Eine Analyse von Konsensmechanismen in öffentlichen Blockchain-Netzwerken

Niklas Wilhelm
Fakultät für Informatik
Technische Hochschule Rosenheim
Rosenheim, Deutschland
niklas.wilhelm@stud.th-rosenheim.de

Abstract*—Bei einer Blockchain handelt es sich um ein verteiltes System mit zahlreichen potenziellen Anwendungsmöglichkeiten. Für einen breitflächigen Einsatz der Technologie ist es allerdings notwendig, derzeit bestehende Leistungs- und Sicherheitsprobleme zu lösen. Da der in der Blockchain eingesetzte Konsensmechanismus einen erheblichen Einfluss auf diese Faktoren hat, ist er für die Weiterentwicklung der Technologie von großer Bedeutung. Diese Studienarbeit beschäftigt sich mit sowohl etablierten als auch neuartigen Ansätzen hinsichtlich der Konsensbildung und untersucht ihre Auswirkungen auf die Leistung und Sicherheit von öffentlichen Blockchains. Die in der Arbeit analysierten Konsensansätze umfassen Proof of Work (PoW), Proof of Stake (PoS) und Federated Byzantine Agreement (FBA). Um einen Vergleich mit realen Werten zu ermöglichen, wurde für jeden Ansatz ein exemplarisches Konsensprotokoll ausgewählt. Nakamoto's PoW-Protokoll wird als Referenz für PoW herangezogen, während Tendermint und das Stellar Consensus Protocol (SCP) als Vertreter für PoS bzw. FBA dienen. Die Analyseergebnisse zeigen unter anderem auf, dass Tendermint und das SCP gegenüber dem Nakamoto-PoW einen drastisch reduzierten Energieverbrauch und einen deutlich höheren Durchsatz ermöglichen. Des Weiteren wurde aufgedeckt, dass das SCP zu einer stark zentralisierten Netzwerkstruktur führen kann, was erhebliche Auswirkungen auf die Ausfallsicherheit der Blockchain hätte.

I. Einführung

Eine Blockchain ist ein verteiltes System, das Transaktionen speichert und verwaltet. Die Transaktionen sind hierbei in kryptographisch gesicherten und miteinander verknüpften Blöcken organisiert. Eine Blockchain kommt ohne zentrale Instanzen aus und bietet Vorteile wie Ausfall- und Manipulationssicherheit sowie Transparenz [1].

Das Interesse an Blockchain wurde 2008 durch den Erfolg von Bitcoin [2] geweckt. Bei Bitcoin handelt es sich um ein Zahlungsnetzwerk, das digitales Geld verwaltet. Es ermöglicht Benutzern, Transaktionen ohne Beteiligung von zentralen Vermittlern wie Banken durchzuführen. Aufgrund zahlreicher weiterer potenzieller Anwendungsbereiche hat die Technologie in den letzten Jahren von der Wissenschaft und der Industrie eine hohe Aufmerksamkeit erfahren. Sie wird als eine der fundamentalen Technologien betrachtet, die zu einer Revolutionierung der Systemlandschaften führen kann [1]. Zu den betroffenen Bereichen gehört nicht nur die Finanzbranche, sondern u.a. auch das Lieferketten- und Identitätsmanagement, die Medienindustrie, der juristische Sektor oder das Internet der Dinge [3]. Durch die intensive Forschung an der Blockchain-Technologie wurden Leistungs- und Sicherheitsprobleme aufgedeckt, die gelöst werden müssen, bevor die Technologie breitflächig Anwendung finden kann. Eine kritische Komponente eines Blockchain-Netzwerks bildet der Konsensmechanismus, der eine Übereinkunft zwischen den teilnehmenden Knoten hinsichtlich des Zustands der Blockchain sicherstellt [4]. Er hat einen erheblichen Einfluss auf die Leistung und Sicherheit der Blockchain und spielt deshalb bei der Weiterentwicklung der Technologie eine zentrale Rolle.

In den vergangenen Jahren wurden zahlreiche Konsensmechanismen entwickelt, um die Leistungs- und Sicherheitsprobleme zu lösen. Die große Anzahl an Alternativen macht es allerdings schwierig, für einen bestimmten Anwendungszweck den am besten geeigneten Konsensmechanismus zu identifizieren [5]. Vor allem bei öffentlichen Blockchain-Netzwerken, die für alle Personen zugänglich und dadurch anfällig für Angriffe sind, ist die Auswahl des richtigen Mechanismus von großer Relevanz. Es ist folglich eine systematische Analyse erforderlich, um die Eignung von Konsensmechanismen für öffentliche Blockchain-Netzwerke zu bewerten.

In dieser Studienarbeit liegt der Fokus auf den Konsensmechanismen Proof of Work, Proof of Stake und Federated Byzantine Agreement. Diese Mechanismen werden bereits in verschiedenen öffentlichen Blockchain-Netzwerken eingesetzt und bieten deshalb hinreichend Daten für eine Analyse. Beispiele für solche Netzwerke sind Bitcoin [2] (Proof of Work), Cosmos Hub [6] (Proof of Stake) und Stellar [7] (Federated Byzantine Agreement). Das Ziel besteht darin, die Funktionsweise der ausgewählten Konsensmechanismen anhand der genannten Blockchain-Netzwerke zu erläutern und ihre Leistungs- und Sicherheitseigenschaften zu analysieren.

II. Blockchain

A. Grundlagen

Wie bereits im Kapitel I erwähnt, handelt es sich bei einer Blockchain um ein verteiltes System, das Transaktionen in miteinander verketteten Blöcken speichert. Nach Yao et al. [8] besteht eine Blockchain aus der Infrastrukturschicht, der Netzwerkschicht, der Datenschicht, der Konsensschicht und der Applikationsschicht (siehe Abbildung 1).

Die Infrastrukturschicht stellt die grundlegende Infrastruktur für die Blockchain bereit und umfasst Hardware-Ressourcen, Netzwerkkomponenten und die Deployment-Umgebung [8].

Abbildung 1. Blockchain-Architektur [8]

Hardware-Ressourcen wie Server und Speichermedien sind für den Betrieb der Knoten notwendig, während Netzwerkkomponenten wie Router eine Verbindung zwischen ihnen gewährleisten. Die Deployment-Umgebung, dazu gehören u.a. virtuelle Maschinen und Docker-Container, ermöglicht Flexibilität und Skalierbarkeit des Blockchain-Systems.

Die Netzwerkschicht, auch als Propagationsschicht bekannt [9], steuert die Kommunikation zwischen den Knoten auf Basis eines Peer-to-Peer (P2P) Netzwerks. Ein P2P-Netzwerk [10] ist ein dezentrales Netzwerk, in dem teilnehmende Knoten direkt miteinander kommunizieren und sich Ressourcen wie Dateien oder Rechenleistung teilen. Die teilnehmenden Knoten sind gleichberechtigt, handeln autonom und kommen ohne zentrale Instanzen aus. Wenn ein neuer Knoten einem Blockchain-Netzwerk beitreten möchte, dann stellt er über das Transmission Control Protocol (TCP) Verbindungen zu anderen Knoten her. Sobald eine Verbindung aufgebaut ist, empfängt der neue Knoten kontinuierlich Broadcast-Nachrichten von den bereits verbundenen Knoten. Jede Broadcast-Nachricht beinhaltet Adressinformationen aller Knoten, die mit dem sendenden Knoten verbunden sind. Dies ermöglicht dem neuen Knoten, Verbindungen zu sämtlichen Knoten innerhalb der Blockchain aufzubauen. Wenn der neue Knoten mit einem anderen Knoten verbunden ist, synchronisiert er von diesem die Informationen aller Blöcke und gilt anschließend als arbeitsfähig [8].

Die Datenschicht spiegelt die Datenstruktur einer Blockchain wider und variiert je nach Implementierung [8]. Bei Bitcoin [2] enthält der Blockheader beispielsweise Informationen wie den Hash des vorherigen Blocks, den Merkle-Root-Hash und die Nonce. Während der Hash des vorherigen Blocks die Blöcke miteinander verknüpft, repräsentiert der Merkle-Root-Hash den Hash aller Transaktionen im Blockkörper. Die Nonce ist eine Zahl und wird verwendet, um den Konsensmechanismus Proof of Work zu realisieren. Sie wird so gewählt, dass der resultierende Blockhash einen bestimmten Schwierigkeitsgrad erfüllt (siehe Kapitel IV).

Die Konsensschicht ist dafür verantwortlich, eine Übereinkunft zwischen den teilnehmenden Knoten hinsichtlich des Zustands der Blockchain-Daten zu erreichen [11]. Wenn ein Knoten einen Block hinzufügt, sollen auch die anderen Knoten denselben Block zu ihrer Kopie der Blockchain hinzufügen. Dabei ist zu beachten, dass mehrere Knoten gleichzeitig einen neuen Block hinzufügen und im Blockchain-Netzwerk verbreiten können. In solchen Fällen muss die Konsensschicht sicherstellen, dass alle Knoten dieselbe Version der Blockchain verwenden. Die zentrale Komponente der Konsensschicht ist der in der Blockchain verwendete Konsensmechanismus. Er dient dazu, die beschriebene Problematik zu lösen und hat wie bereits erwähnt einen erheblichen Einfluss auf die Leistung und Sicherheit der Blockchain [4].

Die Applikationsschicht ist die oberste Schicht in einer Blockchain-Architektur, auf der Endanwendungen für Benutzer bereitgestellt werden [1]. Auf dieser Schicht wird z.B. durch die Definition von Protokollen die Übertragung von Kryptowährung zwischen Teilnehmern gesteuert oder die Erstellung von Smart Contracts ermöglicht. Nach Wilkens und Falk [12] ist ein Smart Contract ein “Programm auf der Blockchain, das auf Basis einer WENN-DANN-Logik arbeitet, sodass bei Eintritt eines zuvor festgelegten Ereignisses (sog. trigger) automatisch eine ebenfalls zuvor festgelegte Aktion (bspw. eine Transaktion) ausgeführt wird“. In der Applikationsschicht können darüber hinaus Belohnungsmechanismen festgelegt werden [1]. Bei Bitcoin erhalten Knoten, welche neue Blöcke für die Blockchain erzeugen, beispielsweise Bitcoins. Solche Knoten werden auch als Miner bezeichnet [13].

B. Klassifikation von Blockchain-Netzwerken

Nach der Klassifikation von Paul et al. [14] werden Blockchain-Netzwerke in u.a. folgende Typen unterteilt: privat, öffentlich und hybrid. Diese Einteilung hängt primär von dem Grad der Zugangsbeschränkung ab.

Öffentliche bzw. “permissionless“ Blockchains sind vollständig dezentralisiert und für jedermann zugänglich [1]. Jeder Teilnehmer kann Transaktionen durchführen, neue Blöcke zur Blockchain hinzufügen und das vollständige Transaktionsregister (Ledger) einsehen. Ein weiterer wesentlicher Aspekt ist die Möglichkeit, dass jeder Teilnehmer am Konsensprozess mitwirken und dadurch Einfluss darauf nehmen kann, welche Blöcke der Blockchain hinzugefügt werden. Bekannte Beispiele für öffentliche Blockchains sind die in dem Kapitel I erwähnten Netzwerke Bitcoin, Cosmos Hub und Stellar.

Private Blockchains, auch als “permissioned“ Blockchains bezeichnet, kommen nur in privaten Organisationen und Institutionen zum Einsatz [8]. Hierbei ist der Zugang beschränkt und Knoten benötigen eine Autorisierung durch eine zentrale Instanz, um beitreten zu können [15]. Die Knoten in privaten Blockchains sind bekannt und ihre Vertrauenswürdigkeit wird

vorausgesetzt, was zur Verwendung schnellerer und energieeffizienterer Konsensmechanismen führt. Die Kontrolle über die zentrale Instanz ermöglicht eine präzise Berechtigungsvergabe, wodurch z.B. nur bestimmte Knoten Zugang zum Transaktionsregister haben oder Transaktionen durchführen können. Da somit ein gewisses Maß an Zentralisierung besteht, unterscheidet sich eine private Blockchain von der vollständigen Dezentralisierung einer öffentlichen Blockchain [14].

