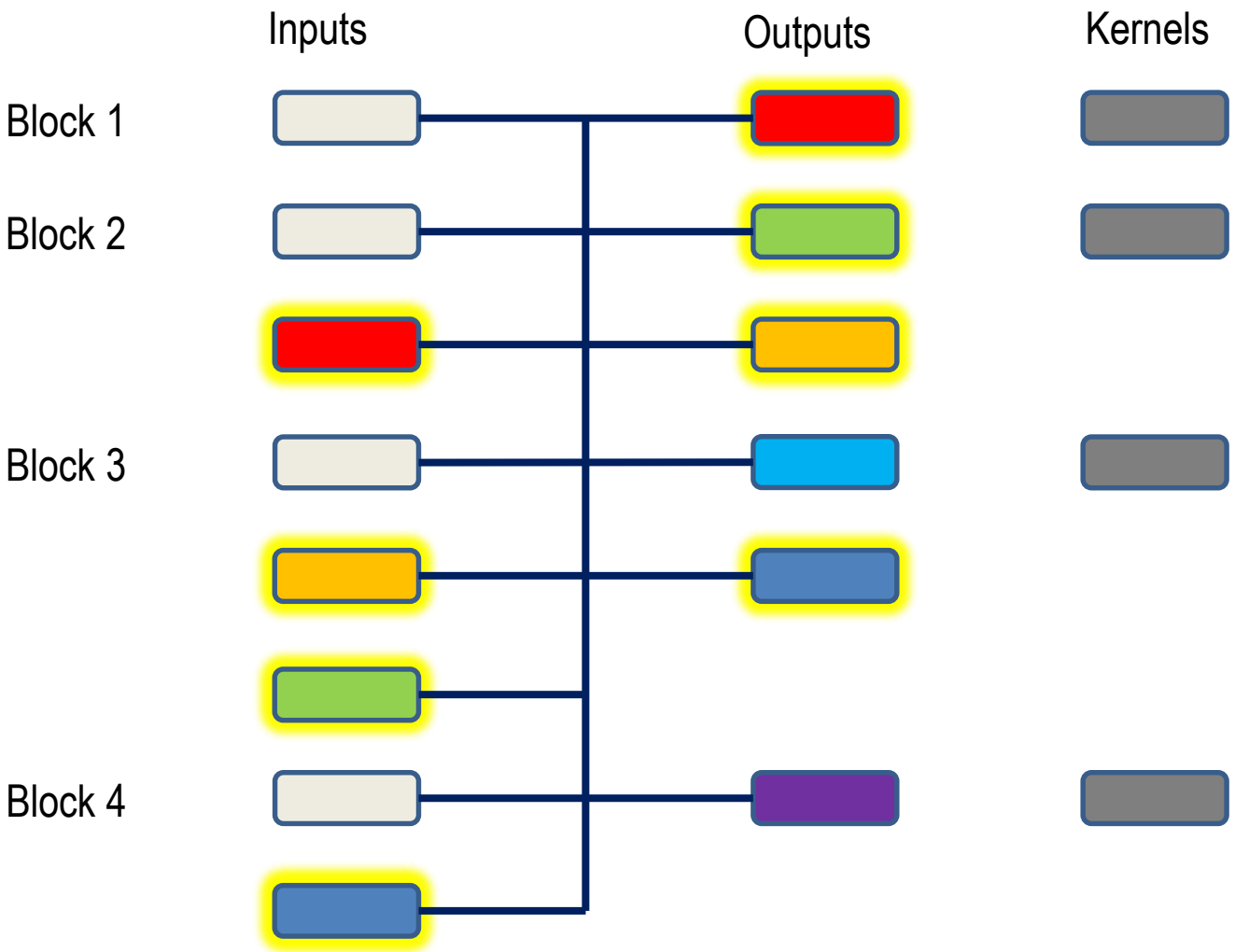
A. Poelstra

AUFBAU DER BLOCKCHAIN



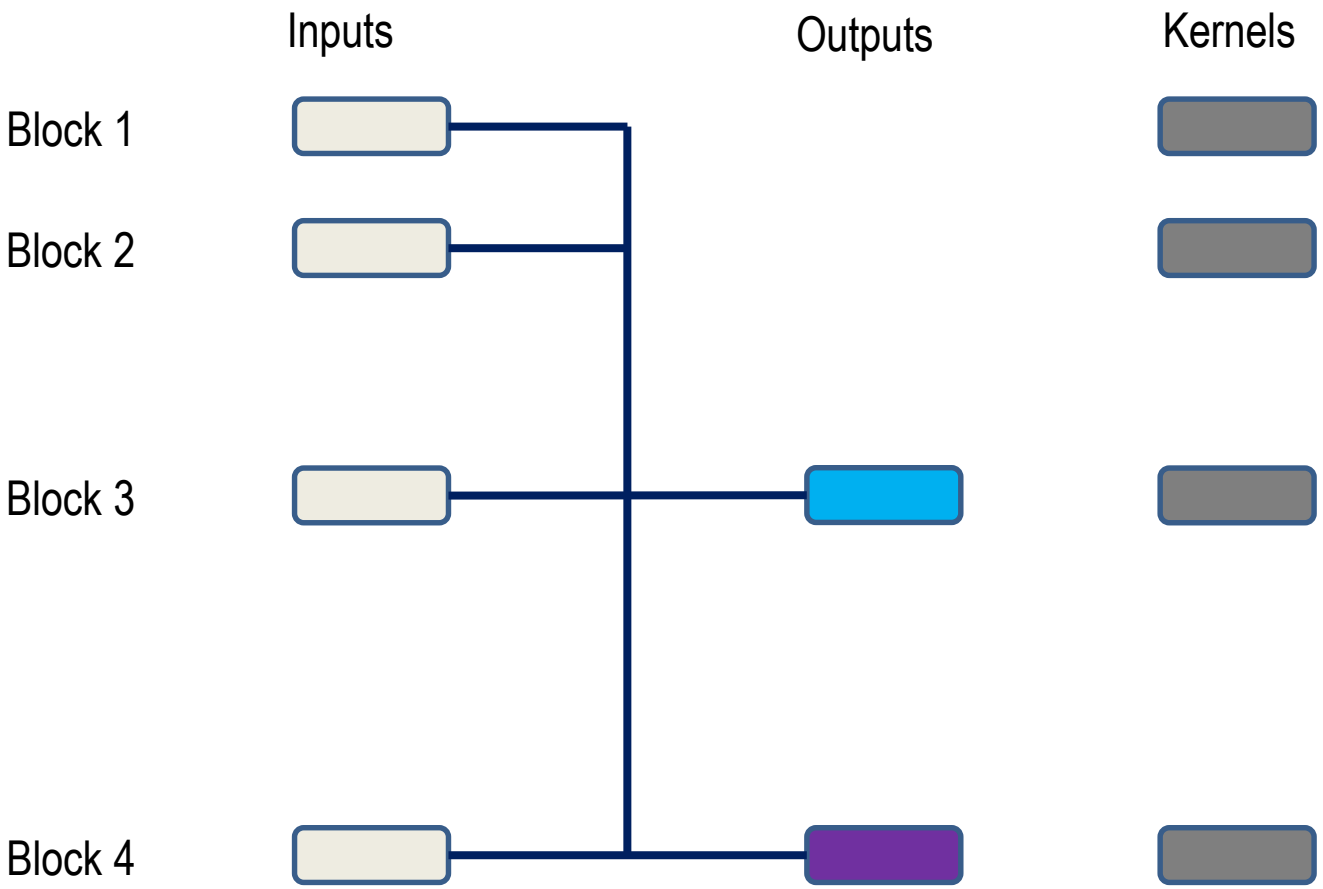
A. Poelstra

AUFBAU DER BLOCKCHAIN



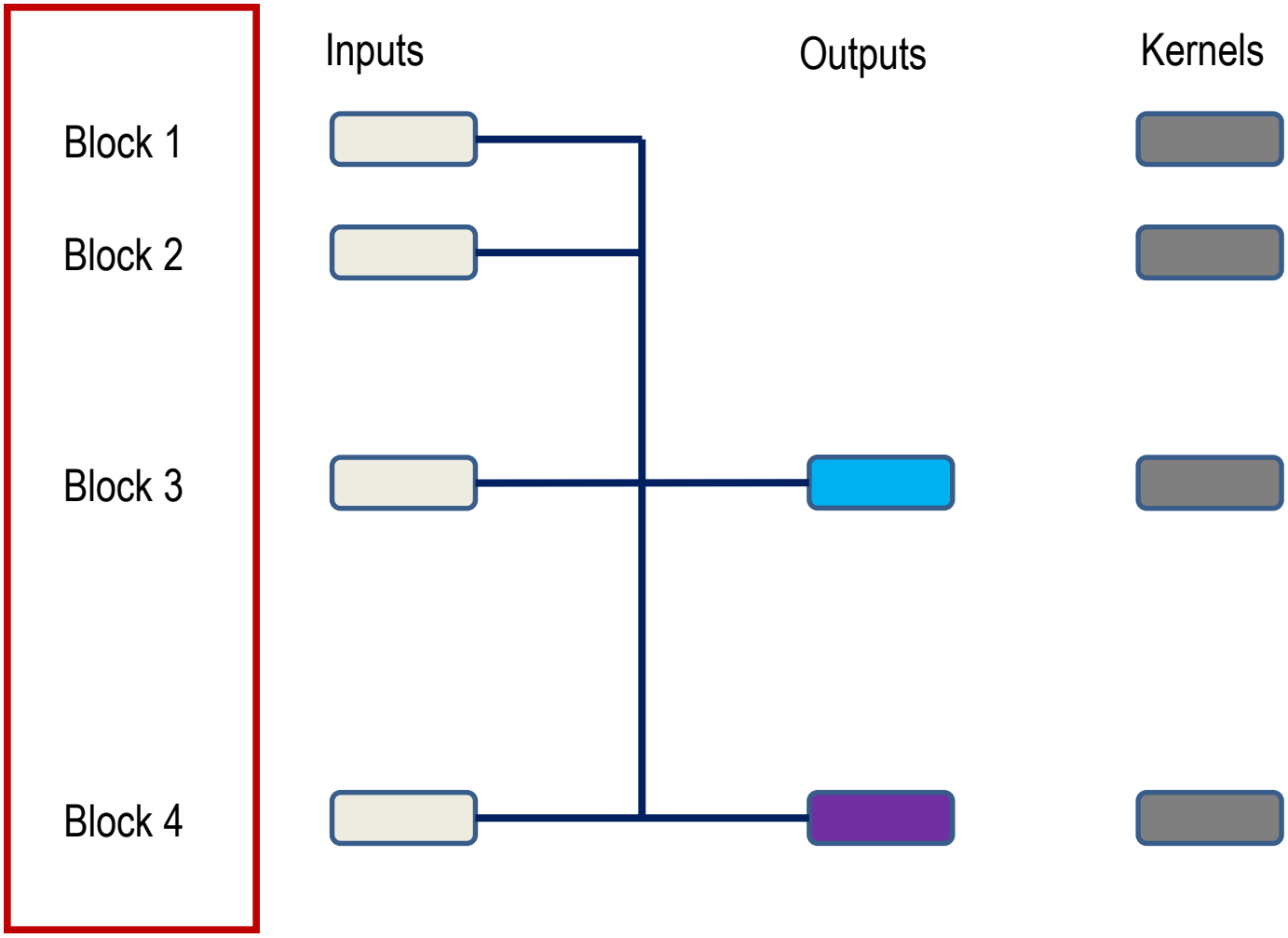
A. Poelstra

AUFBAU DER BLOCKCHAIN



A. Poelstra

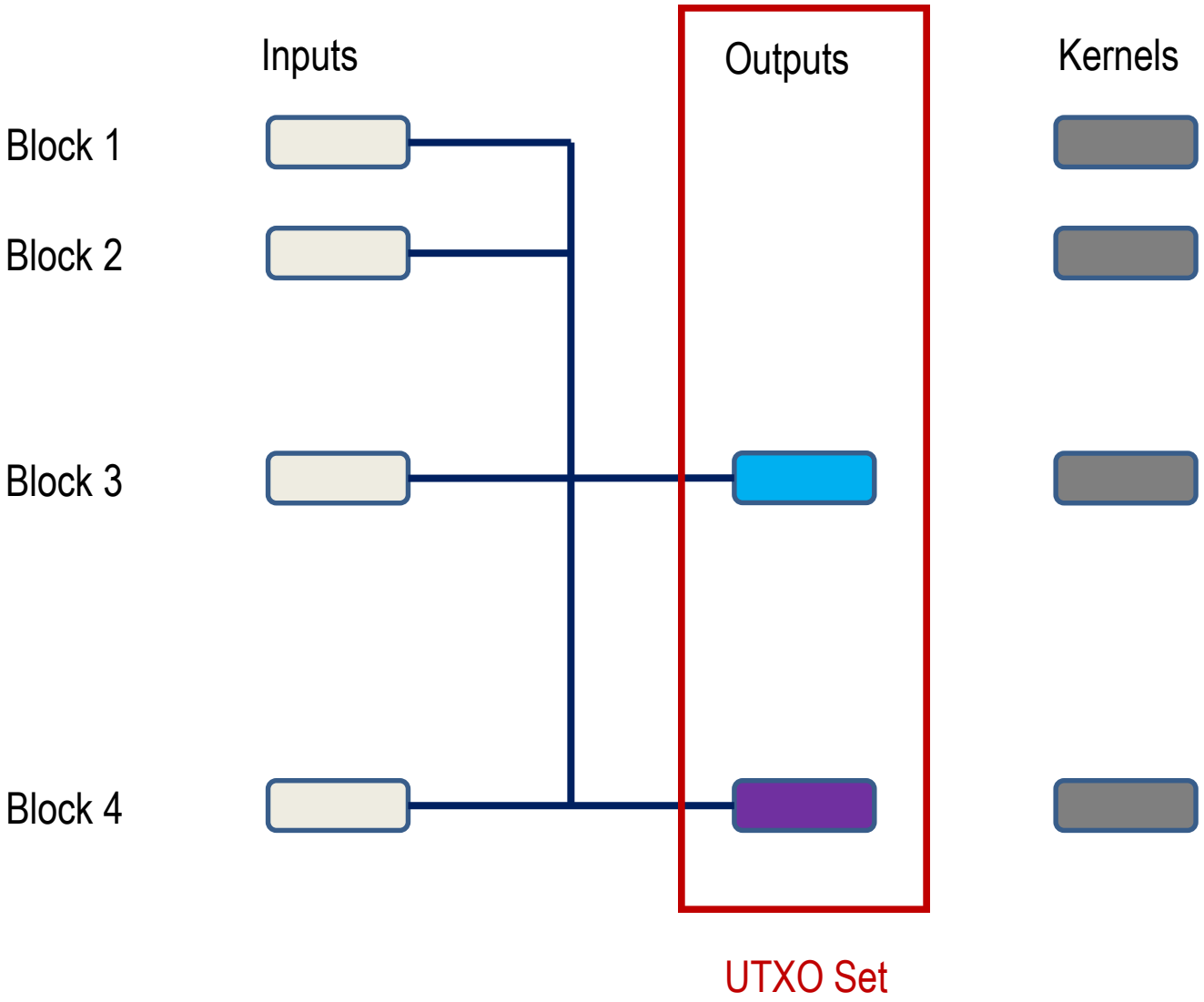
AUFBAU DER BLOCKCHAIN



Block Header

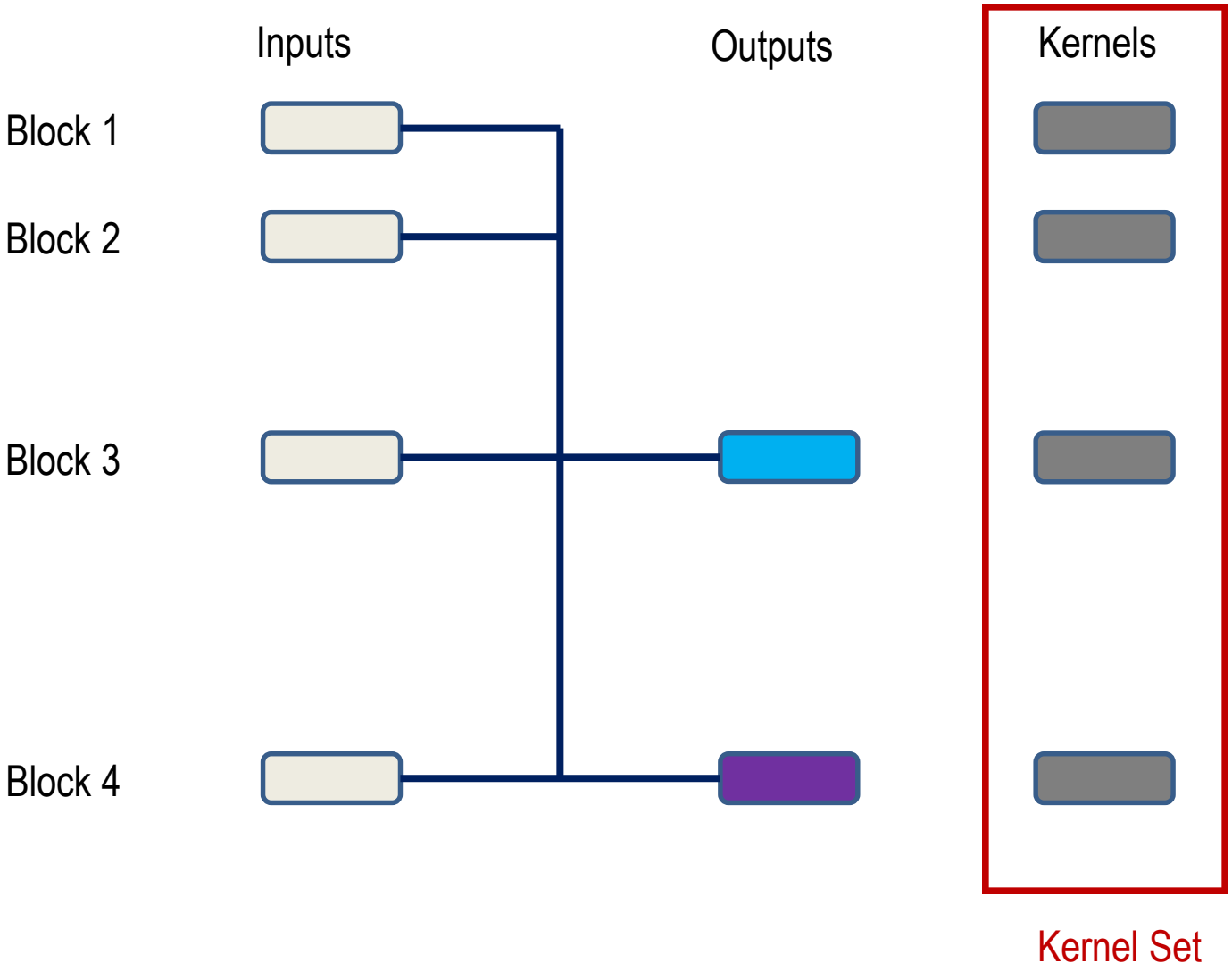
A. Poelstra

AUFBAU DER BLOCKCHAIN



A. Poelstra

AUFBAU DER BLOCKCHAIN



A. Poelstra

PRUNING

- Pruning (Begriff von Grin) benutzt Cut-through für die gesamte Blockchain
- Laut Grin-Team wird das Ledger bei 10 Millionen Transaktionen mit 100.000 unspend Outputs etwa 130 GB betragen, was sich wie folgt aufteilt:
  - 128 GB Transaktionsdaten (In- und Outputs)
  - 1 GB Kernel Daten
  - 250MB Block-Header

- Ledger wird mittels Pruning auf ca. 1,8 GB reduziert:
  - 520MB UTXO Set
  - 1 GB Kernel Daten
  - 250MB Block-Header

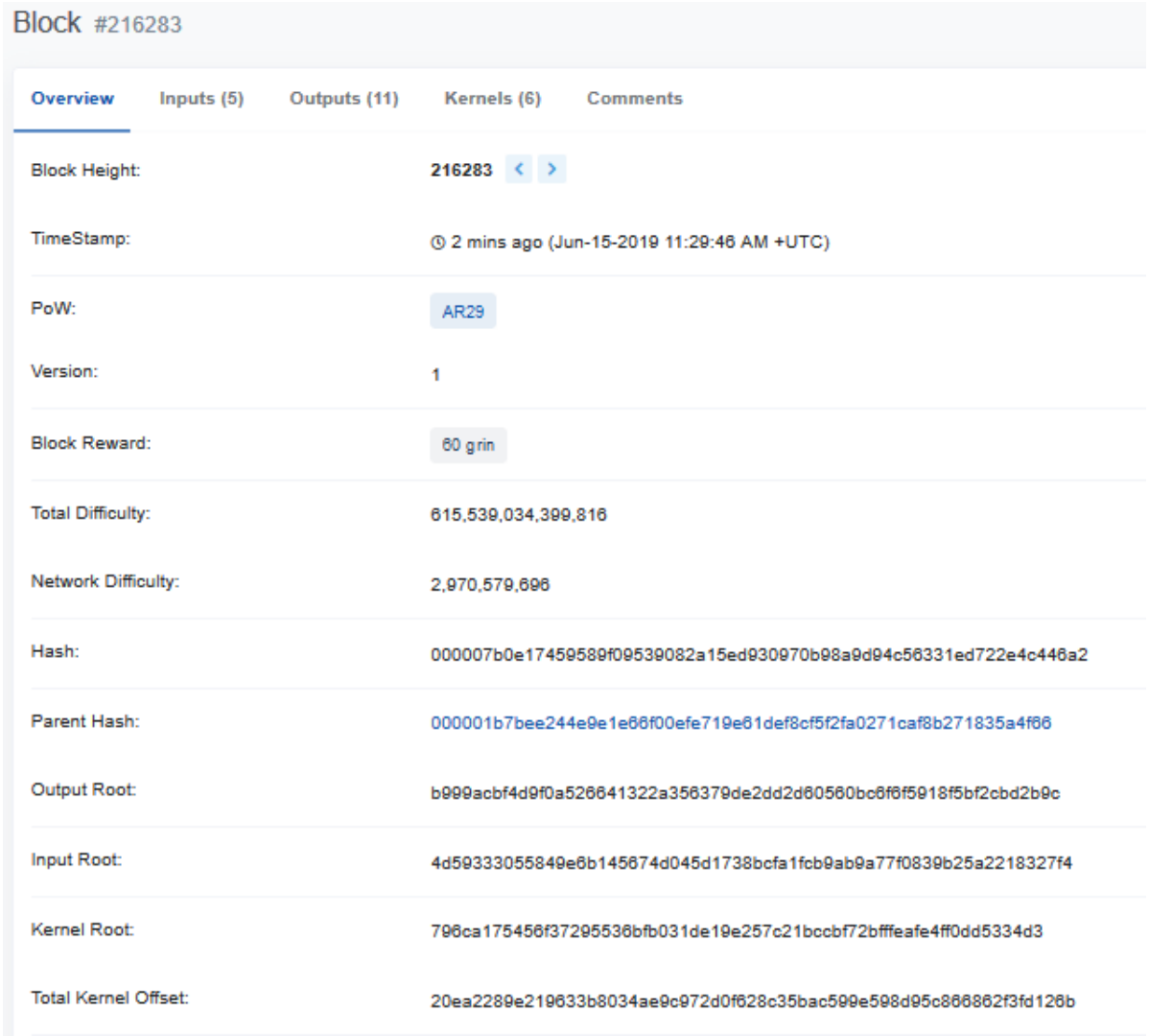
I. Peverell et al., "Pruning Blockchain Data", [online]. Available: https://github.com/mimblewimble/grin/blob/master/doc/pruning.md

