Dr. Egon Bellgardt
Data » Science » Consulting

Eine Einführung in die Technik der Blockchain und des Bitcoin

Version 0

von Egon Bellgardt

17. Februar 2021

Inhaltsverzeichnis

1. Einführung

1.1. Eine kurze allgemeine Definition

Eine Blockchain ist eine bestimmte Art, wie Daten in einer sehr großen Datei gespeichert werden. Die Speicherung erfolgt als **Kette von Datenblöcken** und zwar so, dass nicht autorisierte Versuche, die Daten zu verändern praktisch nicht möglich sind.

Einzelne **Einträge** (z.B. Transaktionen) werden hierzu gesammelt und nach Erreichen einer bestimmten Anzahl zu Blöcken zusammengefasst. Diese Blöcke werden chronologisch aneinander gekettet. Am Anfang dieser Blockchain stehen die frühesten Einträge, am Ende der Blockchain die neuesten Einträge.

Einmal an die Blockchain angefügte Blöcke und deren Inhalt können **nicht mehr geändert** oder gelöscht werden.

Dafür sorgt die Art und Weise der Verkettung der Datenblöcke. Diese erfolgt über **Signaturen** (technisch: eine Folge von Ziffern), die aus dem Inhalt (Ziffern und in Ziffern transformierte Buchstaben und Zeichen) des Blocks errechnet werden.

Dabei geht die Signatur des ersten Blocks in den **nachfolgenden Bock** ein. Die Vorgängersignatur eines Blocks wird also zum Inhalt des jeweiligen Blocks. Damit basiert die Berechnung der Signatur des zweiten Blocks auch auf der Signatur des ersten Blocks. Das setzt sich so über alle Blöcke fort.

Diese Verkettung stellt somit sicher, dass der **Änderungsversuch** eines Angreifers an einem einzigen Block immer auffallen würde. Denn: würde ein einzelner Block geändert, müssten auch alle folgenden Blöcke geändert werden indem neue, stimmige Signaturen berechnet werden. Nur wenn die Signaturen aller Blöcke stimmig sind, ist die Blockchain als Ganzes stimmig.

Technisch ist es so gestaltet, dass

- die **Überprüfung** der Stimmigkeit einer Signatur **extrem schnell** möglich ist,
- die **Berechnung** einer neuen Signatur aber **extrem aufwendig** ist.

Damit will man verhindern, dass Angreifer die gesamte Blockchain verändern können, um deren Stimmigkeit nach einer mißbräuchlichen Änderung wieder herzustellen.

Die gesamte Blockchain wird physisch nicht nur an einer Stelle, sondern **auf den Rechnern aller Teilnehmer** der Blockchain gespeichert. Jede Änderung der Blockchain gelangt damit auf die Rechner aller Blockchain-

Teilnehmer. Das bedeutet auch, dass die Blockchain **öffentlich** ist: grundsätzlich kann jeder Teilnehmer alle Einträge ansehen.

Da verfälschende Änderungen nicht möglich sind, sollte man daher den Inhalten der Blockchain (z.B. Geldguthaben, Besitzrechte oder Identitätsnachweise) grundsätzlich genauso vertrauen können, wie z.B. einer Bank, einem Grundbuchamt oder einem Einwohnermeldeamt.

1.2. Ein Bank-Beispiel als Motivation der Blockchain-Idee

Stellen wir uns zunächst eine vereinfachte Bank (Geldinstitut) vor. Diese speichert alle Transaktionen und sonstigen Tatbestände in einer Datei. Die Datei enthält rechtsverbindliche Tatbestände, auf die sich z.B. ein Bankkunde berufen kann.

Wenn Alexa 10 Euro besitzt und diese 10 Euro an Ben überweist, könnte die Datei der Bank z.B. folgende beiden Einträge enthalten:

```
1. Alexa besitzt 10 Euro

2. Alexa überweist an Ben 10 Euro
```

Die **Bank muss sicherstellen,** dass alle Einträge in dieser Datei stimmig sind und bleiben. Das gelingt dadurch, dass nur die Bank diese Datei verändern darf und auch die Verantwortung für diese Änderungen trägt. Jede Änderung an einer Datei lässt sich nachverfolgen: wann erfolgte die Änderung, was war der Anlass der Änderung, welche Grundlage gibt es zu dieser Änderung (z.B. ein unterschriebenes Überweisungsformular), welcher Bankmitarbeiter hat die Änderung vorgenommen. Dokumentiert wird dies durch die Buchführung der Bank. Die Bank garantiert die Validität der Dateieinträge.

Wir wollen nun die Bank als vertrauenswürdigen Vermittler wegfallen und **alle Bankkunden direkt miteinander interagieren** lassen. Die rechtsverbindlichen Tatsachen sollen aber weiter in einer einzigen Datei gespeichert bleiben. Jeder Teilnehmer (ehemaliger Bankkunde) muss dann natürlich Zugriff auf diese Datei haben, weil ja Transaktionen direkt von Teilnehmer zu Teilnehmer ausgeführt werden sollen und es keinen Vermittler mehr gibt.

Was würde passieren, wenn man das obige einfache Dateiformat beibehalten würde, aber allen Teilnehmern (früher: Bankkunden) freie Zugriffs- und Eingriffsrechte erlauben würde? Ben könnte z.B. den zweiten Eintrag der Datei ändern in

```
2. Alexa überweist an Ben 1000 Euro
```

Auch Alexa könnte den ersten Eintrag ändern in

```
1. Alexa besitzt 100000 Euro
```

Einer solchen Datei- und Zugriffs-Konstruktion könnten also die Teilnehmer - ohne eine Bank, die die Richtigkeit der Einträge garantiert - **kein Vertrauen** schenken, da jeder Teilnehmer unbeschränkten Zugriff hätte und die Einträge jederzeit einfach zu seinen Gunsten verändern könnte.

Vertrauen in die Richtigkeit der Einträge in dieser Datei kann man im Prinzip durch die **Signierung und Verkettung von deren Einträgen,** den Grundprinzipien der Blockchain-Idee, erreichen:

- Jeder Eintrag wird zunächst auf seine Richtigkeit und Zulässigkeit geprüft. Z.B. ist genügend Geldbestand vorhanden, um eine Überweisung auszuführen.
- Jeder Eintrag wird um eine Signatur ergänzt.
- Die Signatur jedes Eintrags wird auch Inhalt des jeweils nachfolgenden Eintrags. Dadurch einsteht eine Verkettung der Einträge.
- Das Erzeugen einer Signatur ist extrem aufwendig.
- Die Überprüfung einer Signatur auf Richtigkeit gelingt extrem schnell.
- Die Zahl der Einträge ist hinreichend groß.