Hybride Blockchains beschreiben die Verknüpfung von öffentlichen und privaten Blockchains [14]. Die hybride Architektur soll die Vorteile beider Blockchain-Typen vereinen und einen Kompromiss zwischen Transparenz und Kontrolle schaffen. So können beispielsweise Transaktionen öffentlich einsehbar sein, während andere Informationen nur von autorisierten Knoten eingesehen werden können. Hybride Blockchains ermöglichen einen hohen Grad an Flexibilität und können auf bestimmte Anwendungszenarien zugeschnitten werden.

III. Konsensmechanismen

A. Grundlagen

Konsensmechanismen, die in verteilten Systemen verwendet werden, stehen vor einer Vielzahl komplexer Probleme. Dazu gehört u.a. der Umgang mit Knotenausfällen, die Bewältigung von Nachrichtenverzögerungen und die Verarbeitung von Nachrichten, die in einer falschen Reihenfolge ankommen. Zudem müssen sie fähig sein, mit Knoten umzugehen, die vorsätzlich einen Schaden anrichten wollen oder eigennützig handeln [4].

Es existieren zahlreiche Konsensmechanismen, die in der Forschungsliteratur vorgeschlagen wurden, um diese Herausforderungen zu bewältigen. Jeder dieser Ansätze trifft eine Reihe von Annahmen in Bezug auf Synchronität, Nachrichtenübermittlung, Ausfälle, bösartige Knoten, Leistung und Sicherheit der ausgetauschten Nachrichten [4]. Die Unterschiede in den Annahmen führen dazu, dass nicht jeder Konsensmechanismus für alle Typen von Blockchain-Netzwerken passend ist. Beispielsweise sind Konsensmechanismen, die eine grundsätzliche Vertrauenswürdigkeit der Knoten voraussetzen und daher bösartige Knoten nur marginal berücksichtigen, für den Einsatz in einem öffentlichen verteilten System ungeeignet.

Unabhängig von den Annahmen und der Eignung für unterschiedliche Blockchain-Netzwerke, bestehen drei grundlegende Anforderungen an Konsensmechanismen [1]:

- Safety / Consistency: Alle Knoten erzeugen die gleiche Ausgabe und die von den Knoten erzeugten Ausgaben sind gemäß den Regeln des Protokolls gültig.
- Liveness / Availability: Alle nicht fehlerhaften Knoten erzeugen eine Ausgabe, was zeigt, dass die Terminierung des Protokolls stattgefunden hat.
- Fault Tolerance: Das Konsensprotokoll arbeitet wie vorgesehen, auch wenn einige Knoten Fehler aufweisen oder ausfallen.

Das Kernprinzip, das in Blockchain-Netzwerken die *Fault Tolerance* ermöglicht, ist die State Machine Replication (SMR) [16]. Bei diesem Ansatz wird ein deterministischer Automat auf alle beteiligten Knoten eines verteilten Systems repliziert. Um die *Safety* in diesem Netzwerk sicherzustellen, müssen alle Knoten dieselben Eingaben in der gleichen Reihenfolge verarbeiten. Zudem muss gewährleistet werden, dass entweder alle Knoten eine Nachricht empfangen oder keiner - ein Konzept, das als Atomic Broadcast bekannt ist [17]. Dies führt dazu, dass alle Knoten die gleichen Zustandsübergänge durchlaufen, wodurch eine durchgehende Konsistenz im Systemzustand aufrechterhalten wird. Der Konsensmechanismus hat die Aufgabe, diesen Atomic Broadcast in der Blockchain zu gewährleisten.

Nach Fu et al. [18] muss bei der Entwicklung von Konsensmechanismen zudem das im Jahr 1985 veröffentliche *FLP Impossibility Theorem* [19] berücksichtigt werden. Das Theorem legt dar, dass es in einem asynchronen verteilten System unmöglich ist, einen Konsensmechanismus zu entwerfen, der sowohl *safe* als auch *live* ist, solange auch nur ein einziger Knotenausfall möglich ist. Unter einem asynchronen System ist zu verstehen, dass keine Zeitschranken für die Ausführung von Operationen oder die Übertragung von Nachrichten existieren. In einem solchen System können Nachrichten unbegrenzt verzögert werden, wodurch es schwierig wird, den Unterschied zwischen einem ausgefallenen Knoten und einem sehr langsamen Knoten zu erkennen. Um das Problem zu vermeiden, basiert ein Großteil der Konsensmechanismen auf einer schwachen Form von synchroner Kommunikation. Bei diesem Kommunikationsmodell muss eine Nachricht innerhalb einer vorgegebenen Zeitschranke ankommen. Ist das nicht der Fall, gilt der sendende Knoten als ausgefallen.

Hinsichtlich der *Fault Tolerance* existieren bei verteilten Systemen zwei wesentliche Modelle: *Crash fault tolerance* (CFT) und *Byzantine fault tolerance* (BFT) [1]. Bei dem CFT-Modell wird ein System als fehlertolerant angesehen, wenn es trotz des Ausfalls eines oder mehrerer Knoten korrekt funktioniert. In diesem Kontext bedeutet ein Ausfall, dass ein Knoten aufgrund von z.B. Software-Bugs oder einer unterbrochenen Netzwerkverbindung aufhört zu reagieren. Sollte solch ein Ereignis eintreten, müssen die übrigen Knoten weiterhin in der Lage sein, den Konsens untereinander zu erreichen und die Systemfunktionen aufrechtzuerhalten. Wenn sich Knoten allerdings unerwartet oder bösartig verhalten, kann bei dem CFT-Modell die Zuverlässigkeit nicht garantiert werden. Daher werden CFT-Konsensmechanismen wie Raft hauptsächlich in privaten Blockchains verwendet [8].

Anders als das CFT-Modell befasst sich das BFT-Modell mit Knoten, die sich unerwartet oder bösartig verhalten [1]. Das Modell ist auf das im Jahr 1982 veröffentlichte *Byzantine Generals Problem* [20] zurückzuführen. Es handelt sich dabei um ein Gedankenexperiment, bei dem Generäle ihre Armeen um eine feindliche Stadt versammelt haben. Die Generäle müssen sich auf einen Angriffs- oder Rückzugsplan einigen und kommunizieren dafür über Boten. Unter den Generälen können sich allerdings Verräter befinden, die eine Übereinkunft verhindern wollen. Die Abbildung 2 zeigt ein beispielhaftes Szenario.

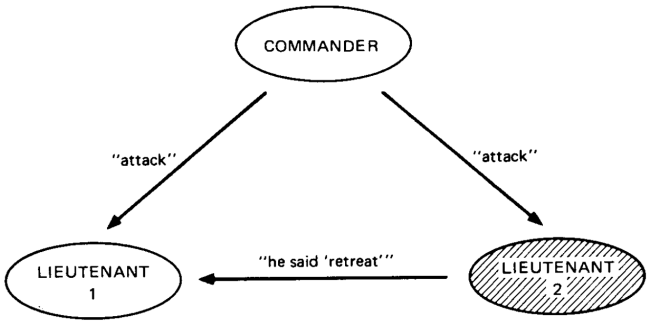


Abbildung 2. Byzantine Generals Problem [20]

Hier sendet der Kommandant einen Angriffsbefehl an zwei Leutnants. Der zweite Leutnant ist ein Verräter und täuscht den ersten Leutnant, indem er ihm zum Rückzug auffordert. Da der erste Leutnant nicht weiß, ob der Kommandant oder der andere Leutnant ein Verräter ist, kann er nicht beurteilen, welche Nachricht die richtigen Informationen enthält. Dadurch kann er keinen Konsens mit dem loyalen Kommandanten erreichen. Zur Lösung des Problems müssen die Generäle einen Algorithmus festlegen, der sicherstellt, dass alle loyalen Generäle zu einem Konsens gelangen und dass eine kleine Anzahl von Verrätern nicht dazu führt, dass ein loyaler General den falschen Plan ausführt.

Wie bereits erwähnt, sind öffentliche Blockchains für jedermann zugänglich und somit anfällig für bösartige Akteure. Um die Sicherheit für diesen Netzwerk-Typ zu gewährleisten, müssen deshalb Konsensmechanismen eingesetzt werden, welche das BFT-Modell integrieren.

B. Klassifikation von Konsensmechanismen

In der Forschungsliteratur existieren zahlreiche Ansätze zur Klassifizierung von Konsensmechanismen. Nguyen und Kim [21] unterscheiden beispielsweise zwischen beweisbasierten und abstimmungsbasierten Konsensmechanismen. Bei beweisbasierten Mechanismen wie Proof of Work oder Proof of Stake befinden sich die Knoten im Wettbewerb miteinander. Die Knoten müssen beweisen, dass sie qualifizierter als die anderen Knoten sind, um einen neuen Block zur Blockchain hinzuzufügen. Abstimmungsbasierte Mechanismen wie Raft treffen ihre Entscheidung bzgl. der Aufnahme eines neuen Blocks auf Grundlage von Stimmen. Ein neuer Block wird nur dann hinzugefügt, wenn mindestens T Knoten dafür stimmen und diesen identischen Block in ihre eigene Blockchain integrieren. Dabei stellt T einen Schwellenwert dar, welcher je nach Blockchain-System variiert. Grundsätzlich gilt allerdings, dass es sich bei T um mehr als 50 Prozent der Knoten handeln muss. Laut Ngyuen und Kim sind beweisbasierte Konsensmechanismen primär für öffentliche Blockchain-Netzwerke geeignet, während in privaten Blockchain-Netzwerken eher abstimmungsbasierte Konsensmechanismen zu bevorzugen sind. Der abstimmungsbasierte Konsensmechanismus Federated Byzantine Agreement, welcher im öffentlichen Blockchain-Netzwerk Stellar zum Einsatz kommt, zeigt allerdings, dass das nicht zwingend der Fall sein muss.

Eine weitere Klassifikation von Konsensmechanismen stammt von Ferdous et al. [1], welche die Konsensmechanismen in anreizbasiert und anreizlos unterteilen. Anreizbasierte Konsensmechanismen belohnen die Knoten, die einen neuen Block zur Blockchain hinzufügen. Der Belohnungsmechanismus dient als Anreiz für die Knoten, sich gemäß dem festgelegten Konsensprotokoll zu verhalten. Mechanismen dieser Kategorie werden vor allem in öffentlichen Blockchain-Netzwerken eingesetzt. Ein Beispiel stellt das Blockchain-Netzwerk Bitcoin dar, dass, wie im Kapitel II erwähnt, erfolgreiche Miner mit Bitcoins belohnt. Auf der anderen Seite stehen anreizlose Konsensmechanismen, die in privaten Blockchain-Netzwerken verwendet werden. Bei diesen Mechanismen gibt es keine Belohnung für die Teilnehmer, die einen neuen Block zur Blockchain hinzufügen. Die Teilnehmer gelten als vertrauenswürdig, weshalb kein Anreiz zur Einhaltung des Protokolls geschaffen werden muss. Raft stellt ein Beispiel für einen anreizlosen Konsensmechanismus dar [18].

Neben solchen grundlegenden Klassifikationen existieren auch umfassende Taxonomien zur Unterteilung von Konsensmechanismen. Beispielsweise haben Lashkari und Musilek [5] eine Hierarchie entwickelt, die acht unterschiedliche Typen von Konsensmechanismen auf der untersten Ebene umfasst. In dieser Studienarbeit wird allerdings auf eine Erläuterung der Typen verzichtet, da die zuvor beschriebenen Klassifikationen als ausreichend betrachtet werden.