BLOCK HEADER AUFBAU

Header Field	Description
Hash	Unique hash of block
Version	Grin version
Previous Block	Unique hash of previous block
Age	Time the block was mined
Cuckoo Solution	The wining cuckoo solution
Difficulty	Difficulty of the solved cuckoo
Target Difficulty	Difficulty of this block
Total Difficulty	Total difficulty of mined chain up to age
Total Kernel Offset	Kernel offset
Nonce	Random number for cuckoo
Block Reward	Coinbase + fee reward for block

W. van Heerden et al. „Mimblewimble-Grin Blockchain Protocol Overview“. [Online]. Available: https://tlu.tarilabs.com/protocols/grin-protocol-overview/MainReport.html

BLOCKAUFBAU



Block #216283

Overview | Inputs (5) | Outputs (11) | Kernels (6) | Comments

Block Height:	216283
TimeStamp:	2 mins ago (Jun-15-2019 11:29:46 AM +UTC)
PoW:	AR29
Version:	1
Block Reward:	60 grin
Total Difficulty:	615,539,034,399,816
Network Difficulty:	2,970,579,696
Hash:	000007b0e17459589f09539082a15ed930970b98a9d94c56331ed722e4c446a2
Parent Hash:	000001b7bee244e9e1e66f00efe719e61def8cf5f2fa0271caf8b271835a4f66
Output Root:	b999acbf4d9f0a526641322a356379de2dd2d60560bc6f6f5918f5bf2cbd2b9c
Input Root:	4d59333055849e6b145674d045d1738bcfa1fcb9ab9a77f0839b25a2218327f4
Kernel Root:	796ca175456f37295536bfb031de19e257c21bccbf72bfffeafe4ff0dd5334d3
Total Kernel Offset:	20ea2289e219633b8034ae9c972d0f628c35bac599e598d95c866862f3fd126b

Blockscan: https://grin.blockscan.com/block/216283

HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

MIMBLEWIMBLE

SCRIPTLESS SCRIPTS

SCHNORR SIGNATUREN

- Schnorr Signaturen können für jede mathematische Gruppe angewendet werden, die den Anforderungen des Diskreten-Logarithmus-Problems entsprechen (secp256k1 - die Elliptische Kurve von Bitcoin - entspricht dieser Anforderung)

- Schnorr Signaturen haben eine Größe von 64 Bytes (Die meisten ECDSA Signaturen sind 71 oder 72 Bytes groß)

- Schnorr Signaturen sind linear, somit besitzen sie die Eigenschaft der Aggregation
  - → Signaturen aus mehreren Eingaben können zu einer einzigen Signatur zusammengefasst werden
  - → Signaturen einer Multisig-Eingabe können zu einer einzigen Signatur verdichtet werden (Anwendung für z.B. Scriptless Scripts)

- Unterschiedliche Ausgabekonditionen können durch Schnorr Signaturen umgesetzt werden (Grundlage für Scriptless Scripts)

J. Clifford. „Scriptless Scripts - A different kind of smart contract". [Online]. Available: https://medium.com/scalar-capital/scriptless-scripts-25e18fd52ede?sk=72eb8b17650316ad92b41179006008a7

SCRIPTLESS SCRIPTS

- Mimblewimble ist so entwickelt wurden, dass es überhaupt keine Skripte unterstützt
- Die gesamte Blockchain basiert auf Signaturen und Commitments

- Andrew Poelstra untersuchte die mathematischen Eigenschaften von Schnorr-Signaturen weiter und nannte die dadurch entdeckte Methode "Scriptless Scripts"

- Scriptless Scripts sind ein Mittel, um Smart Contracts Off-Chain durch den Einsatz von Schnorr-Signaturen auszuführen.
  - Somit können Bedingungen unter den Beteiligten vereinbart werden, ohne dass der Smart Contract irgendwo sichtbar ist
  - Nur die beteiligten Personen wissen von den Bedingungen oder dass ein Vertrag überhaupt existiert, für den Rest der Welt sieht es wie eine normale Transaktionen auf der Blockchain aus

J. Clifford. „Scriptless Scripts - A different kind of smart contract”. [Online]. Available: https://medium.com/scalar-capital/scriptless-scripts-25e18fd52ede?sk=72eb8b17650316ad92b41179006008a7

SCRIPTLESS SCRIPTS

- **Funktionalität**
  - Reichweite und Komplexität von Smart Contracts sollen erhöht werden. (Zur Zeit Einschränkungen durch die Anzahl der OP_CODES, die vom Netzwerk erlaubt sind)
  - Scriptless Scripts verschieben Konkretisierung und Ausführung aus dem Netzwerk zu den Teilnehmern des Smart Contracts
- **Datenschutz**
  - Verlagerung der Konkretisierung und Ausführung des Smart Contracts von On-Chain auf Off-Chain erhöht den Datenschutz
  - Bei On-Chain werden viele Details des Smart Contract dem gesamten Netzwerk mitgeteilt (z.B. Anzahl und Adressen der Teilnehmer, überwiesenen Beträge)
  - Off-Chain weiß Netzwerk nur, dass die Teilnehmer zustimmen, dass Bedingungen ihres Vertrages erfüllt sind und dass betreffende Transaktion gültig ist
- **Effizienz**
  - Scriptless Scripts minimieren die Datenmenge, die eine Verifizierung und Speicherung On-Chain erfordern würde
  - Es entstehen weniger Kosten für vollständige Knoten und niedrigere Transaktionsgebühren für die Benutzer

K. Sardar et al. „Introduction to Sriptless Scripts“. [Online]. Available: https://tlu.tarilabs.com/cryptography/scriptless-scripts/introduction-to-scriptless-scripts.html

CROSS-CHAIN ATOMIC SWAPS

- Was ist ein Cross-Chain Atomic Swap?
  - **Swap**: ein Handel ohne Vermittlerpartei
  - **Atomic**: Im Grunde eine Alles-oder-Nichts-Transaktion. Entweder beide Parteien schließen den Tausch ab oder keine von ihnen
  - **Cross-Chain**: Zwischen verschiedenen Kryptowährungen

- Können vollständig P2P sein, da keine dritte Partei involviert ist
- Sind anonym (niemand sonst weiß, dass dieser Handel stattgefunden hat)

- Nachteil: Trades sind ziemlich langsam (sie können 24 Stunden in Anspruch nehmen)

J. Van der Maarel: „Cross-chain atomic swaps with Grin“. [Online]. https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/06%20-%20jaspervdm%20-%20Atomic%20Swaps.pdf

CROSS-CHAIN ATMIC SWAPS

Mimblewimble Atomic Swaps

1. Bob generiert ein Geheimnis q. Er sperrt die BTC/ETH in einer 2-of-2 Multisignature mit dem Schlüssel von Alice und q. Bob erhält diese nach 24 Stunden zurück.

2. Alice sperrt Grins (mit Unterstützung von Bob) in einer 2-of-2 Multisignature mit den Schlüsseln von Alice und Bob. Alice erhält diese nach 12 Stunden zurück.

3. Alice und Bob arbeiten zusammen und senden die Grins an Bob. Wenn Bob die Transaktion abschließt veröffentlicht er automatisch q.

4. Alice entsperrt die BTC/ETH.

J. Van der Maarel: „Cross-chain atomic swaps with Grin“. [Online]. https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/06%20-%20jaspervdm%20-%20Atomic%20Swaps.pdf

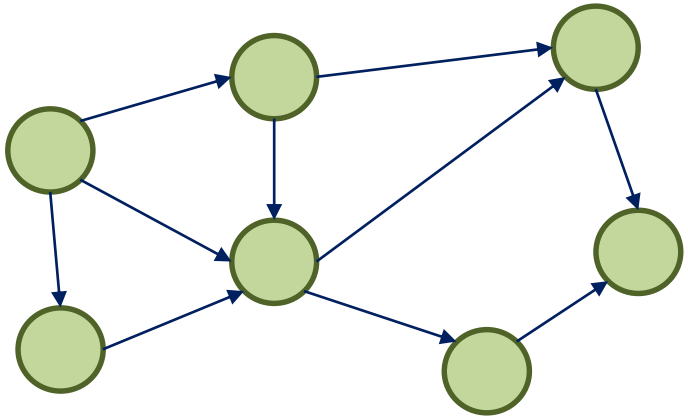
HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

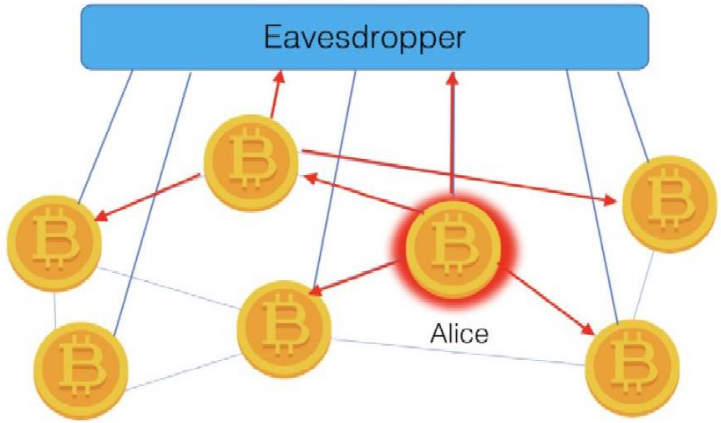
MIMBLEWIMBLE

DANDELION

DANDELION

- Derzeit werden im Bitcoin Netzwerk die Transaktionen mittels dem Prozess “Diffusion” verbreitet
- Jeder Node verbreitet die Transaktion mit seiner eigenen zufälligen Verzögerung im Netzwerk
- Im Laufe der Zeit breitet sich so die Transaktion in dem Netzwerk aus
- Befinden sind allerdings korrupte Knoten im Netzwerk, können diese die jeweiligen Zeitstempel auswerten und zusammen mit der Struktur des Graphen ableiten, welcher Knoten die Quelle einer bestimmten Transaktion war
- Dadurch kann die IP-Adresse mit der jeweiligen Bitcoin-Adresse verknüpft werden

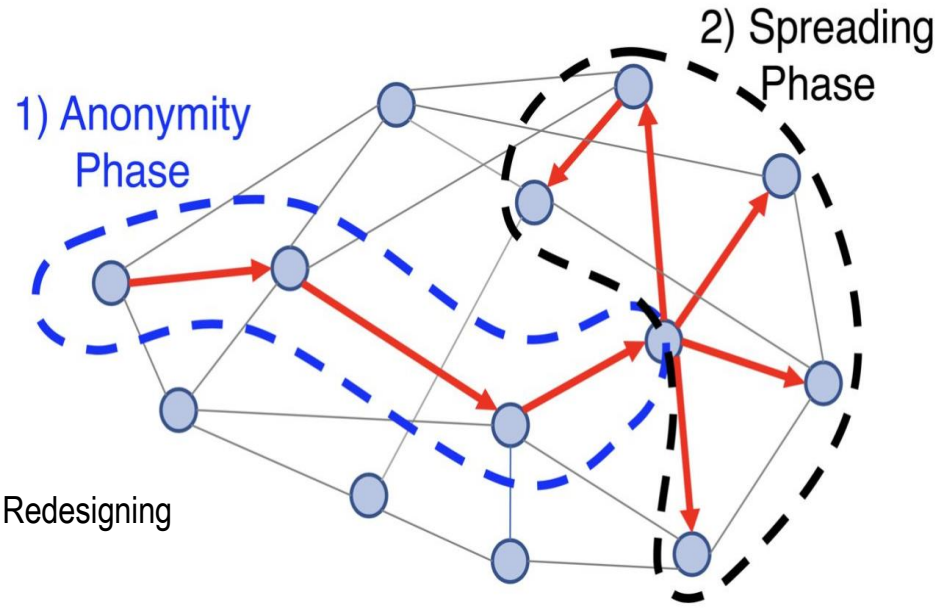




Q. Le Sceller: „Dandelion in Grin: Privacy Preserving Transaction Aggregation Propagation“. [Online]. https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/04%20-%20quentinlesceller%20-%20Dandelion.pdf

DANDELION

- **1. Phase** (Stem-Phase):
  - Der aktuelle Knoten macht einen Münz-Wurf: bei Kopf leitet er die Transaktion an einen einzigen Nachbarn (Dandelion Relay) weiter, bei Zahl geht er in die Fluff-Phase über
  - Parameter für den Münz-Wurf: 90% Stem und 10% Fluff
  - Dieses Verfahren wiederholt sich, bis in die Fluff-Phase übergegangen wird
- **2. Phase** (Fluff-Phase):
  - Hier wird die Nachricht im typischen Format (Diffusion) an den Rest des Netzwerks gesendet



S. Bojja, G. Fanti, P. Viswanath: „Dandelion: Redesigning the Bitcoin Network for Anonymity".

- **Bedeutung für Mimblewimble:**
  - Der Node sendet eine Transaktion an das Netzwerk, es wird direkt an das nächste Dandelion Relay als Stem-Transaktion weiter gesendet
  - Der Dandelion Relay wartet eine bestimmte Zeit (patience timer), um mehr Stem-Transaktionen zu erhalten
  - Am Ende des Timers führt der Relay einen Münz-Wurf für jede Transaktion durch und entscheidet, ob diese als Stem- oder Fluff-Transaktion weitergeleitet wird
  - Bei dem Ergebnis Stem aggregiert er alle seine Stem-Transaktionen und schickt sie an seinen Dandelion Relay weiter
  - Bei Fluff aggregiert er alle Fluff-Transaktionen und schickt diese normal an das Netzwerk

Q. Le Sceller: „Dandelion in Grin: Privacy Preserving Transaction Aggregation Propagation“. [Online]. https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/04%20-%20quentinlesceller%20-%20Dandelion.pdf

HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

MIMBLEWIMBLE

MINING ALGORITHMUS IN GRIN – CUCKOO CYCLE

LARA BISCHOFF

PROOF OF WORKS

- **Hashcash** (wird z.B. bei Bitcoin verwendet)
  - Message m, Difficulty D → Finde n damit Hash(m | n) < D
  - Nonce finden, um bestimmten Hash-Wert zu erhalten
  - Arbeitet symmetrisch

- **Cuckoo Cycle**
  - Arbeitet asymmetrisch
  - Graph-theoretischer Proof of Work Algorithmus
  - Basiert darauf, Kreise einer bestimmten Länge in großen zufälligen Graphen zu finden
  - Die Überprüfung ist dabei trivial und Skalierung beliebig möglich

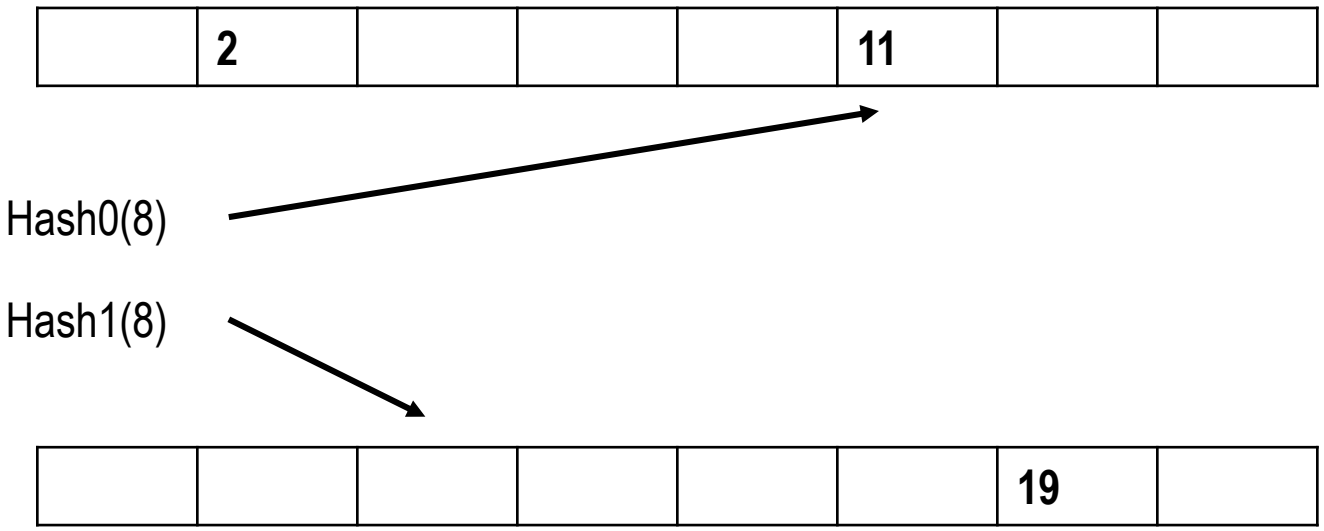
  - Zum Erstellen des Graphen werden Cuckoo-Hashtables verwendet (eingeführt von Rasmus Pagh and Flemming Friche Rodler)

CUCKOO HASHTABLES

- Cuckoo Hashtable bestehen aus zwei Tabellen gleicher Größe mit jeweils einer eigenen Hash-funktion, die einen Schlüssel auf eine Tabellenposition abbildet und dadurch zwei Mögliche Positionen für jeden Schlüssel entstehen.
- Wenn ein neuer Schlüssel eingefügt wird und beide möglichen Positionen bereits belegt sind, wird einer dieser bereits eingefügten Schlüssel herausgeworfen und in seine alternative Position eingefügt, wodurch möglicherweise weitere Schlüssel verschoben werden.
- Dieser Prozess des Verschiebens wird wiederholt, bis entweder ein freier Platz gefunden wurde oder bis sich ein Kreis gebildet hat.
- Dies passiert, wenn sich der Cuckoo Graph gebildet hat.

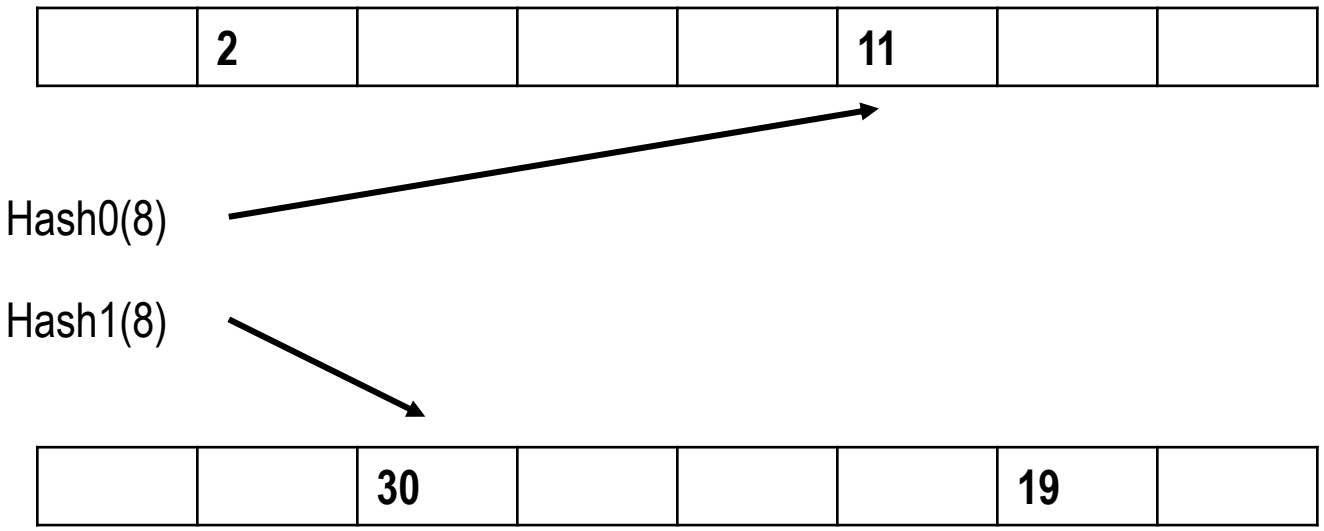
J. Tromp. „Cuckoo Cycle: a memory bound graph-theoretic proof-of-work” Apr 2019.