Die Bedeutung dieser Anforderungen wird nachfolgend erläutert.

In einer Blockchain werden einzelne Einträge (Transaktionen) zu Blöcken zusammengefasst. Es werden so viele Transaktionen in einem Block gebündelt, bis der Block eine bestimmte Maximalgröße erreicht. Bei Bitcoin sind das etwa 1 MB und etwa 2000 Transaktionen.

2. Der Hashwert als digitale Signatur der Blöcke

In einer Blockchain entspricht die Signatur einem Hashwert (kurz auch: Hash) und wird mit Hilfe einer Hashfunktion ermittelt. Eine Hashfunktion errechnet aus beliebigen Einträgen S (z.B. Nummern, Texten, etc.) einen numerischen Hashwert H bestimmter Länge.

Der Hashwert eines Blocks wird aus dem Inhalt eines Blocks, also aus den Zeichen, die der Block enthält, ermittelt.

Eine sehr einfache Hashfunktion H(S) wäre z.B.

$$H(S) = S \text{ MOD } 10.$$

Der Hashwert wäre hierbei der verbleibende Rest einer Division von S durch 10. Ist S also beispielsweise 107, so ergibt sich der Hashwert von S als:

$$H(S) = H(107) = 107 \text{ MOD } 10 = 7.$$

Die Zahl S wird dabei aus allen Zeichen des zu hashenden Inhalts ermittelt. Im Falle von Texteinträgen werden z.B. alle Zeichen durch ihren ASCII oder ANSII-Code repräsentiert. Beispielsweise sind die Zahlen 65, 66 und 67 die ASCII-Codes der Buchstaben A, B und C.

In einer sehr einfachen Hashfunktion könnte man beispielsweise die Summe S dieser Zahlen bilden.[1] Das Wort ABC würde dann durch folgende Summe repräsentiert:

$$S = 65+66+67 = 198.$$

Als Hashwert resultiert im Beispiel der Rest der Division von S durch 10:

$$H(S) = H(198) = 198 \text{ MOD } 10 = 8.$$

In anderen Anwendungen werden Hashfunktionen auch zur Erstellung sogenannter Hashtabellen verwendet. Dabei werden für Datenobjekte, z.B. Einträge in Datenbanken, Hashwerte berechnet. Grundlage der Berechnung sind z.B. die Schlüssel der Datensätze, z.B. eine Kontonummer. Die Berechnung erfolgt so, dass die Hashwerte zu einer bestimmten Position in der Hashtabelle bzw. zu dem Speicherort des Datenobjekts auf dem Datenträger führen.

In der Praxis wird der Hashwert durch eine sogenannte **kryptografische Hashfunktion** berechnet. Es gibt verschiedene kryptografische Hashfunktionen, die sehr viel komplexer definiert sind als im obigen Beispiel. Bei Bitcoin wird die SHA-256-Hashfunktion[2] verwendet.

Kryptographische Hashfunktionen stellen - anders als in der einfachen Hashfunktion oben - sicher, dass zwei unterschiedliche Inhalte auch zwei unterschiedliche Hashwerte generieren und es daher keine **Kollisionen** (verschiedene Inhalte generieren gleiche Hashwerte) gibt. Hashfunktionen sind zudem **Einwegfunktionen,** d.h. aus einem Hashwert kann man nicht den ursprünglichen Inhalt rekonstruieren.

Die n Zeichen eines Blocks werden dabei in eine Zeichenkette der Länge m abgebildet. Ändert sich eines der n Zeichen im Block, ändert sich auch der Hashwert.

Besteht beispielsweise der erste Block einer Blockchain aus dem Eintrag

```
Alexa besitzt 10 Bitcoins.
```

wird dies mit dem SHA-256-Verfahren durch den Hashwert

```
2ed5e28e31be941120179ee996788d5eee9a54fcb212ae6c65eefd2a95adeb97
```

abgebildet.

---

[1] Dieser einfachen Hashfunktion fehlen bestimmte wünschenswerte Eigenschaften. Beispielsweise gibt es Kollisionen: die Wörter CBA und BAC führen zum gleichen Hashwert wie das Wort ABC.

[2] SHA = Secure Hash Algorithm, veröffentlicht im Jahr 2002.

Dieser Hashwert wurde auf

https://www.egon-bellgardt.de/hash_input.php

berechnet. Neben SHA-256 erlaubt dieser HASH GENERATOR auch die Anwendung zahlreicher weiterer Hashfunktionen. Berechnet wird auch die (recht geringe) Hashrate des dabei verwendeten Servers.

Ändert jemand den Eintrag auf

```
Alexa besitzt 1000 Bitcoins.
```

so ergibt sich dieser, völlig andere Hashwert:

```
86a00b845d282a9d6fe0635440ae5808b3a851aa48e9b490393eb14df2a7c8c4
```

Jede noch zu geringfügige Änderung in einem Block führt also dazu, dass sich der Hashwert des Blocks verändert. Für ein gegebenes Verfahren ist die Länge des Hashwerts immer gleich, egal wie groß der Block ist, für den der Hashwert berechnet wird. Auch ein leerer Inhalt, der gar kein Zeichen enthält, hat einen Hashwert; mit SHA-256 z.B.

```
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

Wegen der beabsichtigen Blockverkettung wird der Hashwert des ersten Blocks Bestandteil des zweiten Blocks. In die Berechnung des Hashwerts des zweiten Blocks geht also der eigentliche Inhalt dieses Blocks (z.B. die dort gespeicherten Transaktionen) und der Hashwert des Vorgängerblocks ein. Das setzt sich so für alle Blöcke der Blockchain bis zum neuesten Block fort.

Beispiel für einen zweiten Block, der eine Transaktion (Überweisung von 10 Bitcoins von Alexa an Ben) abbildet ist:

```
Alexa minus 10, Ben plus 10.
```

Der Hashwert dieses Blocks wird errechnet aus dem eigentlichen Eintrag des Blocks („Überweisung von 10 Bitcoins von Alexa an Ben“) und dem Hashwert des vorhergehenden Blocks:

```
Alexa minus 10, Ben plus 10.
8f5b24ed1fe3e718b09f2c6f5431f014082f1b13e2bf3d4cf556dd24dec2639d
```

Als Hashwert dieses Blocks ergibt sich:

```
979751ad7bc3c71337e4a0dde8f0a68960c79d6d6cbb48c088bb2fb229fb682a
```

Ändert sich also im Nachhinein etwas im ersten Block, so stimmen der Hashwert des zweiten Blocks und damit auch die Hashwerte aller nachfolgenden Blöcke nicht mehr, man würde allen Blöcken „ansehen“, dass etwas verändert wurde und die Blockchain ihre Stimmigkeit verloren hat.

Ein Angreifer müsste also nicht nur den ihn interessierenden Block (bei Bitcoin z.B. den Block, in dem eine Einzahlung des Angreifers auf sein Bitcoinkonto erfasst ist) ändern, sondern alle nachfolgenden Blöcke ebenfalls.

Da das Errechnen eines Hashwerts für eine bestimmte Zeichenfolge - ohne weitere Maßnahmen - sehr schnell möglich ist, wäre eine solche Blockchain trotz Signaturen und Verkettung aber noch nicht fälschungssicher. Jeder Teilnehmer könnte beliebige Blöcke (zu seinen Gunsten) verändern und sehr schnell durch Anpassung aller Folgeblöcke die Integrität der Blockchain wieder herstellen.

3. Mining als Garant der Datenintegrität

3.1. Warum Mining?

Eine bestimmte Zeichenfolge (also die Einträge in einem Block) führt immer zu einem eindeutigen Hashwert und ist sehr schnell berechnet. Das erlaubt eine sehr schnelle Überprüfung der Stimmigkeit eines Blocks.

Der Berechnungsprozess der zum Hashwert eines neuen Blocks führt, wird daher **künstlich verlängert.**

Der Prozess der Suche eines zulässigen Hashcodes wird als **Mining** bzw. **Proof of Work** bezeichnet. Derartige Proof of Work-Prozeduren gibt es bereits seit einiger Zeit u.a. zur Eindämmung von Spam-Mails: Vor Absenden einer Email muss eine willkürliche mathematische Aufgabe gelöst werden, die ein paar Sekunden dauert. Einen „normalen" Emailversender stört diese im Hintergrund ablaufende Verzögerung nicht. Für einen Spamabsender, der Millionen von Spams verschicken möchte, aber schon, da es schlicht zu lange dauern würde, bis seine Millionen von Spams versendet würden. Bei einer Verzögerung von nur 2 Sekunden würde das Versenden von 1 Mio. Mails bereits mehr als 23 Tage dauern.

Bei der Blockchain geschieht diese künstliche Verlängerung durch Einführung eines **Nonce** und einer geeigneten Wahl der sogenannten **Network Difficulty.**

3.2. Der Nonce

Jeder Block einer Blockchain wird um einen weiteren, zunächst unbestimmten Eintrag (eine zunächst beliebige Zahlenfolge) erweitert.

Diese Ergänzung wird als **Nonce**[3] bezeichnet und ist eine Zeichenfolge bestimmter Länge. Bei Bitcoin hat der Nonce eine Länge von 32 Bit bzw. 4 Byte und kann damit 2^{32} Werte annehmen ($4{,}3 \cdot 10^9$ bzw. 4,3 Mrd.).

Der Hashwert eines Blocks ergibt sich damit nicht nur aus dem Hashwert des Vorgängers und dem eigentlichen Blockinhalt, sondern aus

- dem Hashwert des Vorgängers,
- dem eigentlichen Blockinhalt und
- dem Nonce.

Je nach dem Wert des Nonce entsteht **ein anderer Hashwert** für den ganzen Block.

Das Mining besteht nun darin, einen Nonce zu finden, der zu einem Hashcode führt, der bestimmte Eigenschaften aufweist. Der Nonce ist

[3] Aus dem Englischen „for the nonce" – übergangsweise.

dabei zunächst unbestimmt und muss während des Minings **so lange zufällig verändert** werden, bis ein Hashwert entsteht, der die geforderten Eigenschaften aufweist.

Dieses **Mining** ist sehr rechenintensiv. Es gibt dabei keinen bestimmten ausgefeilten Algorithmus, sondern es muss einfach solange ausprobiert werden, bis ein gültiger Hashwert entsteht.

Das Ausmaß dieser künstlich geschaffenen Schwierigkeit, einen gültigen Hashcode zu finden, wird als Network Difficulty bezeichnet.

Je höher die **Network Difficulty,** umso aufwendiger wird es für einen Angreifer, die Blockchain zu manipulieren: Er müsste nämlich sehr aufwendig neue Nonces aller dem manipulierten Block nachfolgenden Blöcke generieren.

Wie diese Network Difficulty konkret festgelegt werden kann, wird in den Abschnitten 3.3 und 3.4 gezeigt.

Generell wird der Schwierigkeitsgrad so ausgestaltet, dass der Aufwand eines Angreifers so hoch wäre, dass er den zu erwartenden Ertrag des Angriffs übersteigt und damit eine Manipulation ausgeschlossen ist.

Bei Bitcoin wird die Network Difficulty immer so an die über die Jahre gestiegene Rechnerperformance angepasst, dass der Proof of Work eines Blocks im Durchschnitt etwa 10 Minuten dauert. Das hat einmal der Erfinder des Bitcoin, Satoshi Nakamoto (Pseudonym), so festgelegt.

