A-SIT Technology
E-Government Innovationszentrum

30. Dezember 2019 Studie · Blockchain · Web & Cloud · 10 Min. Lesezeit

Implementierung einer Proof-of-Work-basierten editierbaren Blockchain

Archiviert. Dieser Inhalt wird nicht mehr aktualisiert. Er bleibt zur Information verfügbar.

Stand 2019-12-09
Quelle https://technology2.a-sit.at/veroeffentlichungen/implementierung-einer-proof-of-work-basierten-editierbaren-blockchain/

In diesem Projekt wurde ein Prototyp einer editierbaren Blockchain in Java und Kotlin implementiert. Die Blockchain erlaubt die Korrektur von Blöcken basierend auf einer Mehrheitsentscheidung. Die Implementierung wurde simpel gehalten, enthält aber alle wesentlichen Teile, wie beispielsweise ein P2P-Netzwerk und einen dezentralen Abstimmungsmechanismus. Zusätzlich wurde eine graphische Visualisierungs-lösung basierend auf der DOT-Sprache entworfen.

1. Einleitung

In den letzten Jahren ist das Interesse an Blockchain-basierten Anwendungen enorm gestiegen. Zum Teil dürfte der Hype um Kryptowährungen, wie Bitcoin, dazu beigetragen haben. Verteilte Blockchains wie die Bitcoin-Blockchain erlauben die integritätsgeschützte, dezentrale Ablage von Daten. Da die Mehrheit entscheidet, welche Daten als gültig angesehen werden und in die Blockchain aufgenommen werden, kann auf zentrale vertrauenswürdige Instanzen verzichtet werden. Bitcoin verwendet einen Proof-of-Work Algorithmus, welcher definiert, wie ein gültiger Block aussieht. Alle ehrlichen Teilnehmer halten sich an den Algorithmus, bzw. dessen Regeln. Bei Proofof-Work wird der Ansatz „eine CPU eine Stimme" angewandt. D.h. die Mehrheit im Sinne von Rechenleistung entscheidet, welche (regelkonformen) Daten aufgenommen werden.

Für Details zu den Regeln wird auf das Bitcoin Whitepaper [1] verwiesen, in diesem Dokument werden nur die für dieses Projekt wesentlichen Details vorgestellt. Eine wesentliche Regel besagt, dass alle Blöcke, außer dem Genesis-Block [1] einen Link (eine Referenz) auf ihren Vorgängerblock enthalten müssen. Dadurch ergibt sich eine eindeutige Kette aus Blöcken. Abbildung 1 zeigt einen Ausschnitt einer Blockchain.

Abbildung 1: Eine Blockchain bestehend aus mehreren Blöcken (v. l. n r.). Jeder Block enthält unter anderem eine Referenz auf dessen Vorgängerblock. Eine wesentliche Eigenschaft, die Unveränderbarkeit gespeicherter Daten, ergibt sich durch die Regel, dass die längste Kette [2] aus regelkonformen Blöcken als die derzeit gültige Blockchain angesehen wird. Dadurch bräuchte ein Angreifer mehr Rechenleistung als alle

ehrlichen Nodes zusammen, um nachträglich einen Block zu ändern, da der Angreifer auch alle darauffolgenden Blöcke neuberechnen muss. Diese Unveränderbarkeit ist ein wesentliches Sicherheitsfeature, da es beispielsweise sicherstellt, dass niemand nachträglich eine Transaktion rückgängig machen kann. Zugleich bringt die Unveränderbarkeit jedoch auch Nachteile mit sich. So kann beispielsweise das „Recht auf Vergessenwerden“ nicht umgesetzt werden. Auch können fehlerhafte oder illegale Daten nicht korrigiert bzw. entfernt werden. Im nächsten Abschnitt wird eine editierbare Blockchain vorgestellt, welche diese Probleme löst, ohne die Sicherheit der Blockchain zu verringern.

2. Editierbare Blockchain

Eine wesentliche Eigenschaft von Bitcoin ist, dass die Kryptowährung ohne vertrauenswürdige oder zentrale Instanzen auskommt. Ziel dieses Projektes war es, diese wesentlichen Eigenschaften nicht zu verändern und eine editierbare dezentrale öffentliche Blockchain zu entwickeln, bei der die Mehrheit entscheidet, ob eine Änderung durchgeführt werden soll oder nicht. In Abschnitt 2.1 wird die Funktionsweise der editierbaren Blockchain erklärt, in Abschnitt 2.2 wird der entworfene Abstimmungsmechanismus vorgestellt, in Abschnitt 2.3 wird die Implementierung beschrieben und in Abschnitt 2.4 wird eine automatische Visualisierungslösung basierend auf der Beschreibungssprache DOT präsentiert. Für eine detailliertere und technische Beschreibung wird auf die dazugehörige Publikation verwiesen [2].

2.1. Funktionsweise

Die Idee hinter diesem Ansatz ist, eine zweite Blockchain zu verwenden, welche Korrekturen in der ersten Kette bestätigt bzw. legitimiert. Diese zweite Blockchain startet vom selben Genesis Block wie die herkömmliche Standard Blockchain. In diesem Bericht wird die zweite Blockchain 1 Der erste Block oder Startblock einer Blockchain dient als Vertrauensanker und wir Genesis-Block genannt. 2 Die Kette, in die insgesamt die meiste Arbeit investiert wurde.

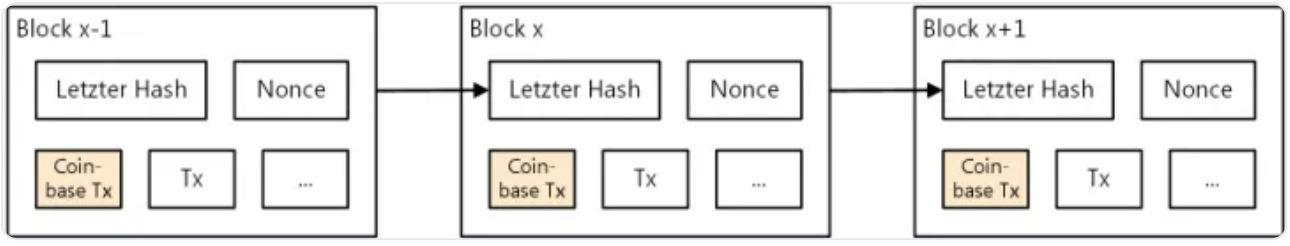


Abbildung 2: Herkömmliche Standard Blockchain ohne Änderungen

Korrekturkette genannt. Abbildung 2 zeigt eine herkömmliche Blockchain ohne Korrekturen. Abbildung 3 zeigt dieselbe Blockchain nach einer Korrektur. Dazu wurde ein Korrekturblock zur zweiten Kette hinzugefügt. Der Korrekturblock enthält Referenzen zu dessen Vorgängerblock, zum Block der bearbeitet werden soll, zu dessen Vorgänger und Nachfolger, sowie eine nicht visualisierte Referenz zum derzeit neuesten Block in der ersten Kette.

