Mathematische Betrachtung

Mining von Bitcoins

Das Bitcoin-Protokoll definiert vor allem zwei wesentliche Voraussetzungen zum Mining von Bitcoin:

(a) alle ca. 10 Minuten soll ein neuer Block gefunden werden.

(b) die Belohnung für einen gefundenen Block beträgt zunächst 50 BTC und wird alle 4 Jahre halbiert.

Zu a): Der Schwierigkeitsgrad D der zu lösenden mathematischen Probleme wird der Gesamtrechenleistung R des Bitcoin-Netzwerks stets so angepasst, dass im Mittel alle 10 Minuten ein Teilnehmer einen neuen Block findet, d.h. berechnet. Die Größen D und R sind also proportional, kurz $D \propto R$.

zu b): Die Belohnung (engl. reward) verringert sich mit zunehmender Laufzeit des Protokolls. Seit Beginn im Jahr 2009 wurden bereits 2 Halbierungen durchgeführt, sodass der aktuelle reward bei 12,5 BTC pro Block liegt.

Wir setzen das Jahr 2009 als $t = 0$ fest und definieren die folgenden Funktionen, wobei t die Zeit in Jahren seit 2009 angibt:

$$b(t) = 6 \cdot 24 \cdot 365 \cdot t = 52560t \tag{1}$$

$$r(t) = \frac{50}{\lceil \frac{t}{4} \rceil} \tag{2}$$

$$B(t) = b(1) \cdot \int_0^t r(x)\, dx \tag{3}$$

Dabei beschreibt $b(t)$ die Anzahl der insgesamt gefundenen Blöcke in der Blockchain (6 Blöcke pro Stunde, 24 Stunden pro Tag, 365 Tage pro Jahr), $r(t)$ gibt den reward für einen zum Zeitpunkt t gefundenen Block an und $B(t)$ liefert die Menge der bis zum Zeitpunkt t geschürften Bitcoins.

Bei der Berechnung der Werte von $B(t)$ tritt das Problem auf, dass die Funktion $r(t)$ nicht auf ganz $\mathbb{R}_{\geq 0}$ stetig ist, sondern an den Stellen $x = 4k$ mit $k \in \mathbb{N}_{>0}$ Sprungstellen aufweist. Die Integrale existieren somit immer nur abschnittsweise und es gibt aufgrund der fehlenden totalen Differenzierbarkeit keine Stammfunktion von $r(t)$. (Auch GeoGebra berechnet die Integralwerte z.T. falsch.)

Lösung ist hier die Darstellung von $\int_0^t r(x)\, dx$ als diskrete Summe. Dies ist leicht möglich, da die Funktion überall treppenartig aussieht, wobei eine Stufe stets die Länge 4 Jahre hat. Wenn wir die Stufen mit den natürlichen Zahlen indizieren hat das Rechteck unter der Stufe mit der fortlaufenden Nummer $k \in \mathbb{N}$ einen Flächeninhalt von $4 \cdot \frac{50}{2^k}$.

Die genaue Anzahl der geschürften Bitcoins nach t Jahren kann in unserem Modell berechnet werden durch:

$$B(t) = b(4) \cdot \sum_{k=0}^{\lfloor \frac{t}{4} \rfloor - 1} r(4k) + b(t - 4 \cdot \lfloor \frac{t}{4} \rfloor) \cdot r(t) \tag{4}$$

Dabei gibt die große Summe die Fläche unter den vollständig durchlaufenen Stufen an und der zweite Summand beschreibt den Restanteil der letzten Stufe. Damit ergibt sich beispielsweise für die Anzahl der geschürften Bitcoin bis zum Beginn des Jahres 2018 ($t = 9$):

$$\begin{aligned} B(9) &= b(4) \cdot \textstyle\sum_{k=0}^{\lfloor \frac{9}{4} \rfloor - 1} r(4k) + b(9 - 4 \cdot \lfloor \frac{9}{4} \rfloor) \cdot r(9) \\ &= b(4) \cdot \textstyle\sum_{k=0}^{2-1} r(4k) + b(9 - 4 \cdot 2) \cdot r(9) \\ &= b(4) \cdot (r(0) + r(4)) + b(1) \cdot r(9) \\ &= 210240 \cdot (50 + 25) + 52560 \cdot 12,5 \\ &= 16,425 \cdot 10^6 \end{aligned}$$

Bis Jahresbeginn 2018 sind also ca. 16,425 Millionen Bitcoin geschürft (worden).