Obwohl der Aufwand des Minings zur Erzeugung eines zulässigen Hashs sehr aufwendig ist, ist das **Überprüfen der Richtigkeit** eines Hashwerts - auch nach Einführung des Nonce - sehr schnell für alle Teilnehmer der Blockchain möglich. Durch diese schnelle Überprüfbarkeit würden fehlerhafte (also z.B. manipulierte Blocks) sehr schnell entdeckt; auch deshalb, weil diese Korrektheitsprüfung von allen Teilnehmern der Blockchain sehr schnell durchgeführt werden kann.

3.3. Network Difficulty einfach erklärt

Um das Ermitteln eines zulässigen Hashwerts zu erschweren, werden an den zu errechnenden Hashwert bestimmte Anforderungen gestellt.

Konkret wird gefordert, dass der Hashwert (der ja mathematisch eine Zahl ist) eine bestimmte Obergrenze nicht überschreitet. Diese Obergrenze wird als **Targetwert** bezeichnet.

Zur Illustration wird in diesem Abschnitt davon ausgegangen, dass der Hashwert nur die zehn Ziffern 0,1,...,9 verwendet. Das in Abschnitt 3.4 verwendete SHA-256-Verfahren benutzt dagegen die 16 hexadezimalen Ziffern.

Ist die Länge des Hashcodes z.B. m=5, so ist beispielsweise der numerische Wert des Hashcodes 00567 (mit zwei führenden Nullen) kleiner als der Wert des Hashcodes 12567. Würde also z.B. gefordert, dass ein zulässiger Hashcode der Länge m=5 mit 2 Nullen beginnt, müsste der Nonce solange zufällig verändert werden, bis ein Hashcode zwischen 00000 und 00999 entsteht.

Das Mining besteht also darin, einen **Nonce zu finden, der zu einem Hashwert führt, der den Targetwert nicht übersteigt.** Im Prinzip läuft das so ab, dass man zufällig einen Nonce erzeugt und prüft, ob der entstehende Hashcode kleiner als der Target ist. Trifft dies zu, ist ein zulässiger Hashcode gefunden.

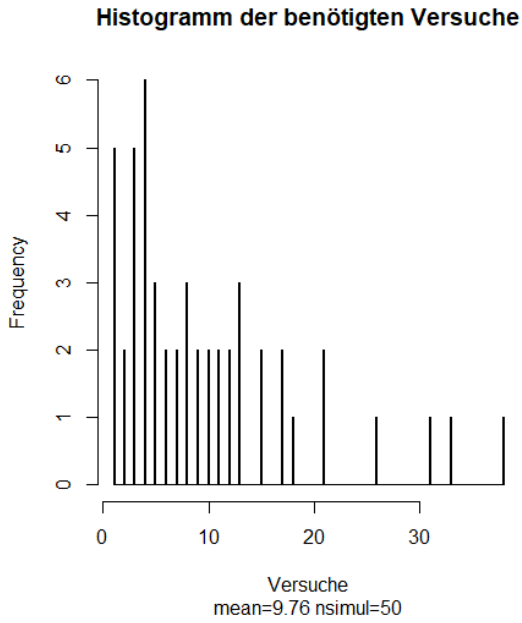
Diese Technik soll zunächst an einem sehr einfachen Beispiel illustriert werden. Dazu sei angenommen, die Hashfunktion liefere 5 Dezimalstellen (m=5) und daher maximal 100000 Werte: 0,1,...,99999.

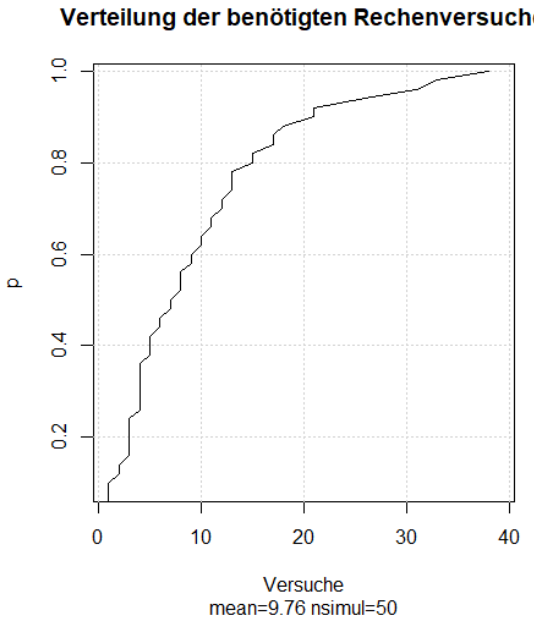
Zunächst soll im Beispiel als **maximaler Targetwert** ein Wert von 9999 (also einschließlich der Null 10000 mögliche Hashwerte) festgelegt werden. D.h. nur einer von 10 - durch zufällige Wahl eines Nonce - generierten 5stelligen Hashwerten wird diesen maximalen Targetwert nicht überschreiten und zu einem akzeptierten Hashwert führen.

Diese Zahl der nötigen Versuche bei Verwendung des maximalen Targetwerts wird in der Variable Z=10 Hashs gespeichert. Man benötigt durchschnittlich 10 Versuche, um einen gültigen Hash zu finden.

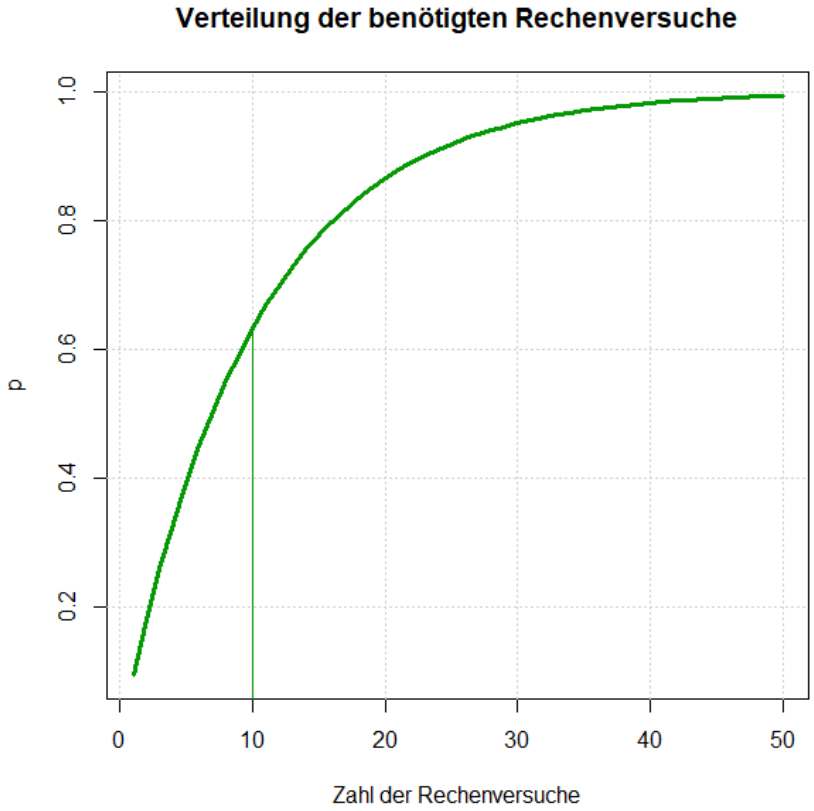
Bei einer tatsächlich durchgeführten Suche kann man einen gültigen Hash zufällig auch nach einer geringeren oder höheren Zahl an Versuchen finden. Die Zahl der nötigen Versuche ist also eine Zufallsvariable.

Das nachfolgende Chart zeigt die Verteilung des Resultats von simulierten 50 Rechengängen (nsimul=50). In dieser Simulation wurden durchschnittlich 9,76 Versuche benötigt, um einen gültigen Hashwert zu finden.





Die Verteilung der benötigten Rechenversuche folgt einer **Exponentialverteilung** (vgl. das nachfolgende Chart). In 63% der Fälle wird man 10 oder weniger Versuche benötigen und in etwa 2% der Fälle mehr als 40 Versuche.



Will man die Schwierigkeit erhöhen, einen zulässigen Hashwert zu finden, also die erforderliche Rechendauer erhöhen, so wird man einen **Targetwert** wählen, der kleiner als der maximale Targetwert ist.

Die sogenannte **Difficulty** ergibt sich dann als

$$D = \text{Maximaler Target} / \text{Target}.$$

Und der **Target** als

$$\text{Target} = \text{Maximaler Target} / D.$$

Entspricht der Target dem maximalen Target, so beträgt die Difficulty D=1.

Will man in unserem Beispiel die Difficulty erhöhen, wählt man einen Target, der kleiner ist als der maximale Target (9999; einschließlich der Null 10000 mögliche Werte), z.B. 999 (einschließlich der Null nur 1000 mögliche Hashwerte). Die Difficulty wäre dann

$$D = \text{Maximaler Target} / \text{Target} = 10000 \text{ Hashs} / 1000 \text{ Hashs} = 10.$$

Ein für einen Block akzeptierter Hashwert darf also den Targetwert von 999 nicht überschreiten.

