HAW HAMBURG

BACHELOR THESIS
Peter Schrüfer

Untersuchung der Blockchain-Technologie für Zuweisungsalgorithmen

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Engineering and Computer Science
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Peter Schrüfer

Untersuchung der Blockchain-Technologie für Zuweisungsalgorithmen

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Angewandte Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Michael Köhler-Bußmeier
Zweitgutachter: Dr. Kilian Schwarz

Eingereicht am: 29. September 2025

Peter Schrüfer

Thema der Arbeit

Untersuchung der Blockchain-Technologie für Zuweisungsalgorithmen

Stichwörter

Blockchain, Aufgabenzuweisung, Smart Contract, Ethereum, Auktion

Kurzzusammenfassung

Diese Bachelorarbeit untersucht die Möglichkeit, Blockchain-Technologie im fundamentalen Problem der Aufgabenzuweisung verteilter Systeme einzusetzen. Der Prozess umfasst die Auswahl einer konkreten Aufgabenzuweisungslösung, deren Entwurf und Implementierung, eine Untersuchung von Möglichkeiten zur Integration von Blockchain-Technologie, den Entwurf und die Implementierung einer Blockchain-basierten Variante und einen anschließenden Vergleich beider Versionen mithilfe durchgeführter Messungen. Eine vergleichende Bewertung der beiden Architekturen zeigt, dass die Blockchain-Lösung durchweg teurer ist. Daraus wird geschlussfolgert, dass der Einsatz einer Blockchain-Lösung einen konkreten Mehrwert, wie erhöhte Sicherheit, rechtfertigen muss, um die höheren Kosten zu kompensieren. ...

Peter Schrüfer

Title of Thesis

Blockchain Technology in Task Allocation Algorithms

Keywords

blockchain, task allocation, smart contract, Ethereum, auction

Abstract

This bachelor's thesis investigates the potential of using blockchain technology for the fundamental distributed systems problem of task allocation. The process involved selecting a specific task allocation solution, designing and implementing it, exploring possibilities for

blockchain integration, and subsequently designing and implementing a blockchain-based variant. A final comparison of both versions was conducted using performed measurements. A comparative evaluation of the two architectures shows that the blockchain solution is consistently more expensive. It is concluded that the use of a blockchain must be justified by a concrete added value, such as increased security, to compensate for the higher costs. ...

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Die Blockchain-Technologie wurde erstmalig durch die Einführung von und die praktische Umsetzung in Bitcoin populär. Die disruptive Technologie diente ursprünglich als dezentrale Grundlage für Kryptowährung, aber ihre Möglichkeiten übersteigen die finanzielle Anwendung, wie sich über die Zeit gezeigt hat. Durch die Entwicklung von erweiterten Konzepten, wie Smart Contracts oder Non-Fungible Tokens (NFTs) werden dezentrale Anwendungen (dApps) in Bereichen wie der Lieferkettenverwaltung oder der Verwaltung geistigen Eigentums realisiert und genutzt.

Ein fundamentales Problem in vielen verteilten Systemen ist die optimale Zuweisung von Aufgaben (task allocation) unter geeigneten Agenten. Es gilt, ein Gesamtziel optimal zu erreichen bei Zuweisung von Arbeit und Ressourcen an die richtigen Agenten, was ein häufiges in verschiedenen Variationen auftretendes Problem darstellt. Daher stellt sich die zentrale Forschungsfrage dieser Arbeit: „Wie kann Blockchain-Technologie genutzt werden, um Aufgabenzuweisung zu unterstützen und welche spezifischen Aspekte sind dabei zu berücksichtigen?“.

Ziel dieser Bachelorarbeit ist es somit, diese Verbindung zu untersuchen. Es sollen Einblicke gewonnen werden, wie Blockchain-Konzepte, insbesondere Smart Contracts, die Aufgabenzuweisung ergänzen können. Dabei werden potenzielle Vorteile, wie erhöhte Transparenz, Nachverfolgbarkeit und Automatisierung, sowie Herausforderungen, wie Skalierbarkeit und Leistungsfähigkeit, aufgezeigt.

2 Technologieeinführung

2.1 Blockchain-Einführung

2.1.1 Blockchain

Eine Blockchain ist eine einfach rückwärts verkettete Liste aus Blöcken, die jeweils als Referenz zum direkten Vorgängerblock einen kryptografischen Hash des Blocks beinhalten [26]. Ein Block kann prinzipiell Beliebiges umfassen, konkret handelt es sich meist um Transaktionen oder Daten. Transaktionen können wirtschaftlicher Natur sein, oder aber auch Aufzeichnungen von Operationen [19].

Eine Blockchain-Technologie ist eine Distributed-Ledger-Technik (DLT) [**?**]. Bei einem Ledger in diesem Kontext handelt es sich um eine Sequenz von Datensätzen (Records) oder Transaktionen. In einer Blockchain-Technologie wird die Blockchain als Ledger verwendet. Dieser Blockchain-Ledger ist verteilt über ein Netzwerk, wobei verschiedene Netzwerkknoten jeweils ein Replika des Ledgers, also der Blockchain, halten. Das Blockchain-Netzwerk ist ein P2P-Netzwerk, das den Blockchain-Ledger nach einem je nach Blockchain-Technologie variierenden Protokoll verwaltet. Damit ein Blockchain-Ledger um einen weiteren Block erweitert werden kann, muss das Netzwerk einen Konsens über den nächsten Block finden. Falls es aufgrund der verteilten Natur dieser Ledger-Technik mehrere alternative Blockchains, sogenannte Blockchain-Forks, gibt, muss ebenfalls ein Konsens über die „wahre“ Blockchain (canonical blockchain) gefunden werden. Der Mechanismus, nach dem Knoten des Netzwerks sich auf den nächsten Block und die wahre Blockchain einigen, wird als Konsensmechanismus der Blockchain-Technologie bezeichnet. Eine Blockchain-Technologie ermöglicht aufgrund ihrer Bestandteile viele Vorteile, wie Dezentralisierung, Resilienz, Manipulationssicherheit, Integrität, Transparenz, Vertraulichkeit, Unabstreitbarkeit, Zurückverfolgbarkeit, Pseudonymität und Automatisierung.

2.1.2 Konsens

Proof-of-Work-Konsens In einem Proof-of-Work-Konsensmechanismus (PoW) darf derjenige Netzwerkknoten den nächsten Block der Blockchain vorschlagen, der einen bestimmten Arbeitsnachweis (proof of work) vorzeigen kann. Dies kann durch Lösung eines kryptografischen Puzzles erfolgen. Dabei ist es von Vorteil, wenn die Verifizierung eines Lösungsvorschlags effizienter ist, als die eigentliche Lösungsfindung des Puzzles. Die Netzwerkteilnehmer befinden sich somit in einem konstanten Wettrennen, um jeweils als Erste den Arbeitsnachweis des aktuellen Puzzles zu finden. Teilnehmer eines PoW-Blockchain-Netzwerks, die an diesem Wettrennen teilnehmen, werden als Miner, die Mining betreiben, bezeichnet. Ein offensichtlicher Nachteil dieses Mechanismus ist der ständige Verbrauch von Rechenkraft zum Mining. Allerdings sichert ein PoW-Mechanismus die Blockchain, indem Manipulationen der Blockchain entsprechend aufwändig werden. Aufgrund der Hash-Links der Blöcke müssen bei Manipulation eines Blockinhalts, entsprechend alle Hashes der Nachfolgerblöcke neuberechnet und diese neuen Blöcke erneut vorgeschlagen werden. Denn ändert sich ein Blockinhalt, so auch der Hash, der sich aus diesem Block errechnen lässt, und da dieser Hash ebenfalls Blockinhalt des nächsten Blocks ist, ändert sich dessen Blockinhalt und Hash ebenfalls etc. Sollte sich ein Knoten dazu entscheiden, eine alternative Blockchain zu kultivieren, muss dieser alle zu verändernden Blöcke neu minen, was ein deutliches Hindernis darstellen kann. Hinzu kommt, dass die längste Blockchain gewinnt (longest chain rule). Es müssen somit genug Knoten an dem Wachstum der manipulierten Blockchain arbeiten, damit diese die wahre Blockchain einholt und sich als wahrer Ledger etablieren kann. Solange also der Großteil der Rechenkraft (51%) nicht unter Kontrolle eines manipulierenden Akteurs steht, ist die Blockchain durch den Mechanismus manipulationssicher.

Proof-of-Stake-Konsens In einem Proof-of-Stake-Konsensmechanismus (PoS) können Netzwerkteilnehmer ihre Kryptowährung auf der Blockchain für das Protokoll einschließen (Staking). Der Netzwerkteilnehmer, der den nächsten Block vorschlagen darf, wird zufällig ausgewählt. Je größer der Anteil („Stake“) eines Netzwerkteilnehmers an abgegebener Kryptowährung, desto wahrscheinlicher ist es, dass dieser zum Blockvorschlag gewählt wird. Damit verbrauchen Knoten nicht Energie zum Minen und sichern dennoch die Blockchain, solange der Großteil (51%) der Stakes von ehrlichen Knoten stammt.