CUCKOO CYCLE



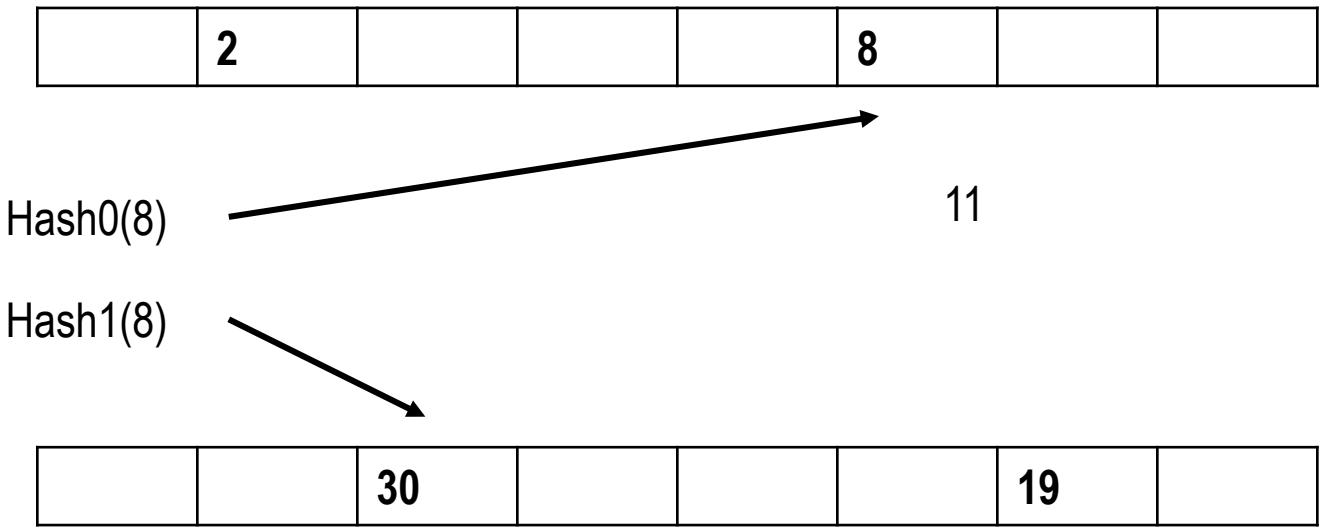
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

CUCKOO CYCLE



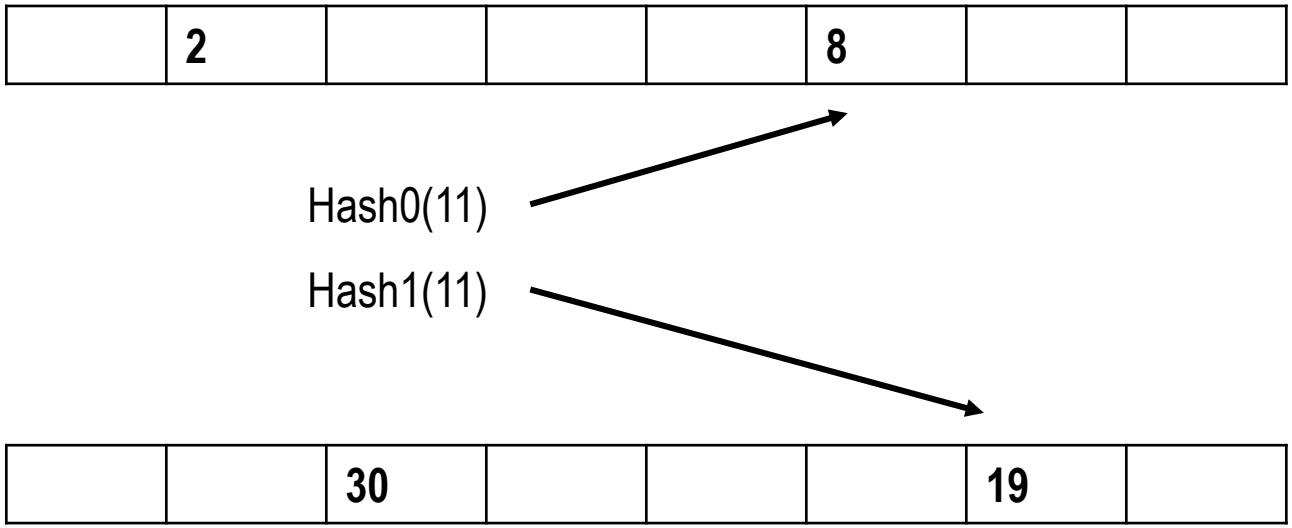
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

CUCKOO CYCLE



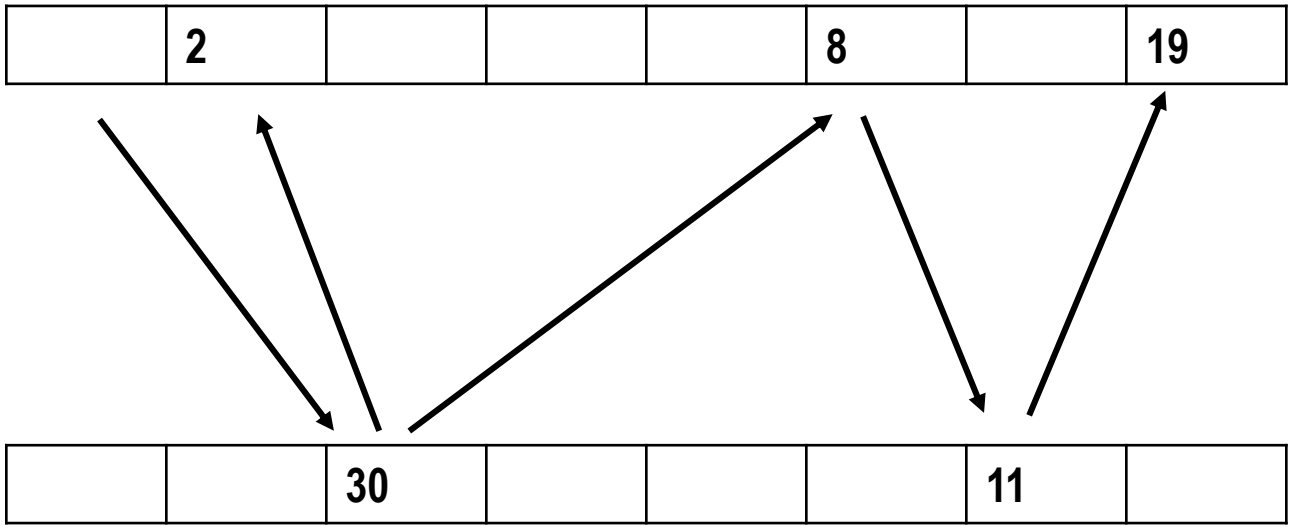
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

CUCKOO CYCLE



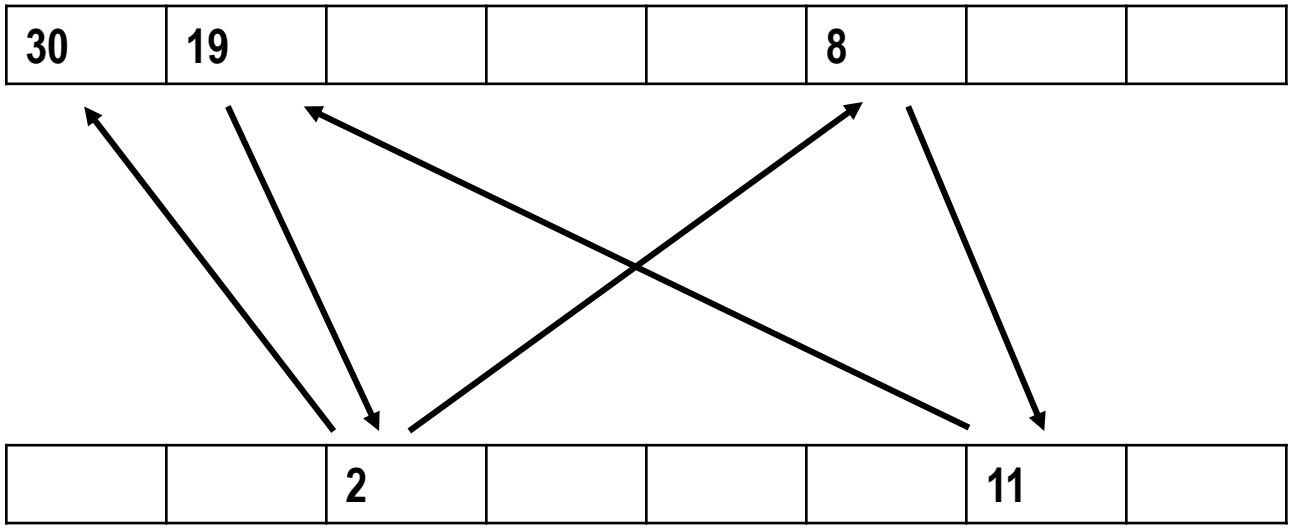
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

CUCKOO CYCLE



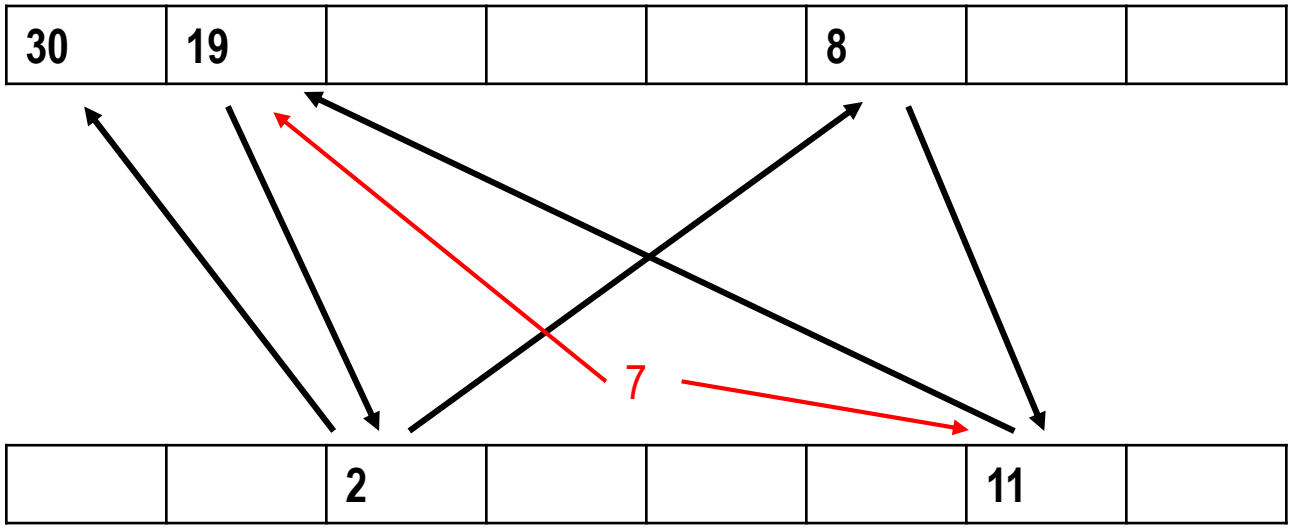
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

CUCKOO CYCLE



J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

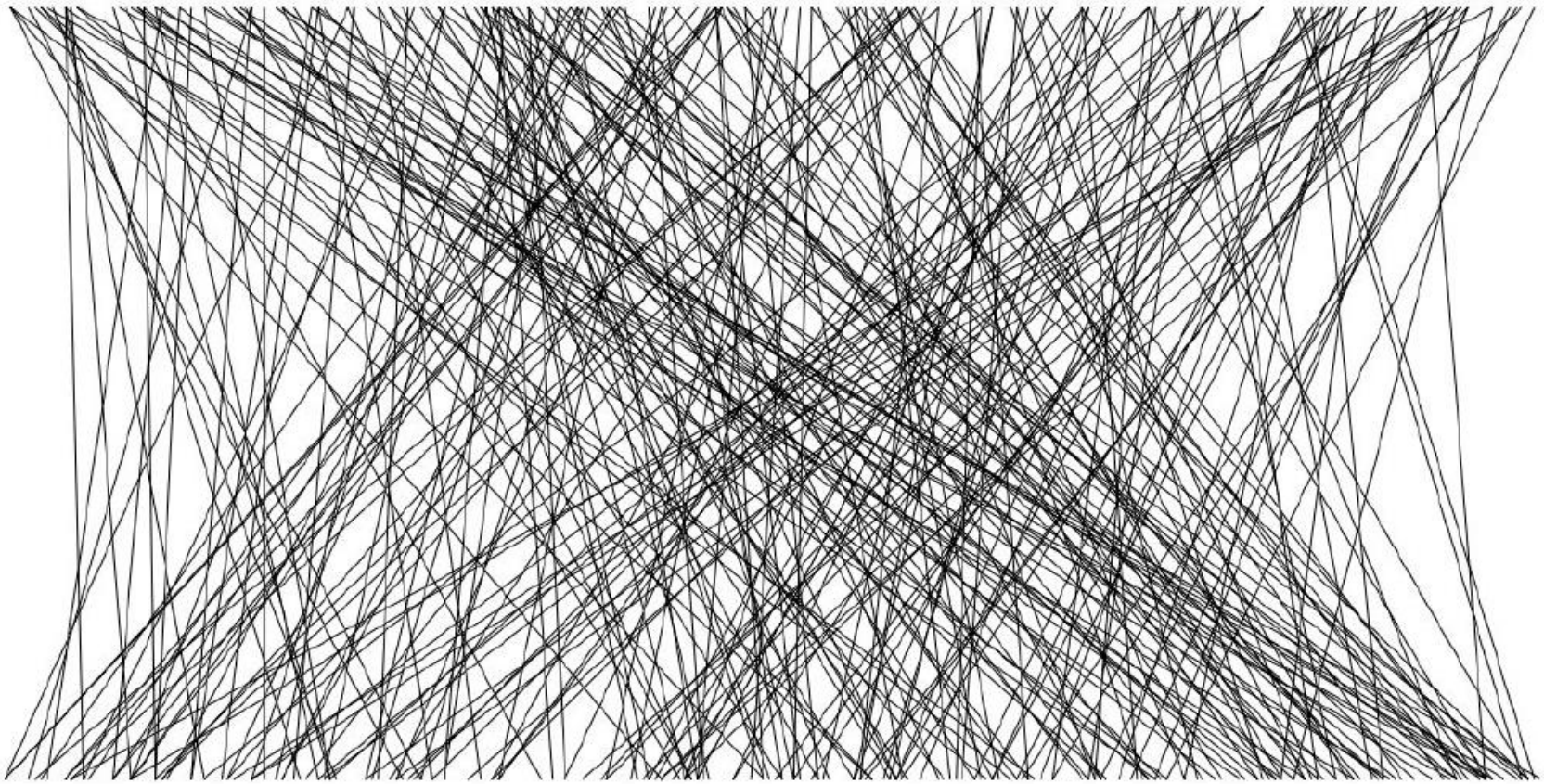
CUCKOO CYCLE



J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

CUCKOO GRAPH

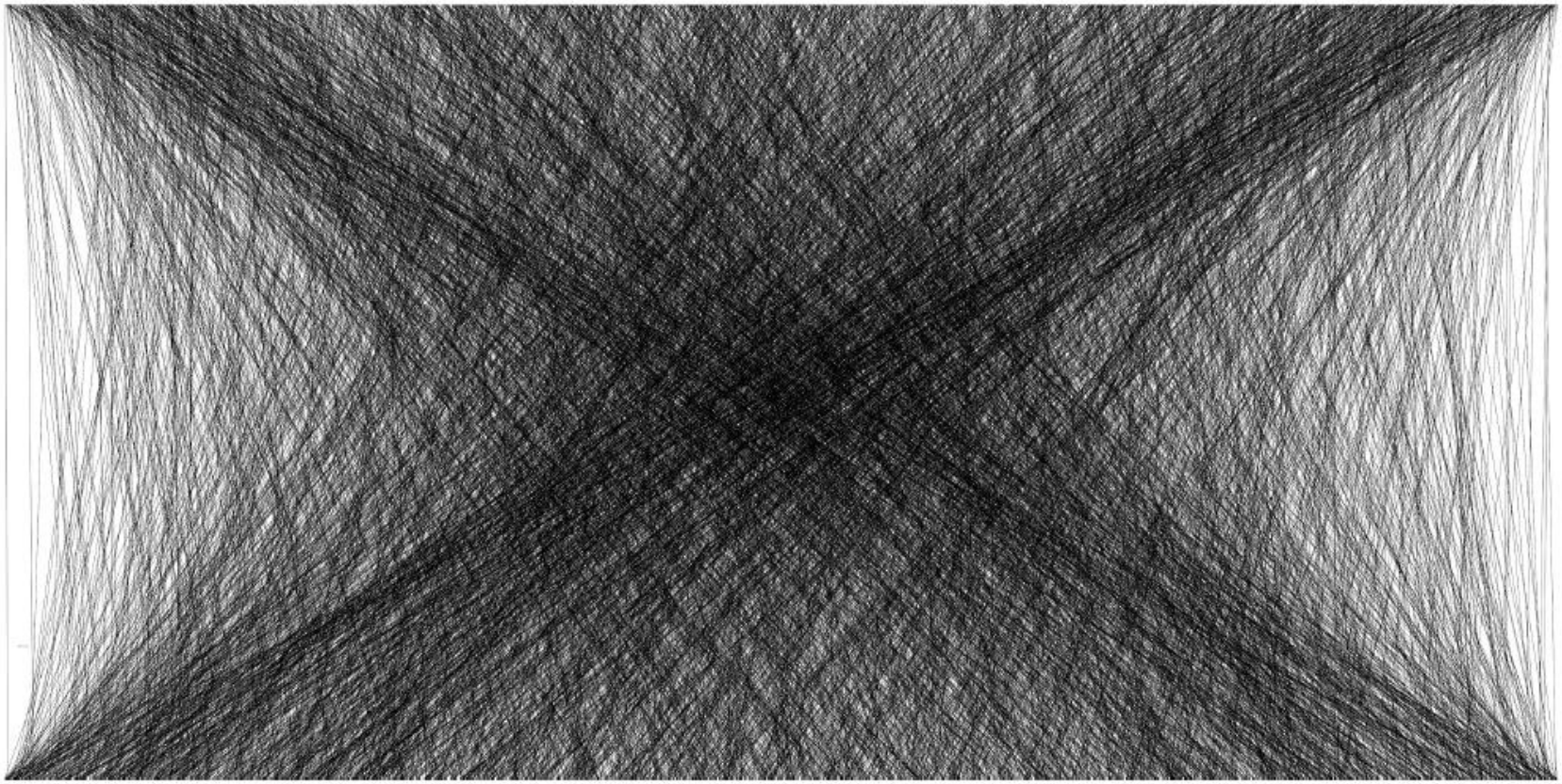
- 2^8 Kanten



J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

CUCKOO GRAPH

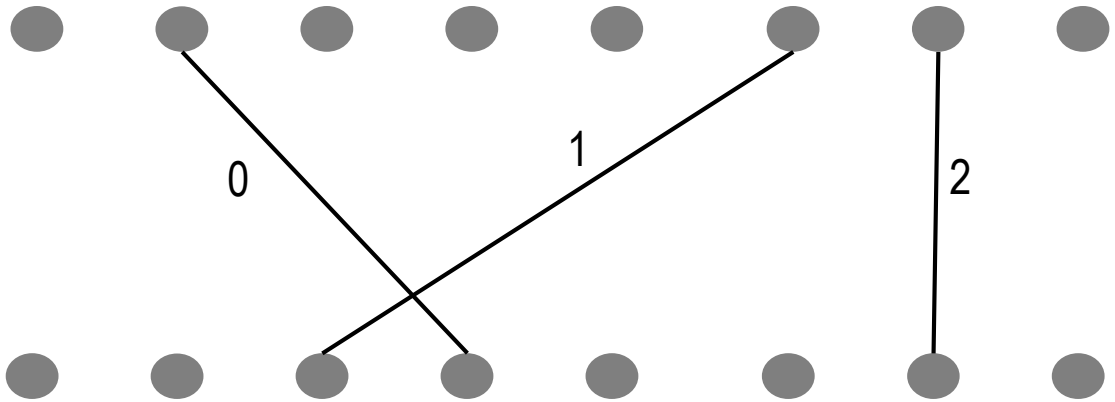
- 2^12 Kanten



J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

CUCKOO GRAPH

- Knotenanzahl N und Kantenanzahl M, $N \leq 2^{64}$ und $M = N/2$
- Das Erzeugen von Kanten erfolgt durch das Setzen von Kantenindizes (0..M) und einer Passwort-basierten Hash-Funktion, SIPHASH
- Jeder Kantenindex wird zweimal durch die SIPHASH-Funktion geleitet, um zwei Kantenendpunkte zu erzeugen, wobei der erste Eingangswert 2 * edge_index und der zweite 2 * edge_index+1 ist.
- Das „Passwort“ für diese Funktion basiert auf dem Hash des Block-Headers
- Der von diesem Algorithmus erzeugte 'Proof' ist eine Reihe von Nonces, die einen Kreis der Länge 42 erzeugen, was von anderen Teilnehmern trivial validiert werden kann.



I. Peverell et al. „Grin‘s Proof-of-Work“. [Online] Available: https://github.com/mimblewimble/grin/blob/master/doc/pow/pow.md

DER MINING LOOP

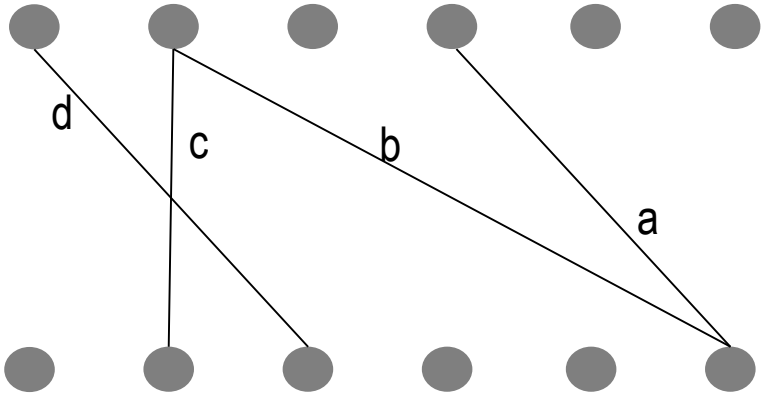
- Der neue Block-Header wird gehashed.
- Der Cuckoo Graph Generator wird mit folgenden Parametern initialisiert:
  - Der Hash des Block-Headers wird als Password für die SIPHASH Funktion benutzt, die die Positionen jeder Kante 0…M im Graphen bestimmt
  - Die Größe des Graphen (Konsensuswert)
  - Ein bestimmter Wahrscheinlichkeitswert (Konsensuswert), der die Anzahl der Kanten im Verhältnis zu der Größe des Graphen angibt
- Der Algorithmus zur Erkennung der Kreise versucht, eine Lösung (d.h. einen Kreis der Länge 42) innerhalb des erzeugten Graphen zu finden.
- Wenn ein Kreis gefunden wird, wird ein Blake2b-Hash des Proofs erstellt und mit dem aktuellen Soll-Schwierigkeitsgrad verglichen

I. Peverell et al. „Grin‘s Proof-of-Work“. [Online]. Available: https://github.com/mimblewimble/grin/blob/master/doc/pow/pow.md.