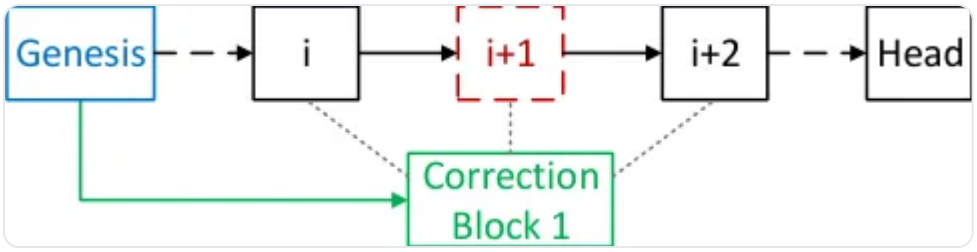


Abbildung 3: Editierbare Blockchain mit einem Korrekturblock, welcher Änderungen am Block i+1 legitimiert.

j	Col2
j	

j+1	Col2
j+1	

j+2	Col2
j+2	

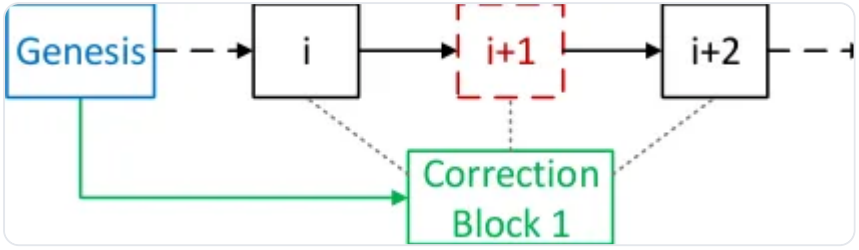


Abbildung 4: Nach der Korrektur werden Blöcke wieder an die Standard Blockchain angehängt.

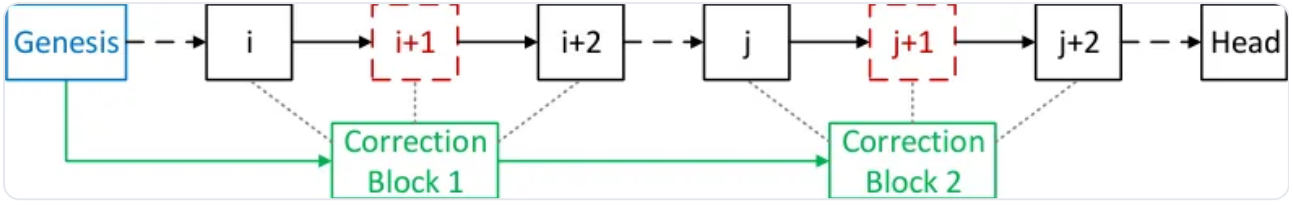


Abbildung 5: Editierbare Blockchain mit zwei korrigierten Blöcken Abbildung 4 zeigt die editierbare Blockchain, nachdem ein paar weitere Blöcke hinzugefügt wurden. Abbildung 5 zeigt die Kette, nachdem ein weiterer Block (j+1) korrigiert wurde. Im Zuge dessen wurde ein weiterer Korrekturblock an die Korrekturkette angehängt. Zugunsten besserer Verständlichkeit wurde der Abstimmungsvorgang nicht visualisiert. Die Idee und die Funktionsweise der Abstimmung werden im nächsten Abschnitt beschrieben.

2.2. Änderungsvorschlag und Abstimmung

Jeder Teilnehmer kann eine Korrektur vorschlagen, indem er oder sie eine Abstimmungstransaktion erstellt und ans Netzwerk schickt. Die Abstimmungstransaktion enthält alle relevanten Daten, beispielsweise welcher Block verändert werden soll und wie er danach aussehen würde. Aus diesen Informationen kann der neue korrigierte Block erstellt werden. Alle Miner überprüfen die Abstimmungstransaktion auf

Regelkonformität und stellen somit sicher, dass nur unterstützte Korrekturen durchgeführt werden. Falls die Transaktion den Regeln entspricht, wird sie in einen Block aufgenommen. Anschließend beginnt eine Abstimmungsphase, in welcher jeder Miner innerhalb des Abstimmungszeitraumes pro erstelltem (gültigen) Block eine Stimme bekommt. Durch diesen Ansatz wird sichergestellt, dass sich die Vertrauensannahmen nicht ändern. Jeder Miner bekommt langfristig gesehen Stimmanteile proportional zur investierten Rechenleistung. Nach dem Ende der Abstimmung werden die abgegebenen Stimmen ausgewertet, um zu sehen, ob die Mehrheit für die Korrektur oder dagegen gestimmt hat. Falls die Mehrheit dafür gestimmt hat, wird als nächstes ein Korrekturblock erstellt, welcher die Korrekturkette verlängert, ansonsten wird ein Standardblock, erstellt, welcher die erste Kette erweitert.

2.3. Implementierung

Der Prototyp einer editierbaren Blockchain wurde in Java und Kotlin umgesetzt. Der Quellcode kann von [3] bezogen werden. Um einen möglichst realistischen Anwendungsfall zu simulieren, wurde ein P2P-Netzwerk [4] integriert. Der Demonstrator wurde gezielt erstellt, um das Konzept zu demonstrieren und wurde möglichst einfach gehalten. Es wurde beispielsweise keine dynamische Anpassung des Schwierigkeitsgrades an die verfügbare Rechenleistung implementiert. Da es sich um einen Demonstrator handelt, sollte die Implementierung nicht für produktive Anwendungen eingesetzt werden. Der Unit-Test demonstriert die Verwendung des Prototyps. Zuerst werden mehrere Blockchain-Knoten erstellt, welche sich zu einem definierten Bootstrap-Knoten verbinden. Nach dem Netzwerk-Bootstrap-Prozess beginnen die Knoten, Blöcke zu minen. Die Blockchain-Knoten erstellen zu zufälligen Zeitpunkten Abstimmungstransaktionen, welche alle Details für die Korrektur eines Blockes enthalten. Nachdem die Abstimmungstransaktionen in einen Block aufgenommen wurden, stimmen die Knoten zufällig für oder gegen die Korrektur. Nach einer definierten Zeitspanne werden alle Miner gestoppt und kurz darauf überprüft der Unit-Test, ob alle Knoten dieselbe lokale Blockchain für derzeit gültig halten. Während des Tests werden sowohl textuelle als auch graphische Visualisierungen erstellt. Abbildung 6 zeigt eine textuelle Repräsentation einer Blockchain. Im nächsten Abschnitt werden die Funktionsweise der graphischen Visualisierung erklärt und ein paar Visualisierungen gezeigt.