Weitere Konsensmechanismen Es gibt viele weitere Konsensmechanismen und Variationen, die erwähnt werden können:

- Proof Of Activity (PoA), welches hybrid aus PoW und PoS ist
- Proof Of Chain
- Proof Of Stake Time
- Proof Of Work Time
- Proof Of Space Time
- Proof Of Devotion
- and many more

2.1.3 Kryptowährung

Es muss einen Grund geben, wieso ein Akteur sich dazu entscheiden sollte, an einem Blockchain-Netzwerk teilzunehmen und ggf. zu minen. Diese Motivation kann durch Richtlinien, Geschäftsregeln oder akademische Natur gegeben sein, oder aber auch monetär. Viele Blockchains belohnen die Knoten, die durch Teilnahme an Konsensverfahren die Blockchain sichern, finanziell. Dafür kann für die Blockchain eine eigene Währung eingeführt werden, die ihren Wert durch die Sicherungsarbeit, wie Minen, erhält. Eine Kryptowährung, die in einer Blockchain-Technologie definiert wird, wird als native Kryptowährung oder nativer Token der Blockchain bezeichnet. Beispielsweise ist die native Kryptowährung der Bitcoin-Technologie Bitcoin und der Ethereum-Technologie Ether. Eine Kryptowährung ist eine digitale Währung, die nicht durch eine zentrale Autorität verwaltet wird. Eine solche Währung lässt sich durch Blockchain realisieren und Blockchain ist mit ihr (Bitcoin) populär geworden. Für jeden neuen Block kann ein Blockchain-Protokoll eine Münze ihrer Kryptowährung ausgeben und den Miner damit belohnen, so wie bei Bitcoin. Zusätzlich können Kosten an Blockchain-Nutzer weitergegeben werden, sodass diese für ihre abgegebenen Transaktionen Kryptowährung zahlen müssen, die wiederum an Miner geht.

2.1.4 Blockchain-Arten

Blockchains lassen sich theoretisch in vier, hauptsächlich drei, Arten unterteilen [18], [33]:

- öffentliche Blockchain (public): Die Blockchain ist prinzipiell jedem mit einer Internetverbindung zugänglich. Der Lesezugriff auf die Blockchain ist der Öffentlichkeit gestattet.
- private Blockchain (private): Die Blockchain ist prinzipiell nicht jedem zugänglich, wie bei einer öffentlichen Blockchain. Der Lesezugriff kann einem Akteur gewährt werden, wenn dieser bestimmte Voraussetzungen erfüllt.
- genehmigungsfreie Blockchains (permissionless): Eine genehmigungsfreie Blockchain ist eine Blockchain, in der jeder Blockchain-Knoten am Prozess der Erweiterung der Blockchain teilnehmen darf. Solche Blockchains haben üblicherweise eine monetäre Motivation, damit möglichst erhöhtes Interesse an der Teilnahme zur Sicherung der Blockchain existiert.
- genehmigungsbasierte Blockchains (permissioned): Eine genehmigungsbasierte Blockchain ist eine Blockchain, in der prinzipiell nicht jeder Blockchain-Knoten am Prozess der Erweiterung der Blockchain teilnehmen darf.

Hierbei können öffentlich und privat als leserechtsbetreffend und genehmigungsfrei oder -basiert als schreibrechtsbetreffend angesehen werden. Lesen der Blockchain meint entsprechend Einsicht in die Blockchain-Inhalte und Schreiben, das Anfügen eines Blocks.

Die hauptsächlichen drei Arten meinen 1) öffentlich, genehmigungsfrei, 2) privat, genehmigungsbasiert, 3) öffentlich, genehmigungsbasiert. Privat, genehmigungsfreie Blockchains scheinen keinen sinnvollen Einsatzzweck und Nutzwert zu bieten und können auf Schneeballsysteme hindeuten.

Es gilt zu beachten, dass mit „public“ oder „permissionless“ jeweils zusammenfassend, sowohl öffentlich als auch genehmigungsfrei gemeint ist. Genauso meint privat auch häufig automatisch genehmigungsbasiert. Dabei kann eine weitere Unterscheidung solcher restriktiver Blockchains getroffen werden. Es wird dann zwischen privater Blockchain und Konsortiumsblockchain unterschieden. Der Zugriff auf eine private Blockchain wird von genau einer Organisation beschränkt und kontrolliert, der Zugriff auf eine Konsortiums-Blockchain durch mehrere Organisationen eines Konsortiums.

2.1.5 Smart Contracts

Ein Smart Contract ist ein Vertrag, dessen Einhaltung automatisch sichergestellt wird, beispielsweise durch Hardware und Software [30]. Im Blockchain-Kontext meint ein Smart Contract ein Computerprogramm, das bei Eintritt bestimmter Bedingungen bestimmte Aktionen durchführt [19] und auf der Blockchain gespeichert ist. Smart Contracts stellen die Basis dezentraler Anwendungen (dApps) und dezentraler autonomer Organisationen (DAOs) dar und können zur Automatisierung und Disintermediation eingesetzt werden. Beispielsweise werden Smart Contracts im E-Kommerz eingesetzt, um automatische Geldüberweisungen zu tätigen, wenn bestimmte (Vertrags-)Bedingungen eintreten. In vielen Blockchain-Technologien können Nutzer einer Blockchain selbst einen Smart Contract genau verfassen und einsetzen. Mit einem Smart Contract kann sogar ein eigener Token eingeführt werden. Der Smart-Contract-Code ist transparent, manipulationssicher und unveränderbar auf der Blockchain gespeichert, was allerdings auch bedeutet, dass Sicherheitslücken schwerwiegende Folgen haben können. Spezielle Blockchain-Knoten sind dafür verantwortlich, durch Transaktionen ausgelösten Smart-Contract-Code auszuführen (Execution-Clients).

2.1.6 Blockchain-Eigenschaften

Was eine Blockchain interessant macht, sind gewisse Eigenschaften, die einer Blockchain unter bestimmten Umständen, wie in einer Blockchain-Technologie, zugeschrieben werden können. Darunter können fallen: Manipulationssicherheit, Unveränderlichkeit, Integrität, Dezentralisierung, Transparenz, Authentizität, Sicherheit, Resilienz, Disintermediation, Unabstreitbarkeit, Nachvollziehbarkeit, Pseudonymität. Dabei wird sich zeigen, dass sich viele nützliche Eigenschaften aus dem Einsatz kryptografischer Konzepte ergeben [19], [33].

Integrität Durch Berechnung des kryptografischen Hashs und Vergleich mit dem im unmittelbaren Nachfolger des Blocks gespeicherten Hash kann die Integrität des Blocks überprüft werden.

Manipulationssicherheit Eine Blockchain ermöglicht eine Form von Unveränderlichkeit, Manipulationssicherheit und Integrität durch den Einsatz kryptografischer Hashes.

Für jeden Block einer Blockchain gilt: Eine Veränderung des Blocks muss eine Veränderung jedes Nachfolgerblocks mit sich ziehen dadurch, dass jeder kryptografische Hash im Nachfolgerblock neu zu berechnen und zu ersetzen ist. Ein Block einer Blockchain ist somit nur in der Hinsicht unveränderbar, ohne dass alle Nachfolger mitgeändert werden müssten. Werden nicht alle Nachfolger bei einer Veränderung angepasst, so lässt sich durch Nachberechnung und Vergleich der kryptografischen Hashes entlang der Kette dies erkennbar machen (tamper-evident).

Resilienz Eine Blockchain-Technologie ermöglicht Resilienz gegen Löschung der Blockchain durch Replikation; verliert ein Blockchain-Knoten seine Kopie, so kann ein Mechanismus eingesetzt werden, durch den der Knoten eine neue Kopie von den restlichen Blockchain-Knoten erhält.

Dezentralisierung Eine Blockchain-Technologie ermöglicht eine Form von Dezentralisierung durch Replikation und Einsatz dezentralisierter Konsensalgorithmen. Es gibt keinen zentralen Knoten, über den andere Knoten auf die Blockchain zugreifen müssen, und der Konsensmechanismus setzt kollaborative Zusammenarbeit voraus. Somit ist die Technologie vor bestimmten Single-Point-of-Failures geschützt.

Transparenz & Nachvollziehbarkeit Ein Einsatz einer Blockchain kann für Transparenz sorgen, indem sie eine Historie aus Transaktionen speichert. Anhand der gespeicherten Historie können neue oder gespeicherte Elemente nachvollzogen oder rekonstruiert werden.

Authentizität & Unabstreitbarkeit Mit einer Blockchain-Technologie kann Authentizität sichergestellt werden. Es kann vorausgesetzt werden, dass Transaktionen signiert sein müssen, damit sie als gültig anerkannt werden können. Durch Verifikation kann jeder Blockchain-Netzwerkteilnehmer die Authentizität der gespeicherten Daten nachprüfen. Damit lässt sich ebenfalls Unabstreitbarkeit erfüllen.

Disintermediation & Vertrauenslosigkeit Ein Kernmerkmal von Blockchain-Technologie ist die Ermöglichung von Disintermediation, also der Beseitigung von Intermediären. Dies wird durch das Konzept der Vertrauenslosigkeit (Trustlessness) erreicht: Die Interaktion zwischen Parteien erfordert kein gegenseitiges Vertrauen, da die Sicherheit und Integrität

der Transaktionen durch krpytografische Verfahren und einen dezentralen Konsensmechanismus garantiert wird - nicht durch eine vertrauenswürdige Drittpartei.