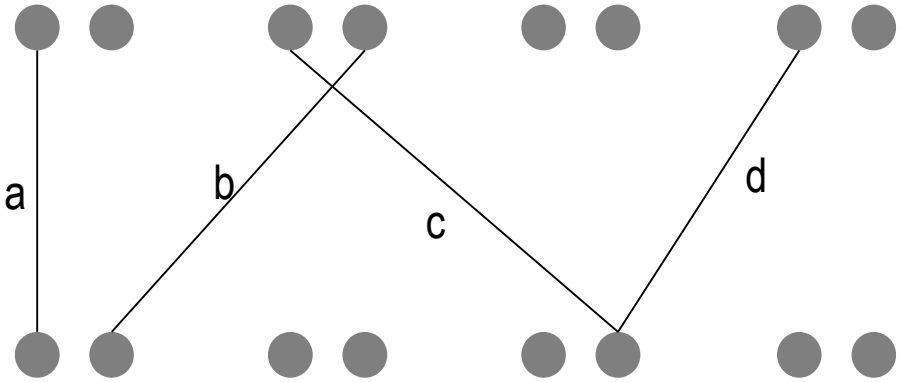
CYCLE FINDING

- a adjazent zu b adjazent zu c
- c NICHT adjazent d

CuckARoo



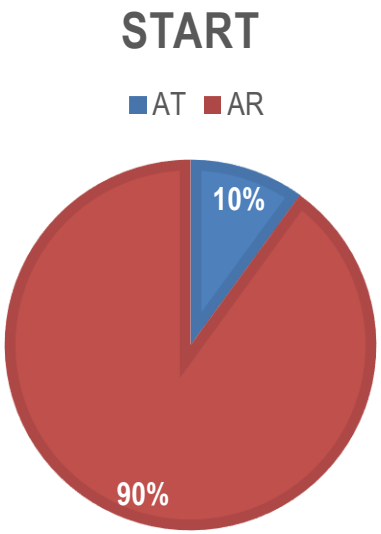
CuckAToo

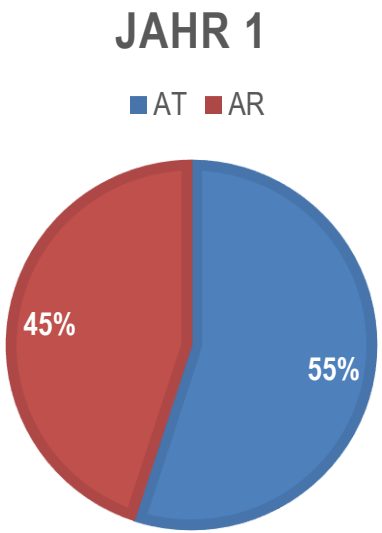


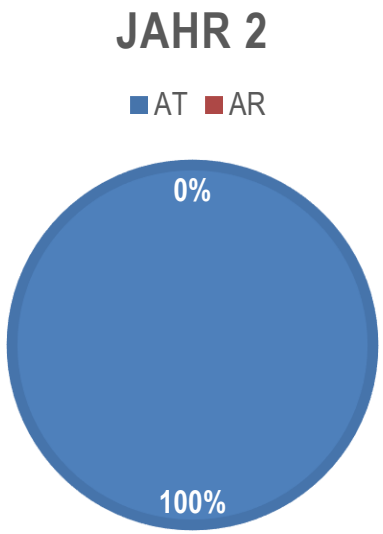
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

GRIN MAINNET PROOF OF WORKS

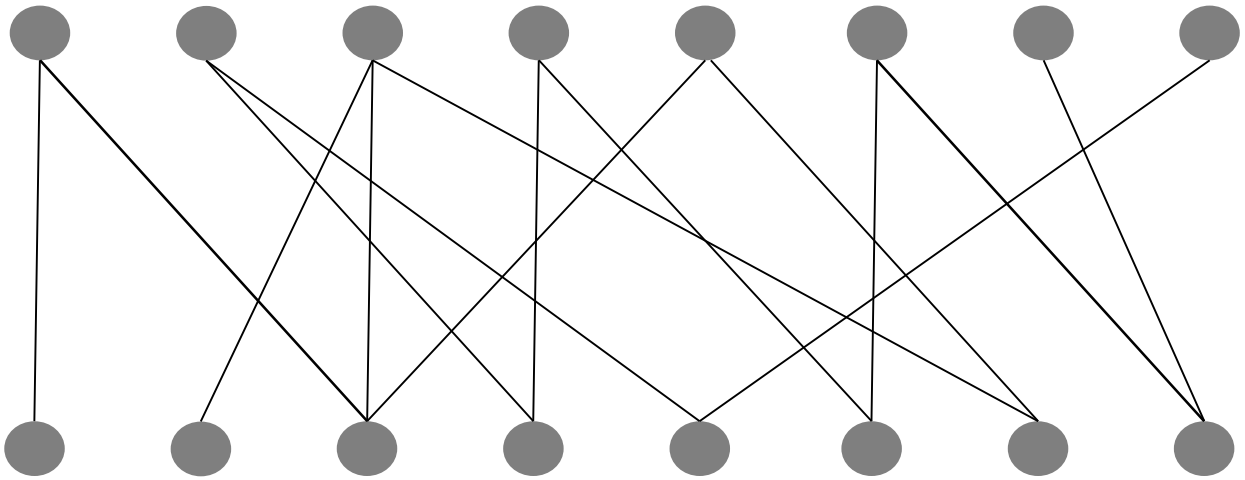
- For GPUs: **CuckARoo29**
- For ASICs: **CuckAToo31+**





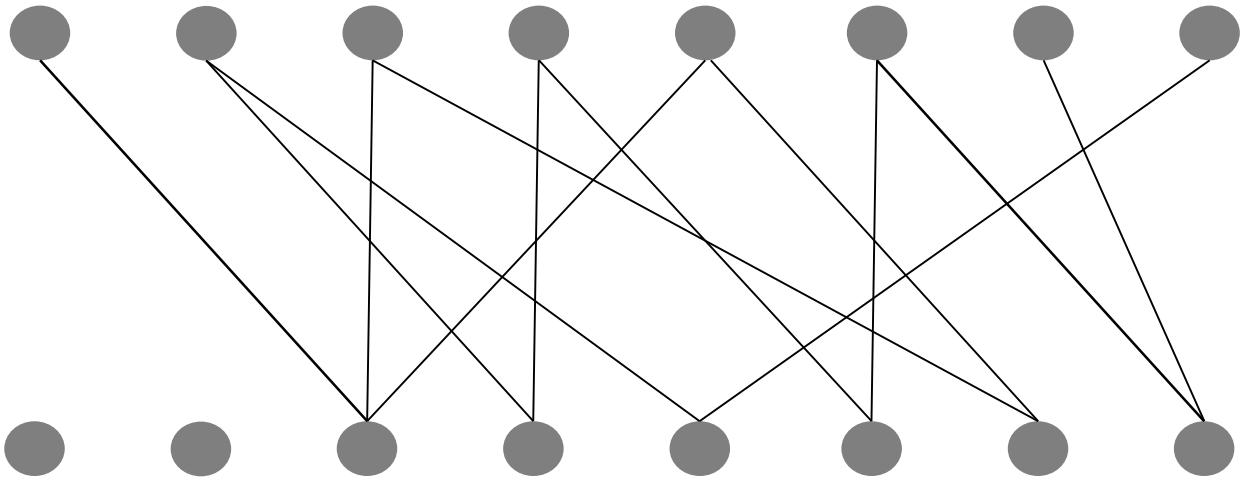


TRIMMING BEI CUCKAROO



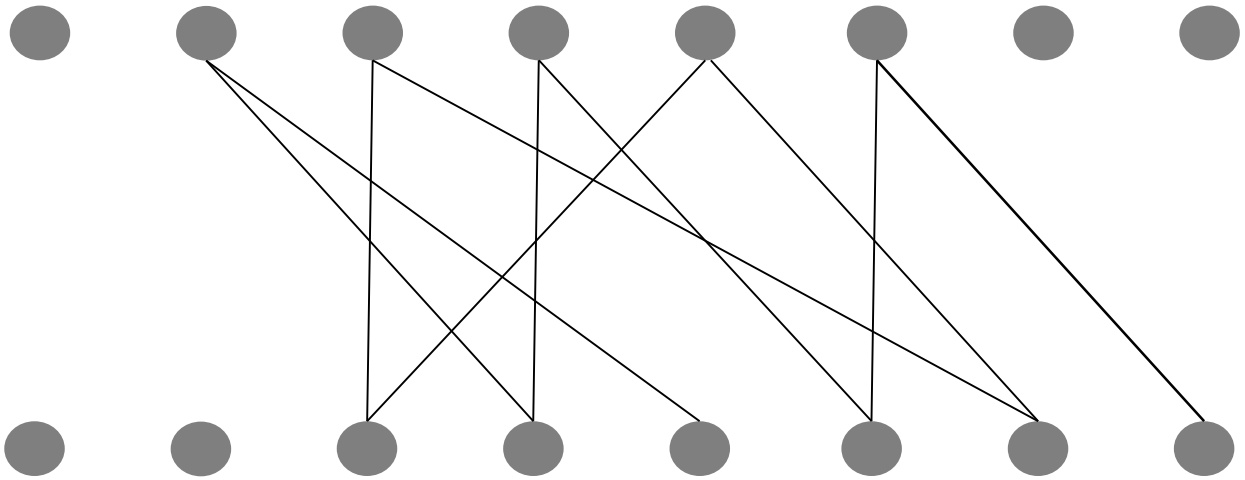
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING BEI CUCKAROO



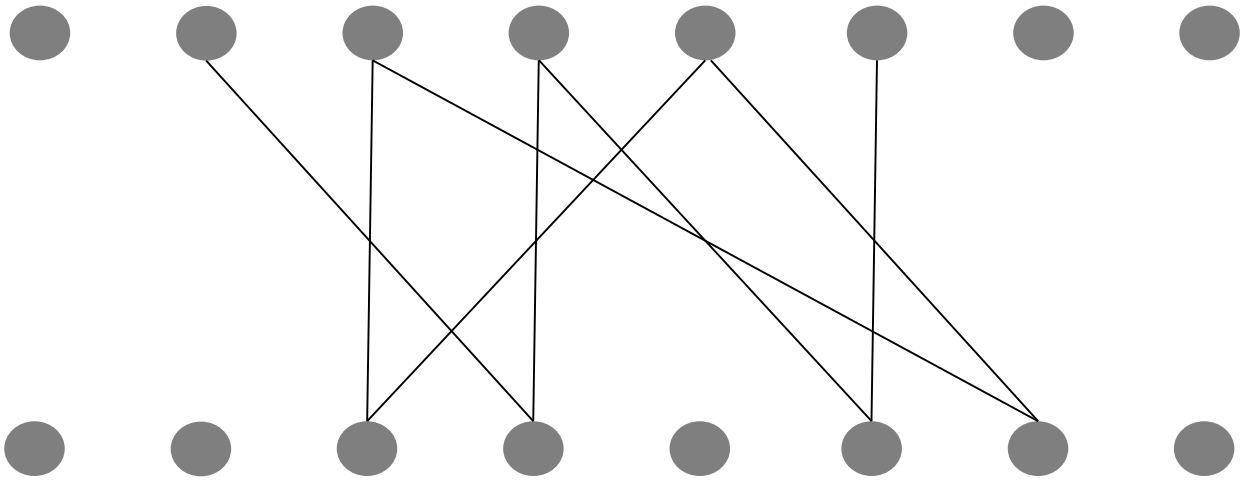
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING BEI CUCKAROO



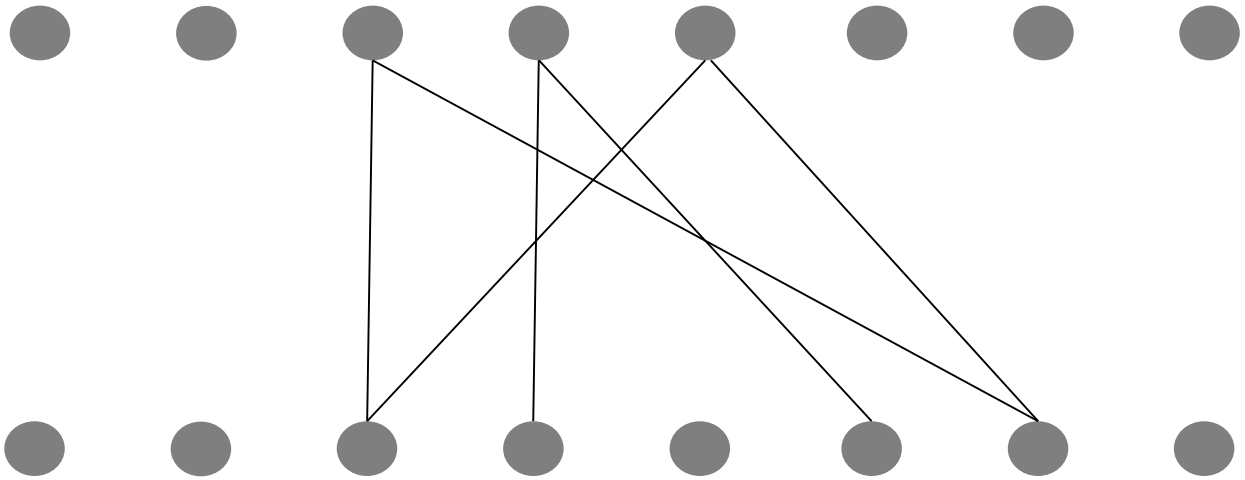
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING BEI CUCKAROO



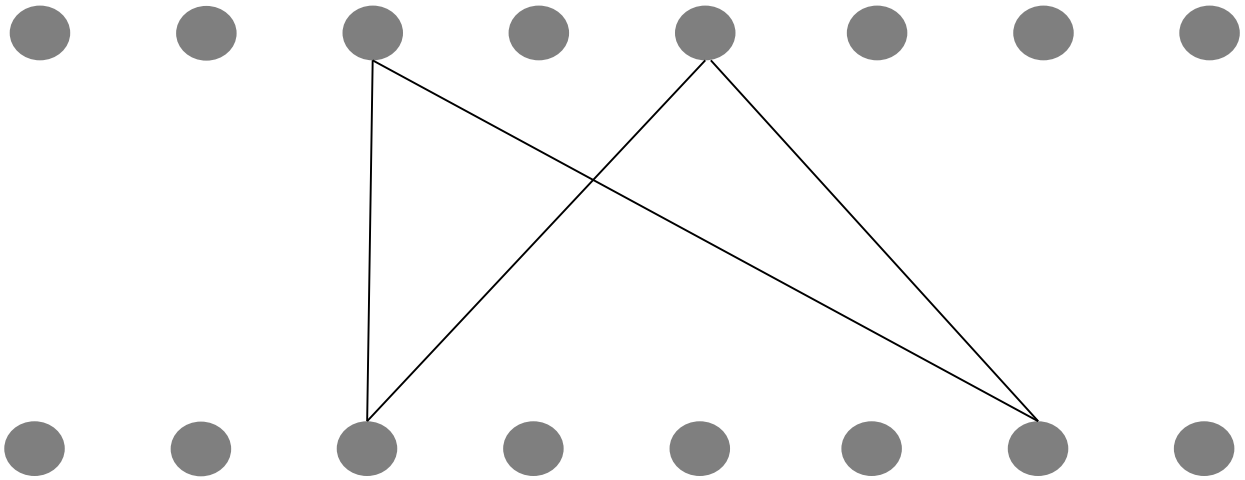
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING BEI CUCKAROO



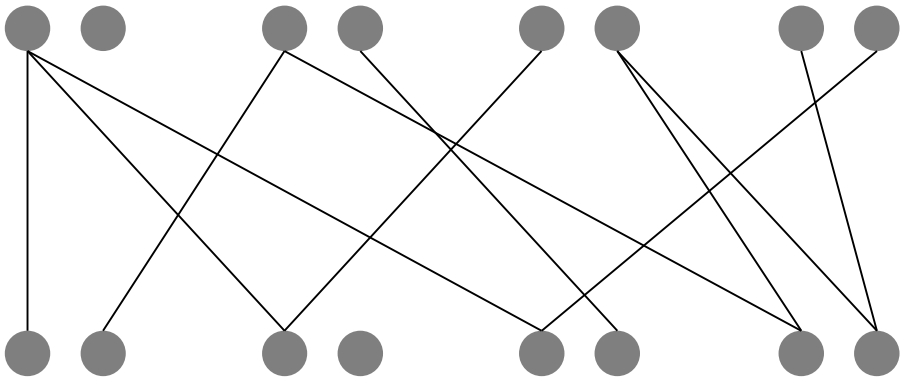
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING BEI CUCKAROO



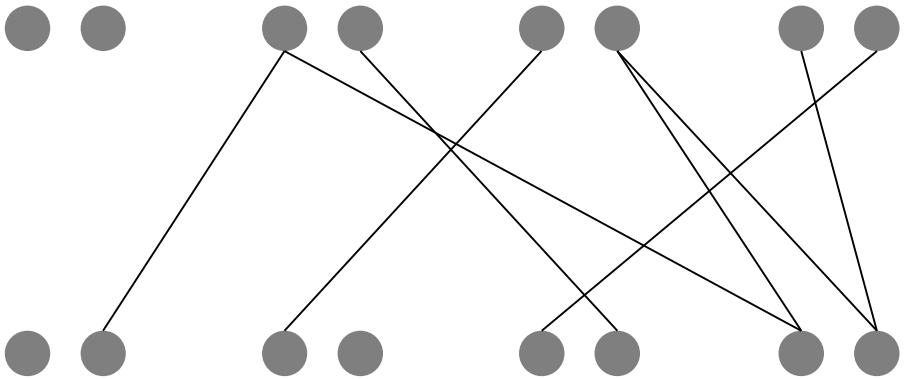
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING BEI CUCKATOO



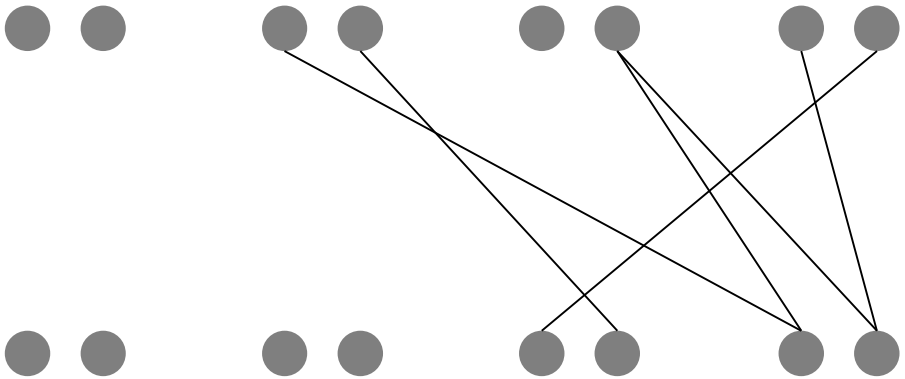
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING BEI CUCKATOO



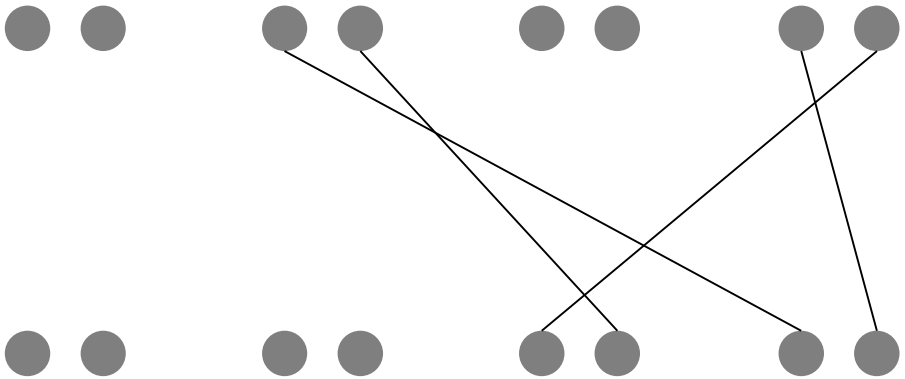
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING BEI CUCKATOO