Die Formel für die Berechnung von $B(t)$ vereinfacht sich bei einer unendliche langen Betrachtungsdauer, da erstens das Restglied, der zweite Summand, wegfällt und zweitens die Anzahl der Summanden in der großen

Summe auch bei Aufteilung in Streifen von jeweils 4 Jahren noch unendlich groß ist. Über eine unendliche Zeit betrachtet sind das Integral über $r(x)$ und die Reihe der Rechtecke also gleich. Es folgt:

$$\begin{aligned}\int_0^\infty r(x)\,dx &= 4 \cdot \sum_{k=0}^{\infty} \frac{50}{2^k} \\ &= 4 \cdot 50 \cdot \sum_{k=0}^{\infty} \frac{1}{2^k} \\ &= 4 \cdot 50 \cdot 2 \\ &= 400\end{aligned}$$

Damit folgt $\lim_{t\to\infty} B(t) = b(1) \cdot \int_0^\infty r(x)\,dx = 52560 \cdot 400 = 21024000 \approx 21 \cdot 10^6$. Es können also (wie im Protokoll erwähnt) nur 21 Millionen Bitcoin geschürft werden. Die kleinste gültige Verrechnungseinheit beträgt dabei $10n$ BTC $= \frac{1}{10^8}$ BTC $= 0,00000001$ BTC $= 1$ *Satoshi*. (Diese Minimaleinheit heißt du Ehren des Bitcoin-Erfinders *Satoshi Nakamoto* ebenfalls *Satoshi*.)
Die Abweichung unserer Rechnung vom genauen Wert liegt einerseits am nicht ganz exakten Beginn und andererseits daran, dass die Kalibrierung der Schwierigkeit D nicht in Echtzeit im Verhältnis zu neu hinzukommenden Rechenleistung erfolgen kann, sodass $b(t)$ nicht perfekt linear ist. In gewissem Sinne kann man diese Durchschnittsschätzung als Regressionsgerade betrachten. Laut Bitcoin-Protokoll soll etwa im Jahr 2140 der letzte Bitcoin geschürft werden. Wir rechnen nach:
$B(2140 - 2009) = B(131) = b(1) \cdot \int_0^{131} r(x)\,dx \approx 4 \cdot \sum_{k=0}^{32} \frac{50}{2^k} \approx 21024000$. Der Zuwachs der Fläche unter $r(x)$ jenseits von 2140 ist also verschwindend gering. Wir erhalten hier bereits schätzungsweise den Wert des uneigentlichen Integrals.

Blockchain

Struktur

Die Blockchain ist eine Baumstruktur, in der mithilfe der gefundenen Blöcke alle jemals durchgeführten Transaktionen festgehalten werden. Diese Datenbank wird dezentral, d.h. bei jedem Teilnehmer gehalten und gepflegt. Sobald ein neuer Block gefunden wird, wird dieser im ganzen Netzwerk den jeweiligen Blockchains hinzugefügt. Es handelt sich also um ein *public ledger*. Die Kontostände liegen also offen, nur der Kontoinhaber ist unbekannt (und nicht bestimmbar).

Jeder Knoten außer der Wurzel, dem sogenannten *Genesisblock* enthält neben den durch ihn bestätigten, d.h. noch offenen, Transaktionen eine digitale Unterschrift (einen kryptographischen Schlüssel) des Erzeugers des Blocks und einen Verweis (einen 8-Byte-Hashwert) auf den Vorgängerblock. Die anderen, weiter hinten liegenden Blöcke sind jedem Block unbekannt. Wenn bisweilen mehrere Blöcke (fast) zeitgleich gefunden werden, kann es zu Abspaltungen, sogenannten *forks* der Blockchain kommen. Hier ist geregelt, dass die jeweils längste Kette von allen Clients als die gültige akzeptiert wird. Dies wird durch das Bitcoin-Protokoll verbindlich geregelt.

Wird eine Transaktion von einem Bitcoin-Konto auf ein anderes durchgeführt, wird diese nach Bestätigung durch die nachfolgenden 6 Blöcke für gültig erklärt. Bis zur letztendlichen Bestätigung einer Transaktion vergeht gemäß der Mining-Geschwindigkeit also etwa eine Stunde.