C. Eigenschaften

Ferdous et al. haben in ihrer Arbeit "Blockchain consensus algorithms: A survey" [1] verschiedene Eigenschaften von Konsensmechanismen erhoben und diese in Form einer Taxonomie dargestellt (siehe Abbildung 3). Laut der Taxonomie lassen sich die Eigenschaften von Konsensmechanismen in vier Hauptgruppen unterteilen: Strukturelle Eigenschaften, Block- und Belohnungseigenschaften, Sicherheitseigenschaften und Leistungseigenschaften.

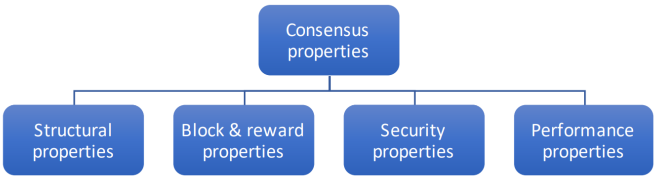


Abbildung 3. Eigenschaften von Konsensmechanismen [1]

Strukturelle Eigenschaften legen fest, wie die verschiedenen Knoten innerhalb eines Blockchain-Netzwerks aufgebaut sind und wie sie in den Konsensalgorithmus einbezogen werden. Ein Beispiel hierfür ist die Eigenschaft "Knotentypen". Die Knotentypen stellen die unterschiedlichen Arten von Knoten dar, die für die Erzielung eines Konsenses erforderlich sind. Die Block- und Belohnungseigenschaften können hauptsächlich als quantitative Metriken zur Unterscheidung

verschiedener Kryptowährungen herangezogen werden. Zum Beispiel repräsentiert die Eigenschaft "Gesamtvorrat" die maximale Menge an verfügbaren Coins einer Kryptowährung. Sicherheitseigenschaften beschreiben sicherheitsrelevante Merkmale des Konsensmechanismus wie mögliche Angriffsvektoren. Leistungseigenschaften können hingegen verwendet werden, um die Leistung eines Konsensprotokolls quantitativ zu messen. Ferdous et al. definieren für diese Hauptgruppe u.a. folgende Eigenschaften:

- Fehlertoleranz: bezeichnet die maximale Anzahl fehlerhafter Knoten, die das jeweilige Konsensprotokoll tolerieren kann.
- Durchsatz: bezieht sich auf die Anzahl der Transaktionen, die das Protokoll in einer Sekunde verarbeiten kann.
- Energieverbrauch: zeigt an, ob der Konsensmechanismus eine signifikante Menge an Energie verbraucht.

In dieser Studienarbeit werden die Konsensmechanismen hinsichtlich der möglichen Angriffsvektoren und der beschriebenen Leistungseigenschaften analysiert und bewertet. Obwohl strukturelle Eigenschaften keinen direkten Einfluss auf die Bewertung eines Konsensmechanismus haben, werden sie bei der Erläuterung der Funktionsweise des jeweiligen Mechanismus behandelt. Die Block- und Belohnungseigenschaften sind hingegen nicht im Fokus dieser Arbeit und werden deshalb nicht weiter betrachtet.

IV. Proof of Work

A. Grundprinzip

Proof of Work (PoW) basiert auf der Interaktion zwischen einem Beweiser und einem Prüfer. Der Beweiser führt für die Erreichung eines bestimmten Ziels eine ressourcenintensive Rechenaufgabe durch und stellt dieses Ergebnis einem Prüfer oder einer Gruppe von Prüfern zur Verfügung. Das Überprüfen der Rechenaufgabe nimmt dabei viel weniger Ressourcen in Anspruch als das Lösen der Aufgabe [1]. Das Prinzip von PoW soll vor allem einen Schutz gegen Angriffe bieten. Ein Angreifer müsste enorm viele Ressourcen aufwenden, um das System zu manipulieren, was i.d.R. unwirtschaftlich und daher abschreckend ist.

Die Grundidee von Proof of Work wurde 1993 von Cynthia Dwork und Moni Naor zur Bekämpfung von Spam vorgeschlagen [22]. Der Absender einer E-Mail muss hierbei eine ressourcenintensive Rechenaufgabe lösen, bevor die E-Mail gesendet werden kann. Da diese Aufgabe für eine einzelne E-Mail nur minimale Verzögerungen verursacht, ist sie für normale Nutzer unproblematisch. Bei Massenversendern von Spam würden hingegen erhebliche Kosten in Form von Rechenzeit und Stromverbrauch entstehen.

Im Kontext von Blockchains muss der Beweiser, also ein Knoten im Blockchain-Netzwerk, eine rechenintensive Aufgabe lösen, um einen neuen Block zur Blockchain hinzufügen zu können [1]. Bei der Aufgabe handelt es sich meistens um eine Hashfunktion, die einen bestimmen Hash erzeugen soll. Der Bereich, in dem sich der Hash befinden muss, wird durch den sogenannten Schwierigkeitsgrad definiert. Der Beweiser führt die Hashfunktion in mehreren Durchgängen aus, wobei in jedem Durchgang ein anderer Eingangswert (oft "Nonce" genannt) verwendet wird. Sobald der Beweiser einen Eingangswert gefunden hat, der einen Hash innerhalb des vorgegebenen Bereichs erzeugt, sendet er den neuen Block zusammen mit diesem Wert an das Netzwerk. Die anderen Knoten im Netzwerk fungieren nun als Prüfer und können mit minimalen Ressourcenaufwand den vorgeschlagenen Block validieren. Sie führen dafür die gleiche Hashfunktion mit dem übermittelten Eingangswert aus und kontrollieren, ob der resultierende Hash innerhalb des geforderten Bereichs liegt. Ist dies der Fall, wird der neue Block zur Blockchain hinzugefügt.

Nach Ferdous et al. [1] gibt es drei Haupttypen von Proof of Work Konsensmechanismen: *Compute-bound*, *Memory-bound* und *Chained*. Bei *Compute-bound* PoW liegt der Fokus auf der Verwendung von Prozessoreinheiten wie der Central Processing Unit (CPU) oder der Graphical Processing Unit (GPU). Je mehr Leistung die Prozessoreinheit aufweist, desto schneller kann die Rechenaufgabe gelöst werden. Um die Rechenzeit beträchtlich zu verbessern, können zudem anwendungsspezifische integrierte Schaltungen (ASICs) verwendet werden. Dadurch werden allerdings Allzweckcomputer beim Mining benachteiligt, wodurch es zu einer ungewollten Zentralisierung des Blockchains-Netzwerks kommen kann. Für eine höhere Resistenz gegen ASICs und einem Schutz gegen Zentralisierung wurden *Memory-bound* und *Chained* PoW vorgeschlagen. Während bei *Memory-bound* PoW die Lösung einer Aufgabe von der Leistungsfähigkeit des Arbeitsspeichers abhängt, verkettet *Chained* PoW mehrere aneinandergereihte Hashfunktionen.

B. Bitcoin

Für die Erläuterung des in Bitcoin eingesetzten *Compute-bound* PoW Konsensprotokolls ist es notwendig, die im Kapitel II erwähnte Datenstruktur zu vertiefen. Die Abbildung 4 zeigt alle Daten, die in einem Bitcoin-Block enthalten sind. Der Blockheader enthält eine Versionsnummer, einen Zeitstempel, die Merkle-Root, den Hash-Wert des vorherigen Blocks, die Nonce und das Target. Der Blockkörper speichert sowohl eine Liste von Transaktionen als auch deren Anzahl [13].

Die Versionsnummer (*nVersion*) stellt die Blockformatversion dar und soll sicherstellen, dass alle Teilnehmer das gleiche Datenformat verwenden. Blöcke, welche nicht der aktuellen Blockformatversion entsprechen, werden von dem Netzwerk nicht akzeptiert. Der Zeitstempel (*nTime*) dokumentiert die Erstellungszeit des Blocks und spielt vor allem bei der Festlegung des Schwierigkeitsgrads, dem Target, eine Rolle [1]. Alle 2016 Blöcke wird mithilfe des Zeitstempels die durchschnittliche Zeit berechnet, die zum Minen eines Blocks benötigt wird. Wenn diese Zeit unter 10 Minuten liegt, wird die Schwierigkeit erhöht, um den Prozess des Minings zu verlangsamen. Wenn es im Gegensatz dazu länger als 10 Minuten dauert, wird die Schwierigkeit verringert. Zwei weitere Bestandteile des Blockheaders sind die Merkle-Root (*HashMerkleRoot*) und der Hash des vorherigen Blocks (*HashPrevBlock*). Die Merkle-

Field name	Type (Size)
nVersion	int (4 bytes)
HashPrevBlock	uint256 (32 bytes)
HashMerkleRoot	uint256 (32 bytes)
nTime	unsigned int (4 bytes)
nBits	unsigned int (4 bytes)
nNonce	unsigned int (4 bytes)
#vtx	VarInt (1-9 bytes)
vtx[]	Transaction (Variable)

Abbildung 4. Bitcoin Datenstruktur [13]

Root ist ein Hash, der aus den Hashes aller im Block enthaltenen Transaktionen erstellt wird. Der Hash-Wert des vorherigen Blocks bildet die Verbindung zwischen den einzelnen Blöcken und stellt sicher, dass sie in einer festgelegten Reihenfolge angeordnet sind. Der Hash-Wert eines Blocks wird mithilfe einer doppelten SHA-256 Funktion wie folgt berechnet [13]:

$$SHA256^2(nVersion||HashPrevBlock||HashMerkleRoot||nTime||nBits||nNonce) \quad (1)$$

Da sich der Hashwert eines Blocks aus der Kombination aller im Blockheader enthaltenen Daten ergibt, wird er auch von der Nonce (*nNonce*) beeinflusst. Die ressourcenintensive Rechenaufgabe, die Miner zur Erzeugung eines neuen Blocks lösen müssen, besteht bei Bitcoin darin, den korrekten Wert der Nonce zu ermitteln. Hierbei probieren Miner unterschiedliche Werte aus, bis der Hash-Wert des neu generierten Blocks innerhalb des vorgegebenen Bereichs liegt. Dieser Bereich wird durch das Target (*nBits*) des vorherigen Blocks definiert, das in Form einer enkodierten achtstelligen Hexadezimalzahl angegeben ist. Um das Target aus der enkodierten Form $0xh_0h_1h_2h_3h_4h_5h_6h_7$ abzuleiten, wird folgende Formel herangezogen [13]:

$$0xh_2h_3h_4h_5h_6h_7 \times 2^{8\times(0xh_0h_1-3)} \quad (2)$$

Die Nonce muss so gewählt werden, dass der hexadezimale Wert von dem Hashwert des Blocks kleiner oder gleich dem Target ist. Durch dieses Prinzip gewährleistet die Bitcoin-Blockchain die Manipulationssicherheit der in dem Block enthaltenen Daten [1]. Die Änderung einer beliebigen Information innerhalb eines Blocks führt dazu, dass die Hashwerte von dem Block selbst und von allen nachfolgenden Blöcken nicht mehr im vorgegebenen Bereich liegen und somit als ungültig betrachtet werden. Dass auch nachfolgende Blöcke aufgrund einer Manipulation ungültig werden, liegt an dem in der Hashfunktion berücksichtigten *HashPrevBlock*. Wenn sich der *HashPrevBlock* ändert, ändert sich folglich auch der Hash des aktuellen Blocks, was wiederum den Hash des nächsten Blocks beeinflusst - es entsteht eine Kettenreaktion.