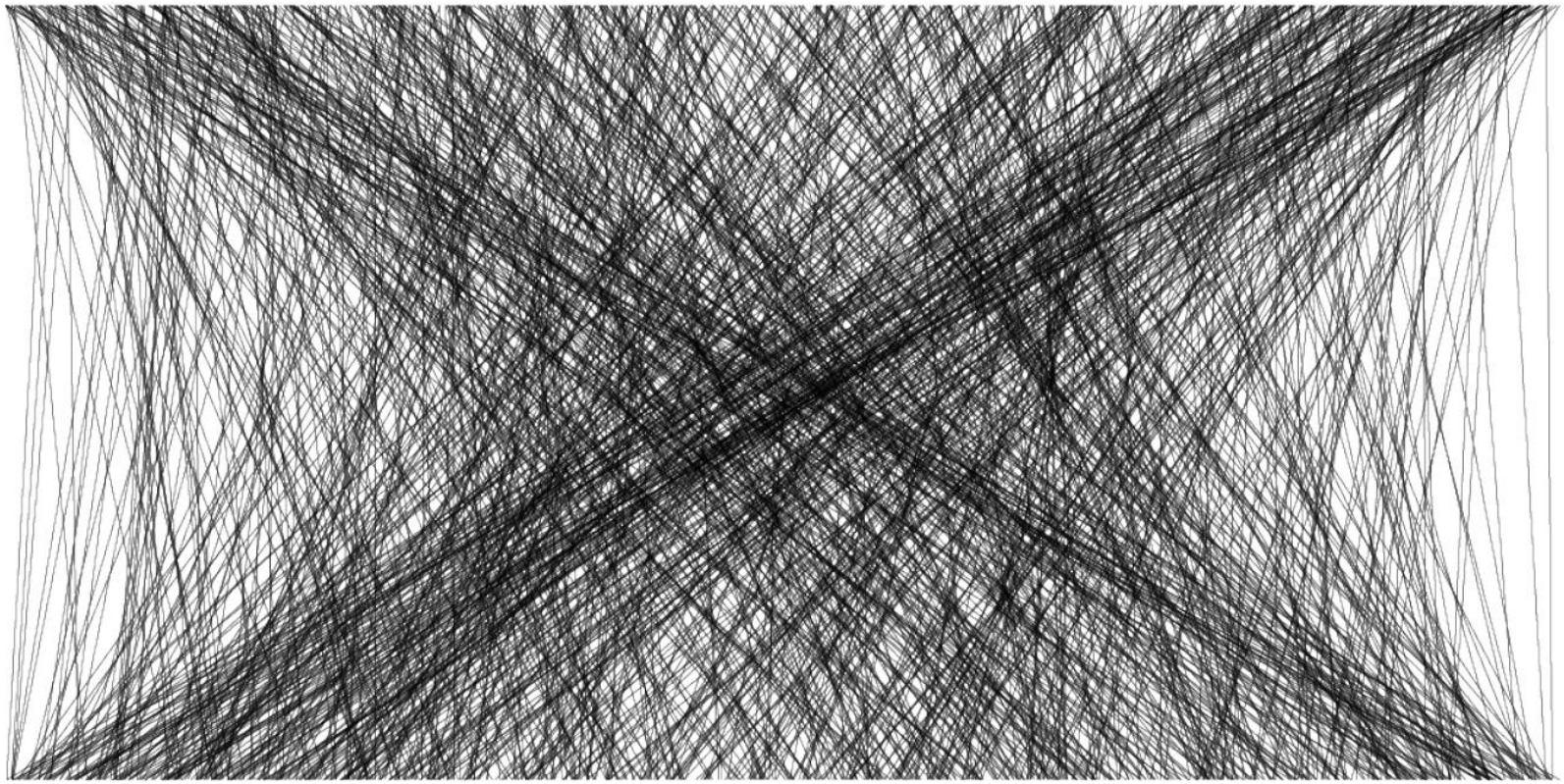
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING BEI CUCKATOO



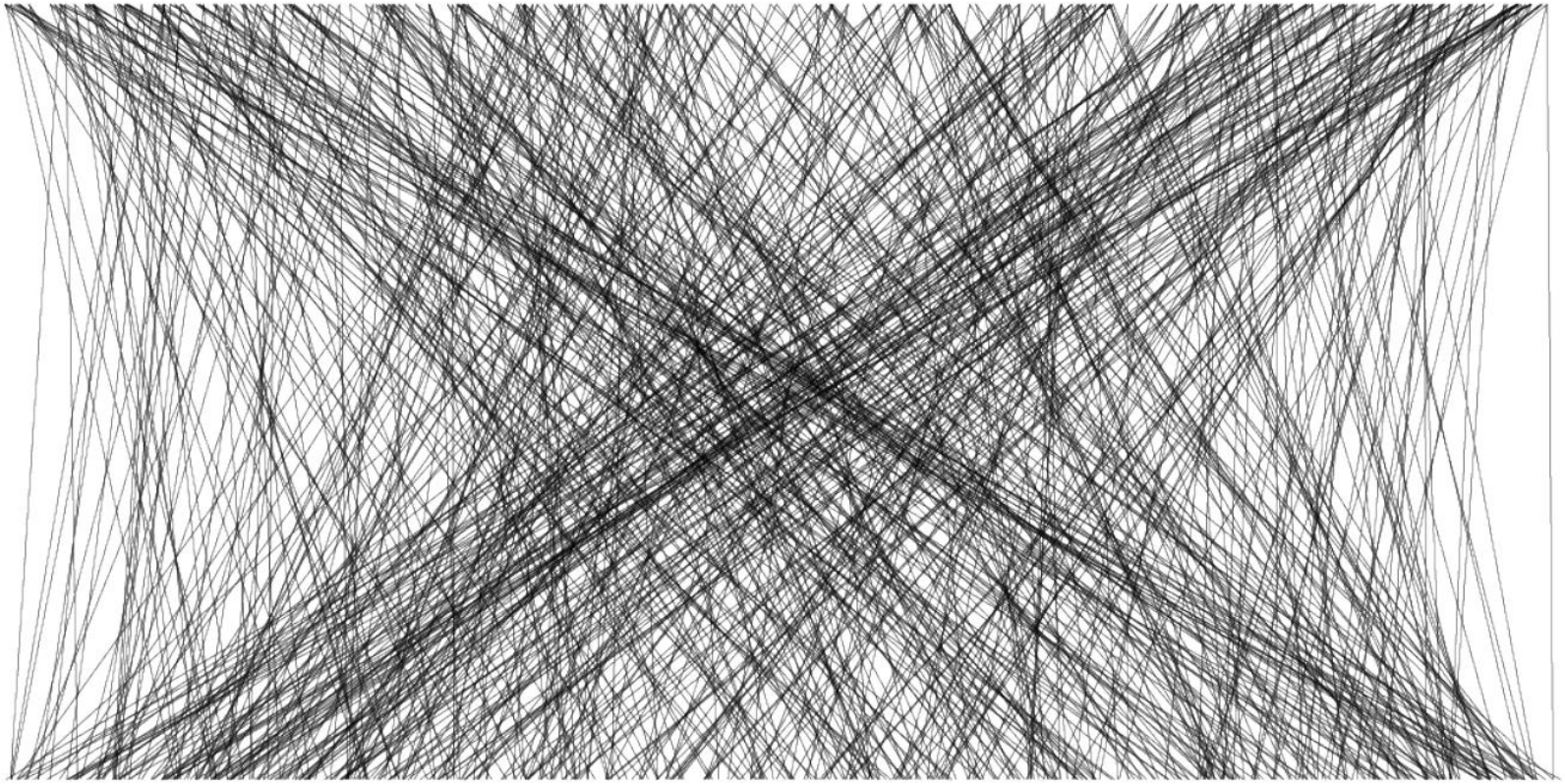
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



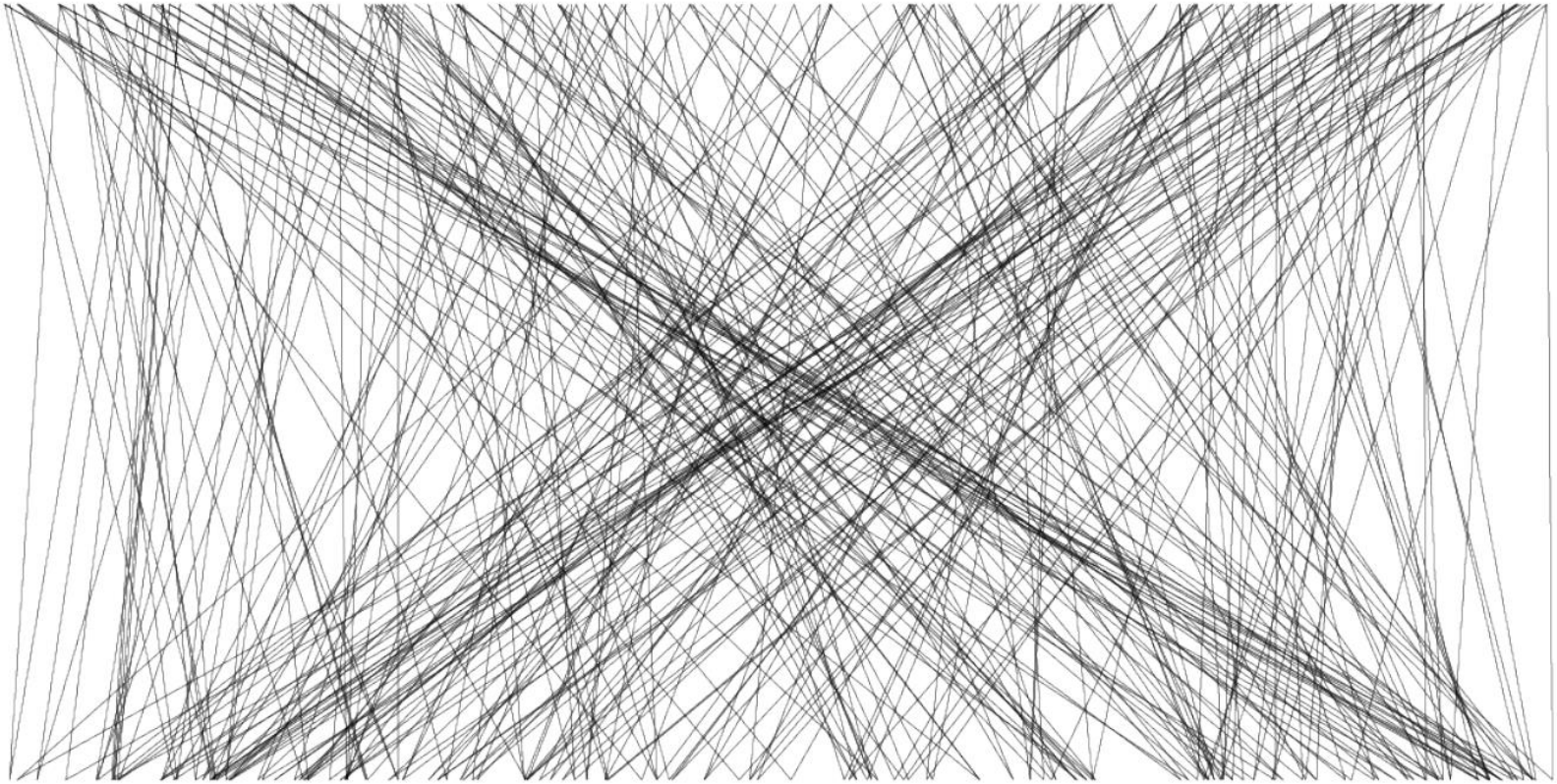
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



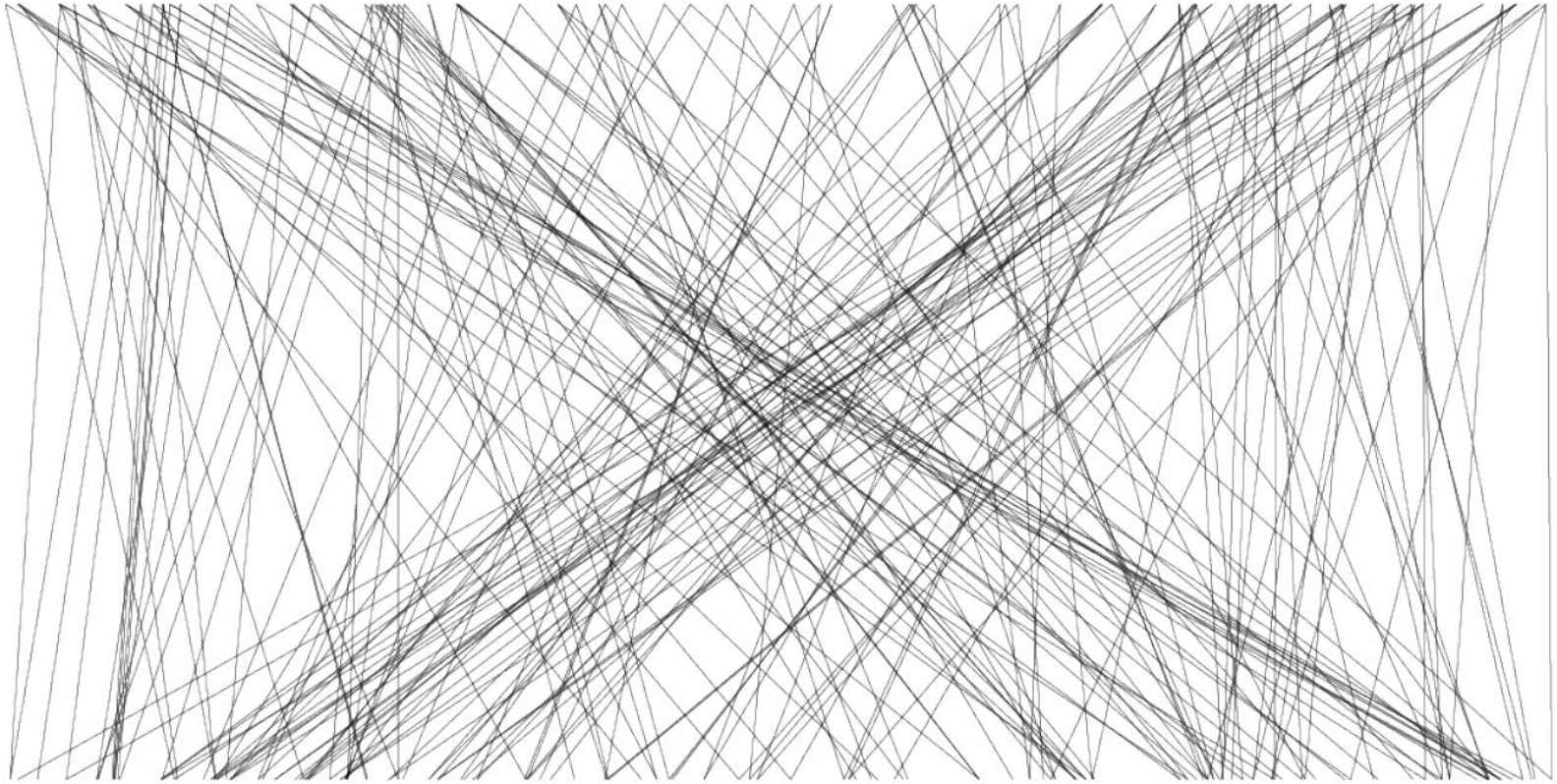
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



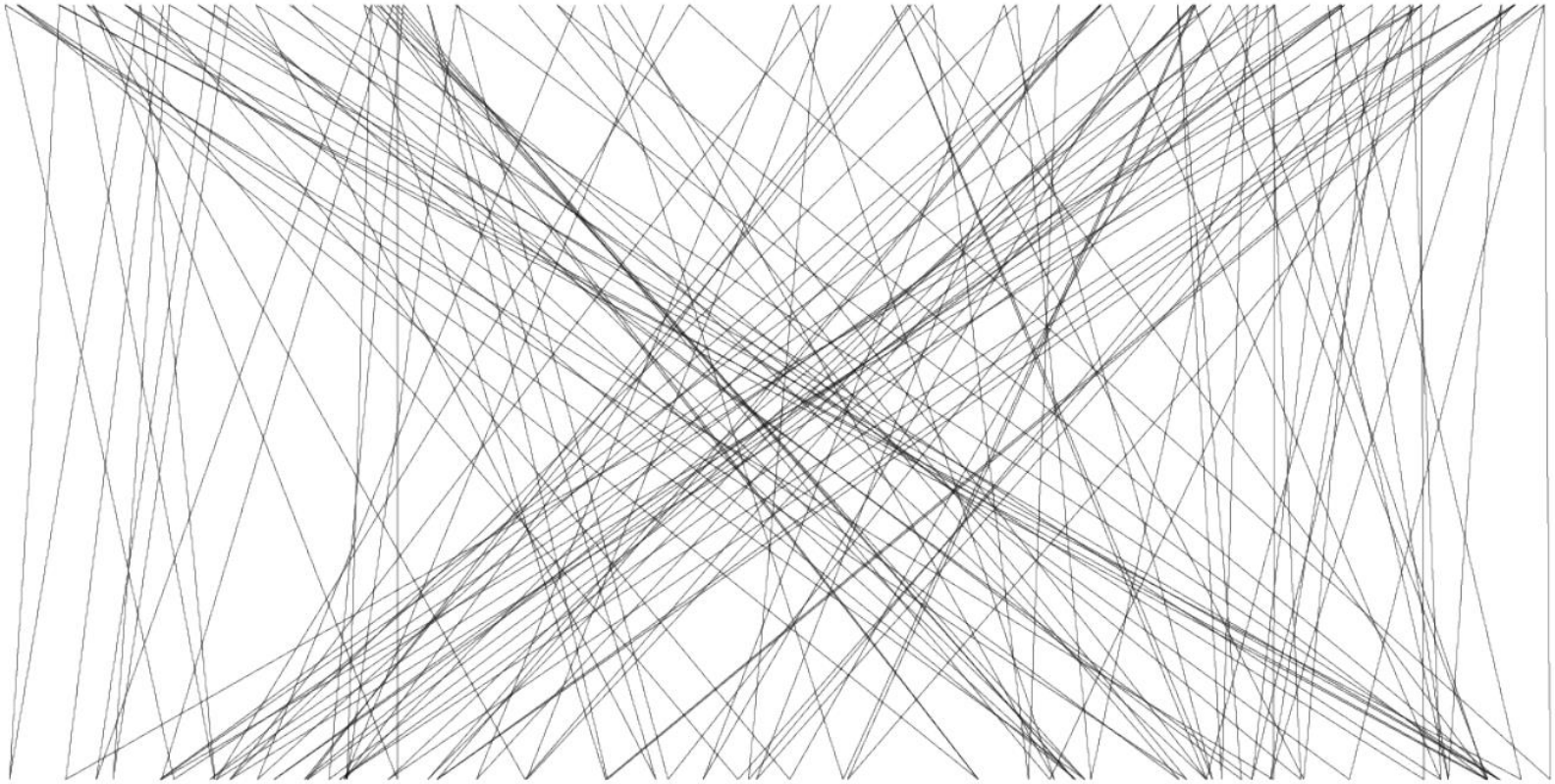
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



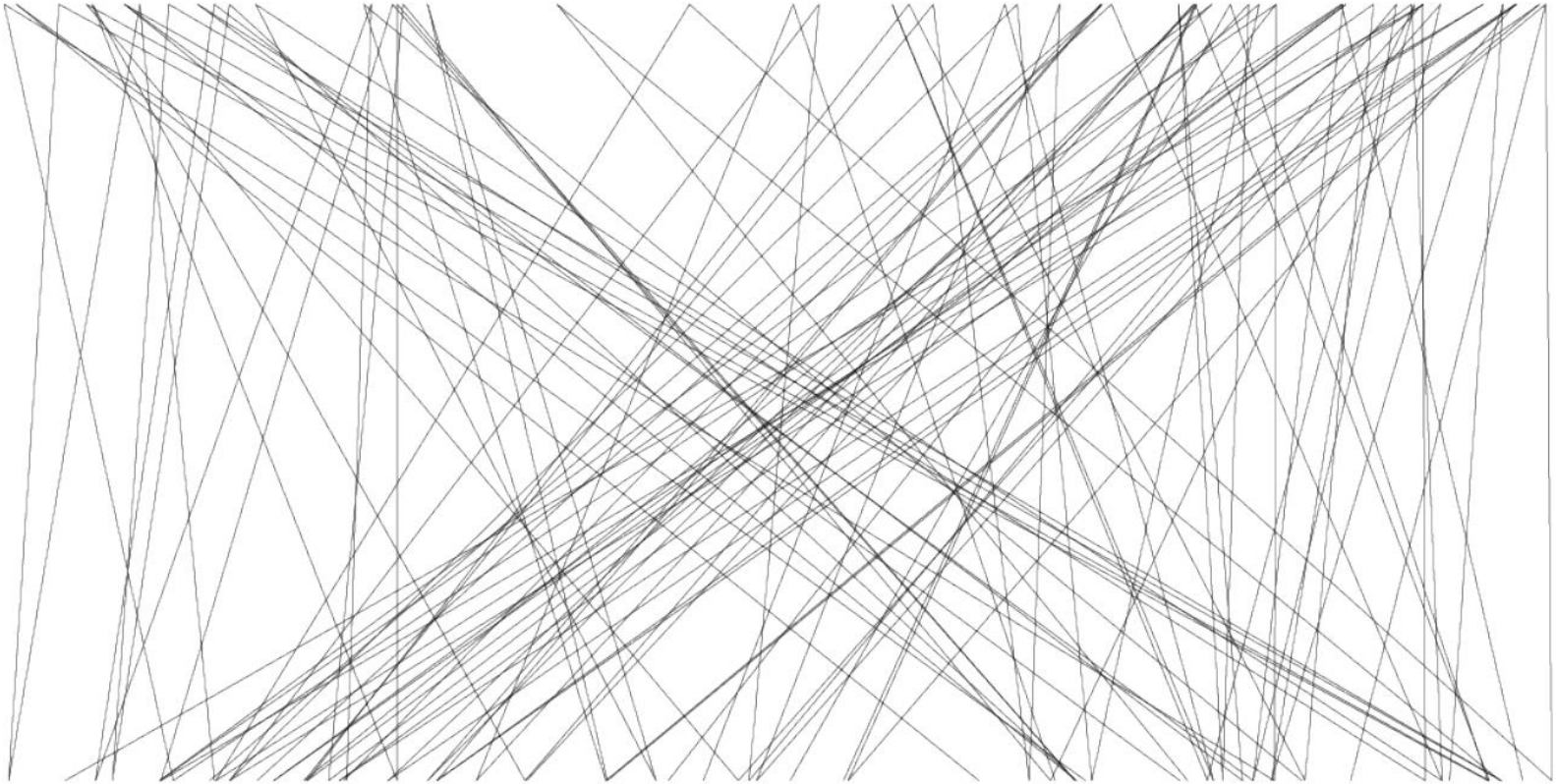
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