Pseudonymität & Anonymität Jeder Nutzer besitzt ein Schlüsselpaar aus öffentlichem und privatem Schlüssel für ein Blockchain-Netzwerk. Aus dem öffentlichen Schlüssel wird die Adresse des Nutzers im Netzwerk abgeleitet. Über diese Adresse interagiert ein Nutzer mit einer Blockchain. Somit ist dem Blockchain-Netzwerk keine wahre Identität bekannt, sondern nur ein Pseudonym in Form einer öffentlichen Adresse. Dies garantiert allerdings nicht Anonymität, obwohl es Blockchain-Systeme gibt, die Anonymität herstellen können.

2.1.7 Anwendung

P2P-Geldtransfer Eine wohlbekannte Anwendung von Blockchain und die eigentlich vorgesehene ist die Ermöglichung von Peer-to-Peer-Geldtransfers, d.h. ohne Einbezug von Banken und sogar mit Gewährleistung von Pseudonymität und globaler Reichweite. Blockchain-Technologie ermöglicht die Herstellung einer vertrauenslosen Kryptowährung [15].

Provenance Tracking Provenance Tracking meint die Nachverfolgung und Verifizierung der Herkunft eines Gegenstands. Blockchains können hier angewendet werden, um unveränderliche Aufzeichnungen über Besitzer und Charakteristiken der Gegenstände transparent zu verwalten. Zertifizierungen angesehener Institutionen können dem Gegenstand zugeordnet werden und dienen somit der Verifikation. Ein Beispielkonzept ist Everledger, eine private, genehmigungsbasierte Blockchain für Provenance Tracking von Luxusgütern, Diamanten, Edelsteinen, Wein etc. Everledger integriert IoT, NFC/RFID zum physikalischen Tracking von Waren. Mehr in [5].

Decentralized Storage In Decentralized Storage werden Dateien verschlüsselt, fragmentiert und auf Knoten eines Netzwerks zur Speicherung verteilt und repliziert. Blockchains können genutzt werden, um Metadaten zu speichern, wie Existenz, Ort und Zugriffsrechte der Dateifragmente. Storj ist ein solches dezentralisiertes Cloud-Speichernetzwerk, welches ein auf Ethereum basierendes Blockchain-Token (STORJ) einführt. STORJ wird genutzt, um Knoten für das Speichern und Bereitstellen der Dateifragmente zu bezahlen.

Die Tokens können in andere Währungen konvertiert werden und Nutzer zahlen für das Speichern ihrer Daten. Mehr Informationen im Whitepaper [11].

Weitere Anwendungen Es gibt viele weitere Anwendungen von Blockchain, wie in DeFi, in der Medizin, im Energiesektor etc. (siehe [15]).

2.2 Aufgaben-Zuweisung-Einführung

2.2.1 Aufgaben-Zuweisung

Aufgaben-Zuweisung ist an sich ein sehr genereller Begriff, in dem es darum geht, Aufgaben unter potenziellen Aufgabenträgern aufzuteilen. Es gibt viele unterschiedliche Varianten, die sich darin unterscheiden, wie die Aufgaben zugewiesen werden, wer die Aufgaben zuweist, welche Aufgaben zugewiesen werden und was diese für Beziehungen haben. Es existieren vielerlei Anwendungen, wie in Cloud-Computing, Militäroperationen, physical disasters management, crowd-sourcing platforms usage, smart grids, resource allocation in manufacturing und search-and-rescue-Missionen [28].

2.2.2 Herausforderungen

Herausforderungen, die im Zusammenhang mit Aufgabenzuweisungen auftreten können, sind:

- Heterogenität der Agenten
- Heterogenität der Aufgaben
- Abhängigkeit der Aufgaben
- Zerlegbarkeit der Aufgaben
- Auswahl aus mehreren Alternativen
- Dynamik der Aufgaben, Agenten
- Existenz nicht-kooperierender Agenten
- Deadlines der Aufgaben

- informierte Zuweisung
- Optimierung des Systems

Heterogenität der Agenten Ein Agent ist eine Entität, die Aufgaben zur Bearbeitung übernehmen kann. Je nach Anwendung ist ein Agent z. B. ein Software-Agent oder ein physikalischer Roboter. Agenten können sich in ihren Fähigkeiten unterscheiden, die Voraussetzung der Bearbeitung bestimmter Aufgaben bilden und je nach Ausprägung eine unterschiedliche Bearbeitungseffizienz ermöglichen. Eine mögliche Herausforderung in der Aufgabenzuweisung ist es, diese Merkmale für eine optimierte Zuweisung zu berücksichtigen [28].

Heterogenität der Aufgaben Eine mögliche Herausforderung in der Aufgabenzuweisung stellt die Heterogenität der Aufgaben dar [21]. Jede Aufgabe erfordert einen bestimmten Satz von Fähigkeiten zur erfolgreichen Bearbeitung, die der Zugewiesene vorzuweisen hat [17]. Einige Aufgaben können zudem voraussetzen oder ermöglichen, dass Agenten Koalitionen zur gemeinsamen Bearbeitung der Aufgabe bilden. Eine reale Instanz einer solchen Gruppenaufgabe, wäre z. B. das Anheben und Transportieren eines schweren Objekts, wobei die Agenten jeweils nicht genug Transportfähigkeit besitzen, um das Objekt selbst zu transportieren, und sie somit ihre Kräfte zur Bewältigung der Aufgabe kombinieren müssen [27]. Es gilt, den jeweils passenden Agenten für die zuzuweisenden Aufgaben zu finden.

Abhängigkeit der Aufgaben Aufgaben können auf verschiedene Weisen voneinander abhängen. Die Herausforderung ist, diese Abhängigkeiten zu berücksichtigen. Einige Aufgaben können eine Ausführungsreihenfolge voraussetzen, z. B. in Form einer partiellen Ordnung auf der Menge der zuzuweisenden Aufgaben [27]. Andererseits zeigen sich Abhängigkeiten in der Überschneidung der für die Aufgaben benötigten Ressourcen: Eine Aufgabe kann eine Ressource benötigen, die aktuell von der Bearbeitung einer anderen Aufgabe beansprucht wird, sodass die Aufgabe erst danach erfüllt werden kann [27].

Berücksichtigung der Zerlegbarkeit der Aufgaben Einige Aufgaben lassen sich in Teilaufgaben zerlegen. Dies ermöglicht parallele Erfüllung von Aufgaben für eine erhöhte Leistungsfähigkeit. Allerdings gilt zu beachten, dass zwischen den Teilaufgaben komplexe Abhängigkeiten existieren können, die es zu berücksichtigen gilt. In [21] wird zusätzlich

berücksichtigt, dass es eine Mindest- und Maximalanzahl von Teilaufgaben derselben Aufgabe gibt, die ein Agent annehmen muss, wenn diesem denn eine Teilaufgabe der Aufgabe zugewiesen werden soll.

Auswahl aus mehreren Alternativen Häufig reicht es nicht aus, nur geeignete Agenten zu finden, sondern man muss einen guten oder optimalen Agenten aus einer Menge Alternativen zur Aufgabenübernahme wählen [25]. Diese Herausforderung ergibt sich auch aus der Heterogenität der Agenten. Je nach Metrik gibt es Agenten, die in einer Hinsicht besser als andere geeignete Agenten sind.

Dynamik der Aufgaben und Agenten In einer dynamischen Umgebung können stets Aufgaben und Agenten dazukommen, die den Vorgang der Zuweisung beeinflussen können. Eine dynamische Aufgaben-Zuweisung ist in der Lage, solche Veränderungen in ihrem Prozess zu berücksichtigen [25].

Kollaboration Aufgaben können Kollaboration von Agenten ermöglichen oder erfordern. Ein Aufgaben-Zuweisungs-Algorithmus kann dementsprechend eine Koalitionsbildung der Agenten enthalten. Dabei kann berücksichtigt werden, dass bestimmte Agenten aufgrund bestimmter Bedingungen, wie geografischer, keine Koalitionen eingehen können [27].

Deadlines Einige Aufgaben können Deadlines mit sich bringen, eine erfolgreiche Zuweisung muss dann berücksichtigen, dass Agenten ihre zugewiesene Aufgabe vor Ablauf der Deadline abgearbeitet haben [25].

informierte Zuweisung Je nach Situation müssen eingehende Aufgaben sofort zugewiesen werden, während für andere Aufgaben genug Information existiert, um zeitlich geplante Zuweisung zu einem späteren Zeitpunkt durchzuführen (instantaneous vs. time-extended assignment, siehe [21]).

Optimierung Ein zusätzliches Ziel der Aufgaben-Zuweisung kann sein, eine Optimierung der Leistungsfähigkeit beispielsweise in Form von Aufgaben-Ausführungs-Zeit, inaktiver Zeit von Agenten, Anzahl durchgeführter Aufgaben in einer bestimmten Zeitspanne, Zuverlässigkeit der Aufgaben-Ausführungen, einem bestimmten Nützlichkeitswert oder bestimmten Kosten [28].