Sollte ein Angreifer versuchen, einen Block zu manipulieren und die veränderte Version der Blockchain im Netzwerk zu verbreiten, dann würden die anderen Miner durch das Ausführen der Hashfunktion auf die Manipulation aufmerksam werden. Aufgrund des ungültigen Hashwerts wird die modifizierte Blockchain nicht anerkannt und die Integrität des Netzwerks bleibt gewahrt.

Ein weiterer relevanter Aspekt ist, dass mehrere Knoten gleichzeitig einen neuen Block erzeugen und diesen im Blockchain-Netzwerk verbreiten können. Das führt dazu, dass mehrere Zweige in der Blockchain entstehen, was als Fork bezeichnet wird [1]. Um sicherzustellen, dass letztendlich nur ein Zweig bleibt und alle anderen verworfen werden, lässt das Bitcoin-Protokoll die Zweige wachsen. Sobald ein Zweig länger als die anderen ist, entscheiden sich alle Miner für den längsten Zweig und verwerfen die übrigen. Es gilt, dass nur die Miner des längsten Zweiges mit Bitcoins belohnt werden. Miner, welche neue Blöcke für einen verworfenen Zweig erzeugt haben, gehen leer aus. Nachdem der Fork auf diese Weise gelöst wurde, herrscht wieder ein Konsens im Netzwerk.

V. Proof of Stake

A. Grundprinzip

Das Prinzip des Proof of Stake (PoS) Konsensmechanismus, das erstmals 2012 in einem Blockchain-System zum Einsatz kam [23], unterscheidet sich grundlegend vom PoW-Ansatz. Bei Proof of Stake stehen die Knoten nicht in einem Wettbewerb zueinander, um die schnellste Lösung einer anspruchsvollen Rechenaufgabe zu ermitteln. Es findet stattdessen ein Auswahlverfahren statt, bei dem der Gewinner das Recht erhält, den nächsten Block zu erzeugen [1].

Um an dem Auswahlverfahren teilnehmen zu können, müssen die Knoten zum einen beweisen, dass sie einen bestimmten Betrag der Kryptowährung besitzen. Zum anderen müssen sie einen Teil dieses Betrags als Pfand, das auch als Stake bezeichnet wird, in einem Treuhandkonto hinterlegen. Ein Knoten, der in einer PoS-Blockchain seine Coins auf diese Weise bindet, wird als “Stakeholder“ bezeichnet [1]. Der Stakeholder, der das Auswahlverfahren gewinnt und den nächsten Block erzeugen darf, wird auf eine von zwei möglichen Arten belohnt. Bei der ersten Methode enthält jede Transaktion eine Gebühr, die als Anreiz dient, die Transaktion in den Block aufzunehmen. Der Stakeholder darf hierbei alle Gebühren einsammeln, die sich in den Transaktionen innerhalb seines Blocks befinden. Bei der zweiten Methode bekommt der Stakeholder eine Belohnung, welche eine Zinszahlung des von ihm hinterlegten Betrags repräsentiert.

Nach Shifferaw und Lemma [24] gibt es beim Auswahlverfahren zwei zentrale Ansätze: *Randomized block selection* und *Coin age selection*. Die *Randomized block selection* verwendet eine Formel, die bei der Auswahl des Stakeholders sowohl die Größe des Stakes als auch zusätzliche Parameter berücksichtigt, wodurch eine gewichtete Zufallsauswahl stattfindet. Bei der *Coin age selection* wird ein Stakeholder ausgewählt, der das höchste *Coin Age* aufweist. Dieser Wert

errechnet sich aus der Multiplikation der Menge der gestakten Coins und der Zeit in Tagen, die diese Coins bereits gestaked sind. Zur Vermeidung der Dominanz einzelner Stakeholder wird das *Coin Age* nach dem Erstellen eines Blocks zurückgesetzt.

Bei Proof of Stake werden die Erzeuger eines neuen Blocks i.d.R. Validator und nicht Miner genannt [1]. Das liegt daran, dass der Schwerpunkt bei der Blockerstellung primär auf der Validierung der in dem Block enthaltenen Transaktionen liegt. Während diese Aufgabe auch in PoW-Blockchains erledigt wird, ist sie hinsichtlich des Ressourcenverbrauchs von geringerer Bedeutung als das Erraten der richtigen Nonce. Die Validierung einer Transaktion umfasst u.a. die Kontrolle der Signatur [25]. Wenn ein Sender eine Transaktion tätigen will, erzeugt er einen Hash der Transaktion und verschlüsselt diesen mithilfe seines privaten Schlüssels. Dieser verschlüsselte Hashwert stellt die Signatur des Senders dar. Der Validator entschlüsselt diese Signatur mittels des öffentlichen Schlüssels des Senders und vergleicht sie mit dem von ihm selbst berechneten Hash der Transaktion. Stimmen beide Hashwerte überein, kann davon ausgegangen werden, dass die Transaktion unverändert ist.

Wenn ein betrügerischer Validator einen Block mit einer manipulierten Transaktion verbreitet, dann können andere Validatoren durch eine eigenständige Validierung der Transaktionen diese Manipulation erkennen. In einem solchem Fall würde der Betrüger als *bad* gekennzeichnet werden und dadurch seine gestakten Coins verlieren [24]. Der Validator, der den Betrug erkannt hat, bekommt diese Coins übertragen. Aufgrund dieses Prinzips spielt die Anzahl der gestakten Coins sowohl bei der *Randomized block selection* als auch bei der *Coin age selection* eine zentrale Rolle. Je größer der Stake, desto größer der potenzielle Verlust für einen Betrüger, was die Sicherheit des Netzwerks erhöht. Der Stakeholder verliert darüber hinaus nicht nur seine Coins, sondern auch das Recht, zukünftig am Auswahlverfahren teilzunehmen.

Wird ein Block von anderen Knoten erfolgreich validiert und in die Blockchain aufgenommen, dann erhält der Erzeuger des Blocks eine Belohnung gemäß einem der beiden genannten Belohnungsmechanismen. Im Vergleich zu PoW-Blockchains fällt die Belohnung allerdings geringer aus, da PoS ressourcenschonender als PoW ist [1].

B. Cosmos Hub

Cosmos [26] stellt ein vernetztes Blockchain-Ökosystem dar, in dem mehrere unabhängige Blockchains integriert sind. Das Ökosystem hat das Ziel, die Interoperabilität zwischen den integrierten Blockchains zu ermöglichen, damit sie auf sichere und effiziente Weise Informationen austauschen können. Die Blockchains, welche auch als Zonen bezeichnet werden, kommunizieren dafür mit dem sogenannten Cosmos Hub über das inter-blockchain communication (IBC) Protokoll (siehe Abbildung 5).

Der Cosmos Hub dient als Knotenpunkt für den Austausch der Informationen und wird als öffentliche Blockchain betrieben. Der Konsens wird durch das PoS-Konsensprotokoll

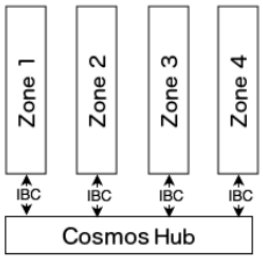


Abbildung 5. Cosmos Hub [26]

Tendermint erreicht, welches von dem Programm *Tendermint Core* zur Verfügung gestellt wird.

Um die Generierung eines Blocks im Rahmen des Tendermint-Konsensprotokolls zu ermöglichen, ist es zunächst erforderlich, eine bestimmte Anzahl von Validatoren auszuwählen. Diese ausgewählten Validatoren werden als “aktive Validatoren“ bezeichnet, da sie zur Teilnahme am Konsensprozess berechtigt sind. Im Cosmos Hub sind derzeit 175 aktive Validatoren vorgesehen [6].

Die Auswahl der aktiven Validatoren basiert auf der Menge der gestakten ATOM Coins, welche die native Währungseinheit des Cosmos Hub darstellen [6]. Der Stake eines Validators setzt sich dabei nicht nur aus den von ihm selbst gestakten Coins zusammen. Andere Knoten, auch bekannt als Delegatoren, können ihre Coins an Validatoren delegieren und dadurch deren Stake erhöhen. Im Gegenzug erhalten die Delegatoren einen Anteil an der Belohnung, die dem Validator zusteht. Sie tragen dabei allerdings auch ein Risiko: Wenn sich ein Validator nicht gemäß dem Konsensprotokoll verhält, dann werden ihm und den zugeordneten Delegatoren Coins entzogen. Aus diesem Grund prüfen Delegatoren im Vorfeld die Vertrauenswürdigkeit und Zuverlässigkeit der Validatoren. Ein Validator behält seinen Status als aktiver Validator bei, solange er mithilfe seiner eigenen und den ihm delegierten Coins zu den 175 Stakeholdern mit dem höchsten Stake im Netzwerk gehört.

Die Abbildung 6 gibt einen Überblick, wie mithilfe der aktiven Validatoren ein Konsens im Blockchain-Netzwerk erreicht werden kann. Die Erzeugung, Validierung und Bestätigung eines neuen Blocks findet bei Tendermint in Runden statt [28]. Bei jeder Runde wird anhand des Round-Robin Verfahrens aus den aktiven Validatoren ein sogenannter Proposer ausgewählt. Bei dem in der Abbildung dargestellten Propose-Schritt ist er dafür zuständig, einen neuen Block zu erzeugen und diesen den aktiven Validatoren “vorzuschlagen“.

Im Prevote-Schritt sendet jeder aktive Validator, der einen vorgeschlagenen Block erfolgreich validiert hat, mittels einer Broadcast-Nachricht einen signierten *Prevote* an alle anderen aktiven Validatoren. Erhält ein Validator keinen Block oder kann diesen nicht erfolgreich validieren, sendet er stattdessen einen speziellen signierten *Nil-Prevote*.

Der nachfolgende Precommit-Schritt richtet sich nach den erhaltenen Prevote-Nachrichten. Hat ein Validator Prevote-Nachrichten von mehr als 2/3 der Validatoren erhalten, sendet

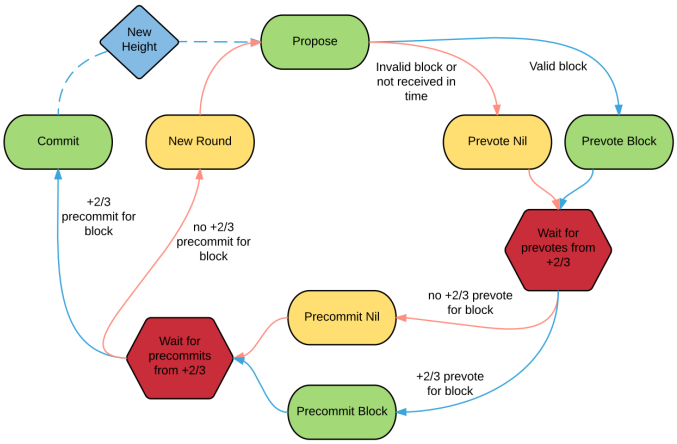


Abbildung 6. Tendermint Konsensprotokoll [27]

er mittels einer Broadcast-Nachricht einen signierten *Precommit*. Bei dem Erhalt von Prevote-Nachrichten von weniger als 2/3 der Validatoren sendet er stattdessen einen *Nil-Precommit*.

Sollte ein Validator Precommit-Nachrichten von weniger als 2/3 der Validatoren erhalten, beginnt für die gleiche *Block Height* eine neue Runde mit einem anderen Proposer. Unter *Block Height* ist die Position in der Blockchain zu verstehen, an der ein neu erzeugter Block hinzugefügt werden soll. Wenn ein Validator hingegen Precommit-Nachrichten von mehr als 2/3 der Validatoren erhält, dann geht er in den finalen Commit-Schritt über. Hierbei sendet er mittels einer Broadcast-Nachricht einen signierten *Commit*.