Manipulation

Wir betrachten ein Bitcoin-Netzwerk mit T Teilnehmern, von denen $M \subseteq T$ Manipulatoren und $L \subseteq T$ loyal sind. Natürlich gilt $T = M \cup L$ und $M \cap L = \emptyset$. Die Manipulatoren könnten nun versuchen einen Block aus der Vergangenheit, der bereits in der Blockchain enthalten ist, auszutauschen durch einen Block, der ihre eigenen möglicherweise illegalen Transaktionen für gültig erklärt und für immer im Netzwerk verankert. Da aber alle Teilnehmer, Manipulatoren wie auch loyale, nur die längste aktuell vorliegende Kette der Blockchain als gültig akzeptieren, müssen zur Manipulation des k-ten Blocks in einer Blockchain der Länge $n \in \mathbb{N}$ mit $k < n$ (mindestens) die folgenden Bedingungen erfüllt sein:

- Gruppe M muss Fälschungen der fehlenden $n - k$ Blöcke erzeugen.
- Gruppe M muss dafür sorgen, dass sie schneller Blöcke produzieren kann als Gruppe L.

Während nämlich M den gefälschten Block k und die $n - k$ Folgeblöcke berechnet, werden die L weiter ca. alle 10 Minuten Blöcke finden und sie der echten längsten Kette hinzufügen. Auch dieser Rückstand muss von M aufgeholt werden. Anschaulich ist klar, dass eine Manipulation der Blockchain schwieriger wird, je

weiter eine Transaktion (ein Block) in der Vergangenheit liegt, da umso mehr gefälschte Blöcke erzeugt werden müssen. (Die gefälschten Blöcke müssen trotzdem den Anforderungen an gültige Blöcke entsprechen, d.h. sie sind prinzipiell nicht schneller erzeugbar.) Wir betrachten den einfachsten Fall, nämlich eine Manipulation des gerade gefundenen letzten Blocks in der Blockchain. Sei dieser Block b_0. Da die Mining-Geschwindigkeit ausschließlich von der verfügbaren Rechenleistung abhängt, definieren wir $R(M) = \sum_{t \in M} R(t)$ und analog $R(L) = \sum_{t \in L} R(t)$ als verfügbare anteilige Rechenleistung der Gruppen M und L (mit $R(T) = 1$). Will M die längste Kette nun durch Austauschen des Blocks b_0 verfälschen, so muss M in der Lage sein, die gefälschten Blöcke b'_0 und b'_1 (dessen Nachfolger) schneller zu finden als L in der Lage ist, einen korrekten Block b_1 zu finden, der die echte Kette verlängert.
Genauer: M muss eine Folge $\langle b'_i \rangle_{i \in \mathbb{N}}$ finden, sodass zu irgendeinem Zeitpunkt t diese Folge länger ist als die Folge $\langle b_i \rangle_{i \in \mathbb{N}}$ der echten Blöcke. Damit dies gelingen kann, muss offenbar $R(M) > R(L)$ gelten. Die benötigte Rechenleistung $R(M)$ kann dabei wie folgt nach unten abgeschätzt werden:

$$\begin{array}{rrcl} & R(T) & = & 1 \\ \iff & R(M) + R(L) & = & 1 \\ \iff & R(L) & = & 1 - R(M) \end{array}$$

Nehmen wir nun an, dass $R(M) > R(L)$ gilt. Dann folgt mit obiger Rechnung:

$$\begin{array}{rrcl} & R(M) & > & R(L) \\ \iff & R(M) & > & 1 - R(M) \\ \iff & 2 \cdot R(M) & > & 1 \\ \iff & R(M) & > & 0,5 \end{array}$$

Für einen derartigen erfolgreichen Angriff auf die Blockchain muss M also über mehr als 50% der gesamten Rechenleistung im Bitcoin-Netzwerk verfügen, um die anderen theoretisch zwingen zu können, die gefälschte Kette als längste Kette der Blockchain zu akzeptieren und die darin enthaltenen Transaktionen durch eigene Blöcke zu bestätigen und zu konservieren.