2.2.3 Aufgaben-Zuweisungs-Algorithmen

Nach [28] lassen sich Aufgaben-Zuweisungs-Algorithmen unterscheiden:

- marktbasiert, auktionsbasiert
- spieltheoriebasiert
- optimierungsbasiert
- lernbasiert

Marktbasierte Zuweisung Algorithmen dieser Kategorie basieren auf Marktprinzipien und die Agenten bieten in Auktionen um Aufgaben, die sie für sinnvoll betrachten. Agenten basieren ihre Entscheidungen auf ihrer Wahrnehmung ihrer lokalen Umgebung und eigens berechneten Kosten und Nutzen, um zu möglichst hohem Nutzen und niedrigen Kosten Aufgaben zu übernehmen. Das Contract-Net-Protokoll ist die erste Plattform für Auktionen und Aufgabenzuweisung. Sie ist standardisiert und viele weitere Algorithmen basieren auf diesem Werk.

Spieltheoriebasierte Zuweisung Spieltheoriebasierte Algorithmen sehen Agenten als Spieler, in denen die Aufgabenzuweisung durch eine Spielstrategie modelliert wird. Es gibt kooperative, sowie nichtkooperative Spiele, in denen Spieler Koalitionen zur Aufgabenlösung bilden können.

Optimierungsbasiert Ziel einer optimierungsbasierten Methode ist es, eine Zielfunktion zu optimieren. Dabei kann die Zielfunktion Constraints mitberücksichtigen. Es gibt deterministische und stochastische Varianten, wobei stochastische Methoden Zufallselemente einführen, z. B. evolutionäre Algorithmen.

Lernbasiert In lernbasierter Aufgabenzuweisung lernen Agenten mit Machine-Learning-Mechanismen aus ihren vergangenen Entscheidungen, sowie den Entscheidungen anderer Agenten im System, um auch in unvorhersehbaren Umgebungen ihre Aufgabenzuweisung zu optimieren.

2.2.4 Anwendungen

Es gibt ein breites Anwendungsfeld für Aufgabenzuweisungsmechanismen, die betrachtet werden können. Ein Anwendungsgebiet von Aufgabenzuweisung liegt z. B. in Multi-Roboter-Systemen. Dabei gibt es beispielsweise die Aufgabenzuweisung unter unbemannten Luftfahrzeugen unter Berücksichtigung ihrer Heterogenität [34]. Im Cloud-Computing findet Aufgabenzuweisung statt auf Computational Grids [24] oder auf Cloud-Computing-Plattformen in Form von Resource Allocation zwischen Ressourcen-Anbieter und Ressourcen-Konsumenten [16]. Ein weiterer Anwendungsfall ist die Zuweisung zu menschlichen Ressourcen in Organisationen [22].

3 Entwurf einer Blockchain-Anwendung zur Aufgabenzuweisung

3.1 Vorgehen zur Erschließung der Anwendung

Für die Integration von Blockchain in ein geeignetes Aufgabenzuweisungsverfahren, ist zunächst zu klären, wie sich solche Möglichkeiten finden lassen. Damit Blockchain-Konzepte genutzt werden können, müssen ihre Eigenschaften Vorteile einbringen können, die sonst fehlten. Entsprechend gilt es auch, einen Überblick über die Verfahren zur Verteilung von Aufgaben zu erlangen, um Möglichkeiten zuordnen zu können.

3.1.1 Untersuchung der Aufgabenzuweisungslösungen

Unterteilung von Aufgabenzuweisungslösungen Nach [28] lassen sich Techniken zur Aufgabenzuweisung unterteilen in auktionsbasierte, spieltheoriebasierte, optimierungsbasierte und lernbasierte Techniken. Auktionsbasierte Techniken basieren auf wirtschaftlichen Prinzipien und Agenten verhandeln miteinander durch Teilnahme an Auktionen. Agenten geben Gebote ab, bestimmt nach ihrer Wahrnehmung ihrer lokalen Umgebung. Auktionen können durch Auktionatoren verwaltet werden. Ein Auktionator empfängt alle Gebote einer Auktion, setzt Auktionsregeln durch, liefert Informationen an die Bieter und berechnet die endgültige Zuweisung der Auktionsobjekte [31]. Auktionsbasierte Verfahren weisen generell moderate Rechenanforderungen auf. Spieltheoriebasierte Techniken sehen Agenten als Spieler eines Spiels. Die Spielstrategie eines Spielers entscheidet über seine Aufgabenzuordnung, und Spieler versuchen jeweils, ihre Payoff-Funktion zu optimieren, die ihre Belohnung (z. B. Kosten, Nutzwert) für ein Spiel modelliert. In kooperativen Spielen bilden Agenten Koalitionen und kooperieren, um eine optimale Lösung zu finden, während in non-kooperativen Spielen jeder für sich agiert. Optimierungsbasierte Techniken kommen aus der mathematischen Optimierung, das heißt.

es gilt, hinsichtlich eines Kriteriums eine beste Lösung aus einer Menge Alternativen zu wählen. In diese Kategorie gehören deterministische Methoden, wie die ungarische Methode, und Heuristiken, wie biologisch inspirierte Algorithmen. Lernbasierte Techniken versuchen, aus der Historie von Aufgabenzuweisungen, Handlungen anderer Agenten und Zuständen der Umgebung zu lernen, um optimale Aufgabenzuweisungen zu finden.

Chancen der Blockchainnutzung Durch genauere Betrachtung der Komponenten der jeweiligen Verfahren können erste Überlegungen über die Nutzung von Blockchains und Smart Contracts angestellt werden. Zunächst lässt sich feststellen, dass die Speicherung von endgültigen Zuweisungen, unabhängig von der Verfahrensart, für die Herstellung von Transparenz und Nichtabstreitbarkeit der einzelnen Zuweisungen zu den Agenten nützlich sein kann. Allerdings kann es auch Verfahren geben, die selbst von einer verfügbaren Historie von Informationen profitieren können, wie beispielsweise lernbasierte Verfahren. Agenten, die dem Netzwerk neu beitreten, können die bisherige Historie einsehen und somit Lernarbeit aufholen und mit einem optimaleren Entscheidungsprozess loslegen. Mithilfe der Blockchain-gespeicherten Daten können sichere Ruf-basierte Systeme gebaut werden. Über die endgültigen Zuweisungen hinausgehend, können Zwischenzustände bestimmter Verfahren in die Blockchain gespeichert werden. Diese Informationen können nun zusätzlich der Nachvollziehbarkeit und Auswertung vergangener Entscheidungen dienen. Smart Contracts dienen der Automatisierung von Prozessen, der Ersetzung von womöglich unvertrauenswürdigen Drittparteien, der Herstellung von Dezentralisierung und Sicherheit, dass bestimmter Code unter bestimmten Bedingungen ausgeführt wird und damit bestimmte (Vertrags-)Bedingungen durchgesetzt werden. Somit kommen allgemein für eine Anwendung von Smart Contracts zentralisierte Verfahren infrage oder Verfahren, die auf Dritten basieren und ineffiziente Prozesse darstellen, wie bereits in dem Kapitel zu Anwendungen von Smart Contracts vorgestellt. Auktionen bieten auf den ersten Blick gleich eine Möglichkeit zur Verwendung von Smart Contracts. Ein Smart Contract könnte nämlich die Funktionen eines Auktionators übernehmen, sodass dieser automatisiert wird, Auktionsprozesse dezentralisiert werden, Vertrauen in die Blockchain delegiert wird, Auktionsregeln transparent gemacht werden und gleichzeitig relevante Daten in die Blockchain gespeichert werden.

Limitationsfaktoren Allerdings gilt es zu berücksichtigen, dass Smart Contracts Limitationen aufweisen können und Herausforderungen mit sich bringen, die beachtet werden sollten. Das Speichern eines entwickelten Smart Contracts in eine Blockchain ist eine

weitere Transaktion, die in der Blockchain gespeichert wird, die wiederum Bytecode des Smart Contracts abspeichert. Transaktionen kosten bei ihrer Ausführung Gas und bei Speicherung des Smart Contracts entstehen entsprechend Gaskosten, die mit der Größe des Smart Contracts zusammenhängen. Die Kosten selbst sind dabei nicht die einzige Herausforderung, denn jeder Block besitzt eine Maximalgröße in Form eines Block-Gas-Limits. Dieser Maximalwert gibt an, wie viel Gaskosten, die Transaktionen des Blocks in ihrer Summe maximal verursachen dürfen. Übersteigt eine einzelne Transaktion das Block-Gas-Limit, so kann diese Transaktion und somit die Speicherung des Smart Contracts nicht akzeptiert werden. Nach einer erfolgreichen Integration eines Smart Contracts in die Blockchain entstehen Gaskosten auf leicht andere Weise. Nutzer, die Transaktionen an das Blockchain-Netzwerk senden, tragen Gaskosten in Form von transaction fees, die durch ihre Transaktionen anfallen. Ist die Smart-Contract-Logik komplex, können entsprechend hohe Gaskosten für Nutzer entstehen, die eine Smart-Contract-Anwendung einschränken könnten. Ein weiterer Aspekt, den es zu berücksichtigen gilt, ist die Sicherheit des Smart Contracts. Ist der Smart Contract öffentlich einsehbar und fehlerhaft, könnten Akteure Sicherheitslücken ausnutzen, um verheerenden Schaden anzurichten. Je nach Anwendung muss ein Smart Contract gegen potenzielle Angreifer sicher sein.