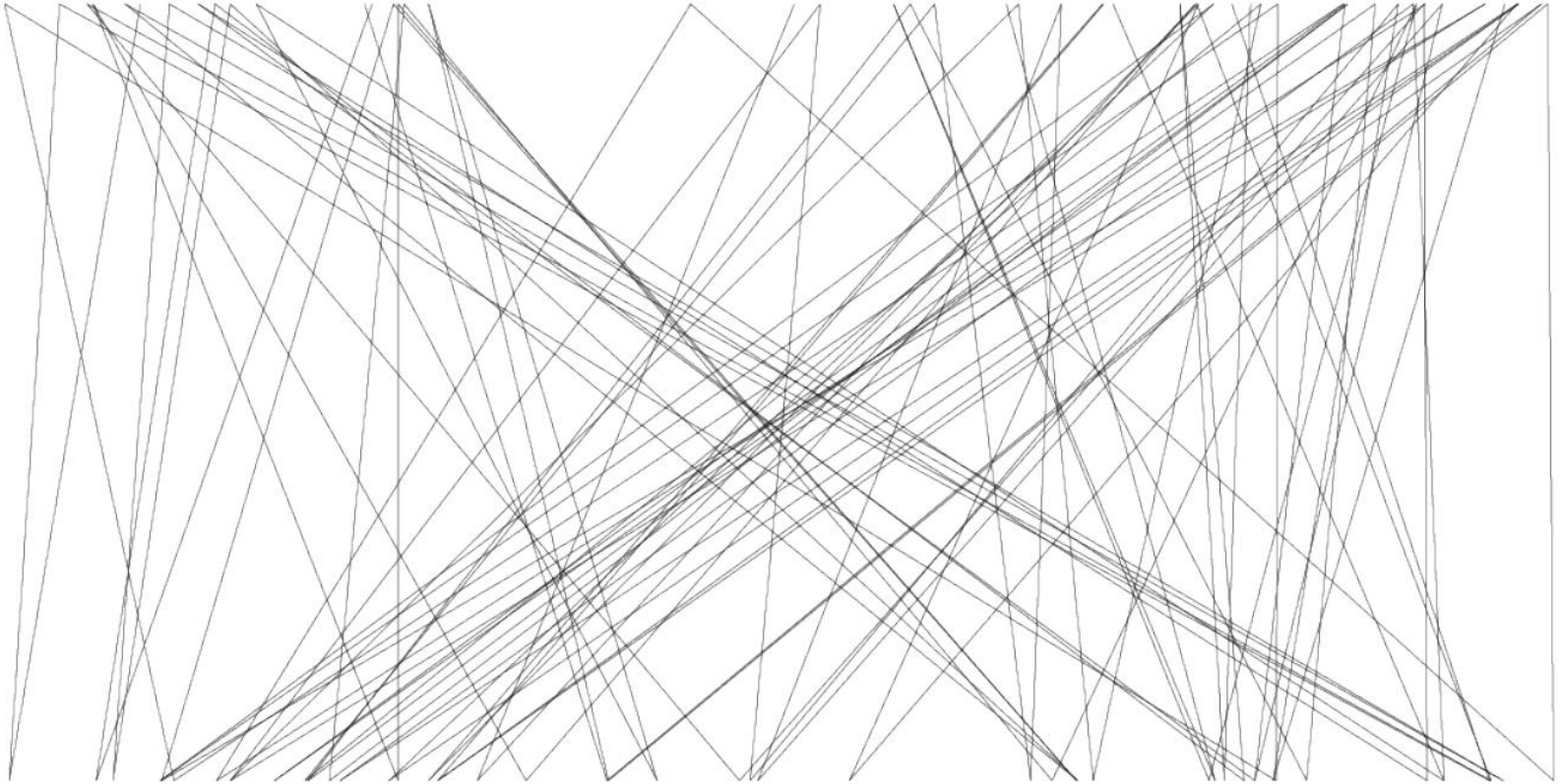
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



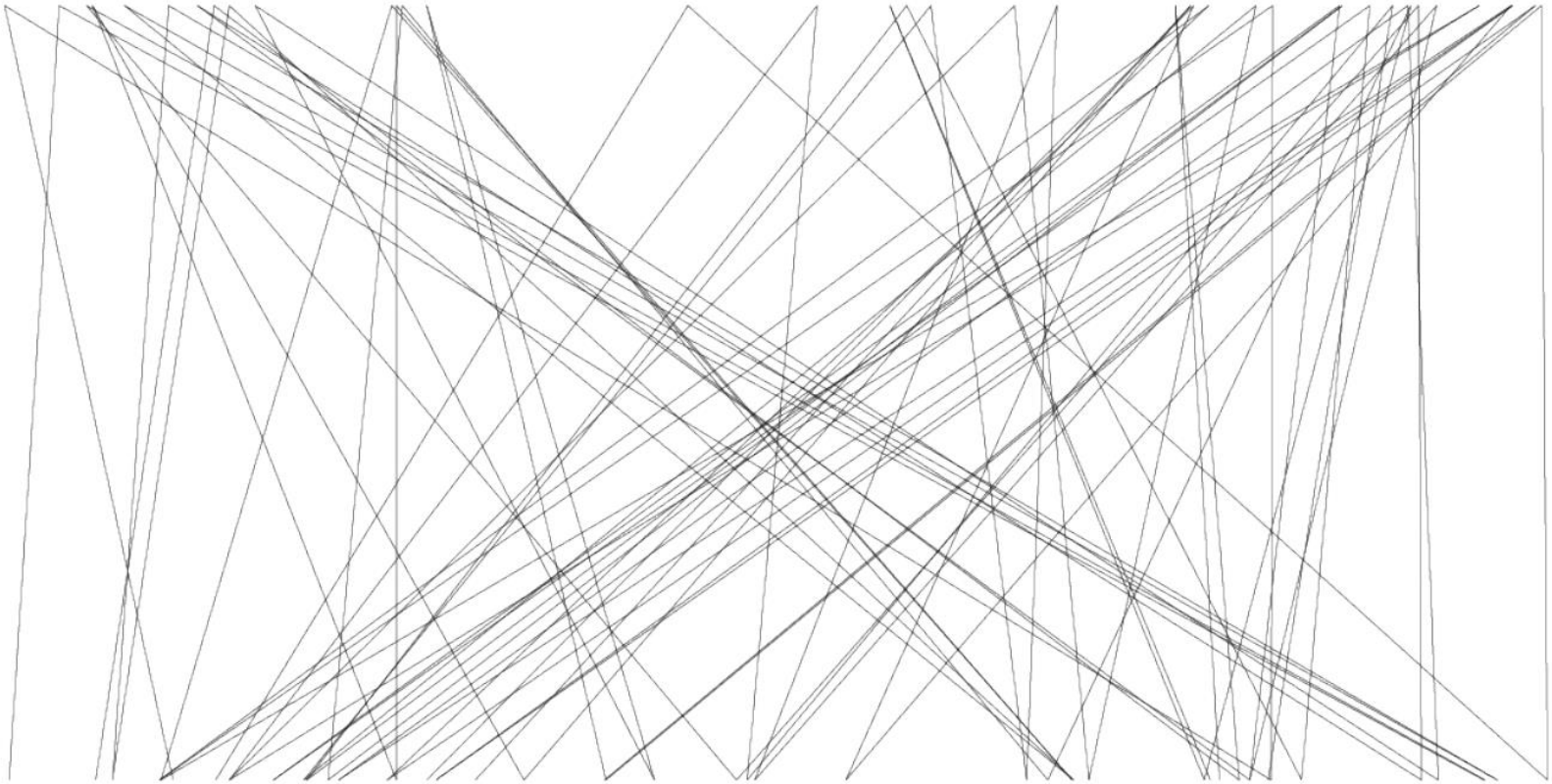
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



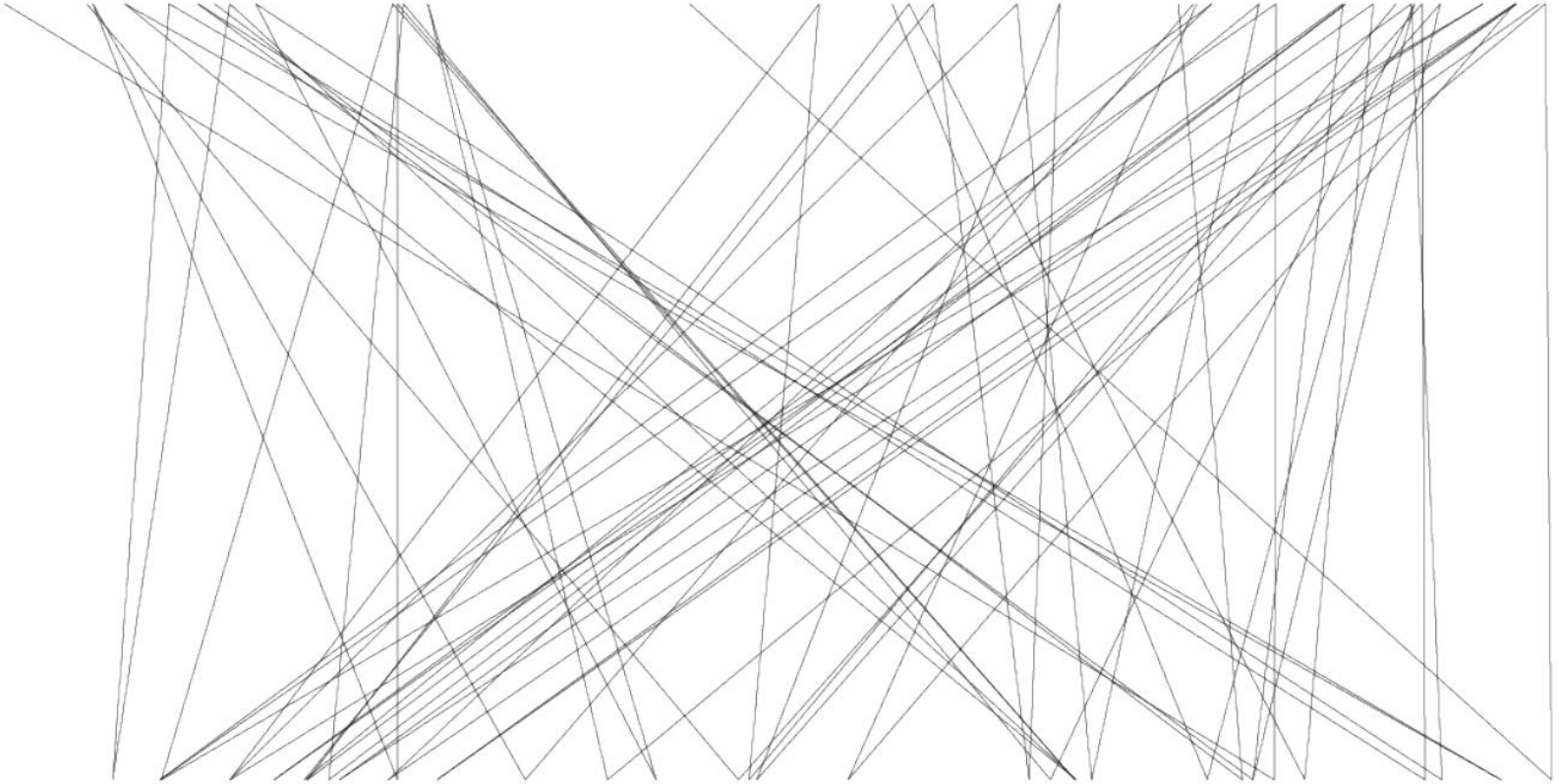
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



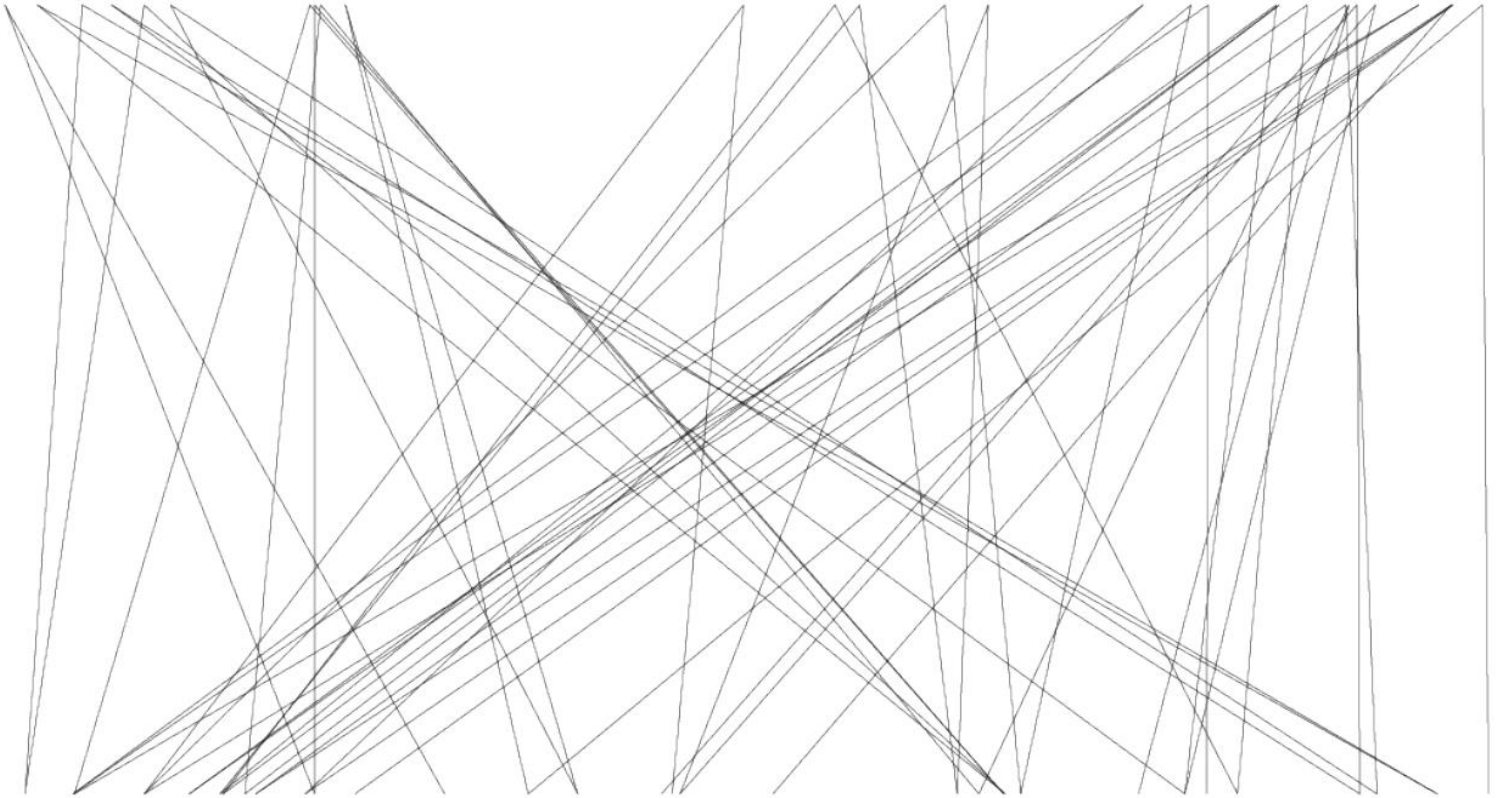
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



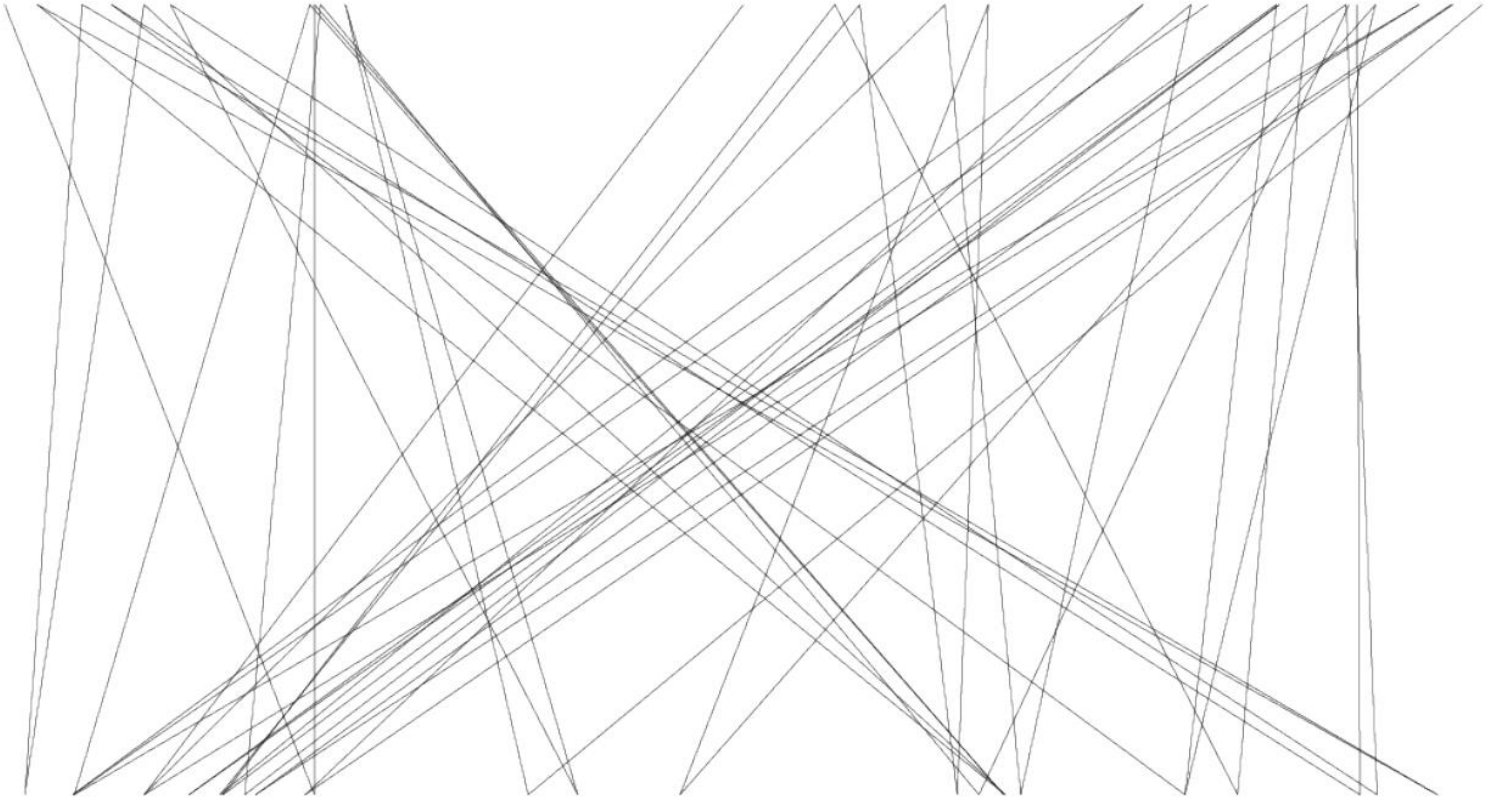
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



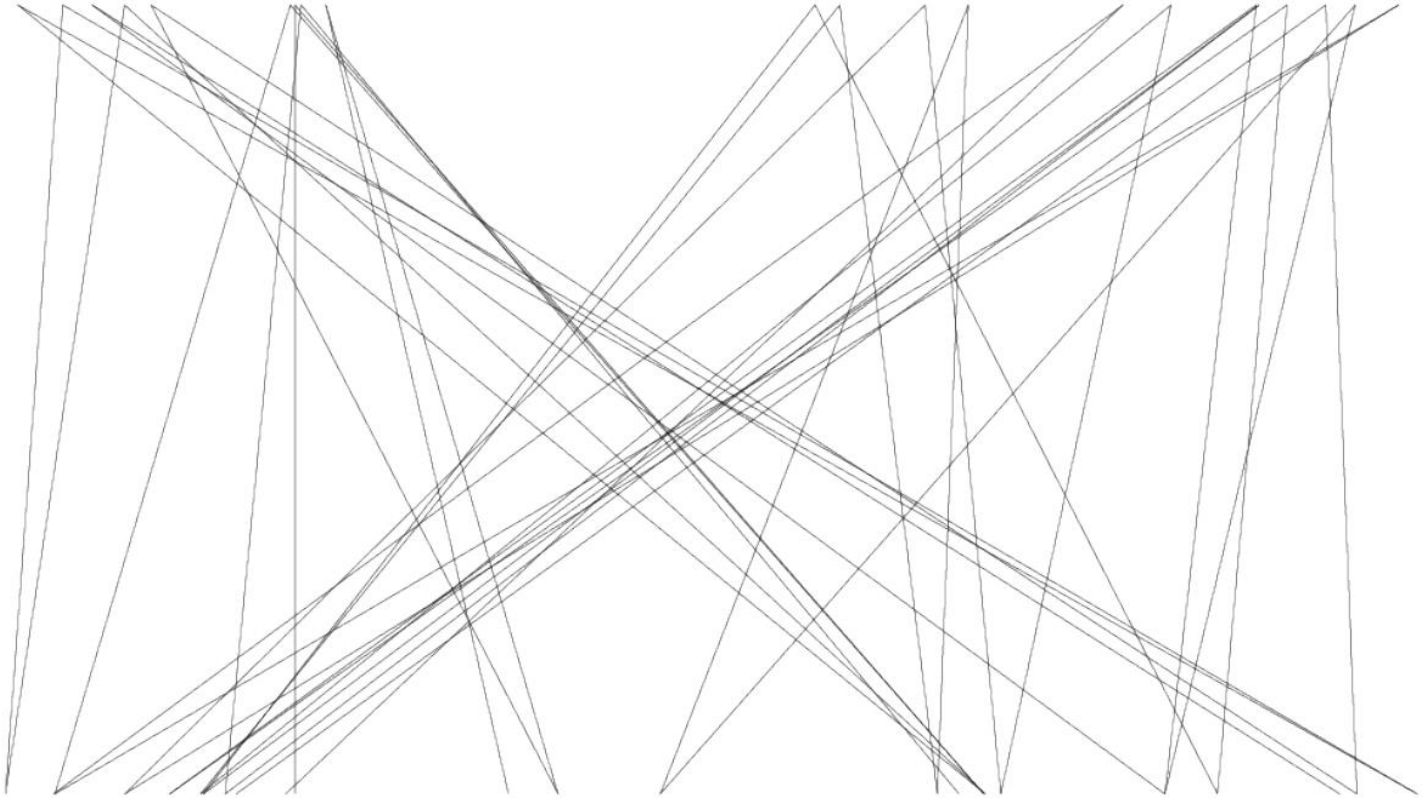
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