0 SB [hash=54…62, previousHash=00…00, lastRedactionBlockHash=00…00, merkleRoot=47…2d, transactions=[PTX [txId=47…2d, outputs=[TxOutput [recip=f7…46, value=100]]]], nonce=2534] 1 VB [hash=00…f3, previousHash=54…62, lastRedactionBlockHash=00…88, merkleRoot=null, transactions=[], nonce=0] 2 SB [hash=00…49, previousHash=00…f3, lastRedactionBlockHash=54…62, merkleRoot=85…74, transactions=[PTX [txId=85…74, outputs=[TxOutput [recip=48…b0, value=100]]]], nonce=4889] 3 VB [hash=00…69, previousHash=00…49, lastRedactionBlockHash=00…6c, merkleRoot=null, transactions=[], nonce=0] 4 SB [hash=00…e1, previousHash=00…69, lastRedactionBlockHash=54…62, merkleRoot=d0…d5, transactions=[PTX [txId=e1…17, outputs=[TxOutput [recip=48…b0, value=100]]], ElectionTransaction{voteQuestion=VoteQuestion{hashOfBlockToReplace='00…f3', hashOfPreviousBlock='54…62', hashOfSubsequentBlock='00…49', txs=[PTX [txId=bd…47, outputs=[TxOutput [recip=f7…46, value=100]]]]}}], nonce=7809] 5 SB [hash=00…f1, previousHash=00…e1, lastRedactionBlockHash=54…62, merkleRoot=9d…d9, transactions=[PTX [txId=fe…10, outputs=[TxOutput [recip=48…b0, value=100]]], VotingTX [election=79…45, vote=true]], nonce=1894] 6 SB [hash=00…47, previousHash=00…f1, lastRedactionBlockHash=54…62, merkleRoot=a80…71, transactions=[PTX [txId=b9…b6, outputs=[TxOutput [recip=48…b0, value=100]]], VotingTX [election=79…45, vote=true]], nonce=278] 7 SB [hash=00…d7, previousHash=00…47, lastRedactionBlockHash=54…62, merkleRoot=d8…8d, transactions=[PTX [txId=6b…26, outputs=[TxOutput [recip=48…b0, value=100]]], VotingTX [election=79…

45, vote=true]], nonce=10487] 8 SB [hash=00…ec, previousHash=00…d7, lastRedactionBlockHash=00…88, merkleRoot=c4…0d, transactions=[PTX [txId=c4…0d, outputs=[TxOutput [recip=48…b0, value=100]]]], nonce=8786] 9 SB [hash=00…57, previousHash=00…ec, lastRedactionBlockHash=00…88, merkleRoot=70…ff, transactions=[PTX [txId=70…ff, outputs=[TxOutput [recip=48…b0, value=100]]]], nonce=1790] 10 SB [hash=00…22, previousHash=00…57, lastRedactionBlockHash=00…88, merkleRoot=be…35, transactions=[PTX [txId=be…35, outputs=[TxOutput [recip=48…b0, value=100]]]], nonce=1037] 11 SB [hash=00…57, previousHash=00…22, lastRedactionBlockHash=00…88, merkleRoot=0f…19, transactions=[PTX [txId=81…35, outputs=[TxOutput [recip=48…b0, value=100]]], ElectionTransaction{voteQuestion=VoteQuestion{hashOfBlockToReplace='00…69', hashOfPreviousBlock='00…49', hashOfSubsequentBlock='00…e1', txs=[PTX [txId=b2…25, outputs=[TxOutput [recip=48…b0, value=100]]]]}}], nonce=2929] 12 SB [hash=00…6f, previousHash=00…57, lastRedactionBlockHash=00…88, merkleRoot=ab…20, transactions=[PTX [txId=77…e1, outputs=[TxOutput [recip=48…b0, value=100]]], VotingTX [election=f3…78, vote=true]], nonce=4680] 13 SB [hash=00…e0, previousHash=00…6f, lastRedactionBlockHash=00…88, merkleRoot=e7…f6, transactions=[PTX [txId=d9…fb, outputs=[TxOutput [recip=48…b0, value=100]]], VotingTX [election=f3…78, vote=true]], nonce=781] 14 SB [hash=00…b0, previousHash=00…e0, lastRedactionBlockHash=00…88, merkleRoot=46…63, transactions=[PTX [txId=3f…14, outputs=[TxOutput [recip=48…b0, value=100]]], VotingTX [election=f3…78, vote=true]], nonce=13046] *Abbildung 6: Textuelle Visualisierung der ersten Kette. Standardblöcke wurden mit „SB“, korrigierte Blöcke mit „VB“ und Zahlungstransaktionen mit „PTX“ abgekürzt.*

2.4. Visualisierung

Aufgrund der grundlegenden Änderungen an der Blockchain-Struktur konnte keine der bestehenden Visualisierungslösungen eingesetzt werden. Da es sich bei Blockchains um gerichtete Graphen handelt, wurde die Beschreibungssprache DOT [3] als Basis ausgewählt. Mittels DOT kann die Struktur eines gerichteten oder ungerichteten Graphen beschrieben werden. Zusätzlich erlaubt DOT 3 http://www.graphviz.org/doc/info/lang.html die Form und Farbe von Knoten und Kanten zu definieren. Abbildung 7 zeigt die textuelle Beschreibung eines einfachen Graphen sowie die gerenderte Version. Als Rendering-Engine wurde „DOT“ verwendet.

|digraph G { compound=true; concentrate=true; graph [fontsize=10 fontname="Verdana"]; node [shape=record fontsize=10 fontname="Verdana"]; H000a4215895b3c0613d9aa0f8adcbbe4004d1573ec98951c915dab6e2799ea 85 [label="{<hash> 000a4…9ea85 |{<prev> 54a3a…98562|…|<last2> 54a3a…98562}}",color=green]; }|Col2| |—|—| | digraph G { compound=true; concentrate=true; graph [fontsize=10 fontname="Verdana"]; node [shape=record fontsize=10 fontname="Verdana"]; H00079cc303279a6a30601a7df3ee71871987355c577f2e7c36988031b8dc320 8 [label="{<hash> 00079…c3208|{<prev2> 54a3a…98562|<other> …|<last1> 00010…bf167}| {<prev1> 00076…f1326 | <subs1> 000a5…36178}}",color=red]; }| | |digraph G { compound=true; concentrate=true; graph [fontsize=10 fontname="Verdana"]; node [shape=record fontsize=10 fontname="Verdana"]; H000709e19402fd31ec2c80538ad4ea7b545e0f2ceeed684a002a9130d27dd59 8 [label="{<hash> 00070…dd598 |{<prev> 0000a…99e98|…|<last2> 00011…66262}}",color=orange]; }| |