Man muss damit durchschnittlich

$$D \cdot Z = 10 \cdot 10 \text{ Hashs} = 100 \text{ Hashs}$$

ermitteln, bis ein gültiger Hashwert gefunden wird, d.h. nur einer von 100 zufällig erzeugten Hashwerten (durch zufällige Veränderung des Nonce) ist auch ein gültiger Hashwert.

Würde unser (sehr ineffizienter) Rechner nur 2 Hashs je Sekunde berechnen können (**Hashrate** = 2 Hashs / Sekunde), ergäbe sich eine durchschnittliche Rechendauer von

$$D \cdot Z / \text{Hash Rate} =$$

$$10 \cdot 10 \text{ Hashs} / 2 \text{ Hashs/Sekunde} =$$

$$50 \text{ Sekunden}.$$

Da sich die Hashrate über die Jahre durch den Einsatz immer leistungsfähigerer Rechner erhöhen wird, wird man auch die Difficulty von Zeit zu Zeit erhöhen müssen, damit sich die Rechendauer nicht verkürzt.

Verdopppelt sich in unserem Beispiel durch die Verfügbarkeit neuer Rechner die Hashrate auf 4 Hashs je Sekunde, so wird man die Difficulty von 10 auf 20 erhöhen müssen, damit weiterhin eine **mittlere Rechendauer von 50** Sekunden erforderlich ist. Als Targetwert ergibt sich damit ein Wert von

Target = Maximaler Target / D =

10000 Hashs / 20 =

500 Hashs,

was bedeutet, dass nunmehr ein zulässiger Hashwert den Wert von 499 nicht überschreiten darf.

Alle Angaben zu diesem Beispiel sind noch einmal in der nachfolgenden Übersicht zusammengefasst.

Übersicht zur Network-Difficulty

Länge das Hashcodes	m=5 Stellen: 100.000 Hashs = [0,1,2,...,99.999]
Maximaler Target	10.000 Hashs = [0,1,2,...,9.999]
Benötige Rechenversuche*	Z = 10

Network-Difficulty	**D = 10**
Target	Maximaler Target / D = 10.000 Hashs / 10 = 1.000 Hashs = [0,1,2,...,999]
Benötigte Rechenversuche*	Z·D = 10·10 = 100

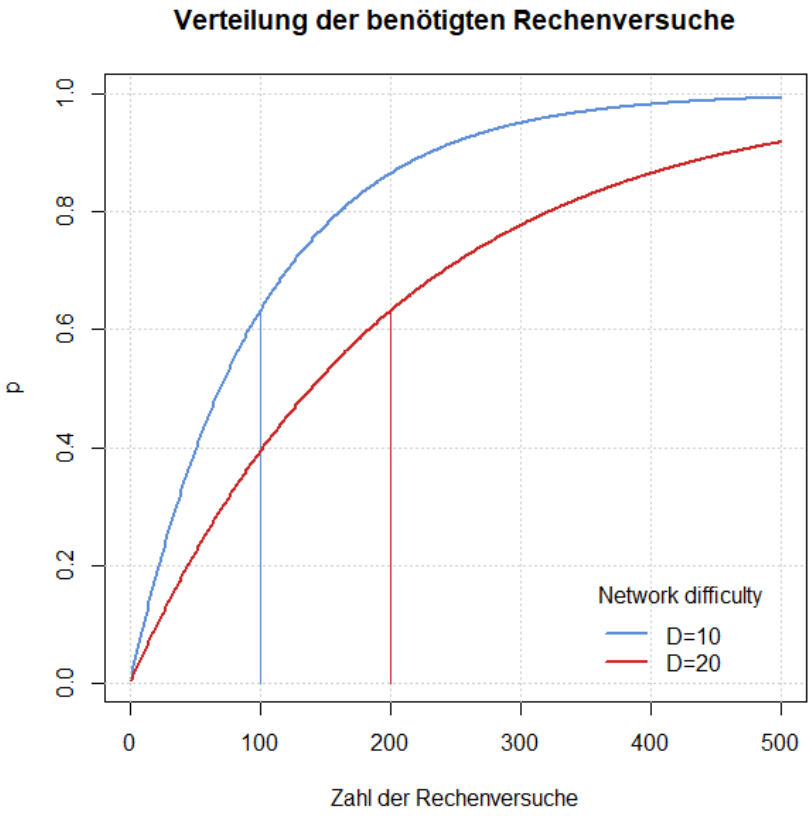
Hashrate	2 Hashs / sec
Durchschnittliche Rechendauer	Z·D/Hashrate = 10·10/2 = 50 sec

Gestiegene Hashrate und Anpassung der Network-Difficulty

Hashrate ist gestiegen	4 Hashs / sec
Durchschnittliche Rechendauer bei bisheriger Network Difficulty	Z·D/Hashrate = 10·10/4 = 25 sec
Neue **Network-Difficulty**	**D_2 = 20**
Neuer Target	Maximaler Target / D_2 = 10.000 Hashs / 20 = 500 Hashs = [0,1,2,...,499]
Benötigte Rechenversuche*	Z·D_2 = 10·20 = 200
Durchschnittliche Rechendauer	Z·D_2/Hashrate = 10·20/4 = 50 sec

*) Durchschnitt.

Das nachfolgende Chart zeigt die statistische Verteilung der erforderlichen Zahl an Rechenversuchen für das obige Beispiel und die beiden verwendeten Network Difficulties von 10 und 20 mit der erwarteten durchschnittlichen Zahl an Rechenversuchen von 100 (D=10) und 200 (D=20).



3.4. Network Difficulty des Bitcoin

Das Grundprinzip zur Network-Difficulty wurde im vorherigen Abschnitt skizziert. Hier wird das Vorgehen bei Bitcoin erläutert. Durch den dabei verwendeten viel längeren Hashcode sind die nötigen Ausführungen dazu sehr viel technischer. Da es sich dabei im Prinzip um das gleiche Konzept handelt, kann der Abschnitt daher auch übersprungen werden.

Ein Hashwert bei Bitcoin besteht wegen des verwendeten **SHA-256-Verfahrens** aus 256 Bits (Nullen und Einsen) und kann daher maximal 2^{256} Werte annehmen. In dezimaler Schreibweise entspricht dies $1{,}16 \cdot 10^{77}$ möglichen Werten, also einer Zahl mit 78 Stellen.

Typischerweise wird der Hash in **hexadezimaler Schreibweise** dargestellt. Während das gebräuchliche (und oben in Abschnitt 3.3 verwendete) dezimale Zahlensystem die 10 Ziffern 0, 1, ..., 9 kennt, gibt es im hexadezimalen System 16 Ziffern. Die ersten 10 Ziffern werden dabei mit den üblichen Ziffern 0 bis 9 dargestellt. Die Ziffern(!) 10, 11, 12, 13, 14 und 15 durch die Buchstaben A, B, C, D, E und F.

Beispiele:

- dezimal 9 = $9 \cdot 10^0$ = hexadezimal 9 = $9 \cdot 16^0$
- dezimal *10* = $1 \cdot 10^1$ = hexadezimal *A* = $10 \cdot 16^0$
- dezimal *15* = $1 \cdot 10^1 + 5 \cdot 10^0$ = hexadezimal *F* = $15 \cdot 16^0$
- dezimal *18* = $1 \cdot 10^1 + 8 \cdot 10^0$ = hexadezimal *12* = $1 \cdot 16^1 + 2 \cdot 16^0$
- dezimal *35* = $3 \cdot 10^1 + 5 \cdot 10^0$ = hexadezimal *23* = $2 \cdot 16^1 + 3 \cdot 16^0$
- dezimal *255* = $2 \cdot 10^2 + 5 \cdot 10^1 + 5 \cdot 10^0$ = hexadezimal *FF* = $15 \cdot 16^1 + 15 \cdot 16^0$

Auf Grundlage des SHA-256-Verfahrens gibt es 2^{256} verschiedene **mögliche Hashs** - in hexadezimaler Schreibweise:

```
FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
```

Der **maximale Targetwert** wird dabei - in hexadezimaler Schreibweise - festgesetzt auf:

```
0000 0000 FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
```

Da Bitcoin das übliche Floating Point-Zahlenformat[4] verwendet, wird dieser maximale Target noch verringert zu:

0000 0000 FFFF 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Das ergibt für den maximalen Target (errechnet aus den 4 hexadezimalen F-Ziffern der vorstehenden hexadezimalen Zahl):

$$15 \cdot 16^{55} + 15 \cdot 16^{54} + 15 \cdot 16^{53} + 15 \cdot 16^{52} =$$

$$2{,}695954 \cdot 10^{67} \approx 2^{224} \text{ Hashs.}$$

Hinweis: Mit meinem HEX-DEC COMPUTER auf

https://www.egon-bellgardt.de/hexdec_input.php

lassen sich hexadezimale Zahlen in dezimale Zahlen umwandeln. Auch einfache Rechenoperationen mit zwei hexadezimalen Zahlen sind möglich.

Ein zufällig generierter Hashwert, also ein einzelner Hashwert aus 2^{256} möglichen Hashwerten, überschreitet daher mit einer Wahrscheinlichkeit von

$$2^{224} / 2^{256} = 2^{224-256} = 2^{-32}$$

den maximalen Target *nicht.* D.h. einer von $Z=2^{32}$ Hashwerten wird den maximalen Target nicht überschreiten. Dieser maximale Target entspricht einer **Difficulty** von 1, da gilt

$$D = \text{Maximaler Target} / \text{Target} = 2^{224} / \text{Target}$$

und

$$\text{Target} = \text{Maximaler Target} / D = 2^{224} / D$$

Eine Erhöhung der Difficulty verringert also den Target und damit den maximalen Wert des zulässigen Hashwerts.

Im Durchschnitt muss man zufällig $D \cdot Z = D \cdot 2^{32}$ Hashwerte berechnen, um einen akzeptierten Hashwert für einen Block zu finden.

Im **Februar 2021** betrug die **Bitcoin Difficulty** etwa

$$D = 21{,}4 \cdot 10^{12}$$

(= 21,4 T; T=Terra)

[4] Analoges Beispiel in dezimaler Schreibweise: Ein Zahlenformat darf z.B. maximal den Wert 9000 annehmen und nicht den Wert 9999.

Daraus ergibt sich folgender **Target:**[5]

$$2^{224} / D =$$

$$2^{224} / 21{,}4 \cdot 10^{12} =$$

$$1{,}26 \cdot 10^{54}.$$

Ein akzeptierter Hashwert eines Blocks darf diesen Target nicht überschreiten.

Man muss durchschnittlich

$$D \cdot Z \text{ Hashs} =$$

$$21{,}4 \cdot 10^{12} \cdot 2^{32} \text{ Hashs} =$$

$$9{,}19 \cdot 10^{22} \text{ Hashs}$$

berechnen um einen validen Hashwert zu finden.

Die **Hashrate** im Bitcoin-Netzwerk beträgt im Februar 2021 etwa

$$160 \cdot 10^{18} \text{ Hashs je Sekunde}$$

(= 160 EH/s, EH=Exa-Hash)

Daher dauert die **Suche** nach einem validen Hashwert durchschnittlich

$$D \cdot Z / \text{Hashrate} =$$

$$21{,}4 \cdot 10^{12} \cdot 2^{32} \text{ Hashes} / 160 \cdot 10^{18} \text{ Hashs/Sekunde} =$$

$$574{,}45 \text{ Sekunden} =$$

$$9{,}57 \text{ Minuten}.$$

Bei einer **Difficulty** von $D_1=1$ entspräche der aktuelle Target dem maximalen Target und eine Suche würde bei der aktuellen Hashrate nur

$$D_1 \cdot Z / \text{Hashrate} =$$

$$1 \cdot 2^{32} \text{ Hashs} / 160 \cdot 10^{18} \text{ Hashs/Sekunde} =$$

$$2{,}68 \cdot 10^{-11} \text{ Sekunden}$$

dauern.

Die Network Difficulty wird bei Bitcoin alle 2016 Blöcke überprüft und ggf. so angepasst, dass die Rechendauer von etwa 10 Minuten aufrecht erhalten wird.

[5] In hexadezimaler Notation:
0000 0000 0000 0000 000d 21b9 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Beispielsweise betrug die **Difficulty Anfang 2016** noch $0{,}104 \cdot 10^{12}$ (0,104 T) und die durchschnittliche Hashrate $0{,}743 \cdot 10^{18}$. Daraus ergab sich damals eine durchschnittliche Suchdauer von

$$D \cdot Z \,/\, \text{Hashrate} =$$

$$0{,}104 \cdot 10^{12} \cdot 2^{32} \text{ Hashs} \,/\, 0{,}743 \cdot 10^{18} \text{ Hashs/Sekunde} =$$

$$601{,}2 \text{ Sekunden} =$$

$$10{,}02 \text{ Minuten.}$$

Die **Hashrate** hat sich von Anfang 2016 bis Anfang 2021 auf das 205fache erhöht. Das entspricht einer Steigerung von etwa 0,29% pro Tag oder 9,2% pro Monat.