Technische Betrachtung

Grundlagen des Mining-Prozesses

Als Mining bezeichnet man das Erzeugen von neuen Blöcken für die Blockchain, in denen Transaktionen bestätigt werden. Das Mining erfolgt durch die Teilnehmer im Bitcoin-Netzwerk; diese konkurrieren um den Betrag, der für das erfolgreiche Mining an die Teilnehmer ausgeschüttet wird. Bei Bitcoin wird im Durchschnitt alle ca. 10 Minuten ein Block gefunden. Die Entlohnung betrug dabei zu Beginn 50 BTC. Diese Belohnung halbiert sich alle 4 Jahre, um der gestiegenen Rechenleistung der Teilnehmer entgegenzuwirken. Die Gesamtmenge alle Bitcoins ist durch das Protokoll (derzeit) auf 21 Millionen Einheiten begrenzt. Mining kann prinzipiell durch jeden Computer erfolgen. Wegen der gestiegenen Anforderungen entwickeln sich dazu fähige Geräte nach dem Schema: CPU $\rightarrow$ GPU $\rightarrow$ FPGA $\rightarrow$ ASIC.

Um das Mining sicher und skalierbar zu machen, müssen einige Voraussetzungen erfüllt werden.

- Das mathematische Problem muss eine Einwegfunktion sein.
- Der Problemsteller darf das Problem nicht durch A-priori-Kenntnisse lösen können.
- Der Schwierigkeitsgrad muss (nahezu) beliebig skalierbar sein.

Ansätze

Wir betrachten fünf mögliche Entwürfe zum Mining. (Die Implementierung erfolgte in Java.)

PrimeMiner

- Challenge: prüfe, ob $p \in \mathbb{N}$ eine Primzahl ist
- Miner: suche einen nichttrivialen Teiler von p. Falls keiner gefunden wird, ist es eine Primzahl.
- Verifikation: wie Miner.
- Schwierigkeit: schwer (Miner), schwer (Verifikation)
- Mangel: Verifikation ist genauso aufwendig wie Suche der Miner.

DigitMiner

- Challenge: suche $n \in \mathbb{N}$ so, dass $n < 10^k$, wobei k den Schwierigkeitsgrad darstellt.
- Miner: rate Zahlen aus z.B. $[0; 10^9]$ und prüfe, ob die Bedingung erfüllt ist.
- Verifikation: prüfen, ob Stellenanzahl der Antwort korrekt ist.
- Schwierigkeit: leicht (Miner), leicht (Verifikation)
- Mangel: Suche benötigt kein Raten, sondern das Problem kann quasi sofort gelöst werden.

FactorMiner

- Challenge: finde die Teiler von $n = p\cdot$.
- Miner: faktorisiere p.
- Verifikation: prüfe, ob p durch die Antwort ohne Rest teilbar ist.
- Schwierigkeit: schwer (Miner), leicht (Verifikation)
- Mangel: Der Problemsteller kennt die Lösung, d.h. die Faktorisierung a priori.

PolynomMiner

- Challenge: wähle zufällig Koeffizienten für ein Polynom $p(x)$. Gesucht sind die Nullstellen.
- Miner: verwende numerische Verfahren, um eine Nullstelle x_n zu finden.
- Verifikation: prüfe, ob $p(x_n) = 0$ gilt.
- Schwierigkeit: schwer (Miner), leicht (Verifikation)
- Mangel: Skalierbarkeit nur eingeschränkt gegeben, da effiziente Verfahren zur numerischen Lösung von ganzrationalen Funktionen existieren.

RealMiner

- Challenge: finde zu einem gegebenen Nonce einen Hashwert fester Länge, der eine bestimmte Anzahl von führenden Nullen aufweist (Schwierigkeit)
- Miner: suche den Hash ausgehend vom Nonce durch Testen verschiedener Eingaben.
- Verifikation: bilde den Hash der Eingabe
- Schwierigkeit: schwer (Miner), leicht (Verifikation)
- Mangel: kein struktureller Mangel bekannt, aber erheblicher Ressourcenverbrauch.

Vor- und Nachteile im Allgemeinen

Vorteile von Bitcoin

- direkte Kontrolle über Zahlungsvorgänge.
- keine dritte Partei benötigt.
- weitgehende Anonymität bei den Zahlungen.
- Inflationssicherheit durch begrenzte Verfügbarkeit.

Nachteile von Bitcoin

- Totalverlust des Kapitals bei Verlust der Schlüssel.
- derzeit noch geringe Akzeptanz.
- Risiko eines 50+1-Angriffs.
- hoher Ressourcenverbrauch.
- Praktikabilität (Dauer einer Transaktion relativ hoch).
- Aufteilbarkeit begrenzt auf $21 \cdot 10^6 \cdot 10^8$ diskrete Einheiten.