Erhält ein Validator nun Commit-Nachrichten von mehr als 2/3 der Validatoren, setzt er den Zeitstempel seines Commits auf die aktuelle Zeit und verbleibt für einen festgelegten Zeitraum inaktiv. Das soll es langsameren Validatoren ermöglichen, den Block in die Blockchain aufzunehmen und den *Commit* ebenfalls mittels einer Broadcast-Nachricht zu senden. Nach dieser Phase der Inaktivität startet für die nächste *Block Height* eine neue Runde des Konsensprozesses.

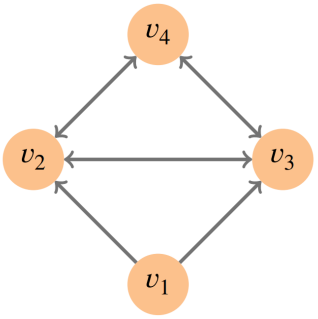
VI. Federated Byzantine Agreement

A. Grundprinzip

Das Federated Byzantine Agreement (FBA) wurde erstmals 2015 von David Mazières zusammen mit dem Stellar Consensus Protocol (SCP) vorgeschlagen [29]. Bei dem Stellar Consensus Protocol handelt es sich um eine konkrete Implementierung des FBA, die derzeit u.a. in den öffentlichen Blockchains Stellar und MobileCoin zum Einsatz kommt. FBA ist im Gegensatz zu PoW und PoS kein beweisbasierter, sondern ein abstimmungsbasierter Konsensmechanismus. Wie im Kapitel III erwähnt, wird bei dieser Kategorie der Konsens auf Grundlage von Stimmen erreicht.

Traditionelle abstimmungsbasierte Konsensmechanismen wie Raft sind grundsätzlich ungeeignet für öffentliche Netzwerke. Das hat den Grund, dass in einem solchen Netzwerk die Vertrauenswürdigkeit der teilnehmenden Knoten nicht sichergestellt werden kann, wodurch eine Abstimmung über alle Knoten hinweg nicht umsetzbar ist. Das FBA [7] führt deshalb das Konzept der sogenannten Quorum-Slices ein. Ein Quorum-Slice repräsentiert die Menge an Knoten, die benötigt wird, um einen einzelnen Knoten von einer Aussage zu überzeugen. Eine typische Aussage könnte zum Beispiel "Commit transaction set X" sein. Ist ein Knoten von dieser Aussage überzeugt, nimmt er die Transaktionsmenge X in seine Blockchain auf. Es gilt, dass jeder Knoten im Netzwerk selbst entscheiden darf, welchen Knoten er vertraut und zu seinem Quorum-Slice hinzufügt. Ein Knoten kann zudem mehrere Quorum-Slices mit unterschiedlichen Kombinationen von vertrauenswürdigen Knoten pflegen, um im Falle von Knotenausfällen trotzdem noch von einer Aussage überzeugt werden zu können.

Quorum-Slices sind Teilmengen eines Quorums. Unter einem Quorum ist die Menge an Knoten zu verstehen, die für die Erreichung eines Konsenses benötigt wird. Die Abbildung 7 veranschaulicht die Beziehung zwischen mehreren Quorum-Slices und einem Quorum.



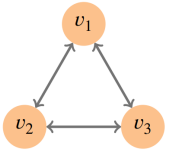
$\mathbf{Q}(v_1) = \{\{v_1, v_2, v_3\}\}$
$\mathbf{Q}(v_2) = \mathbf{Q}(v_3) = \mathbf{Q}(v_4) = \{\{v_2, v_3, v_4\}\}$

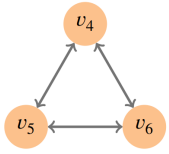
Abbildung 7. Quorum und Quorum-Slices [7]

Der Knoten *v1* hat zu seinem Quorum-Slice Q die Knoten *v1*, *v2* und *v3* hinzugefügt. Wenn diese Knoten einer Aussage zustimmen, dann wird *v1* ihr ebenfalls zustimmen. Die Quorum-Slices von *v2* und *v3* enthalten allerdings *v4*, was bedeutet, dass *v2* und *v3* ohne dem Mitwirken von *v4* nicht selbst von einer Aussage überzeugt werden können. Um einen Konsens zu erreichen, ist deshalb die Menge $\{v1, v2, v3, v4\}$ erforderlich. Diese Menge repräsentiert das Quorum.

Ein auf FBA basiertes Protokoll kann nur einen Konsens gewährleisten, wenn eine sogenannte *Quorum Intersection* vorhanden ist. Darunter ist zu verstehen, dass einzelne Quoren in mindestens einem Knoten überlappen. Diese Überlappungen verhindern die Bildung von sogenannten *Disjoint Quorums*, bei denen sich einzelne Quoren auf unterschiedliche Aussagen einigen können und somit eine Konsensbildung über das gesamte Netzwerk nicht möglich wird (siehe Abbildung 8).

$\mathbf{Q}(v_1) = \mathbf{Q}(v_2) = \mathbf{Q}(v_3) = \{\{v_1, v_2, v_3\}\}$





$\mathbf{Q}(v_4) = \mathbf{Q}(v_5) = \mathbf{Q}(v_6) = \{\{v_4, v_5, v_6\}\}$

Abbildung 8. Disjoint Quorums [7]

Bei FBA erfolgt die Konsensbildung hinsichtlich einer Aussage in zwei Schritten. Im ersten Schritt geben die Knoten ihre Stimmen für eine bestimmte Aussage ab. Sollte die Abstimmung erfolgreich sein, bestätigen die Knoten diese Aussage im zweiten Schritt. Aus der Sicht eines einzelnen Knoten lässt sich die Erreichung eines Konsenses in drei Phasen unterteilen: *unknown*, *accepted* und *confirmed*.

Zu Beginn befinden sich die Knoten in der Phase *unknown*, da noch keine Annahme zum Status der Aussage getroffen werden kann. Sie könnte am Ende des Konsensprozesses den Zustand wahr, falsch oder sogar "stuck" annehmen. Der Zustand stuck wird z.B. erreicht, wenn in einem Quorum die eine Hälfte der Knoten für und die andere Hälfte gegen eine Aussage stimmt und somit eine Pattsituation entsteht.

Ein Knoten geht erst von der Phase *unknown* in die Phase *accepted* über, wenn die Abstimmung, also der erste Schritt, erfolgreich war. Das Akzeptieren einer Aussage bedeutet allerdings noch nicht, dass sie als wahr angesehen wird und der Knoten auf Grundlage dieser Aussage handeln darf. Das liegt u.a. daran, dass die Aussage noch den Zustand stuck annehmen kann, wodurch ein vorzeitiges Handeln eine Inkonsistenz im Netzwerk verursachen würde. Um das Problem zu lösen, existiert der oben erwähnte zweite Schritt, bei denen die Knoten in die Phase *confirmed* übergehen. Hier wird die Aussage nun als wahr angesehen und die Knoten dürfen auf ihrer Grundlage handeln.

Es ist zu betonen, dass es sich bei der geschilderten Vorgehensweise nur um die Konsensbildung hinsichtlich einer einzigen Aussage handelt. Im Falle von Stellar müssen mehrere Aussagen durchlaufen werden, um einen einzigen Block zur Blockchain hinzuzufügen.

B. Stellar

Bei Stellar [30] handelt es sich um ein im Jahr 2014 gegründetes öffentliches Blockchain-Netzwerk, das von Unternehmen und Privatpersonen genutzt werden kann, um länderübergreifende Zahlungen abzuwickeln. Stellar ermöglicht die Umwandlung von Werten jeglicher Art, darunter Fiatwährungen wie Euro, Kryptowährungen wie Bitcoin oder auch tokenisierte Vermögenswerte wie Immobilien. Die native Währungseinheit Lumen wird in diesem Zusammenhang als Transfermittel verwendet. Wenn eine Person eine Transaktion tätigt, dann wird die Ausgangswährung zunächst in Lumen umgewandelt. Die Transaktion wird anschließend von der Blockchain verarbeitet und schließlich beim Empfänger in die Zielwährung umgemünzt. Neben dem Versenden und der Umwandlung von bereits bestehenden Werten ermöglicht Stellar zudem die Erstellung eigener Tokens. Unter einem Token ist die digitale Repräsentation eines beliebigen Vermögenswertes zu verstehen [31].

Das von Stellar eingesetzte SCP [7] beruht auf zwei Subprotokollen: dem Nominierungs- und dem Abstimmungsprotokoll. Im Rahmen des Nominierungsprotokolls schlagen die teilnehmenden Knoten verschiedene Transaktionsmengen, sogenannte "transaction sets", vor. Das Vorschlagen bzw. das Nominieren einer Transaktionsmenge *X* erfolgt hierbei über die Aussage *Nominate transaction set X*.

Wenn die Abstimmung hinsichtlich der Aussage erfolgreich verläuft und die Knoten sie als wahr ansehen, dann wird die Transaktionsmenge *X* in die Liste der *candidate values* aufgenommen. Diese Liste repräsentiert Kandidaten für das Abstimmungsprotokoll. Der *candidate value*, der die größte Anzahl an Transaktionen aufweist, wird als *composite value* bezeichnet und dem Abstimmungsprotokoll übergeben.

Das Abstimmungsprotokoll zielt nun darauf ab, den übergebenden *composite value* in die Blockchain zu integrieren. Um das zu erreichen, muss für folgende zwei Aussagen eine erfolgreiche Abstimmung erfolgen: *Prepare transaction set X* und *Commit transaction set X*. Die Prepare-Aussage verifiziert, dass jeder Knoten die gleiche Transaktionsmenge ausgewählt hat und für dessen Aufnahme in die Blockchain bereit ist. Die Commit-Aussage gewährleistet anschließend, dass der *composite value* tatsächlich aufgenommen wird.

Der beschriebene Prozess, der die Nominierung, Abstimmung und Integration in die Blockchain umfasst, wiederholt sich in Stellar alle drei bis fünf Sekunden [30].

VII. Leistung

In diesem Kapitel werden die Konsensprotokolle Nakamoto-PoW, Tendermint und SCP anhand folgender Leistungseigenschaften verglichen: Fehlertoleranz, Durchsatz und Energieverbrauch. Unter Nakamoto-PoW ist hierbei das in Bitcoin eingesetzte Konsensprotokoll zu verstehen, da es von Satoshi Nakamoto in dem Whitepaper "Bitcoin: A Peer-to-Peer Electronic Cash System" [2] erstmals vorgeschlagen wurde.

Nach Ferdous et al. [1] gibt die Fehlertoleranz die maximale Anzahl fehlerhafter Knoten an, die das in dem Netzwerk verwendete Konsensprotokoll tolerieren kann. Für den Nakamoto-PoW kann nach dieser Definition keine konkrete Grenze fehlerhafter Knoten angeben werden. Das liegt daran, dass die Fehlertoleranz in diesem Fall nicht von der Anzahl der Knoten, sondern von der Rechenleistung abhängt. Laut Bach et al. [32] ist sie bei PoW-Konsensmechanismen nämlich nur gewährleistet, wenn mehr als 75 Prozent der Gesamtrechenleistung des Netzwerks aus sich korrekt verhaltenen Knoten besteht.

Tendermint, das in Cosmos Hub eingesetzte Konsensprotokoll, weist hingegen eine Fehlertoleranz von $3f+1$ auf [27]. Die Variable f stellt hierbei die maximale Anzahl fehlerhafter Validatoren dar, die toleriert werden können. Ein Netzwerk, das beispielsweise den Ausfall von bis zu drei Validatoren tolerieren soll, muss gemäß der Formel aus mindestens zehn Validatoren bestehen. Das ist auch anhand des im Kapitel V beschriebenen Konsensprozesses ersichtlich. Um einen Konsens zu erreichen, müssen mindestens 2/3 der Validatoren jeweils einen Prevote, Precommit und Commit durchgeführt haben. Nach Florian et al. [29] weist das Stellar Consensus Protocol ebenfalls eine Fehlertoleranz von $3f+1$ auf.