Auch Blockchains selbst können Limitationen aufweisen, beispielsweise durch limitierte Geschwindigkeit in der Verarbeitung von Transaktionen durch das Blockchain-Netzwerk (transactions per second, TPS), ihren Grad der Dezentralisierung und Sicherheit, ihren Grad der Unterstützung von Smart Contracts und ihren Speicherbedarf, sowie die Reife ihres Ökosystems. Ein generelles Problem, das Blockchains aufweisen können, ist dadurch geschuldet, dass Blockchains zur Speicherung jeglicher Daten genutzt werden, die nichts miteinander zu tun haben müssen und ohne semantisch erkennbare Ordnung gespeichert sind. Dies kann es schwer machen, nach bestimmten Informationen zu suchen, vor allem bei einer immer größer werdenden Blockchain und gesuchten Daten, die über mehrere Blöcke verteilt gesammelt werden müssen [26].

3.1.2 Auktionsbasierte Aufgabenzuweisung

Unter Berücksichtigung bekannter Limitationen von Blockchain und Smart Contracts, sowie direkt erkennbarer Einsatzmöglichkeiten in Auktionatoren, wird hier die Entscheidung getroffen, auktionsbasierte Aufgabenzuweisung genauer zu untersuchen. Dafür wird im Folgenden das Konzept der Auktion differenziert.

Auktionen

Eine Auktion ist eine Verhandlung über den Kaufpreis von Gütern. Ein Auktionator ist die Entität, die eine Auktion durchführt und verwaltet. Sie empfängt abgegebene Gebote von Agenten, setzt Gebotsregeln durch, liefert relevante Informationen an teilnehmende Agenten und berechnet eine Allokation der Auktionsobjekte nach einer Funktion der gesammelten Gebote.

Auktionen unterscheiden sich beispielsweise in der Menge der zu versteigernden Objekte. Dabei stehen Einzelauktionen sogenannten Multiunit-Auktionen gegenüber. Multiunit-Auktionen können dabei homogene (Mehrgutauktion) oder heterogene Objekte (kombinatorische Auktion) versteigern. Dabei gilt es ebenfalls festzulegen, ob mehrere Objekte zeitgleich (simultaneous auction) versus sequentiell (sequential auction) versteigert werden.

Einige Auktionen sehen vor, dass entweder nur Käufer oder nur Verkäufer an der Verhandlung teilnehmen dürfen. Bei abgeschlossener Auktion des Auktionsobjekts gibt es natürlich einen Verkäufer, dieser steht aber innerhalb der Auktion nicht in Konkurrenz mit anderen Verkäufern und umgekehrt genauso. Dadurch unterscheiden sich einseitige (single-sided auction) und zweiseitige (double-sided auction/double auction) Auktionen.

Auktionen definieren Regeln, nach denen Bieter innerhalb der Auktionen handeln dürfen. Je nach Auktion dürfen beispielsweise aufeinanderfolgende Gebote entweder unterschiedlicher oder jeweils derselben Agenten nur steigen (ascending-bid auction) oder fallen (descending-bid auction). Da in Auktionen über Kaufpreise verhandelt wird, muss aber auch festgelegt werden, zu welchem Preis Objekte letztendlich gekauft werden. In Erstpreisauktionen wird der höchste Preis durch den Gewinner eines Objekts gezahlt, während in Zweitpreisauktionen (Vickrey-Auktion) der Gewinner nur den Preis des zweithöchsten Gebots zu zahlen hat (etc. für Drittpreisauktion). Eine Abweichung von der Erstpreisauktion, fördert das Abgeben wahrheitsgemäßer Gebote von den Agenten. Für Details, siehe [23].

Allgemein ist von Auktionsobjekten die Rede. Damit können dementsprechend auch Güter oder in diesem Falle Aufgaben gemeint sein. Auktionen finden u. a. Anwendung in Aufgabenzuweisungs-Mechanismen, indem Aufgaben an Agenten versteigert werden. Die Zuweisung einer Aufgabe wird modelliert durch einen Kauf der Aufgabe innerhalb einer Auktion, d. h., ein gewinnender Verkäufer übernimmt die Bewältigung der gewonnenen

Aufgabe und der gewinnende Käufer gewinnt eine Zusage für die Erledigung oder das Ergebnis der erledigten Aufgabe.

Es gibt somit eine Menge verschiedener Auktionen, welche in Betracht gezogen werden können. Theoretisch könnte die Nutzung von Blockchains für Einzelauktionen von Aufgaben untersucht werden, jedoch würde sich das kaum von Einzelauktionen genereller Güter unterscheiden. Daher sind hier Multiunit-Auktionen von Interesse. In kombinatorischen Auktionen ersteigern Agenten Bündel heterogener Güter, deren Nutzwert in Kombination nicht unbedingt gleich der Summe ihrer einzelnen Nutzwerte entspricht. Da es extrem viele mögliche Kombinationen gibt, aus denen eine optimale gefunden werden soll, sind solche Auktionen sehr aufwändig. Da kombinatorische Auktionen gegenüber Mehrgutauktionen generell deutlich erhöhte Komplexität aufweisen, die einen Smart Contract sehr teuer oder gar unmöglich machen könnte, bezieht sich diese Arbeit fürs Erste vorausschauend nur auf Mehrgutauktionen. Auktionen, wie die Vickrey-Auktion ermöglichen eine vereinfachte Betrachtung von Bietverhalten. Da nur der zweithöchste Preis zu zahlen ist, ist es rational, möglichst den wahren Wert für ein Auktionsobjekt zu bieten.

Unterteilung auktionsbasierter Techniken

Laut [28] lassen sich auktionsbasierte Aufgabenzuweisungsalgorithmen hauptsächlich in CBBA-basiert oder CNP-basiert unterteilen. Der Consensus-Based Bundle Algorithm (CBBA) ist ein konkreter Aufgabenzuweisungsalgorithmus, in dem Agenten über zwei Phasen Aufgabenbündel erstellen und sich untereinander zuordnen [20]. In der ersten Phase werden Auktionen eingesetzt, um Aufgaben zu ersteigern. In der zweiten Phase werden Konsensalgorithmen genutzt, um konfligierende, ersteigerte Aufgaben eindeutig zuzuordnen. Durch Konsens wird global konsistente Information hergestellt. Es wird zwischen beiden Phasen iteriert, bis die Lösung gefunden wird. Das Contract-Net-Protocol [29] (CNP) ist ein Protokoll, das als Basis vieler weiterer Zuordnungsalgorithmen dient. Knoten eines Netzwerks kommunizieren miteinander, um in einer Top-down-Herangehensweise Aufgaben zu zerteilen und zuzuweisen. Es ist ein High-Level-Protokoll für Kommunikation in verteiltem Problemlösen. Ein sogenannter Manager kündigt eine Aufgabe an, die ein verfügbarer und passender Knoten (Contractor) übernehmen soll. Knoten filtern Aufgaben-Ankündigungen nach denen, die sie übernehmen können. Jeder interessierte Knoten liefert ein Gebot für die Aufgabe an den Steller, welcher ein gültiges und am besten passendes Gebot akzeptiert. Der Knoten, der die Aufgabe erhält, kann

diese nun bearbeiten oder sie in Teilaufgaben zerlegen und diese ebenfalls auslagern und ankündigen, bis ein Aufgabenabhängigkeitsnetzwerk vollständig gebildet wurde.

3.1.3 Wahl des Algorithmus

Da der CBBA durch die iterierenden Phasen und kombinatorischen Task-Bundles relativ komplex wirkt, wird in dieser Arbeit im Folgenden eine CNP-basierte Technik betrachtet, wieder mit Rücksicht auf potenzielle Limitationen von Blockchain und Smart Contracts. CNP hat das Problem, dass durch den Top-down-Ansatz ein Agent die Verantwortung einer Auslieferung zusagt, ohne Sicherheit über ihre Einlieferungen zu haben. Es wird also erst eine Beziehung eingegangen, in der ein Agent ein bestimmtes Gut liefert, obwohl noch keine Beziehungen etabliert wurden, die die benötigten Eingabegüter des Agents garantieren. Zudem berücksichtigt CNP nicht die Konsequenz limitierter Ressourcen, sodass trotz Existenz einer möglichen Lieferkette, diese nicht gefunden wird. CNP funktioniert Top-down und gierig, somit kann es passieren, dass ein Agent einer Aufgabe zusagt, obwohl ein solcher Agent erneut in einer tieferen Ebene des Netzwerks notwendig ist. Für eine entsprechende Lösung hätte ein alternativer Agent die Aufgabe erhalten müssen und der begehrte Agent tiefer im Netzwerk die notwendige Teilaufgabe. Daher schlagen [32] ein Marktprotokoll vor, das die Probleme des CNP mit Mehrgutauktionen und generalisierten Vickreyauktionen adressiert, wobei es sich nun um simultane, zweiseitige Auktionen handelt mit mehreren Käufern und Verkäufern. Dabei werden von den Käufern/Verkäufern separate Auktionatoren eingeführt, deren Ersetzung durch Smart Contracts innerhalb dieser Arbeit untersucht wird.