3.5. Das Bitcoin-Mining

Der **Aufwand** derjenigen Blockchain-Teilnehmer, die dieses Mining durchführen, die **Miner,** wird bei Bitcoin durch Bitcoins entschädigt. Bei Bitcoin ist dies gleichzeitig die einzige Art und Weise, **wie neue Bitcoins überhaupt entstehen:** Sie stellen schlicht eine Aufwandsentschädigung zur Sicherstellung der Richtigkeit der Blockchain dar (sogenannter **Block Reward**). Man spricht daher auch kurz vom „Minen von Bitcoins" bzw. vom „Schürfen von Bitcoins". Es ist also im Prinzip ein „Geldverdienen" durch die Berechnung zulässiger Hashwerte. Das verdiente Geld entsteht aber sozusagen aus dem Nichts, es ist damit eine alternative Art der Geldschöpfung.

Die ersten Miner bei Bitcoin erhielten pro Block 50 Bitcoins als Block Reward. Dieser Wert halbiert sich bei Bitcoin alle 210000 Blöcke **(Bitcoin Halfing).** Im Februar 2021 beträgt der **Block Reward** bei Bitcoin 6,25 Bitcoins je Block.

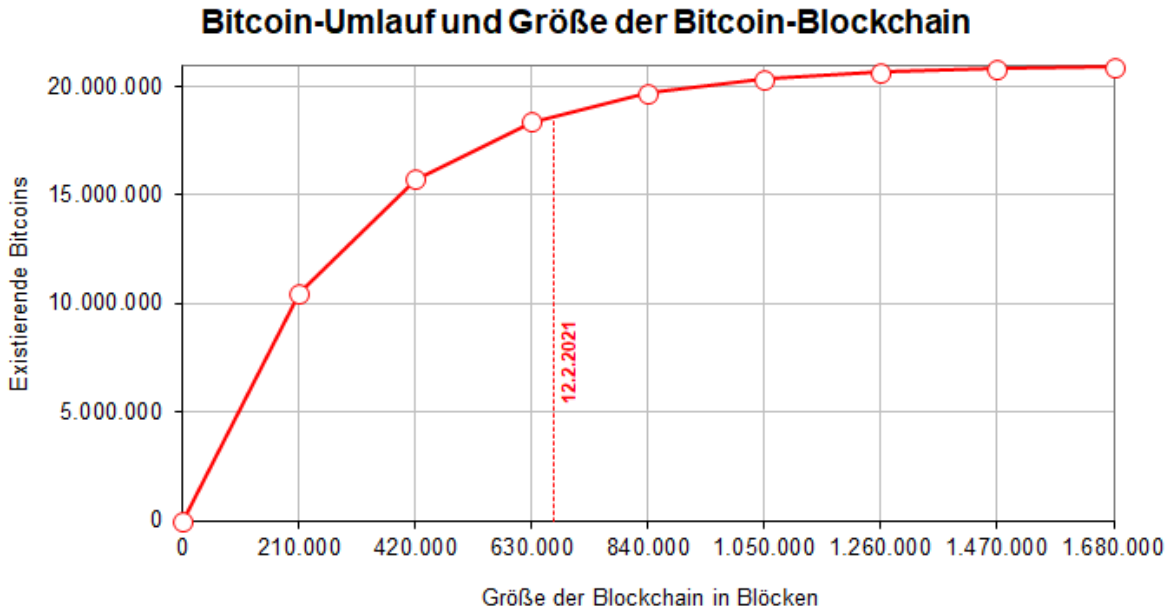
Das letzte Halfing erfolgte am 11. Mai 2020. Seither wurden durchschnittlich 146 Blöcke pro Tag erstellt. Damit sind pro Tag etwa 912 Bitcoins neu entstanden.

Am 12. Februar 2021 bestand die Blockchain bei Bitcoin aus **670362 Blöcken.**

Ab einer Blockzahl von 840000 folgt das nächste Bitcoin Halfing. Dann erhalten die Miner nur noch 3,125 Bitcoins je Block.

Dieser Rückgang sorgt dafür, dass es eine maximale Zahl von Bitcoins gibt. Diese Maximalzahl ist also von vorne herein sozusagen mathematisch festgelegt und beträgt 21 Mio.

Am 12. Februar 2021 waren **18,63 Mio. Bitcoins im Umlauf.** Daher können in der Zukunft nur noch 2,37 Mio. Bitcoins neu geschürft werden.



Einmal durch die Miner geschaffene Bitcoins können auch gehandelt werden, z.B. an andere Teilnehmer verkauft werden. Durch dieses Kaufen und Verkaufen entsteht ein bestimmter **Kurswert des Bitcoin,** z.B. ausgedrückt in Euro.

Die neuesten produzierten Bitcoinblöcke kann man sich auf

https://www.blockchain.com/de/explorer

ansehen.

Der **Kurswert** des Bitcoin betrug am 12. Februar 2021 39157 Euro je Bitcoin. Bewertet zu diesem Kurs beträgt der Wert aller Bitcoins 729 Mrd. Euro.

Binnen Jahresfrist hat sich der Kurs damit mehr als vervierfacht (Faktor 4,15) und innerhalb der letzten 2 Jahre mehr als verzwölffacht (Faktor 12,25). Das entspricht durchschnittlichen monatlichen Wertsteigerungen von 12,6% bzw. 11,0%.

Anfang Februar 2021 betrug die **physische Größe** der Bitcoin-Blockchain 328 GigaByte.

Organisatorischer Ablauf des Mining

Neue Transaktionen werden in einem Pool gesammelt. Die Miner greifen auf diesen Pool zu, prüfen die Zulässigkeit der Transaktion (z.B. reicht der Kontostand des Absenders für eine Überweisung), bilden einen geeigneten Block und beginnen den Proof of Work, also die Suche nach einem geeigneten Nonce. Eine Transaktion wird dann dem Block zugeordnet, dessen Miner den Proof of Work zuerst abgeschlossen hat.

So arbeiten grundsätzlich immer mehrere Miner am Einbinden der Transaktionen in Blöcke. Ist der Proof of Work durch einen Miner abgeschlossen, sind damit auch alle in diesem Block enthaltenen Transaktionen in die Blockchain eingebaut. Sie können nicht mehr Inhalt anderer Blöcke sein.

Miner, die ebenfalls an Blöcken arbeiteten, die mindestens eine dieser Transaktionen zum Inhalt hatten, müssen den Proof of Work an diesem Block beenden.

Das führt zu einer Art Wettbewerb unter den Minern und natürlich zu einer massiven Verschwendung von Rechenzeit. Daher gehen Miner auch dazu über, sich geeignet abzusprechen, wer wann welche Transaktionen einbaut.

3.6. Alternativen zum Proof of Work

Proof of Stake

Nicht der schnellste Teilnehmer, sondern ein zufällig gewählter Teilnehmer (Forger), darf den Block einbauen.