Es ist zu erwähnen, dass es sich bei der Fehlertoleranz um *Byzantine fault tolerance* und nicht um *Crash fault tolerance* handelt (siehe Kapitel III). Unter einem fehlerhaften Knoten

ist also auch zu verstehen, dass sich dieser unerwartet oder bösartig verhält, und nicht nur z.B. aufgrund eines Software-Bugs keine Nachrichten mehr sendet.

Der Durchsatz bezieht sich auf die Anzahl der Transaktionen, die das Protokoll in einer Sekunde verarbeiten kann [1]. Für einen Vergleich wird typischerweise die Metrik *Transaktionen pro Sekunde (tps)* herangezogen. Bei Nakamoto's PoW bestimmt sich der Durchsatz primär anhand zwei festgelegter Parameter: der Größe eines Blocks und dem Intervall, in dem neue Blöcke erzeugt werden [33]. Die Blockgröße kann bei Bitcoin zwischen zwei und vier Megabyte liegen und gibt indirekt die Anzahl der Transaktionen vor, die sich einem Block befinden dürfen. Das Intervall für das Erzeugen eines neuen Blocks pendelt sich durch den anpassbaren Schwierigkeitsgrad bei 10 Minuten ein (siehe Kapitel IV). In dem Zeitraum vom 01.05.2023 bis 01.06.2023 wurde mit diesen Parameterwerten ein durchschnittlicher Durchsatz von etwa *6 tps* erreicht [34].

Decker und Wattenhofer liefern in ihrer Arbeit "Information propagation in the bitcoin network" [35] mehrere Gründe, warum sich eine Veränderung der Parameterwerte zur Verbesserung des Durchsatzes als schwierig gestaltet. Nach ihnen haben beide Aspekte einen wesentlichen Einfluss auf die in einem Blockchain-Netzwerk auftretenden Forks. Diese können u.a. Sicherheitsprobleme wie das Double Spending hervorrufen. Unter Double Spending ist der Versuch eines Benutzers zu verstehen, dieselben Coins in zwei oder mehr Transaktionen gleichzeitig zu übertragen. Eine von Gervais et al. [33] durchgeführte Simulation zeigt allerdings auf, dass der Durchsatz auf bis zu *70 tps* erhöht werden kann, bevor solche Probleme zum Tragen kommen. Eine realisierbare Kombination wäre beispielsweise eine Blockgröße von einem Megabyte und ein Blockintervall von einer Minute.

Bei Tendermint hängt der Durchsatz hauptsächlich von der Anzahl der aktiven Validatoren ab [26]. Der Grund dafür ist, dass alle Validatoren miteinander kommunizieren müssen, um eine Zwei-Drittel-Mehrheit zu erreichen. Der Cosmos Hub hat derzeit noch nicht die maximale Kapazität an Transaktionen erreicht, die theoretisch mit Tendermint erreichbar sind. Für den Durchsatzvergleich wird deshalb die im Cosmos-Whitepaper [26] durchgeführte Simulation herangezogen. Bei 64 Validatoren, die auf sieben Rechenzentren in fünf Kontinenten verteilt waren, wurde ein maximaler Durchsatz von *4000 tps* erzielt. Zur Steigerung der Sicherheit ist bei Cosmos Hub die Anzahl der aktiven Validatoren inzwischen auf 175 angestiegen, weshalb ein aktuellerer Durchsatz durch die von Cason et al. [36] durchgeführte Simulation gegeben ist. Die Simulation ergab, dass bei 175 aktiven Validatoren immer noch mehrere hundert tps erreicht werden. Die Stellar-Blockchain ist wie der Cosmos Hub nicht voll ausgelastet. Zur Beurteilung des Durchsatzes wird daher auf die Arbeit von Gorkey et al. [37] Bezug genommen. Laut ihren Angaben kann Stellar bei tausenden von Validatoren einen Durchsatz von bis zu *4000 tps* erreichen.

Das letzte Bewertungskriterium hinsichtlich der Leistung bildet der Energieverbrauch. Aktuellen Schätzungen zufolge beträgt dieser bei Bitcoin pro Transaktion 737,33 Kilowattstunden, was in etwa dem durchschnittlichen Energieverbrauch eines US-amerikanischen Haushalts über einen Zeitraum von 25 Tagen entspricht [38]. Ein anderer Vergleich unterstreicht diese erhebliche Energiezufuhr: Die Energie, die für eine einzige Bitcoin-Transaktion aufgewendet wird, könnte zur Verarbeitung von 496.000 VISA-Transaktionen genutzt werden. Darüber hinaus beläuft sich der geschätzte Gesamtenergieverbrauch von Bitcoin im Jahr 2021 auf 204 Terawattstunden. Dieser Wert entspricht etwa 180 Prozent gegenüber des im gleichen Jahr geschätzten Energieverbrauchs der Niederlande [39].

Der Energieverbrauch des Cosmos Hub ist deutlich geringer. Der Betrieb der Blockchain mit 125 Validatoren summiert sich auf einen geschätzten Jahresverbrauch von 466.470 Kilowattstunden [40]. Selbst bei einem Betrieb von 10.000 Cosmos Hub Blockchains würde der Gesamtenergieverbrauch somit lediglich 2,3 Prozent des Verbrauchs von Bitcoin erreichen. Eine ähnliche Effizienz zeigt das Stellar-Netzwerk. Laut dem Stellar Sustainability Report [41] beträgt der geschätzte Jahresverbrauch bei derzeit etwa 80 Validatoren 481.324 Kilowattstunden, wobei der Energieaufwand pro Transaktion auf nur 0.173 Wattstunden geschätzt wird. Im Vergleich dazu beträgt der Energieverbrauch von Visa pro Transaktion um die 1,5 Wattstunden [42], was etwa das 8,7-Fache des Energieverbrauchs von Stellar darstellt.

VIII. Sicherheit

In diesem Kapitel werden die Konsensprotokolle Nakamoto-PoW, Tendermint und SCP hinsichtlich ausgewählter Angriffsvektoren untersucht. Der erste betrachtete Angriffsvektor ist der von Douceur [43] beschriebene Sybil-Angriff. In einem öffentlichen verteilten System ohne eine zentrale Instanz kann nicht sichergestellt werden, dass eine Entität auch wirklich nur eine Identität besitzt. Es besteht die Möglichkeit, dass eine einzelne unbekannte Entität eine Vielzahl von Identitäten in Form von Knoten erzeugt. Ein Sybil-Angriff bezeichnet das Einschleusen dieser Mehrfachidentitäten in ein verteiltes System zur Erlangung unerlaubter Vorteile. Ein solcher Angriff kann bei einer Blockchain erfolgreich sein, wenn die Anzahl oder die Rechenleistung der Mehrfachidentitäten die zuvor behandelte Fehlertoleranz überschreitet [1]. Für den Nakamoto PoW bedeutet dies, dass die kumulierte Rechenleistung der Angreifer-Identitäten mehr als 25 Prozent der Gesamtrechenleistung des Netzwerks übersteigen muss. In Bezug auf Tendermint und SCP ist es erforderlich, dass die Angreifer-Identitäten mehr als ein Drittel der am Konsensprozess beteiligten Validatoren darstellen. Bei Tendermint muss deshalb jede einzelne Identität zu den Top-Stakeholdern im Netzwerk gehören. Im Falle von SCP ist ein erfolgreicher Sybil-Angriff nur dann möglich, wenn sich gutartige Validatoren dazu entscheiden, die Angreifer-Validatoren in ihre Quorum-Slices aufzunehmen. Ansonsten würden die bösartigen Validatoren zwar unter sich einen Konsens erreichen, dieser würde aber aufgrund von Disjoint Quorums von dem Rest des Netzwerks ignoriert werden.

Ein alternativer Angriffsvektor ist das Selfish-Mining [44]. In einer PoW-Blockchain tendieren Miner dazu, sich zu sogenannten Mining-Pools zusammenzuschließen. Innerhalb dieser Pools agieren Miner gemeinsam, was ihnen eine deutlich erhöhte Erfolgsrate bei der Blockerzeugung gegenüber einzelnen Minern ermöglicht. Die Belohnung wird entsprechend unter den Mitgliedern des Pools aufgeteilt. Selfish-Mining bezeichnet das opportunistische Verhalten von Mining-Pools, an einer privaten Blockchain zu arbeiten und diese zu einem späteren Zeitpunkt im Netzwerk zu propagieren. Auf diese Weise erhält der betroffene Mining-Pool Belohnungen, die unverhältnismäßig hoch zu der zur Verfügung gestellten Rechenleistung sind. Unter den drei untersuchten Konsensprotokollen ist lediglich der Nakamoto-PoW für ein Selfish-Mining-Angriff anfällig [1]. Er lässt sich verhindern, wenn die kumulative Rechenleistung des Mining-Pools nicht mehr als 25 Prozent der Gesamtrechenleistung des Netzwerks übersteigt [44].

Um den Angriff zu verdeutlichen, wird im Folgenden ein Selfish-Mining-Pool als “Alice“ und ein gutartiger Mining-Pool als “Bob“ bezeichnet. Zu Beginn arbeitet Alice genau wie Bob an der öffentlichen Blockchain. Wenn Alice einen neuen Block entdeckt, fügt sie diesen allerdings nicht der öffentlichen Blockchain hinzu und propagiert diesen an das Netzwerk, wie Bob es tun würde. Sie erstellt stattdessen über einen Fork eine private Blockchain, fügt dort den gefundenen Block hinzu und arbeitet daran weiter. Alice beobachtet gleichzeitig die öffentliche Blockchain. Sollte Bob einen neuen Block entdecken und diesen im Netzwerk propagieren, reagiert Alice umgehend und versucht ebenfalls den vorher entdeckten Block zu propagieren. In diesem Szenario besteht die Möglichkeit, dass sich Bob‘s Block im Netzwerk durchsetzt und Alice Rechenleistung verschwendet hat. Wenn Alice jedoch vor Bob einen zweiten neuen Block in Folge findet, dann hat sie sichergestellt, dass sie die Belohnung für das Mining erhält. Denn sobald Bob einen neuen Block entdeckt, propagiert Alice sicherheitshalber ihre private Blockchain im Netzwerk. Da die private Blockchain um einen Block länger ist, setzt sie sich gemäß dem Nakamoto-PoW gegenüber der kürzeren öffentlichen Blockchain durch und wird von den vertrauenswürdigen Knoten übernommen. Solange es Alice schafft, einen Vorsprung von zwei Blöcken zu halten, kann sie an der privaten Blockchain weiterarbeiten und erst dann an das Netzwerk propagieren, wenn der Vorsprung auf einen Block reduziert wird. Der erzielte Vorteil ist, dass Alice, abgesehen von dem oben erwähnten Szenario, keine Rechenleistung verschwendet. Ein gutartiger Miner wie Bob verwendet viel Rechenleistung, um einen Block zu finden. Wenn es jemand anderem zuerst gelingt, muss er das Mining abbrechen und für die nächste Block Height von vorne beginnen. Solange Alice jedoch Selfish-Mining betreibt und einen Vorsprung von zwei Blöcken aufweist, resultiert ihre aufgewendete Rechenleistung mit hoher Wahrscheinlichkeit in einem neuen vom Netzwerk akzeptierten Block.

Ein weiterer untersuchter Angriffsvektor ist das Nothing-at-Stake Problem [45], das bei PoS-Konsensprotokollen durch Forks auftreten kann. Bei PoW-basierten Blockchains ist das Auftreten von Forks ein häufiges und erwartetes Ereignis. Netzwerke wie Bitcoin sind so konzipiert, dass sich letztendlich die längste Kette durchsetzt. Im Gegensatz dazu gelten Forks bei PoS-Blockchains als Indikatoren für Netzwerk- oder Knotenprobleme. Sie können beispielsweise dann entstehen, wenn aufgrund eines Netzwerkausfalls ein Teil des Netzwerks isoliert wird und dieser seine eigene Version der Blockchain weiterentwickelt.