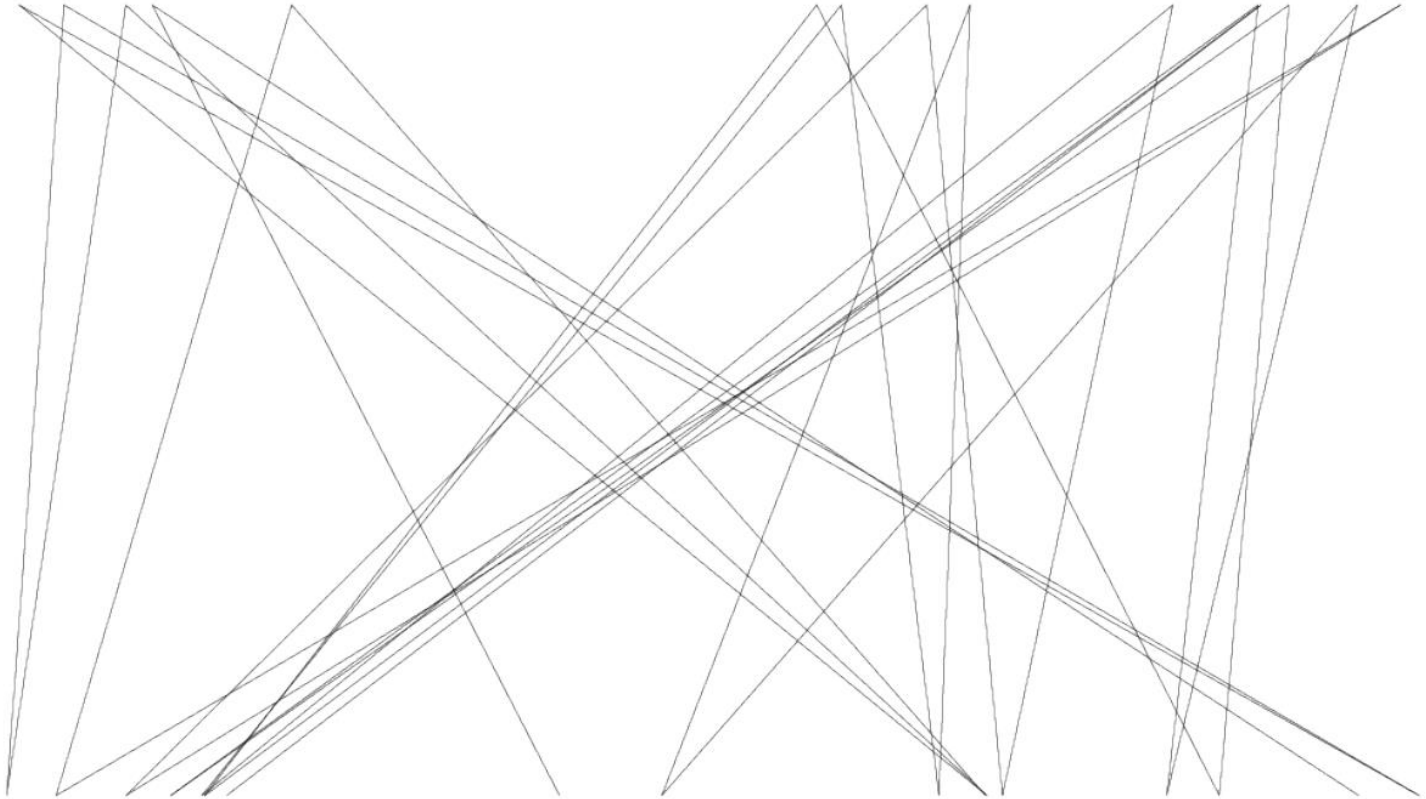
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



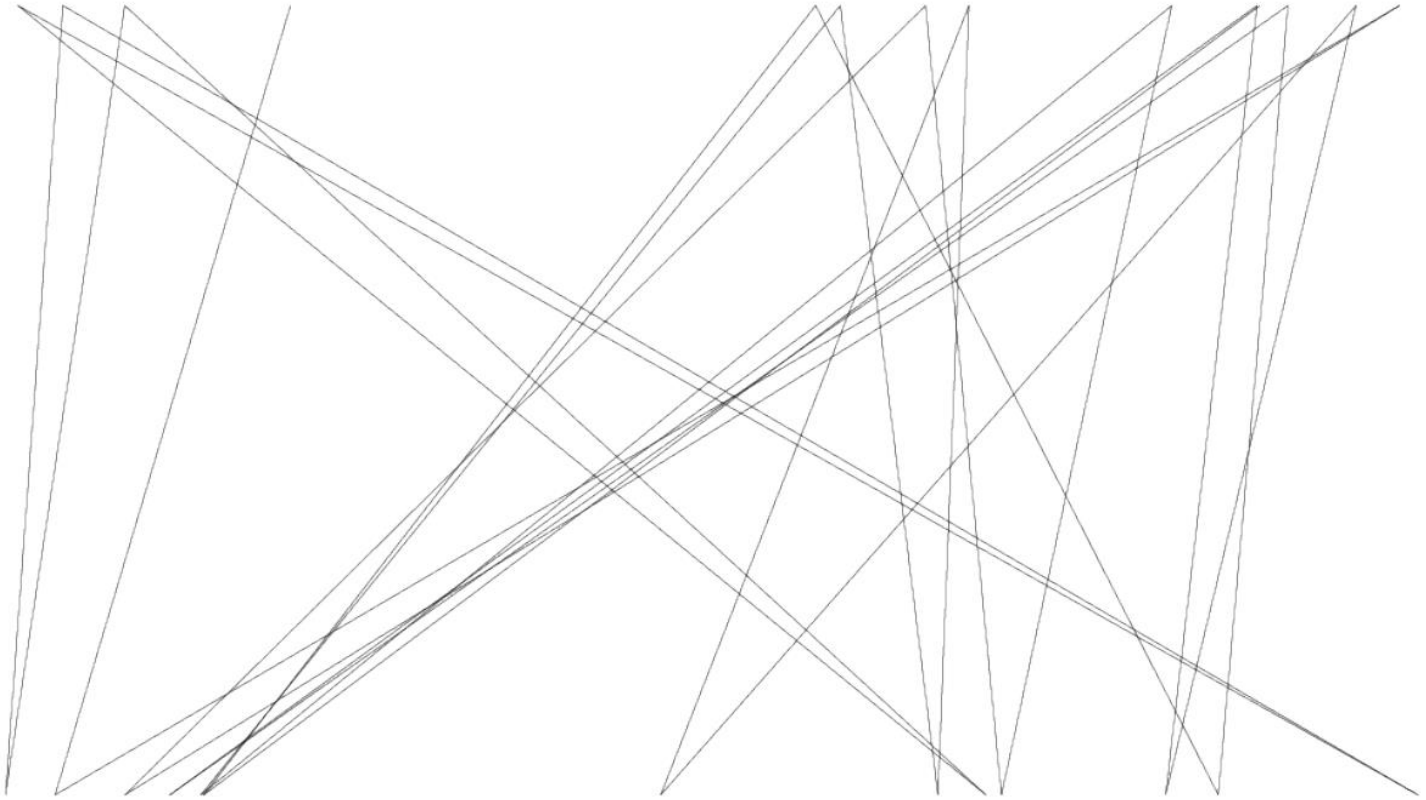
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



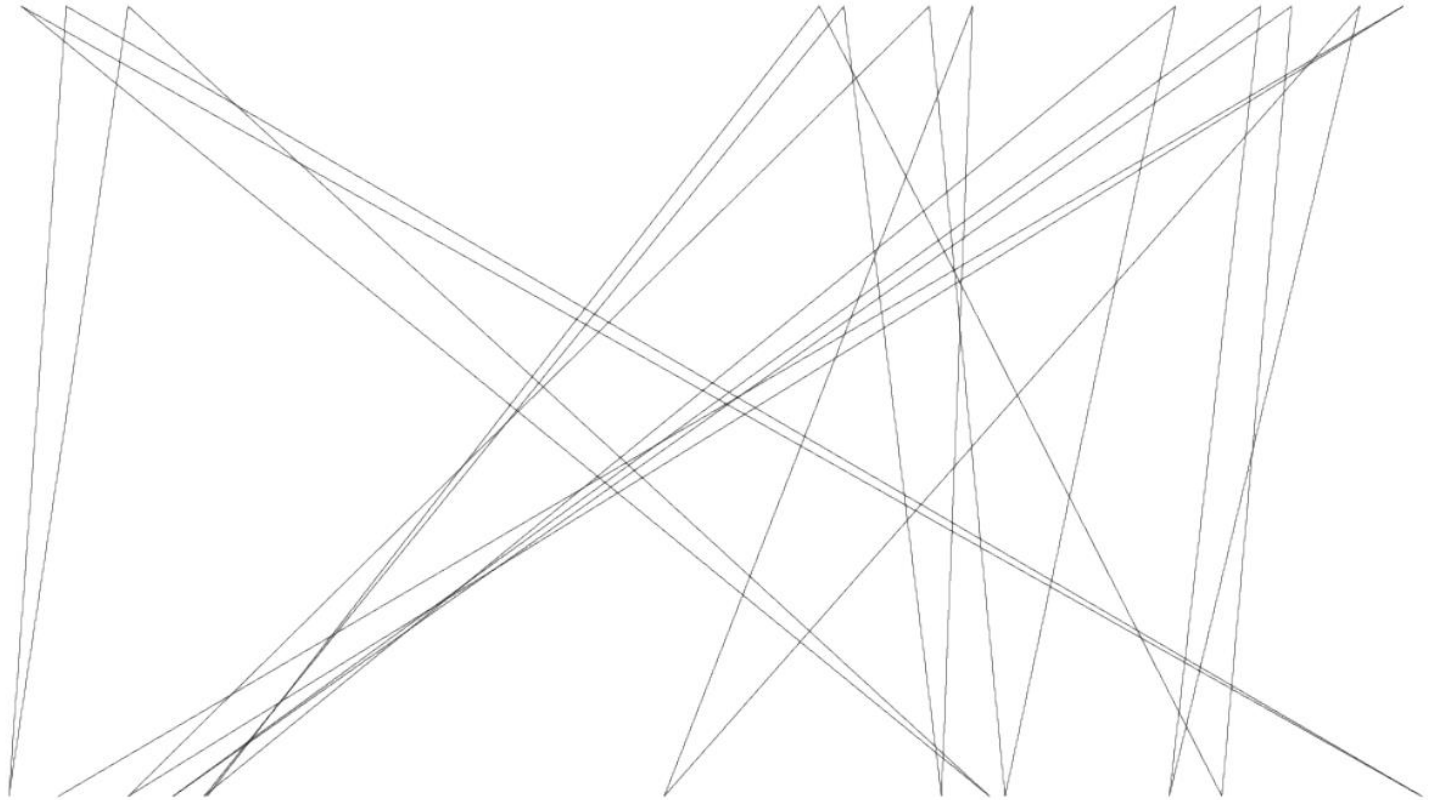
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



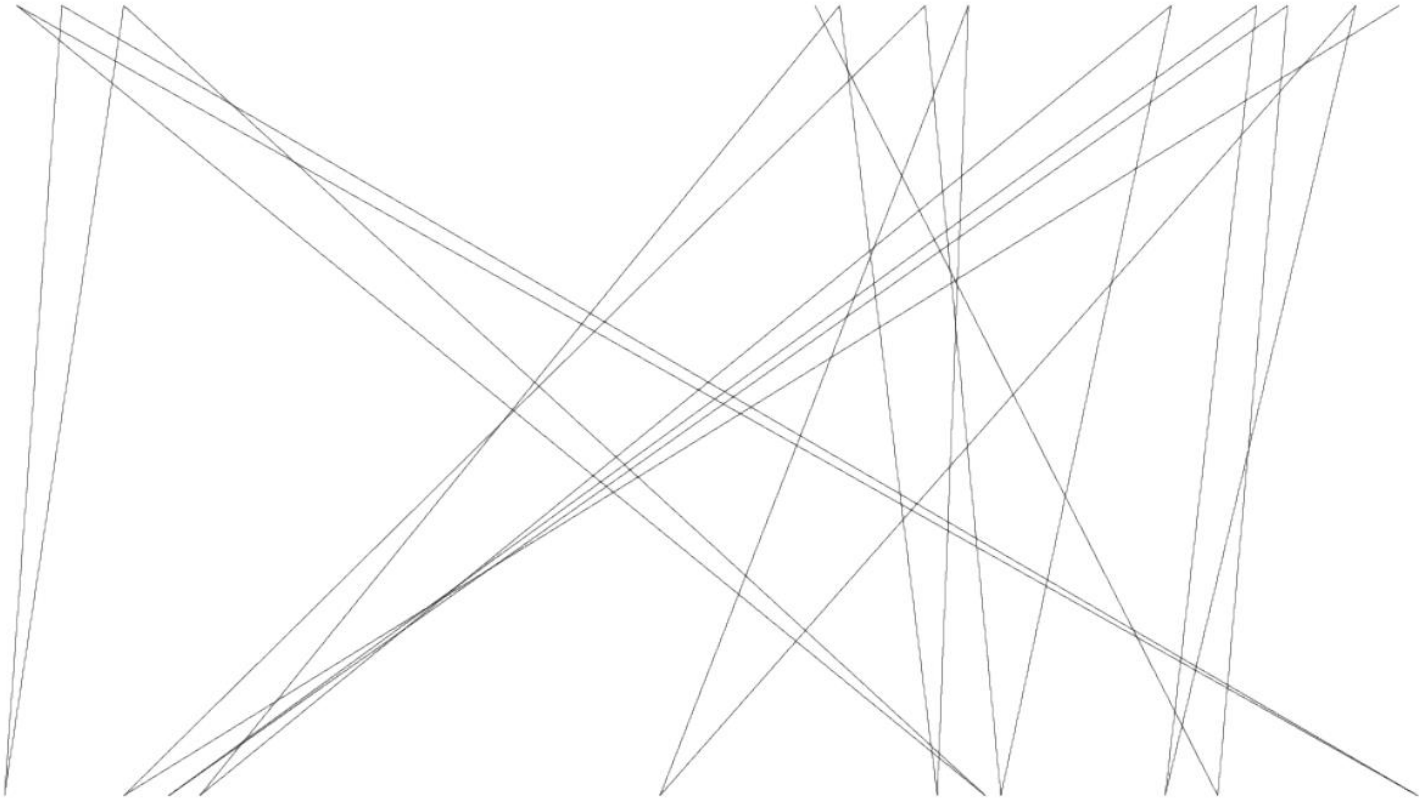
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



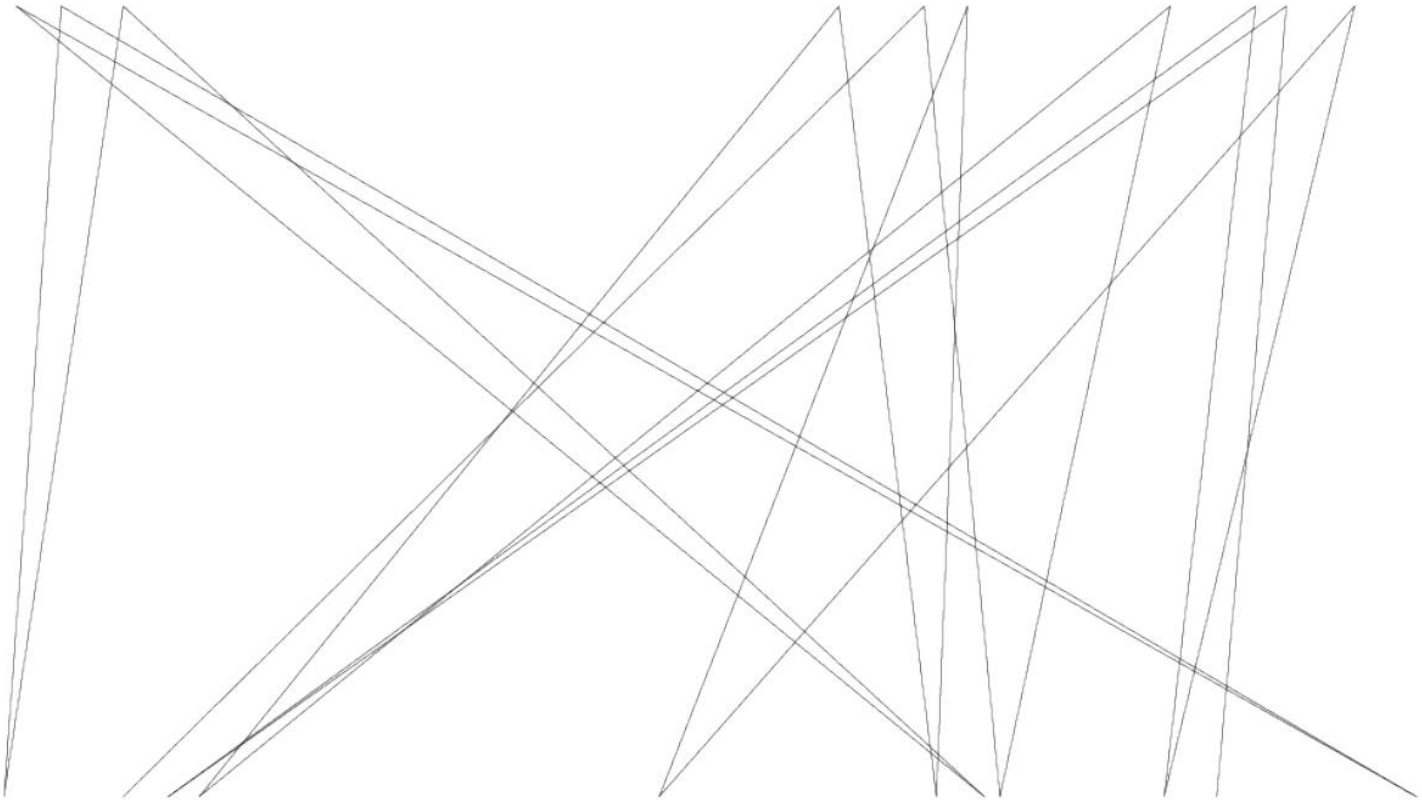
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



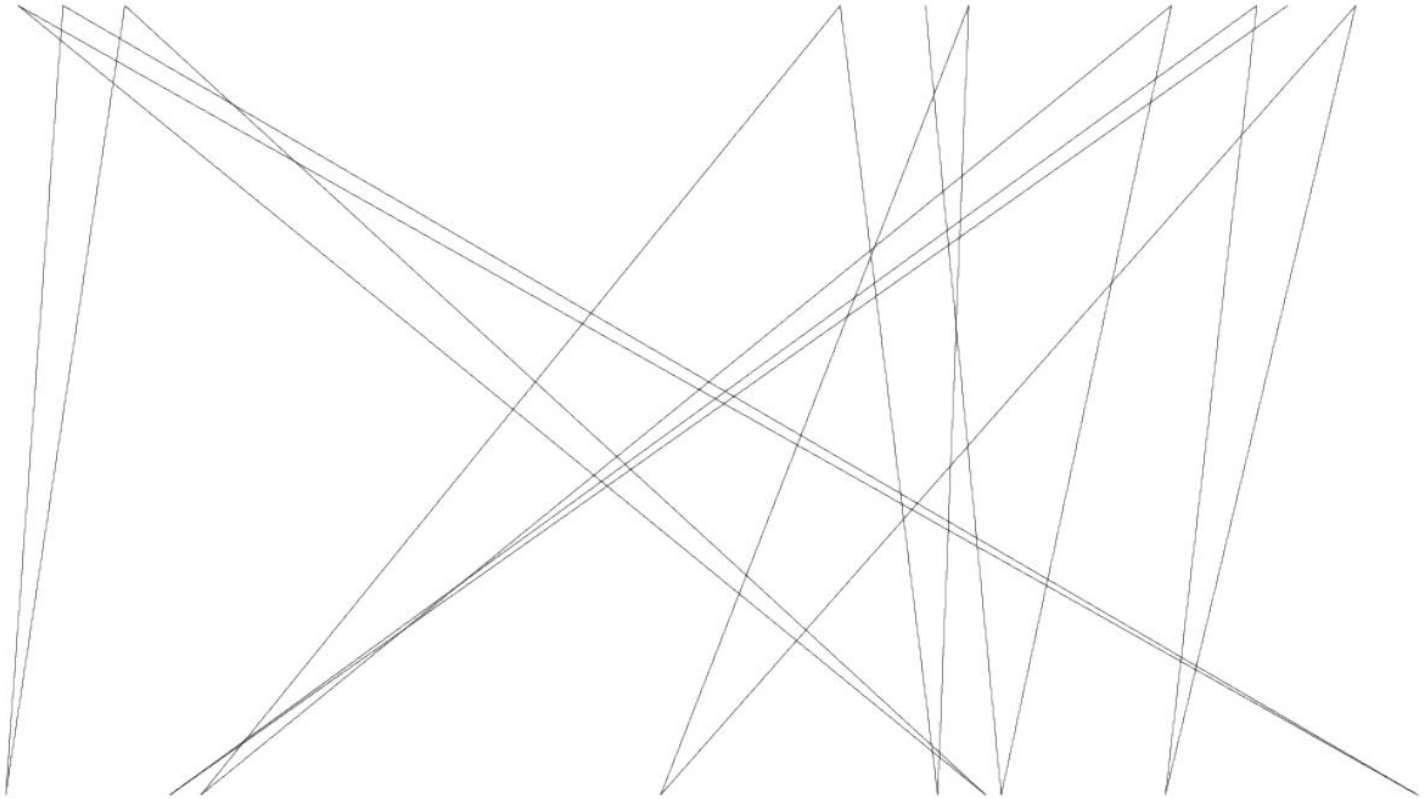
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