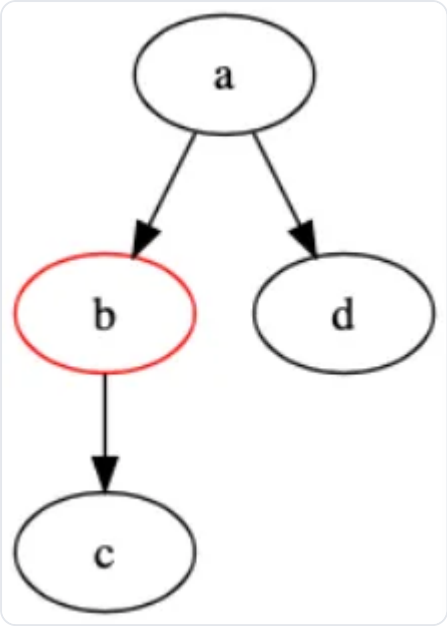
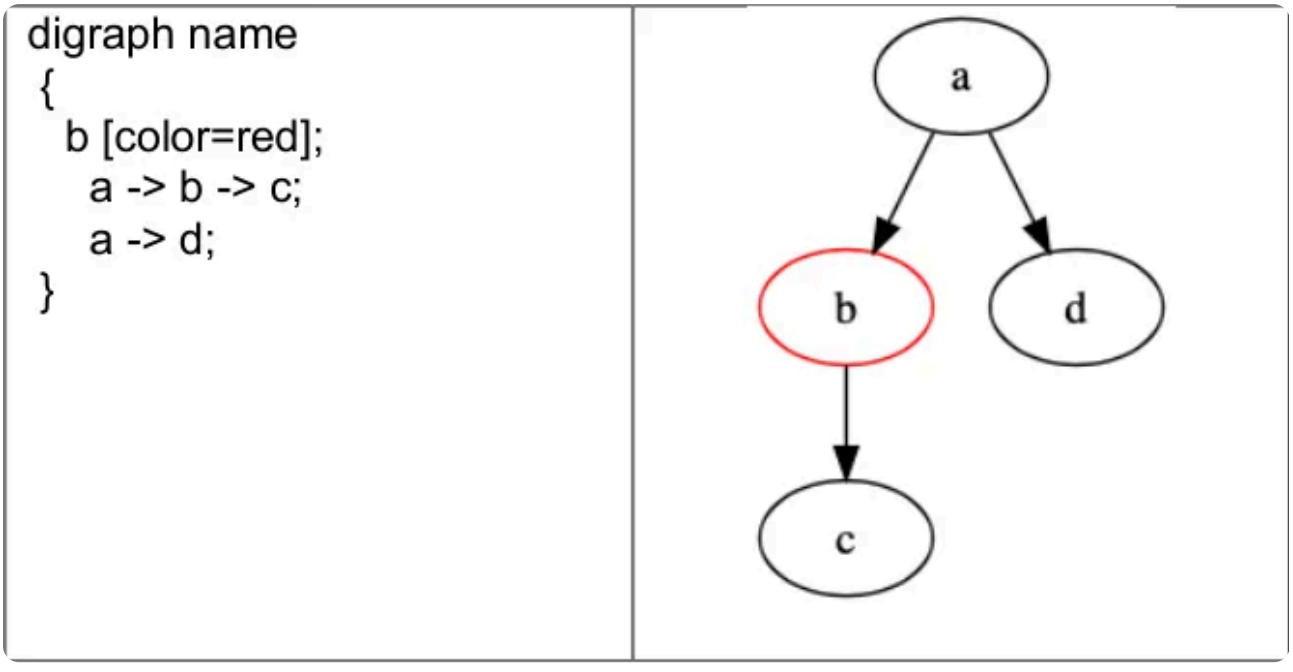


Abbildung 7: Links ist die textuelle Beschreibung eines Graphens zu sehen und rechts die gerenderte Version. Im Zuge dieses Projektes wurde ein Konverter geschrieben, welcher beliebige editierbare Blockchains in die DOT Sprache konvertiert. Abbildung 8 zeigt die Visualisierung von herkömmlichen Blöcken, Korrekturblöcken und korrigierten Blöcken. Herkömmliche Blöcke werden zur besseren Erkennbarkeit in grün dargestellt. Sie bestehen aus zwei Zeilen, die obere Zeile zeigt den Hashwert des Blocks, die untere Zeile enthält zwei Referenzen, links den Hashwert des Vorgängerblockes und rechts den Hashwert des letzten Korrekturblockes. Korrekturblöcke werden in rot und dreizeilig visualisiert. Die erste Zeile zeigt ausschließlich den Hashwert des Blocks, während die zweite und dritte Zeile jeweils zwei Werte enthalten. Die zweite Zeile enthält links eine Referenz auf den letzten Korrekturblock und rechts auf den letzten Block in der ersten Kette. Die dritte Zeile zeigt links auf den Vorgängerblock und rechts auf den Nachfolgeblock des zu korrigierenden Blocks. Korrigierte Blöcke werden in orange visualisiert und bestehen aus zwei Zeilen: Die obere Zeile zeigt den Hashwert des Blockes vor der Korrektur. In der zweiten Zeile wird im linken Feld der Hashwert des Vorgängerblockes angezeigt und im rechten Feld der Hashwert des zur Korrektur gehörenden Korrekturblocks.



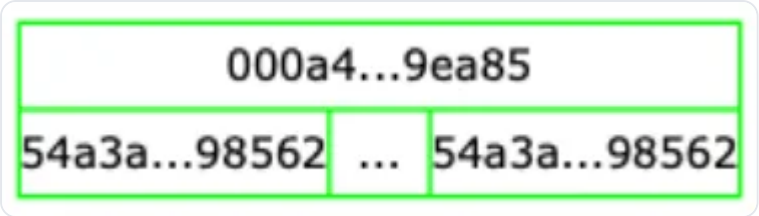


Abbildung 8: Visualisierung von herkömmlichen Blöcken (oben), Korrekturblöcken (mittig) und korrigierten Blöcken (unten). In der linken Spalte ist jeweils der DOT-Quellcode zu sehen, in der rechten Spalte die konvertierte Visualisierung. Abbildung 9 zeigt den DOT-Quellcode und die Visualisierung für eine simple Blockchain. Der schwarze Block ist der Genesis-Block.