Das Erzeugen eines Blocks umfasst bei PoS-Blockchains nicht das ressourcenintensive Mining, sondern primär das Validieren von Transaktionen. Sollte ein Netzwerkausfall einen Fork auslösen, können Validatoren somit ohne erhebliche zusätzliche Kosten auf jedem vorhandenen Zweig arbeiten. Im Falle eines Forks wird davon ausgegangen, dass die Validatoren dies aus Eigeninteresse auch tun werden werden. Sie wissen nicht, welcher Zweig sich letztendlich durchsetzen wird. Daher sammeln sie von allen Zweigen die Belohnungen, um unabhängig vom Ergebnis zu profitieren. Dieses Verhalten kann jedoch dazu führen, dass die Zweige nebeneinander bestehen bleiben und kein Konsens erreicht wird (siehe Abbildung 9).

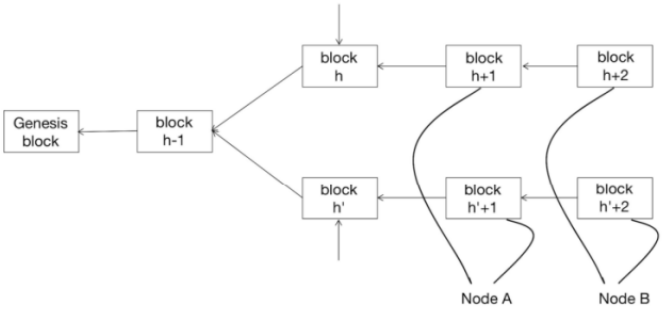


Abbildung 9. Nothing-at-Stake Problem [45]

Das in Cosmos Hub eingesetzte Konsensprotokoll Tendermint löst das Nothing-at-Stake Problem wie folgt: Es bestraft Validatoren, die auf mehreren Forks arbeiten [26]. Unter Bestrafung ist in diesem Fall zu verstehen, dass ein Teil des eingesetzten Stakes entzogen wird. Dies stellt einen starken finanziellen Anreiz für die Validatoren dar, sich an die Regeln des Protokolls zu halten.

Der letzte in dieser Studienarbeit untersuchte Angriffsvektor bezieht sich auf den von Kim et al. veröffentlichten Konferenzbeitrag “Is Stellar As Secure As You Think?” [46]. Die Autoren beschäftigen sich mit der von SCP verursachten dezentralen Struktur des Netzwerks und die Sicherheitsprobleme, die dadurch entstehen. Die Abbildung 10 zeigt die Quorum-Slices Struktur des Stellar-Netzwerks von Januar 2019. Jeder Kreis repräsentiert einen Knoten, wobei die Größe des Kreises proportional zu der Anzahl der eingehenden Kanten ist. Zur Veranschaulichung: Wenn der Knoten A zu seinem Quorum-Slice den Knoten B hinzufügt, dann existiert für B eine eingehende Kante. Nach Kim et al. kann die Anzahl der eingehenden Kanten Aufschluss über dessen Einfluss geben.

Sie ist nämlich proportional zu dem ihm entgegengebrachten Vertrauen. Der Darstellung zufolge tragen unter den 62 Validatoren die drei *sdf_validator* Knoten das meiste Vertrauen, gefolgt von *eno* und *tempo.eu.com*.

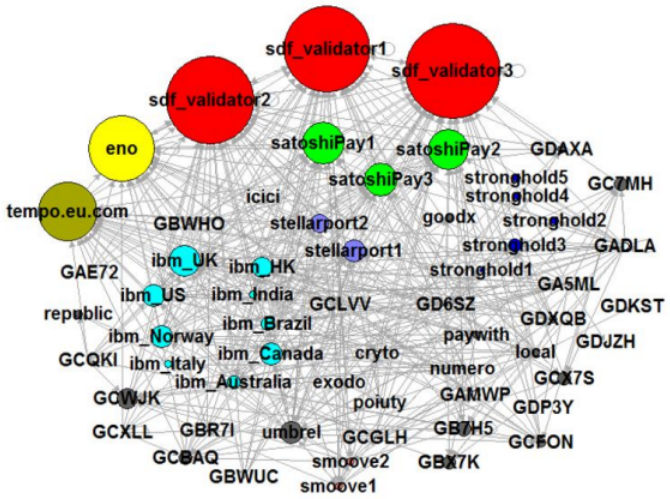


Abbildung 10. Struktur der Quorum-Slices [46]

Zur genauen Bestimmung des Einflusses eines Knotens müssen die folgenden drei Faktoren berücksichtigt werden [46]:

1) Die Anzahl der Quorum-Slices, die den Knoten enthalten.
2) Ob sich der Knoten in dem Quorum-Slice eines einflussreichen Knotens befindet.
3) Der Schwellenwert des Quorum-Slice, der den Knoten enthält.

Der Schwellenwert bezieht sich in diesem Kontext auf das Verhältnis zwischen der Anzahl der vertrauenswürdigen Knoten und der Anzahl der Knoten im Quorum-Slice [7]. Wie bereits im Kapitel VI erwähnt, kann ein Knoten mehrere Quorum-Slices mit unterschiedlichen Kombinationen von Knoten pflegen. Dies ermöglicht es, auch bei einem Knotenausfall von einer Aussage überzeugt werden zu können. Wenn ein Knoten beispielsweise insgesamt zehn vertrauenswürdige Knoten kennt und einen Quorum-Slice mit acht Knoten pflegt, dann beträgt der Schwellenwert 80 Prozent. Kim et al. zufolge gilt: Je höher der Schwellenwert des Quorum-Slice, desto größer ist der Einfluss des Knotens.

Um den Einfluss quantitativ zu erfassen, haben die Autoren zwei Metriken herangezogen: PageRank (PR) und NodeRank (NR). PR wurde erstmals 1998 von Sergey Brin und Lawrence Page vorgeschlagen und wird typischerweise für das Ranking von Webseiten in der Google-Suchmaschine verwendet [47]. NR ist eine von Kim et al. speziell entwickelte Metrik zur Messung des Einflusses von Knoten in Stellar. Während PR die Faktoren 1) und 2) berücksichtigt, erfasst NR alle drei Faktoren und bietet daher für diesen Zweck eine höhere Aussagekraft. Die folgende Abbildung zeigt den normalisierten PR und NR für die oben dargestellte Struktur der Quorum-Slices:

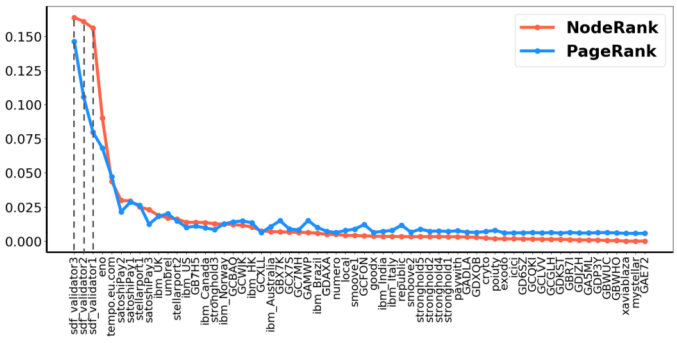


Abbildung 11. PageRank und NodeRank [46]

Wie zu erkennen ist, verfügen die *sdf_validator* Knoten im Vergleich zu den anderen Knoten über den weitaus höchsten NodeRank und haben folglich einen erheblichen Einfluss auf das Netzwerk. Aufgrund dieser Tatsache wird Stellar als ein Netzwerk mit hoher Zentralisierung angesehen, was hinsichtlich der Sicherheit mehrere Probleme hervorruft. Eines der kritischsten Probleme ist das sogenannte “Cascading Failure“ [46]. Es bezeichnet eine Situation, in der der Ausfall einiger Knoten einen schrittweisen Ausfall anderer Knoten auslöst. Nach der Analyse von Kim et al. würde beispielsweise der Ausfall der Knoten *sdf_validator 3* und *sdf_validator 2* zu einem Zusammenbruch des Netzwerks führen.

Die Autoren sehen die Ursache für die zentralisierte Struktur der Quorum-Slices darin, dass der Konsensprozess auf dem Vertrauen der Knoten basiert. In einem solchen Modell kommt es zur Voreingenommenheit: Die Knoten, die bereits hohes Vertrauen im Netzwerk genießen, werden den Quorum-Slices hinzugefügt. Bei der Analyse stellten Kim et al. daher fest, dass mindestens einer der drei *sdf_validator* Knoten in jedem bestehenden Quorum-Slice des Netzwerks enthalten ist.

IX. Fazit

In dieser Studienarbeit wurden die Konsensmechanismen Proof of Work, Proof of Stake und Federated Byzantine Agreement anhand der Protokolle Nakamoto-PoW, Tendermint und SCP analysiert. Im Fokus der Analyse stand die Funktionsweise sowie Aspekte der Leistung und Sicherheit.

Der in Bitcoin eingesetzte Nakamoto-PoW verwendet eine rechenintensive Aufgabe, um den Konsensprozess zu steuern. Die am Prozess beteiligten Knoten konkurrieren miteinander und versuchen, eine bestimmte Zahl zu finden, bei der der Hash-Wert eines neuen Blocks innerhalb eines vorgegebenen Lösungsraums liegt. Zur Vermeidung von Forks und dadurch entstehende Probleme wie dem Double Spending wird der Lösungsbereich so begrenzt, dass durchschnittlich nur alle 10 Minuten ein Block erstellt wird. Dieses Vorgehen wirkt sich erheblich auf den Durchsatz aus, der mit dem Nakamoto-PoW erreicht werden kann. Gemäß durchgeführten Simulationen beträgt der maximale Wert 70 Transaktionen pro Sekunde und liegt somit weit unter den mit Tendermint und SCP realisierbaren Durchsatz. Im Hinblick auf den Energieverbrauch

schneidet der Nakamoto-PoW ebenfalls schlecht ab. Aktuellen Schätzungen zufolge entspricht der Stromverbrauch einer Bitcoin-Transaktion dem durchschnittlichen Stromverbrauch eines US-amerikanischen Haushalts über einen Zeitraum von 25 Tagen.

Zwei zentrale Schwachstellen des Nakamoto-PoW sind die Anfälligkeit für Sybil-Angriffe und das Selfish-Mining. Während bei einem Sybil-Angriff ein Angreifer zahlreiche Identitäten in das Netzwerk einschleust, werden beim Selfish-Mining opportunistisch handelnde Mining-Pools gebildet. Beide Szenarien zielen darauf ab, die Konsensbildung zu Gunsten des Angreifers zu beeinflussen. Sie können allerdings erst dann Erfolg haben, wenn die kumulierte Rechenleistung der Angreifer-Identitäten bzw. der im Mining-Pool mitwirkenden Knoten mehr als 25 Prozent der Gesamtrechenleistung des Netzwerks übersteigt.