3.2 W.E. Walsh Aufgabenzuweisung

Eine konkrete Technik zur Aufgabenzuweisung wird mithilfe von Blockchain und Smart Contracts implementiert und untersucht. Dabei handelt es sich um das Marktprotokoll nach [32]. Dieses Verfahren ist ein auktionsbasiertes Verfahren, welches zwei Probleme des CNP adressiert. Ein Marktprotokoll ist ein Protokoll, das aus zwei Komponenten besteht, dem Auktions-Mechanismus und den Gebotsregeln. Der Auktions-Mechanismus legt die Art und Weise der Auktionen fest, während die Gebotsregeln beschreiben, nach welchen Regeln Bieter ihre Gebote stellen [32]. In CNP werden Auktionen genutzt, um Aufgaben Agenten zuzuordnen, und ein Agent führt selbst die Auktion durch für die

zu versteigernde Aufgabe. CNP ermöglicht Kommunikation zur verteilten Lösung von Problemen. Verteiltes Problemlösen (distributed problem solving) ist ein kooperatives Lösen von Problemen verschiedener, verteilter Knoten, die jeweils unterschiedliche Wissensquellen (knowledge-source, kurz KS) beinhalten. Die Knoten müssen miteinander kooperieren, da kein Knoten genug Wissen besitzt, das gesamte Problem selbst zu lösen [29].

CNP unterscheidet zwei Rollen von Agenten:

- Manager: Agent, der verantwortlich ist für die Überwachung der Ausführung einer Aufgabe und die Verarbeitung des Ergebnisses. Manager machen Aufgabenankündigungen, evaluieren einkommende Gebote und verteilen angekündigte Aufgaben an Bieter.

- Contractor: Agent, der verantwortlich für die eigentliche Ausführung einer Aufgabe ist, Aufgaben-Ankündigungen filtert und Gebote zur Übernahme angekündigter Aufgaben abgibt. Ein Contractor kann selbst Manager werden, wenn der Contractor auf KSs anderer Agenten zur Lösung der eigenen Aufgabe angewiesen ist.

Das Protokoll unterscheidet drei Arten von Agenten:

- Konsument: Das einzige, was ein Konsument macht, ist, eine Aufgabe zu stellen, um einen Agenten zu finden, der sie übernimmt. Der Konsument stellt sozusagen das zentrale Problem dar, das gelöst werden soll und welches der Konsument selbst nicht zu lösen weiß oder kann.

- Produzent: Ein Produzent sammelt angekündigte Aufgaben und versucht, geeignete Aufgaben zu übernehmen. Die KS eines Agents können u.a. Produktionsregeln enthalten, welche Wissen darstellen, bestimmte (Teil-)Probleme bzw. Aufgaben zu lösen. Produzenten zerlegen ein Problem in Teilprobleme und stellen diese wiederum dem Netzwerk durch Aufgabenankündigung, wie der Konsument.

- Lieferant: Ein Lieferant sammelt, wie ein Produzent, angekündigte Aufgaben, jedoch besitzt dieser eine KS, die Lösung und direkte Lieferung bestimmter gestellter Aufgaben ermöglicht. Ein Lieferant zerlegt ein Problem nicht weiter und stellt somit keine Teilaufgaben.

Zudem gibt es den Auktionator, einen Agenten, der verantwortlich für die Durchführung einer Auktion ist. Der Agent sammelt die einzelnen Gebote der Konsumenten, Produzenten und Lieferanten, berechnet und liefert relevante Informationen an die Bieter und führt letztlich die endgültige Aufgabenzuweisung durch.

Durch Ankündigung einer Aufgabe durch einen Konsumenten, entsteht bei der Propagierung dieser Nachfrage durch das Netzwerk ein Aufgabenabhängigkeitsgraph („task dependency network“), siehe Abbildung 4.1. Produzenten und Lieferanten filtern angekündigte Aufgaben und bestimmen für sich diejenigen Aufgaben, die sie übernehmen können.

Während sich ein Aufgabenabhängigkeitsgraph bildet, verhandeln die Agenten miteinander, um letztendlich eine Aufgabenzuweisung zustande zu bringen. In CNP findet dieser Prozess top-down statt, während im betrachteten Marktprotokoll die einzelnen Zuweisungen bottom-up erfolgen. Die Entstehung des Aufgabenabhängigkeitsgraphs kann in beiden Fällen als top-down angesehen werden, allerdings erfolgt die Lösung des Graphs entweder top-down oder bottom-up. Für jede (Teil-)Aufgabe wird eine Auktion abgehalten, in der sie unter den Agenten versteigert wird. Eine Auktion wird initiiert durch einen Konsumenten oder Produzenten, der die Aufgabe zuweisen möchte. Der Konsument kann als Käufer einer Aufgabenerfüllung/eines Aufgabenergebnisses gesehen werden und ein Lieferant als Verkäufer einer solchen Aufgabenerfüllung/eines Aufgabenergebnisses, kurz der Aufgabe. Sobald Agenten von Auktionen erfahren, an denen sie interessiert sind, können sie Verkaufsgebote abgeben, und um die Aufgaben bieten. Sobald Ruhe in die Auktionen einkehrt, können sie als beendet erklärt werden und die Aufgabenzuweisung endgültig durchgeführt werden.

Die Auktionen werden von Auktionatoren verwaltet und durchgeführt. Unter anderem setzt der Auktionator die Regeln der Auktion durch. Bei den Auktionen des Marktprotokolls handelt es sich um eine Verallgemeinerung der Vickrey-Auktion. Konkret werden Einzelauktion zu zweiseitiger Mehrgutauktion und Zweitpreisauktion zu (M + 1)-Preis-Auktion generalisiert. Mehrgutauktion bedeutet, dass statt einer Aufgabe, dieselbe Aufgabe in mehreren Einheiten (homogene Objekte) versteigert werden kann. (M + 1)-Preis-Auktion heißt, dass eine versteigerte Aufgabe, statt zum zweithöchsten Gebotspreis, zum (M + 1)-höchsten Gebotspreis versteigert wird. M ist dabei die Anzahl der Instanzen der Aufgabe, die in der Auktion versteigert werden. Der zu zahlende Preis trennt gewinnende Gebote von verlierenden Geboten, das heißt: Kaufgebote unter und Verkaufsgebote über der Grenze verlieren bzw. Kaufgebote über und Verkaufsgebote unter der Grenze gewin-

nen. Kaufgebote über der Grenze werden zu Verkaufsgeboten unter der Grenze gematcht. Diese Art Auktion macht es rational für Agenten, für ihren wahren Preis Gebote abzugeben. Zudem müssen Gebote bei den hier betrachteten Auktionen aufsteigend sein, in der Hinsicht, dass das neuere zweier aufeinanderfolgender Gebote desselben Agents höher als das vorige zu sein hat. Ein Auktionator muss diese Bedingungen bei der Verwaltung der Auktion berücksichtigen und gleichzeitig die Agenten mit Zwischenstandsinformationen beglücken. Genauer: Muss der Auktionator nach Überprüfung der Validität eines Gebots, das valide Gebot in den Auktionszustand integrieren und beteiligte Bieter davon benachrichtigen, wie viel sie gewinnen würden, sodass diese ihre Gebote überdenken können? Dafür berechnet und versendet der Auktionator ein sogenanntes „price quote“, hier Preisangebot.

Der Auktionsmechanismus liegt unter der Kontrolle der Systemdesigner, jedoch lässt sich dies nicht immer über das Bietverhalten der Agenten sagen. Es möge vom System bestimmtes Verhalten motiviert werden, allerdings kann dies nicht immer garantiert werden. Diese Eigenschaft kann sich von System zu System unterscheiden, aber für eine Analyse des Systems, kann es von Nutzen sein, Annahmen über das Bietverhalten der Agenten zu treffen. Hierbei werden die Bietstrategien aus [28] übernommen. Konsument und Lieferanten bieten zu ihrem Höchstpreis. Produzenten hingegen haben das Ziel, so zu bieten, dass sie nicht zu viel versprechen und womöglich zu ihrem Verlust handeln. Daher funktionieren ihre Bietregeln, wie folgt:

- Der Produzent führt eine Schätzung der Kosten seiner Eingabe durch. Ist diese Schätzung höher als das letzte Ausgabegebot, das der Produzent platziert hat, so platziert er ein neues Ausgabegebot zu dem erhöhten Preis.

- Initial bietet der Produzent den Nullwert für seine Eingabe-Aufgaben.

- Zeigt das aktuellste erhaltene Preisangebot einer Eingabeaufgabe, dass der Produzent sie verlieren würde, aber das aktuellste erhaltene Preisangebot der zugehörigen Ausgabeaufgabe, dass der Produzent diese gewinnen würde, erhöht der Produzent dann, und nur dann, sein Eingabegebot um einen kleinen endlichen Wert (δ).

Durch diese Verhaltensweise arbeitet sich ein Produzent vorsichtig hoch, um keine Minusgeschäfte versehentlich einzugehen.

3.3 Blockchain-basiertes Marktprotokoll

Abbildung 3.1 zeigt eine Kontextsicht auf die traditionell mit zentralem Auktionsserver entworfene Aufgabenzuweisungsvariante. Die Sicht umfasst den Auktionsserver, die Agenten und den Auktionschecker.