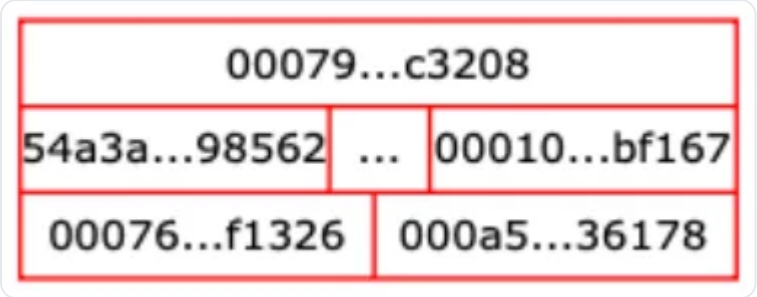
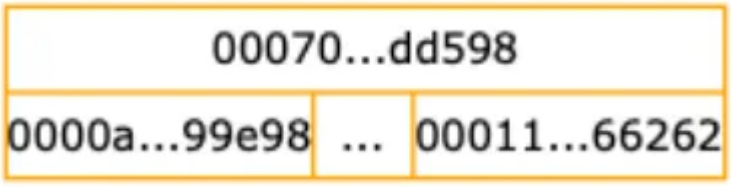
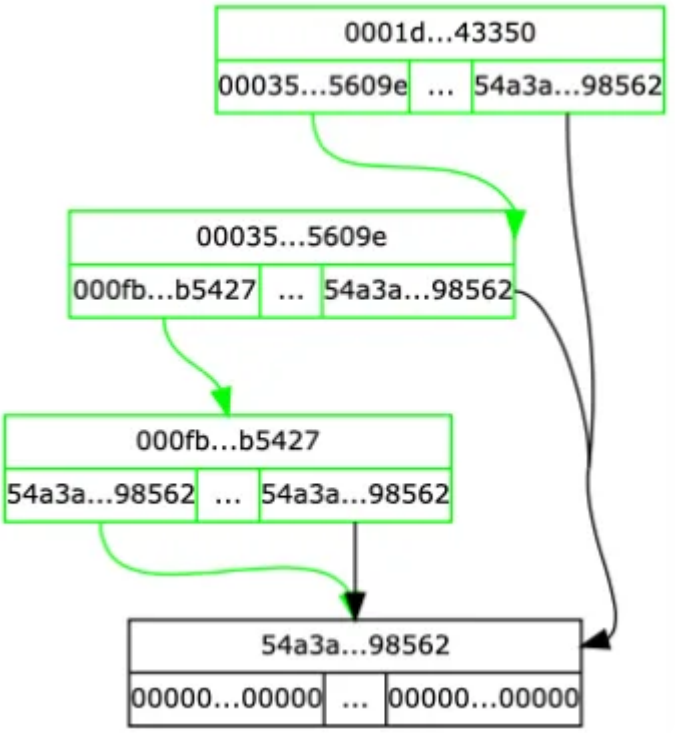
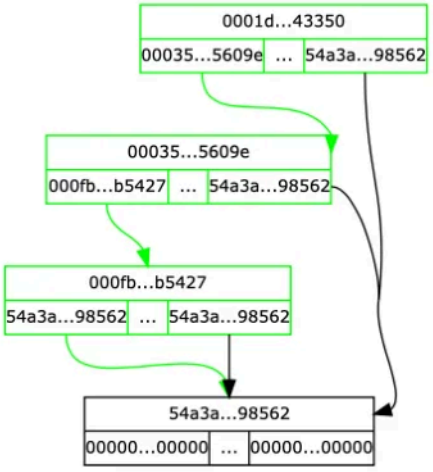


Abbildung 9: DOT-Quellcode und Visualisierung einer simplen Blockchain. Abbildung 10 und Abbildung 11 zeigen den Verlauf einer Blockchain. Abbildung 10 (links) zeigt eine Blockchain vor der Korrektur des zweiten Blockes. In der rechten Abbildung ist dieselbe Blockchain nach dem Hinzufügen eines Korrekturblockes zu sehen. Die grünen Pfeile zeigen jeweils auf den direkten Vorgängerblock, die schwarzen Pfeile zeigen auf den letzten Korrekturblock, bzw. auf den Genesis-Block, sofern noch keine Korrektur durchgeführt wurde. Der pinke Pfeil zeigt den zum Korrekturblock gehörenden korrigierten Block. Der violette Pfeil zeigt den letzten Block in der ersten Kette zum Zeitpunkt der Erstellung des Korrekturblockes. Die orangen Pfeile zeigen auf den Vorgängerblock des zu korrigierenden Blocks. Der blaue Pfeil zeigt auf den Nachfolgeblock des zu korrigierenden Blocks.






```
digraph G {
compound=true;concentrate=true;
    graph [fontsize=10 fontname="Verdana"];
    node [shape=record fontsize=10 fontname="Verdana"];
//Genesis Block
H54a3aaeb515cfb8242fd166faed8a218f0980adeb60ab0490640eba351398562
[label="{<hash> 54a3a...98562 |{<prev> 00000...00000|...|<last2> 00000...00000}}"];
//Block 1
H000fbf92d2bb9ebc0ad16a0275fd1f937a8eba03c23a09aa67e9b1b6699b5427
[label="{<hash> 000fb...b5427 |{<prev> 54a3a...98562|...|<last2>
54a3a...98562}}",color=green];
H000fbf92d2bb9ebc0ad16a0275fd1f937a8eba03c23a09aa67e9b1b6699b5427:prev
->
H54a3aaeb515cfb8242fd166faed8a218f0980adeb60ab0490640eba351398562:has
h [color=green];
H000fbf92d2bb9ebc0ad16a0275fd1f937a8eba03c23a09aa67e9b1b6699b5427:last2
->
H54a3aaeb515cfb8242fd166faed8a218f0980adeb60ab0490640eba351398562:has
h;
//Block 2
H00035e68423ed43c4adb2d16aae024a38c7fed9d2b396393d8a5425b8675609e
[label="{<hash> 00035...5609e |{<prev> 000fb...b5427|...|<last2>
54a3a...98562}}",color=green];
H00035e68423ed43c4adb2d16aae024a38c7fed9d2b396393d8a5425b8675609e:pr
ev ->
H000fbf92d2bb9ebc0ad16a0275fd1f937a8eba03c23a09aa67e9b1b6699b5427:hash
[color=green];
H00035e68423ed43c4adb2d16aae024a38c7fed9d2b396393d8a5425b8675609e:las
t2 ->
H54a3aaeb515cfb8242fd166faed8a218f0980adeb60ab0490640eba351398562:has
h;
//Block 3
H0001d9f42c1fa5ba2f943b7c30323514905775a207ed0ceaefd9bcd09c243350
[label="{<hash> 0001d...43350 |{<prev> 00035...5609e|...|<last2>
54a3a...98562}}",color=green];
H0001d9f42c1fa5ba2f943b7c30323514905775a207ed0ceaefd9bcd09c243350:prev
->
H00035e68423ed43c4adb2d16aae024a38c7fed9d2b396393d8a5425b8675609e:ha
sh [color=green];
H0001d9f42c1fa5ba2f943b7c30323514905775a207ed0ceaefd9bcd09c243350:last2
->
H54a3aaeb515cfb8242fd166faed8a218f0980adeb60ab0490640eba351398562:has
h;
}
```