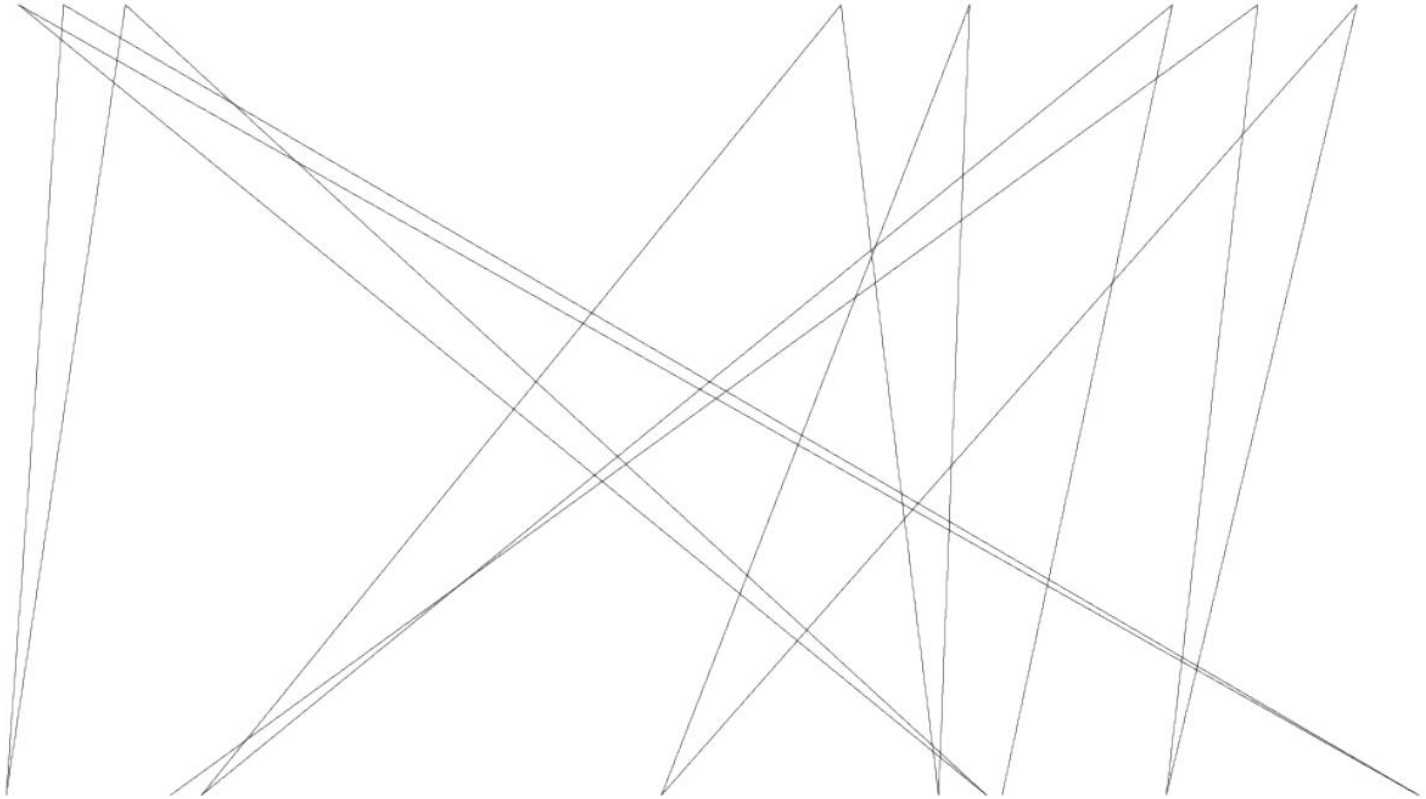
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



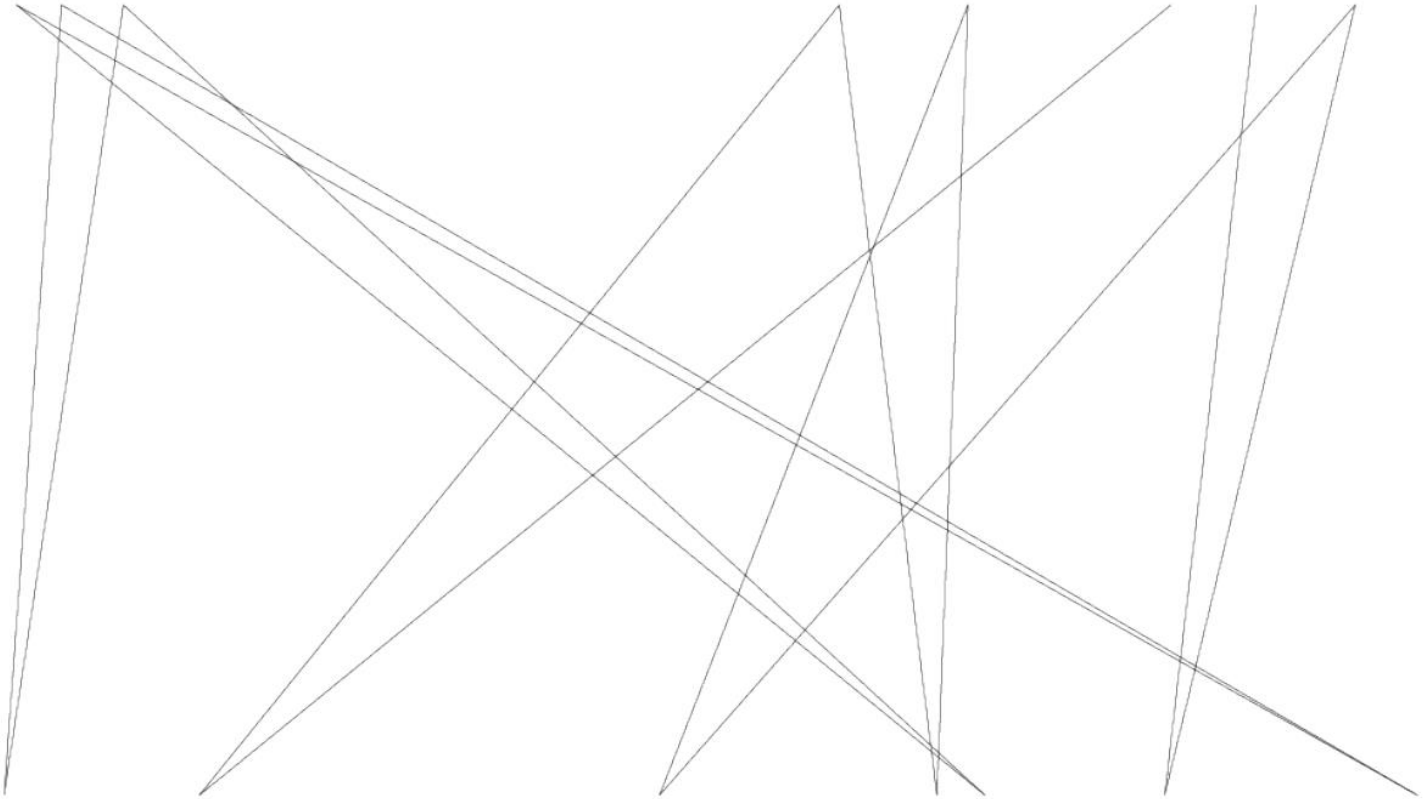
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



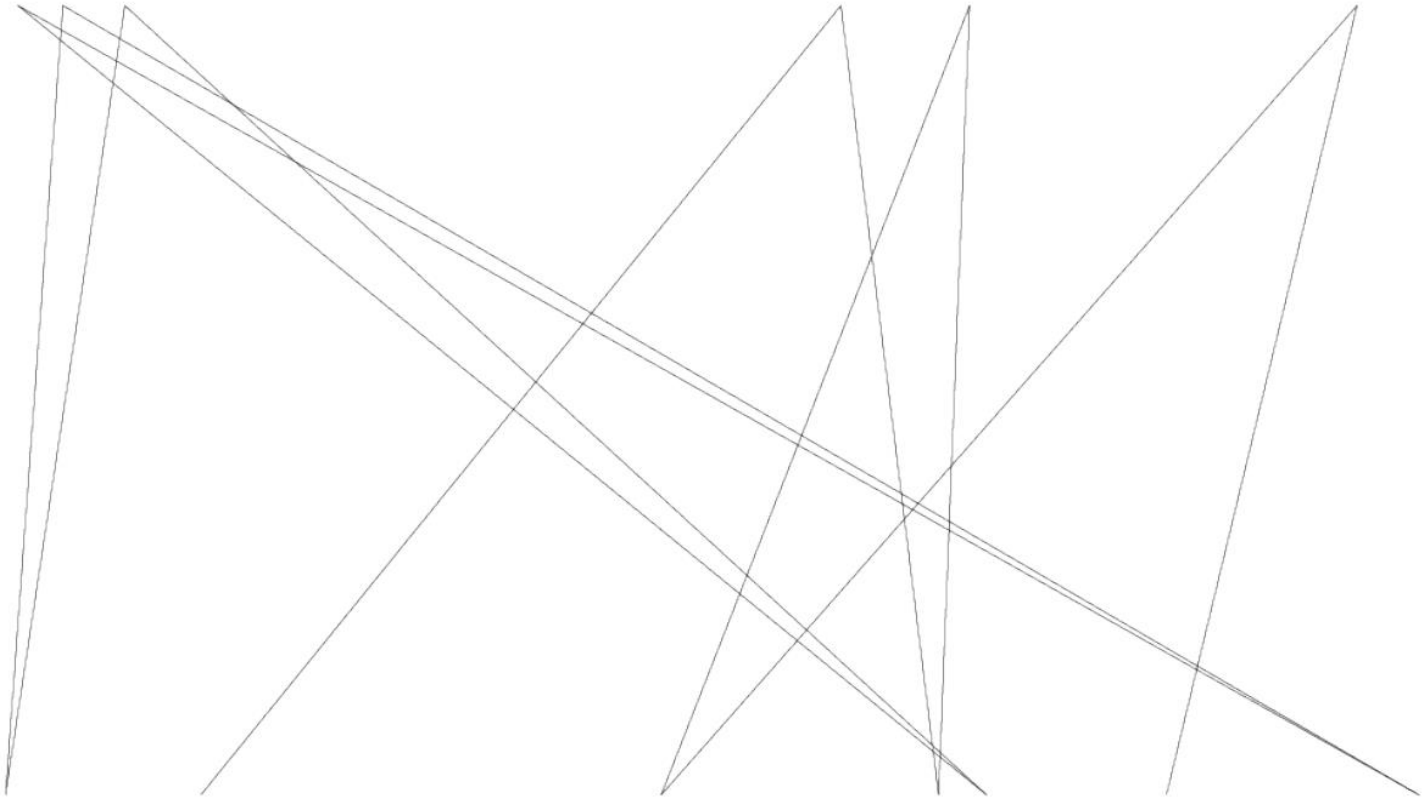
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



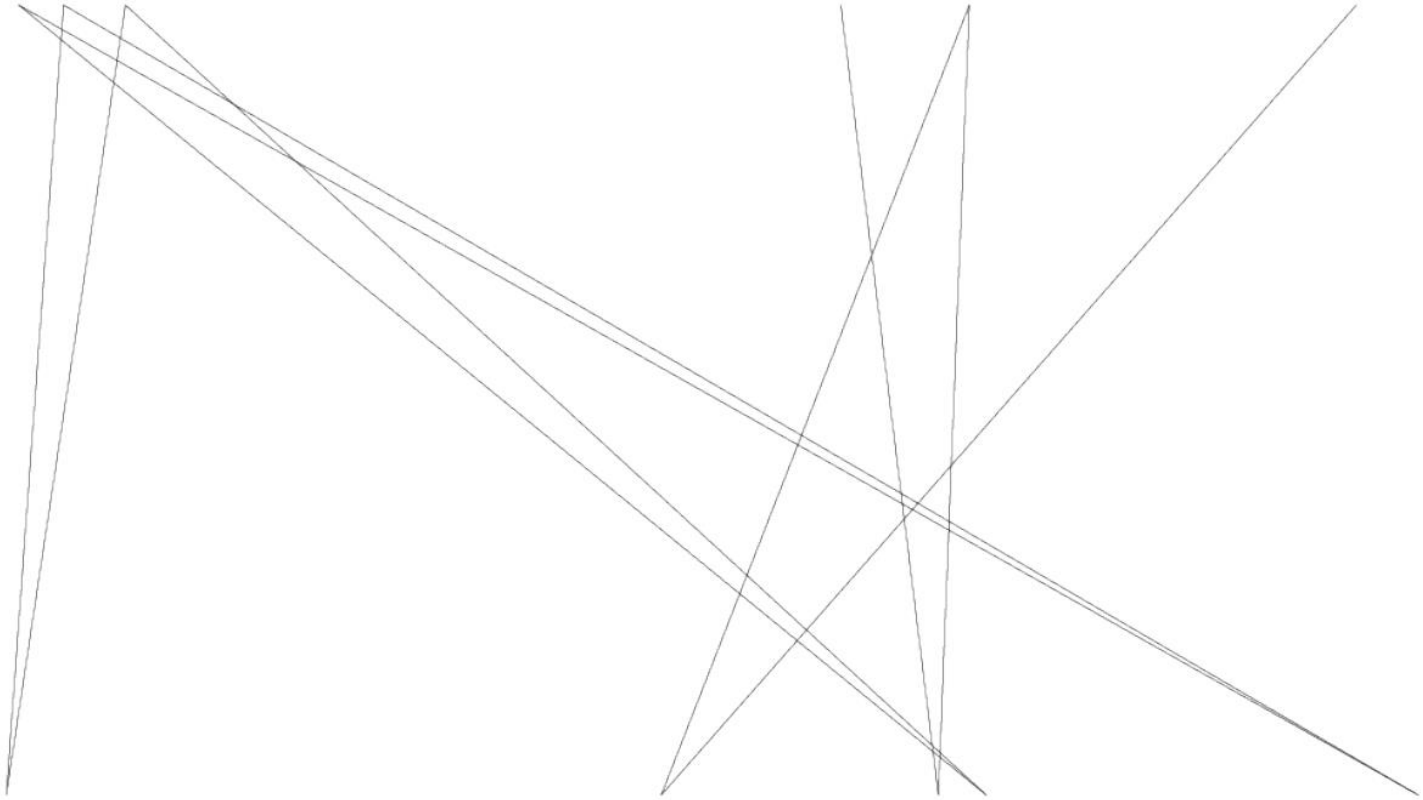
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



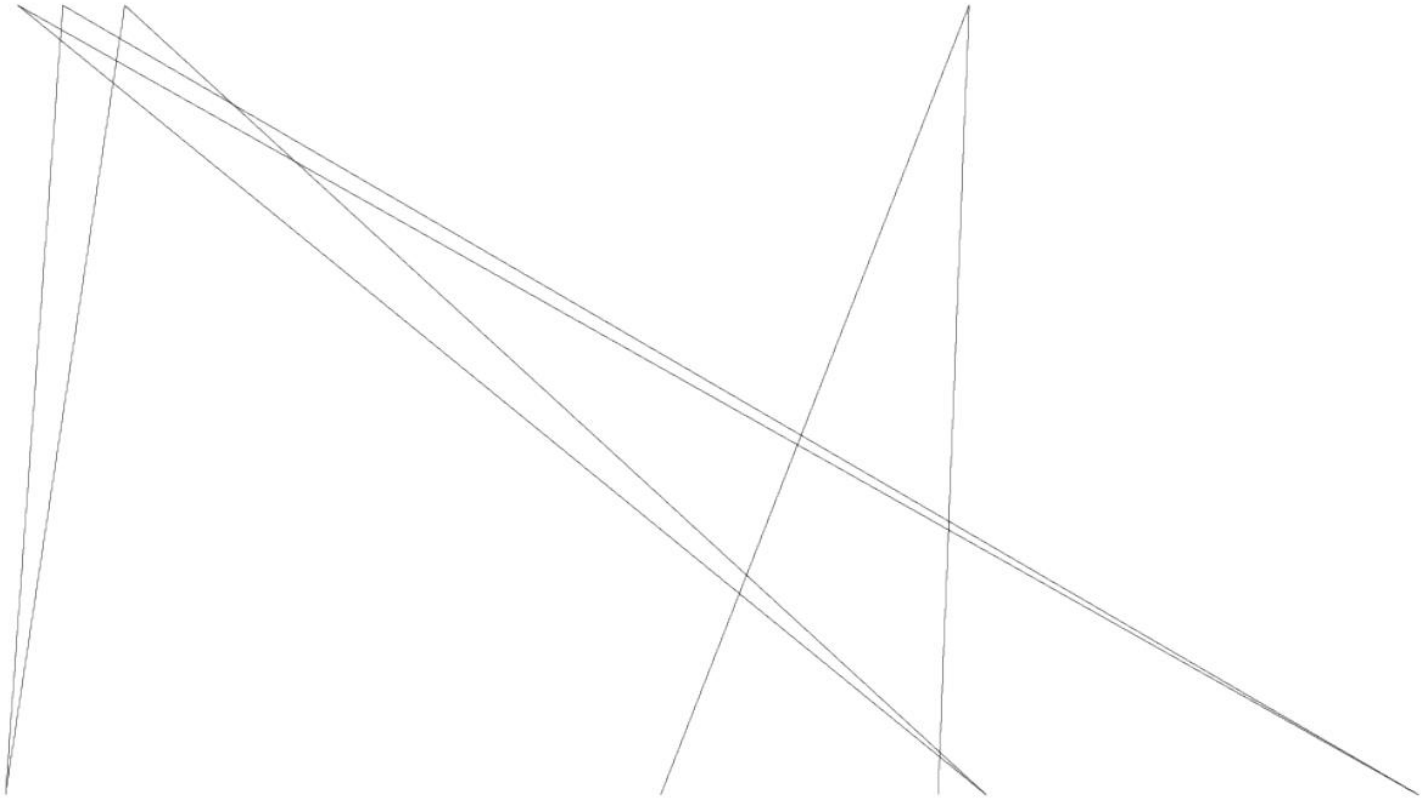
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



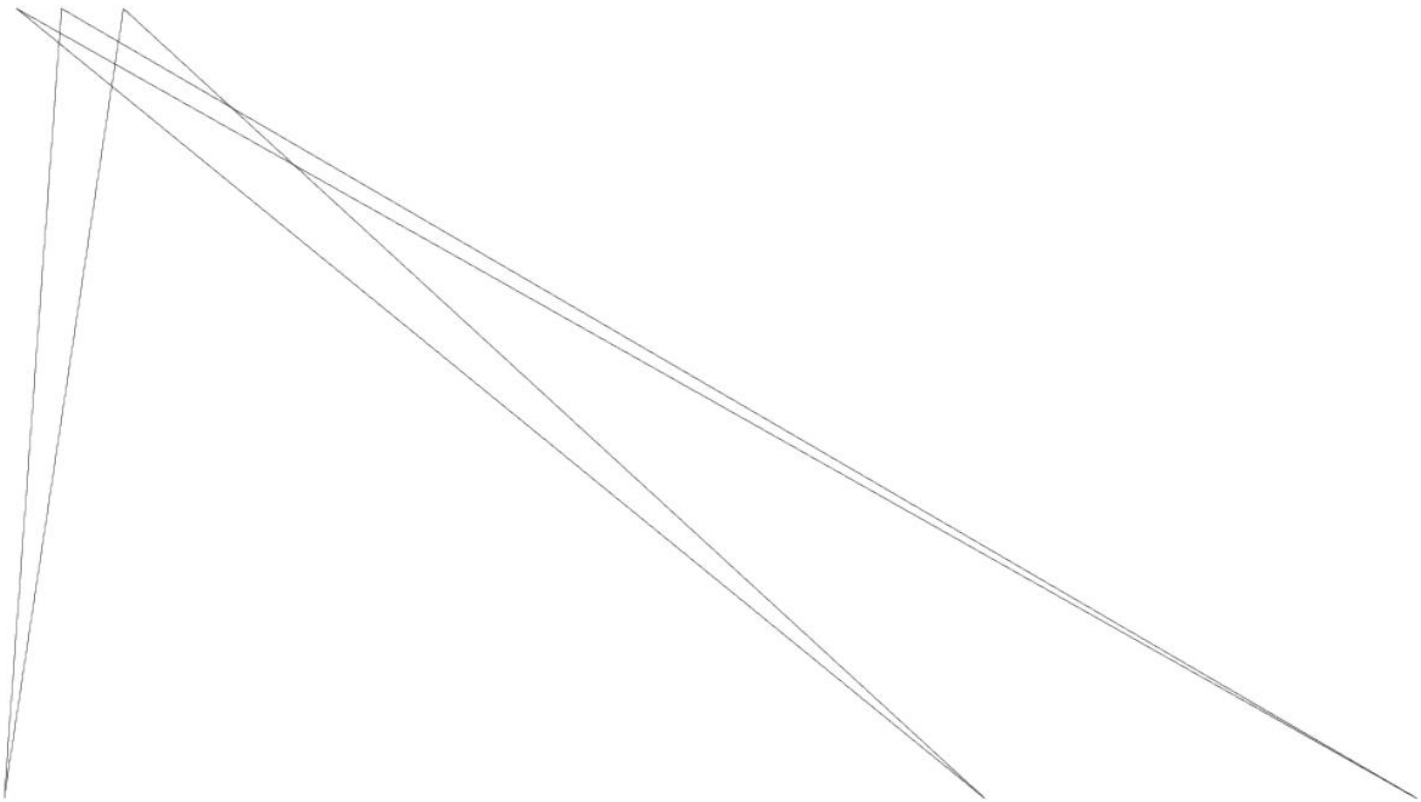
J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf

TRIMMING



J. Tromp. „Grin Proof of Work“. [Online]. Available: https://github.com/mimblewimble/grin-pm/blob/master/presentations/grincon0/07%20-%20tromp%20-%20Cuck(at)oo%20Cycle%20POW.pdf