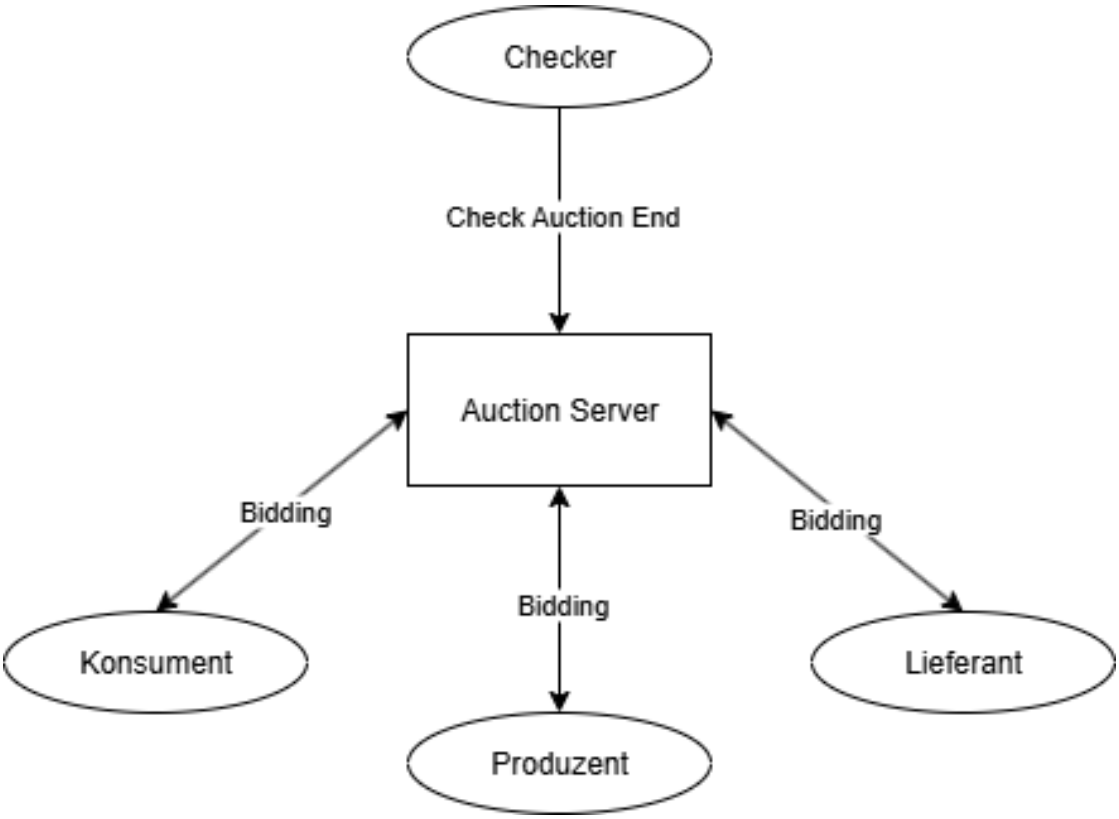

Abbildung 3.1: Auktionsserver

Auktionsserver Der Auktionsserver ist eine zentrale Entität, die eine auktionsbasierte Aufgabenallokation abwickelt. Der Server verwaltet pro Aufgabe eine Auktion und startet Auktionen für neue angekündigte Aufgaben. Es wird sichergestellt, dass der Server ein Gebot zur Zeit verarbeitet, da sonst nebenläufige Fehler auftreten können. Dies sollte nicht vollständig der Vorstellung des Marktprotokolls entsprechen, da es möglich sein sollte, Gebote unterschiedlicher Auktionen parallel zu verarbeiten. Um jedoch die Architektur vergleichbarer zur Blockchain-Variante zu machen, wird eine vollständig sequentielle Verarbeitung vorgesehen. Zusätzlich zu den Zuständen der einzelnen Auktionen verwaltet der Auktionsserver den Zustand des gesamten Algorithmusablaufs.

Agenten Die Agenten umfassen, wie im Algorithmus bereits beschrieben, die drei Unterarten: Lieferanten (supplier), Produzenten (producer) und den Konsumenten (consumer). Gemeinsam ist den Agenten, dass sie allesamt über den Auktionsserver als Mittelmann gegeneinander um Aufgaben bieten. Ein Konsument kündigt über den Auktionsserver Aufgaben mit einem Kaufgebot an. Produzenten platzieren ihre eigenen Gebote und Aufgabenankündigungen, basierend auf Benachrichtigungen über Gebote und Aufgabenankündigungen, die sie von dem Auktionsserver erhalten. Lieferanten platzieren nur Gebote, um Aufgabenankündigungen zu erfüllen. Die Agenten stehen innerhalb des Prozesses einer Aufgabenzuweisung somit nicht miteinander direkt in Kontakt und müssen vorab auch keine Informationen voneinander wissen. Jeder Agent besitzt eigenes Wissen darüber, wie er bestimmte Aufgaben lösen kann. Dadurch bestimmen sich seine Kosten und die Bestimmung von Teilaufgaben, die von anderen Agenten mit speziellem Wissen zu lösen sind. Auf diese Weise wird eine gestellte Aufgabe durch die Agenten in Teilprobleme zerlegt und zugewiesen.

Checker Der Checker ist hier ein einzelner Agent, der als Lösung der Ruheerkennung des Auktionsprozesses dient, da das Marktprotokoll keine eigene vorschlägt. Er sendet regelmäßig eine Erinnerung an den Auktionsserver, der überprüft, ob ein Timeout seit dem letzten Gebot eingetreten ist. Die Nutzung einer solchen Checker-Komponente stellt einen Single-Point-of-failure dar. Eine Überlegung zur Dezentralisierung dieser Lösung, wäre eine Überprüfung durch Agenten, da diese eigenes Interesse an einer Fertigstellung der Auktionen besitzen. Zur Vereinfachung wird dieser Teil der Agenten durch einen Einzelnen repräsentiert.

Zustände Der Zustand einer einzelnen Auktion umfasst Informationen zu Geboten, Bietern und daraus berechenbaren Meta-Informationen, wie der Anzahl der Verkaufsgebote m, relevant für die Bestimmung der einzelnen Preisangaben. Der Auktionsserver verwaltet einen Auktionszustand pro Aufgabe, allerdings nur genau einen Algorithmuszustand. Ein Algorithmuszustand umfasst zusätzlich zu den Auktionszuständen, den Zeitforschritt zur Ruheerkennung und die letzte erhaltene Gebots-ID pro Bieter. Das Marktprotokoll beinhaltet einen Gebots-ID-Mechanismus, um Asynchronizität zu berücksichtigen, die sich darin zeigt, dass Nachrichten bzw. Gebote nicht chronologisch bei ihren Empfängern eintreffen.

Funktionen Die Funktionen des Auktionsservers umfassen die Behandlung von Geboten, die Berechnung der Preisinformationen für die Bieter, die Beendigung der Auktionen und Aufgabenzuweisung.

3.3.1 Ansetzstellen einer Integration

Wie sich durch die vorhergehende Betrachtung des Marktprotokolls ergeben hat, gibt es zwei Problemstellen, an denen nun mit Blockchain-Technologien angesetzt werden kann:

1. Es könnte Disintermediation, Automatisierung, Vertrauensfreiheit bezogen auf die Auktionen hergestellt werden. Dafür lassen sich Blockchains mit Smart Contracts verwenden.
2. Es könnte Transparenz, Nachvollziehbarkeit, Manipulationssicherheit bezogen auf die Abläufe und Kommunikationen sichergestellt werden. Dazu können die Blockchain-Eigenschaften genutzt werden durch Speicherung der relevanten Daten in die Blockchain.

Datenspeicherung in einer Blockchain Zur Herstellung von Transparenz und weiteren Blockchain-Merkmalen können die relevanten Daten in einer Blockchain gespeichert und verwaltet werden. Es gilt somit, die relevanten Daten vorerst zu identifizieren. Daten, die im Protokoll verkehren, werden hier gesammelt:

1. angekündigte Aufgaben des End-Konsuments mit Höchstpreis
2. Kauf- und Verkauf-Gebote für bestimmte Aufgaben der Agenten
3. aus Auktionszustand berechnetes Preisangebot
4. Endgültige Aufgaben-Zuweisung durch Beendigung der Auktionen

Es können also einzelne Nachrichten Teil der Blockchain-Historie werden. Dafür wird festgelegt, welche Informationen Teil welcher Nachricht zu sein haben. Ein Gebot auf eine Aufgabe umfasst hier folgende Informationen:

1. eine Bieter-ID
2. Angabe der Aufgabe, für die geboten wird

3. eine Angabe, ob es sich um ein Kauf- oder Verkauf-Gebot handelt
4. die Höhe des Gebots

Ein Preisangebot von Auktion an Agent umfasst:

1. die Gebots-ID des letzten Gebots des Empfänger-Agents
2. den $(M + 1)$-höchsten Kaufpreis der bisher vorgeschlagenen Preise
3. die Anzahl der gewonnenen Güter des Empfänger-Agents

Die Anzahl der gewonnenen Güter ergibt sich nicht unbedingt aus dem aktuellen Kaufpreis, da mehrere Agenten zum gleichen Kaufpreis geboten haben könnten, aber nicht für jeden ein Gut übrig bleibt.