Die Zufallswahl basiert auf bedingten Wahrscheinlichkeiten, die sich bei Kryptowährungen u.a. an der Größe des gehaltenen Guthabens orientieren. Auch das stellt sicher, dass nicht irgendein Teilnehmer, sondern nur der zufällig gewählte Teilnehmer die Blockchain verändern darf.

Es ergibt sich sehr viel weniger Rechenaufwand und damit ein geringerer Energieverbrauch. Teilnehmer mit hohem Guthaben erhalten aber häufiger die Chance, ihr Guthaben weiter zu erhöhen.

Zwar werden Manipulationsversuche eines Forgers mit dessen dauerhaftem Ausschluß sanktioniert, ein Hack ausgesuchter Forger durch Dritte stellt allerdings ein gewisses Sicherheitsrisiko dar.

Proof of Burn

Bei Kryptowährungen müssen hierbei Teilnehmer zunächst eine gewisse Menge an Einheiten dieser Währung (Coins) dauerhaft investieren, um eine Chance auf Blockvalidierung zu erhalten. Die so investierten Coins gelten als „verbrannt", sind unbrauchbar und daher nicht mehr Bestand der umlaufenden Coins der Kryptowährung. Damit erhöht sich die Knappheit der Währung, was als einer der Vorteile des Verfahrens genannt wird.

Die Teilnehmer erwarten dann, dass sich ihre Investitionen durch die für die Blockvalidierung ausgezahlten Belohnungen mit der Zeit rentieren. Das gilt als Anreiz, ehrlich zu arbeiten. Die Überprüfung der Validierungsarbeit dauert aber länger als im Proof of Work.

4. Ein kurzes Resümee

Technische Überlegenheit der Blockchain?

Theoretisch ist die Blockchain eine simple und geniale Idee, um die Integrität von elektronisch gespeicherten Tatbeständen zu sichern.

Durch das erforderliche Mining ergibt sich aber pro Vorgang eine Transaktionsdauer, die sehr viel länger ist als etwa die Transaktionsdauer eines Kreditkartenvorgangs.

Sollen Tatbestände wie etwa die Einträge aller deutschen Grundbücher abgebildet werden, ergibt sich eine extreme Blockchaingröße.

Will man sehr viele Lebensbereiche auf die Blockchaintechnik umstellen, hat jeder Teilnehmer sehr große Datenmengen auf seinem Rechner, da es ja mehrere Blockchains für alle möglichen Lebensbereiche gibt (Bank, Grundstücke, Versicherungen, Einwohnerdaten, etc.).

Also: Man benötigt sehr viel Speicherplatz und hat einen sehr hohen Datentransfer, der z.B. das tägliche mehrfache Streamen von HD-Spielfilmen um ein Vielfaches übersteigen dürfte.

Wegen des hohen Speicherbedarfs und dem hohen Transfervolumen halte ich die Blockchain ohne drastische Effizienzverbesserungen aktuell in vielen der angedachten Anwendungsbereiche noch nicht für zukunftsfähig.

Und: Durch die zunehmende Konzentration der Blockvalidierungen auf die Teilnehmer mit den leistungsfähigsten Rechnern (Proof of Work) bzw. auf Teilnehmer mit dem höchsten Guthaben (Proof of Stake) oder den höchsten Investitionen (Proof of Burn) wird die ursprüngliche Idee einer Validierung durch viele Teilnehmer verwässert und man nähert sich dadurch so gesehen wieder ein wenig den klassischen Lösungen, die nur einen Garanten der Datenvalidität haben.

Ökonomischer Wert des Bitcoin?

Aktuell erlebt der Bitcoin durch Käufe großer Investoren einen Boom. Einige Analysten halten Bitcoin-Investments auch deshalb für ökonomisch durchaus erwägenswert, da die Bitcoin-Menge – genauso wie etwa das Goldvorkommen – nicht vermehrbar ist, es sich also im ein knappes begrenztes Gut handelt, das damit zur Wertaufbewahrung geeignet sei. Im Netz wir wird der Bitcoin teilweise schon als das „knappste Gut der Menschheit" bezeichnet[6].

Welcher ökonomische Wert – jenseits seiner Zahlungsmittelfunktion – hinter dem Bitcoin steht soll hier nicht weiter diskutiert werden. Aber schon bei der Beurteilung seiner Eigenschaft als Wertaufbewahrungsmittel muss - jenseits des aktuellen Hypes - an die hohe Volatilität seines Kurses in der Vergangenheit gedacht werden.

6 https://www.btc-echo.de/wieso-bitcoin-das-knappste-gut-der-menschheit-ist-und-was-das-fuer-den-kurs-bedeutet/