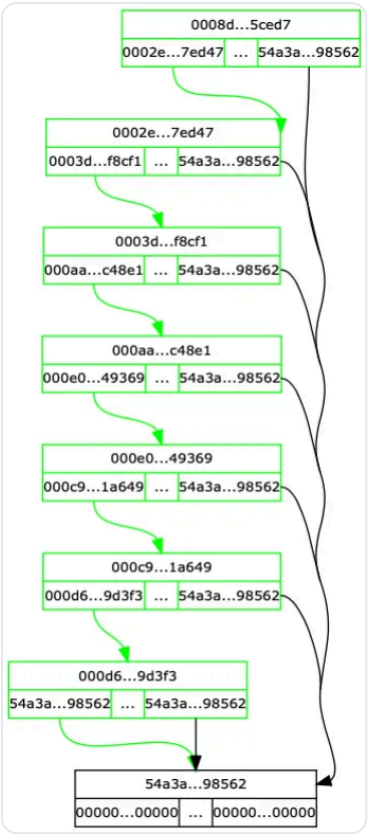
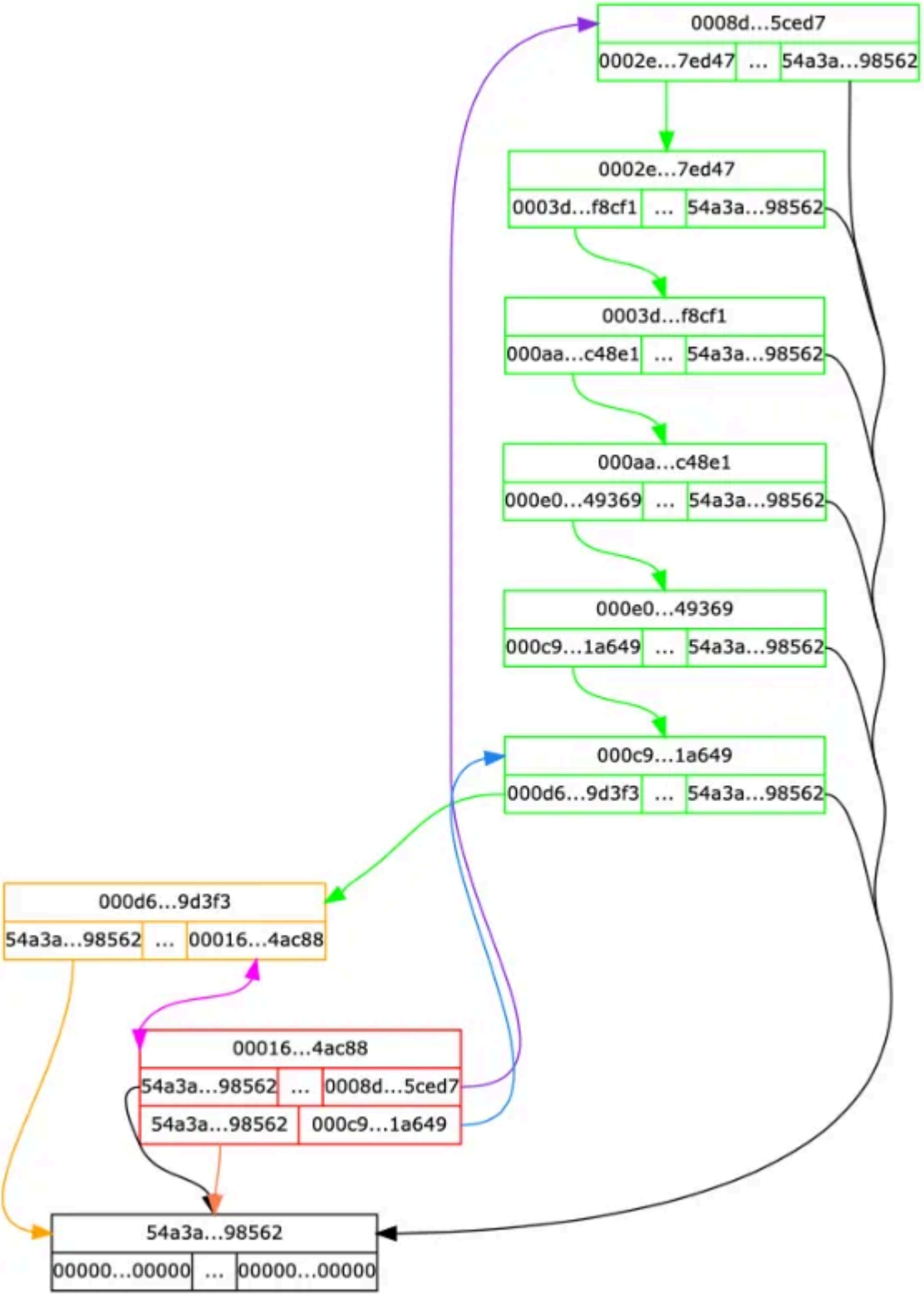
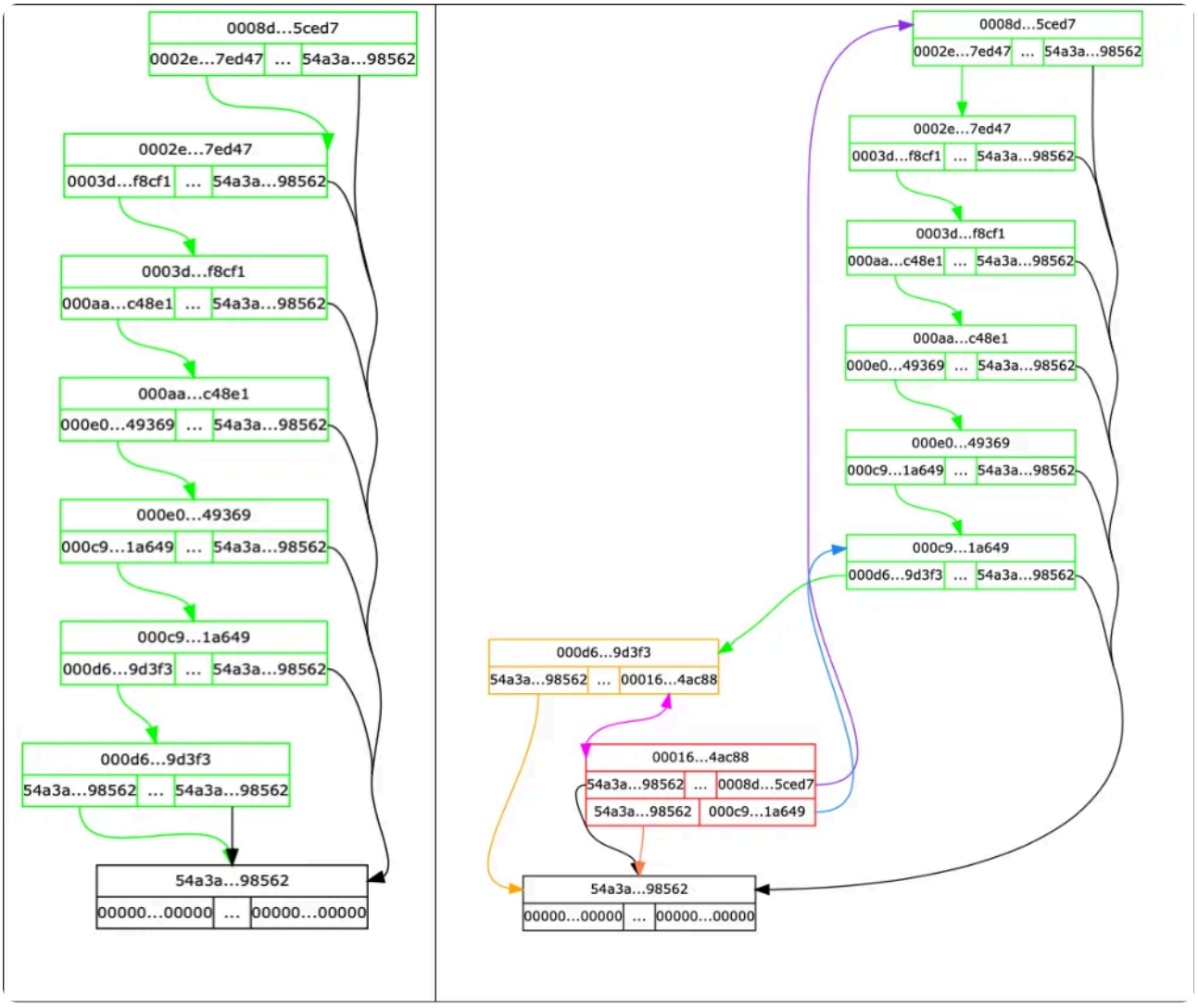


Abbildung 10: Links ist die Blockchain vor der Korrektur des ersten Blocks zu sehen, rechts nach der Korrektur. Der Korrekturblock ist in Rot gehalten und der korrigierte Block in orange. Abbildung 11 zeigt links dieselbe Blockchain nachdem ein weiterer Block hinzugefügt wurde und rechts die Blockchain, nachdem eine weitere Korrektur durchgeführt wurde. Die Abstimmungsprozesse wurden zugunsten einer besseren Verständlichkeit nicht visualisiert.