Bei Tendermint handelt es sich um ein in Cosmos Hub eingesetztes PoS-Konsensprotokoll, bei dem Validatoren Kryptowährung besitzen und einen Teil davon als Pfand hinterlegen. Bei Verstößen gegen die Regeln des Protokolls wird den Validatoren der Pfand, welcher auch als Stake bezeichnet wird, entzogen. Ein größerer Stake dient als effektives Abschreckungsmittel gegen Fehlverhalten, da ein potenzieller Verlust größer ist. Die Validatoren werden deshalb basierend auf der Größe ihres Stakes ausgewählt. Die Konsensbildung erfolgt bei Tendermint in Runden. In jeder Runde wird mittels des Round-Robin Verfahrens ein Validator ausgewählt, der einen neuen Block vorschlagen darf. Der vorgeschlagene Block wird in die Blockchain aufgenommen, sofern unter den Validatoren eine Zwei-Drittel-Mehrheit erreicht wird. Im Gegensatz zum Nakamoto-PoW entstehen auf diese Weise keine regelmäßigen Forks, was eine künstliche Verlangsamung des Blockintervalls überflüssig macht. Dies führt zu einem wesentlich höheren Durchsatz, der bei Cosmos Hub mehrere hundert Transaktionen pro Sekunde betragen kann. Da die Blockerzeugung lediglich das Validieren von Transaktionen und nicht das Lösen einer ressourcenintensiven Aufgabe umfasst, ist der Energieverbrauch zudem deutlich geringer als bei dem Nakamoto-PoW. Selbst bei einem Betrieb von 10.000 Cosmos-Hub Blockchains würde der Gesamtenergieverbrauch lediglich 2,3 Prozent des Verbrauchs von Bitcoin erreichen.

Genau wie bei dem Nakamoto-PoW besteht auch bei Tendermint eine potenzielle Anfälligkeit für Sybil-Angriffe. Ein solcher Angriff kann erfolgreich sein, wenn die Angreifer-Identitäten mehr als ein Drittel der am Konsensprozess beteiligten Validatoren repräsentieren. Dies würde bei Tendermint allerdings bedeuten, dass jede einzelne Identität zu den Top-Stakeholdern im Netzwerk gehören muss. Ein weiterer Angriffsvektor bei PoS-Konsensmechanismen ist das sogenannte Nothing-at-Stake Problem. Obwohl Forks bei dieser Art von Blockchain kein häufiges und erwartetes Ereignis darstellen, können sie aufgrund von z.B. Netzwerkproblemen auftreten. Das Nothing-at-Stake Problem bezieht sich auf die Annahme, dass Validatoren im Falle eines Forks aufgrund geringer Validierungskosten und finanziellem Eigeninteresse auf allen Zweigen arbeiten werden. Dies würde dazu führen, dass der Fork nicht aufgelöst wird und keine Konsensbildung stattfinden kann. Tendermint löst dieses Problem, indem es Validatoren bestraft, die auf mehreren Forks arbeiten.

Das Stellar Consensus Protocol ist eine Implementierung des Federated Byzantine Agreement und kommt u.a. bei der Stellar-Blockchain zum Einsatz. Bei FBA handelt es sich im Gegensatz zu PoW und PoS um keinen beweisbasierten, sondern um einen abstimmungsbasierten Konsensmechanismus. Die Abstimmung hinsichtlich eines vorgeschlagenen Blocks findet dabei über Quorum-Slices statt. Ein Quorum-Slice wird von den Knoten selbst verwaltet und repräsentiert die Menge an Knoten, die ihn von einer Aussage überzeugen können. Durch eine Überlappung von Quorum-Slices ist es möglich, im Netzwerk einen Konsens hinsichtlich einer Aussage zu erzielen. Für das Hinzufügen einer neuen Transaktionsmenge zur Blockchain müssen sich die Knoten gemäß dem Stellar Consensus Protocol bei einer festen Folge von Aussagen einig werden. Die Aussagen beziehen sich u.a. auf das Nominieren, Vorbereiten und Bestätigen der Transaktionsmenge. Der mit dieser Vorgehensweise erzielbare Durchsatz übertrifft deutlich den von Nakamoto-PoW und Tendermint. Bei tausenden von Validatoren können bis zu 4000 Transaktionen pro Sekunde erreicht werden. Der geschätzte Energieverbrauch, den Stellar mit SCP bei etwa 80 Validatoren verursacht, liegt bei 481.324 Kilowattstunden. Er ist somit vergleichbar mit den für den Cosmos Hub geschätzten 466.470 Kilowattstunden.

Das SCP weist zwar enorme Vorteile im Hinblick auf den Durchsatz und Energieverbrauch auf, der auf dem Vertrauen basierende Konsensprozess führt allerdings zu nicht unerheblichen Sicherheitsproblemen. Eine Analyse des Stellar-Netzwerks aus dem Jahr 2019 ergab, dass aufgrund der Voreingenommenheit von Knoten eine hochgradig zentralisierte Struktur von Quorum-Slices entstehen kann. Der große Einfluss von einigen wenigen Knoten kann zu einem Cascading Failure führen, bei der der Ausfall eines Knoten den schrittweisen Ausfall anderer Knoten verursacht. Im Falle der im Jahr 2019 analysierten Struktur würde ein zeitgleicher Ausfall der zwei einflussreichsten Knoten einen Zusammenbruch des Netzwerks verursachen.

LITERATUR

[1] Md Sadek Ferdous, Mohammad Jabed Morshed Chowdhury, Mohammad A. Hoque, and Alan Colman. Blockchain consensus algorithms: A survey. *arXiv preprint arXiv:2001.07091*, 2020.

[2] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized business review*, page 21260, 2008.

[3] J. Schütte, G. Fridgen, W. Prinz, T. Rose, N. Urbach, T. Hoeren, N. Guggenberger, C. Welzel, S. Holly, and A. Schulte. Blockchain - technologien, forschungsfragen und anwendungen. *Blockchain Positionspapier. Fraunhofer, München*, pages 1–39, 2017.

[4] Arati Baliga. Understanding blockchain consensus models. *Persistent*, 4(1):14, 2017.

[5] Bahareh Lashkari and Petr Musilek. A comprehensive review of blockchain consensus mechanisms. *IEEE Access*, 9:43620–43652, 2021.

[6] hub.cosmos.network. Cosmos hub, o. J.

[7] David Mazieres. The stellar consensus protocol: A federated model for internet-level consensus. *Stellar Development Foundation*, 32:1–45, 2015.

[8] Wei Yao, Junyi Ye, Renita Murimi, and Guiling Wang. A survey on consortium blockchain consensus mechanisms. *arXiv preprint arXiv:2102.12058*, 2021.

[9] Bikramaditya Singhal, Gautam Dhameja, and Priyansu Sekhar Panda. *Beginning Blockchain: A Beginner's guide to building Blockchain solutions*, volume 1. Springer, 2018.

[10] Xuemin Shen, Heather Yu, John Buford, and Mursalin Akon. *Handbook of peer-to-peer networking*, volume 34. Springer Science & Business Media, 2010.

[11] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. Blockbench: A framework for analyzing private blockchains. In *Proceedings of the 2017 ACM international conference on management of data*, pages 1085–1100, 2017.

[12] Robert Wilkens and Richard Falk. Smart contracts. *Aufl., Wiesbaden*, 2019.

[13] Krzysztof Okupski. Bitcoin developer reference. In *Eindhoven*. 2014.

[14] P. Paul, P. S. Aithal, and Ricardo Saavedra. Blockchain technology and its types—a short review. *International Journal of Applied Science and Engineering (IJASE)*, 9(2):189–200, 2021.

[15] M. Klein. Die blockchain-technologie: Potentziale und herausforderungen in den netzsektoren energie und telekommunikation, 2019.

[16] Fred B. Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys (CSUR)*, 22(4):299–319, 1990.

[17] Shyh-Wei Luan and Virgil D. Gligor. A fault-tolerant protocol for atomic broadcast. *IEEE Transactions on Parallel & Distributed Systems*, 1(03):271–285, 1990.

[18] Xiang Fu, Huaimin Wang, and Peichang Shi. A survey of blockchain consensus algorithms: mechanism, design and applications. *Science China Information Sciences*, 64:1–15, 2021.

[19] Michael J. Fischer, Nancy A. Lynch, and Michael S. Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)*, 32(2):374–382, 1985.

[20] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. In *Concurrency: the works of leslie lamport*, pages 203–226. 2019.

[21] Giang-Truong Nguyen and Kyungbaek Kim. A survey about consensus algorithms used in blockchain. *Journal of Information processing systems*, 14(1), 2018.

[22] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Advances in Cryptology—CRYPTO'92: 12th Annual International Cryptology Conference Santa Barbara, California, USA August 16–20, 1992 Proceedings 12*, pages 139–147, 1993.

[23] peercoin.net. Peercoin documentation, o. J.

[24] Yenatfanta Shifferaw and Surafel Lemma. Limitations of proof of stake algorithm in blockchain: A review. *Zede Journal*, 39(1):81–95, 2021.

[25] Andreas M. Antonopoulos. *Mastering Bitcoin: unlocking digital cryptocurrencies*. O'Reilly Media, Inc, 2014.

[26] Jae Kwon and Ethan Buchman. Cosmos whitepaper. *A Netw. Distrib. Ledgers*, page 27, 2019.

[27] Ethan Buchman. *Tendermint: Byzantine fault tolerance in the age of blockchains*. PhD thesis, University of Guelph, 2016.

[28] Jae Kwon. Tendermint: Consensus without mining. *Draft v. 0.6, fall*, 1(11), 2014.

[29] Martin Florian, Sebastian Henningsen, Charmaine Ndolo, and Björn Scheuermann. The sum of its parts: Analysis of federated byzantine agreement systems. *Distributed Computing*, 35(5):399–417, 2022.

[30] developers.stellar.org. Stellar, o. J.

[31] Lidia Kurt and Domenic Kurt. *Digitale Assets & Tokenisierung: Grundlagen umfassend verstehen*. Springer, 2022.

[32] Leo Maxim Bach, Branko Mihaljevic, and Mario Zagar. Comparative analysis of blockchain consensus algorithms. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1545–1550, 2018.

[33] Arthur Gervais, Ghassan O. Karame, Karl Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 3–16, 2016.

[34] blockchain.com. Bitoin - transaction rate per second, o. J.

[35] Christian Decker and Roger Wattenhofer. Information propagation in the bitcoin network. In *IEEE P2P 2013 Proceedings*, pages 1–10, 2013.

[36] Daniel Cason, Enrique Fynn, Nenad Milosevic, Zarko Milosevic, Ethan Buchman, and Fernando Pedone. The design, architecture and performance of the tendermint blockchain network. In *2021 40th International Symposium on Reliable Distributed Systems (SRDS)*, pages 23–33, 2021.

[37] Isitan Görkey, Chakir El Moussaoui, Vincent Wijdeveld, and Erik Sennema. Comparative study of byzantine fault tolerant consensus algorithms on permissioned blockchains. 2020.

[38] digiconomist.net. Bitcoin energy consumption index, o. J.

[39] enerdata.net. Netherlands energy information, o. J.

[40] blog.cosmos.network. Why blockchains need cosmos proof-of-stake for a sustainable environment, 2021.

[41] Anand S. Rao, Dan Dowling, Kurt Fields, Tarik Moussa, Jessica Wrigley, Alex Ferraro, Gabriel Blum, Maura Smith, and Tabea Stoeckel. Embracing sustainable blockchain innovation: Understanding the impacts of blockchain technology, 2022.

[42] Visa. Corporate responsibility & sustainability report, 2020.

[43] John R. Douceur. The sybil attack. In *Peer-to-Peer Systems: First InternationalWorkshop, IPTPS 2002 Cambridge, MA, USA, March 7–8, 2002 Revised Papers 1*, pages 251–260, 2002.

[44] Qianlan Bai, Xinyan Zhou, Xing Wang, Yuedong Xu, Xin Wang, and Qingsheng Kong. A deep dive into blockchain selfish mining. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pages 1–6, 2019.

[45] Shi Yan. Analysis on blockchain consensus mechanism based on proof of work and proof of stake. In *2022 International Conference on Data Analytics, Computing and Artificial Intelligence (ICDACAI)*, pages 464–467, 2022.

[46] Minjeong Kim, Yujin Kwon, and Yongdae Kim. Is stellar as secure as you think? In *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 377–385, 2019.

[47] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1):107–117, 1998.