0008d...5ced7
0002e...7ed47 ... 54a3a...98562
0002e...7ed47
0003d...f8cf1 ... 54a3a...98562
0003d...f8cf1
000aa...c48e1 ... 54a3a...98562
000aa...c48e1
000e0...49369 ... 54a3a...98562
000e0...49369
000c9...1a649 ... 54a3a...98562
000c9...1a649
000d6...9d3f3 ... 54a3a...98562
000d6...9d3f3
54a3a...98562 ... 00016...4ac88
00016...4ac88
54a3a...98562 ... 0008d...5ced7
54a3a...98562 000c9...1a649
54a3a...98562
00000...00000 ... 00000...00000



Die Abbildungen zeigen, dass die automatische Platzierung der Blöcke für einfache Blockchains gut funktioniert, jedoch komplexere Blockchains mit Korrekturen nicht optimal dargestellt werden. Leider wird die manuelle Spezifikation von Koordinaten bei DOT nicht unterstützt. Es wurden noch andere Rendering-Engines getestet, welche die manuelle Spezifizierung der Koordinaten erlauben, aber insgesamt lieferte die DOT-Engine die besten automatisch generierten Ergebnisse.

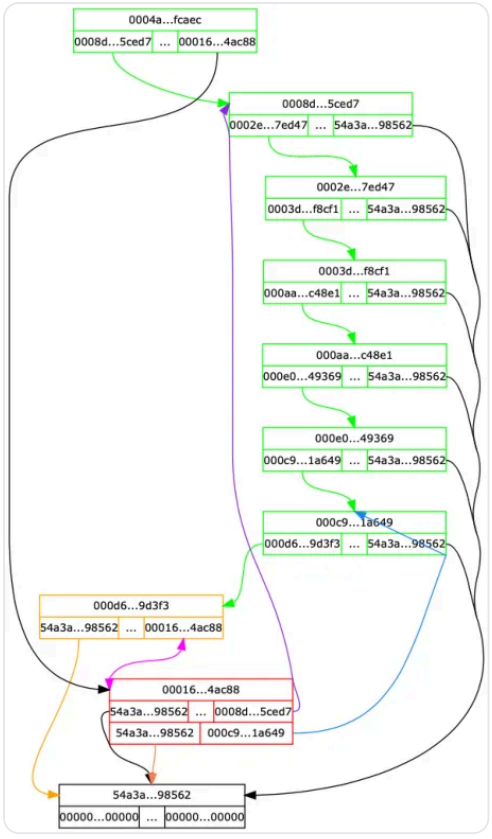
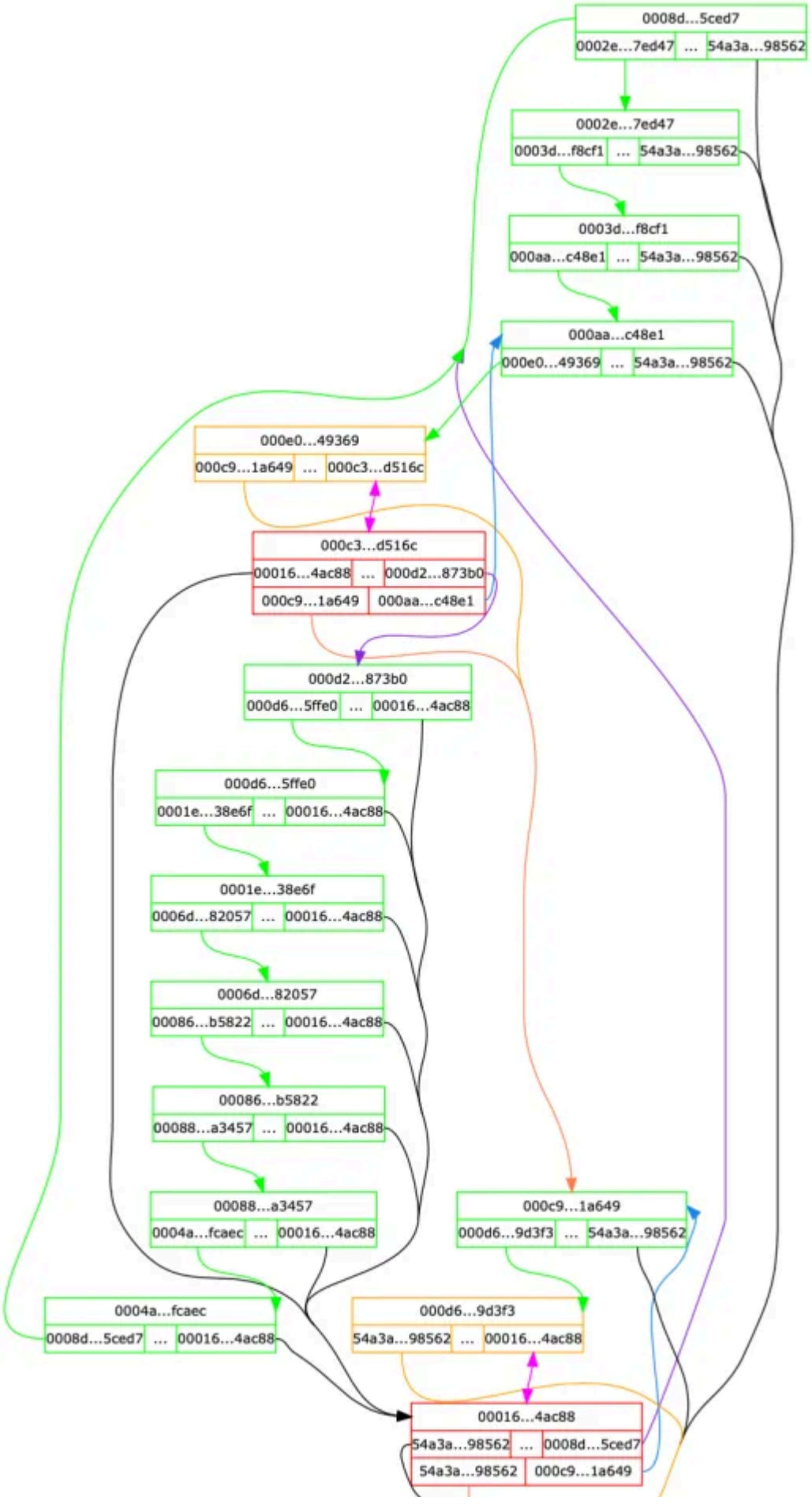
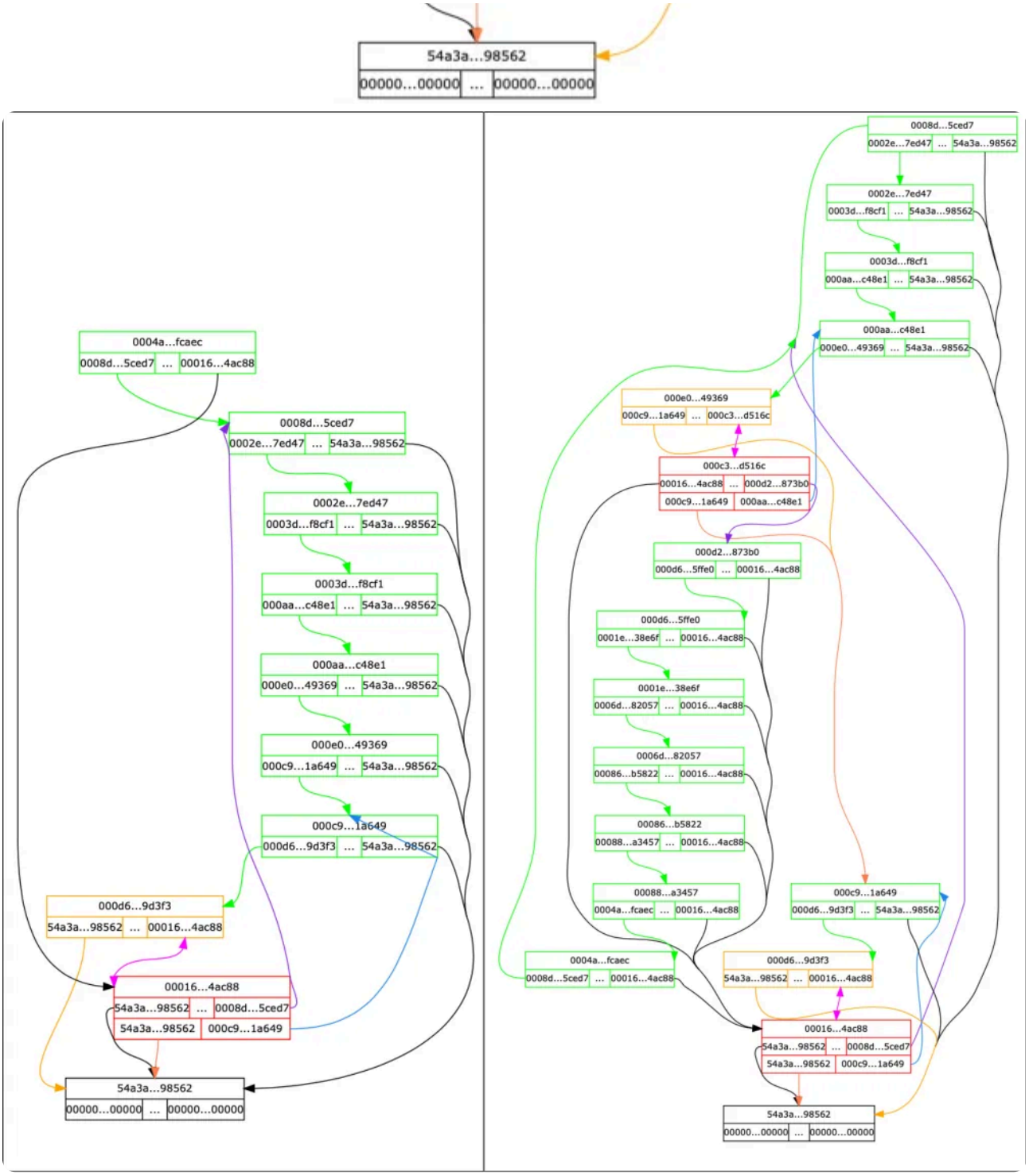


Abbildung 11: Diese Abbildung zeigt die Blockchain aus Abbildung 10 nach dem Hinzufügen eines weiteren herkömmlichen Blocks (links), bzw. nach ein paar weiteren Blöcken und einer zweiten Korrektur (rechts).


0008d...5ced7
0002e...7ed47 ... 54a3a...98562
0002e...7ed47
0003d...f8cf1 ... 54a3a...98562
0003d...f8cf1
000aa...c48e1 ... 54a3a...98562
000aa...c48e1
000e0...49369 ... 54a3a...98562
000e0...49369
000c9...1a649 ... 000c3...d516c
000c3...d516c
00016...4ac88 ... 000d2...873b0
000c9...1a649 000aa...c48e1
000d2...873b0
000d6...5ffe0 ... 00016...4ac88
000d6...5ffe0
0001e...38e6f ... 00016...4ac88
0001e...38e6f
0006d...82057 ... 00016...4ac88
0006d...82057
00086...b5822 ... 00016...4ac88
00086...b5822
00088...a3457 ... 00016...4ac88
00088...a3457
0004a...fcaec ... 00016...4ac88
000c9...1a649
000d6...9d3f3 ... 54a3a...98562
0004a...fcaec
0008d...5ced7 ... 00016...4ac88
000d6...9d3f3
54a3a...98562 ... 00016...4ac88
00016...4ac88
54a3a...98562 ... 0008d...5ced7
54a3a...98562 000c9...1a649



3. Fazit

In diesem Projekt wurden ein Prototyp einer editierbaren Blockchain implementiert, ein dezentraler Abstimmungsmechanismus entworfen und eine graphische Visualisierungslösung erstellt. Der Prototyp zeigt die Funktion des Konzeptes. Durch die Integration des P2P-Netzes ergibt sich ein realistischer Anwendungs- und Testfall. Der Abstimmungsmechanismus erlaubt eine dezentrale Abstimmung unter Berücksichtigung der verfügbaren Rechenleistung. Die Visualisierungslösung funktioniert zwar, aber die automatische Platzierung der Blöcke ist nicht optimal, wodurch die Übersichtlichkeit der Visualisierung verringert wird.

Das Konzept der editierbaren Blockchain wurde auf einer internationalen wissenschaftlichen Konferenz präsentiert und in Form eines Papers [2] unter dem Titel „A Correctable Public Blockchain“ publiziert. Der Quellcode des Demonstrators kann von [3] geladen werden.

Referenzen

[1] S. Nakamoto, „Bitcoin: A Peer-to-Peer Electronic Cash System,“ 2008. [Online]. Available:

https://bitcoin.org/bitcoin.pdf. [Zugriff am 02 03 2018].

[2] A. Marsalek und T. Zefferer, „A Correctable Public Blockchain,“ in *18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering*, Rotorua, 2019.

[3] A. Marsalek, „Quellcode Redactable Blockchain,“ [Online]. Available: https://technology.a sit.at/downloads/3734.

[4] B. Prünster, „s-kad,“ [Online]. Available: https://extgit.iaik.tugraz.at/bpruenster/s-kad. [Zugriff am 2019].