Einsatz eines Smart Contracts Durch den Einsatz eines Smart Contracts könnten die Auktionsvermittler dezentralisiert und automatisiert werden. Kommunikation mit einem Smart Contract funktioniert ereignisgesteuert: Eingehende Transaktionen sind Ereignisse, die Smart-Contract-Funktionen auslösen können, und ein Smart Contract kann Ereignisse emittieren, die wiederum durch Polling abgerufen werden können. Die Funktionen eines Smart Contracts entsprechen den Funktionen des zentralen Auktionsservers, jedoch stellt der Smart Contract selbst keinen Single-Point-of-Failure dar, da mehrere Knoten des Blockchain-Netzwerks dafür verantwortlich sind, Smart-Contract-Code auszuführen. Somit bleibt die Kommunikation von Agent Richtung Auktionsvermittler gleichartig. Kommunikation von Smart Contracts zurück zu Agenten muss allerdings durch Ereignisse erfolgen, die ein Agent abhört. Die Ereignisse bleiben gleich: Aufgabenankündigung, Preisangabe und Aufgabenzuweisung.

Der Smart Contract übernimmt dieselben Funktionen:

1. Überprüfung der gestellten Aufgaben zur Sicherung ihrer Gültigkeit und anschließende Verbreitung der gestellten Aufgaben eines Konsuments zum Start der Auktionen
2. Überprüfung eingehender Gebote zur Sicherstellung ihrer Gültigkeit, Berechnung und Verbreitung der „price quote“-Nachricht für jedes gültige Gebot
3. Beendigung der Auktion, Bestimmung der Auktions-Gewinner und damit endgültige Aufgaben-Zuweisung

Die Abbildung 3.2 zeigt, dass der Auktionsserver durch einen Smart Contract auf ein Blockchain-Netzwerk verteilt wird. Agenten verbinden sich mit Knoten des Blockchain-Netzwerks und senden ihre Gebote als Transaktionen ab. Das Blockchain-Netzwerk garantiert die Ausführung der Auktionslogik, wobei über Blockchain-Ereignisse Informationen an Agenten vermittelt werden.

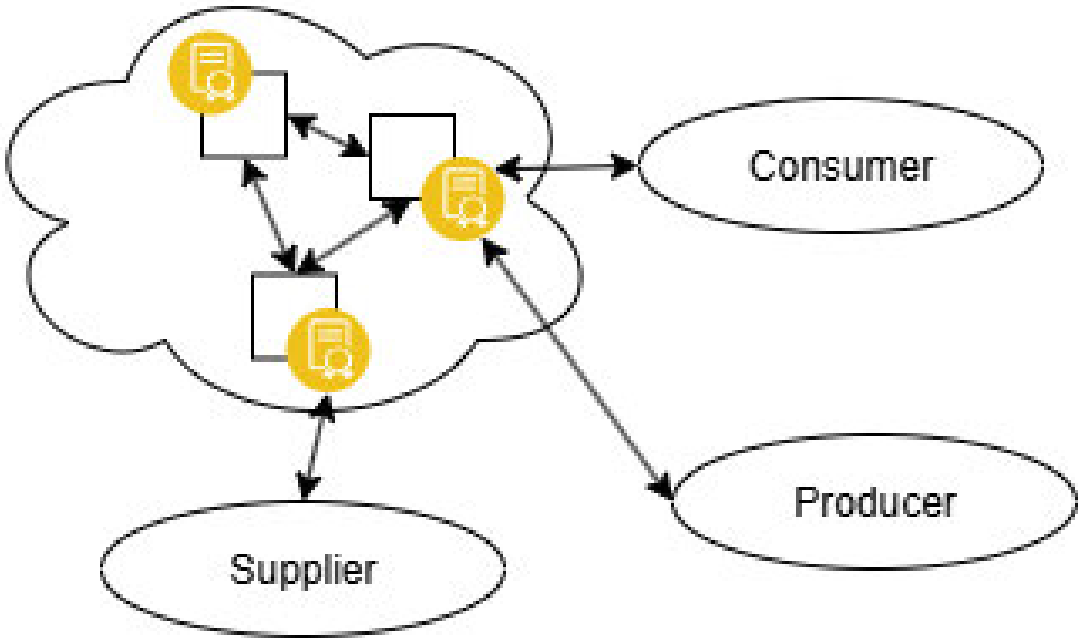

Abbildung 3.2: Smart-Contract-Auktionen

3.3.2 Entwurf des smart contracts

Es gilt nun, einen Smart Contract zu entwerfen, der die besagten Eigenschaften und Funktionen erfüllt. Eine Entscheidung, die getroffen werden muss, liegt zwischen dem Einsatz mehrerer Smart Contracts mit jeweiligen Teilverantwortungen oder eines großen Smart Contracts, also des Scopes. Es könnte einen Smart Contract pro Auktion geben oder einen Smart Contract pro Aufgaben-Zuweisung bzw. Auftrag eines Konsumenten, der alle Auktionen für den Auftrag verwaltet. Überlegungen zu infrage kommenden Faktoren, wie Berechnungskosten, Zeit, Kommunikationsaufwand und weiteren, die sich bei dieser Entscheidung unterscheiden könnten.

Das Deployment eines Smart Contracts auf der Ethereum-Blockchain ist kostspielig im Vergleich zu einzelnen Anweisungen. Dies spricht gegen den Einsatz mehrerer Smart Contracts, vor allem hinsichtlich der Skalierbarkeit der Aufgaben, könnten die Kosten doch schnell den Nutzen übersteigen.

Ein Smart Contract kann nur eine Transaktion zur selben Zeit bearbeiten. Durch Trennung der Auktionen können Gebote für andere Aufgaben parallel verarbeitet werden. Allerdings muss auch berücksichtigt werden, dass die Zeit für das Inkludieren einer Transaktion in einen Block länger dauern könnte, als die sequenzielle Bearbeitung der Transaktionen des letzten Blocks. Dies hängt also von der Anforderung einer konkreten Anwendungsdomäne ab.

Wenn jede Auktion ihren eigenen Smart Contract hat, dann gibt es pro Auktion eine Adresse. Damit könnten Daten auf der Blockchain leichter nachverfolgt und nachvollzogen werden. Jedoch gibt es einen vergrößerten Kommunikationsaufwand: Statt nur einer Verbindung pro Agent, hat jeder Agent so viele Verbindungen, wie es an Aufgabenauktionen beteiligt ist.

Nachrichtenstrukturen im Smart Contract Ein Gebot umfasst eine Gebotshöhe, die Gebotsart (Kauf oder Verkauf), die Aufgabe, für die geboten wird, und die Instanz der Aufgabe. Die Struktur eines Preisangebots besteht aus Empfängeradresse, Aufgabe, Gebotsnummer und der Angabe aller gewinnenden Instanzen des Empfängers an der Auktion. Die schlussendliche Aufgabenzuweisung benötigt hierbei keine eigene Struktur, da die Bieter nach Auktionsablauf ihr aktuellstes Preisangebot ablesen können. Das Signal für ein Auktionsende muss klarstellen, welche Auktion als abgeschlossen gilt. Wenn Nachfrage für eine Aufgabe eintrifft, werden die Agenten benachrichtigt, damit sie mit Verkaufsgeboten reagieren können.

Smart-Contract-Zustand Der Smart Contract muss bestimmte Auktions- und Algorithmusinformationen verwalten. Zur Speicherung dieser Informationen können Smart-Contract-Zustandsvariablen eingesetzt werden.

- *bidList* ist eine Liste aller aktuellen Gebote für eine Auktion. Der Smart Contract speichert somit eine *bidList* pro Aufgabe. Diese wird aufsteigend sortiert, um die Berechnung des $(M + 1)$-Preises zu erleichtern.

- *bidders* ist eine dynamische Liste aller bekannten Bieter für eine Aufgabe, d. h., der Smart Contract verwaltet eine solche Liste pro Aufgabe. Diese ist u. a. notwendig, um an jeden Bieter ein entsprechendes Preisangebot zu adressieren.

- *bidIDs* speichert pro Bieter einer Auktion die Anzahl aller, von diesem Bieter, erhaltenen validen Gebote. Diese wird als Gebots-ID in dem Preisangebot angegeben, damit der Bieter weiß, ob sein aktuellstes Gebot bereits verarbeitet wurde [28].

- *lastBidTime* speichert einen Zeitstempel für das zuletzt eingetroffene Gebot einer beliebigen Auktion des Smart Contracts. Es dient zur Überprüfung des Auktions-Zeitablaufs für die Terminierung des Algorithmus.

- *auctionTasks* ist eine Liste aller Aufgaben, die der Smart Contract kennt und in Geboten berücksichtigt.

Der Smart Contract reagiert auf Transaktionen, die ein Gebot platzieren. Agenten senden als Transaktion, die Platzierung eines Gebots, und interagieren dadurch mit dem Smart-Contract-Auktionator. Der Smart Contract prüft auf das Ende der Auktion, die Validität des Gebots, speichert das Gebot in die Blockchain und kalkuliert und versendet Preisangebote an die bekannten Bieter. [H] [1] place-bidbid *bidTask* ← *bid.task* auction timeout triggered Emit auction end event. !bidValid(bid) Update *lastBidTime* for *bidTask*. Increment last bid id received for that bidder in *bidIDs*. *processBid*(*bid*) *auctionReady priceQuoteHandling*(*bid*)

Der Smart Contract übernimmt die Überprüfung der Validität eines eintreffenden Gebots. Das Gebot gilt als valide, wenn es den gleichen Gebotstyp wie vorige Gebote desselben Bieters aufweist und einen höheren Bietwert, da die Auktionen aufsteigend sind. [H] [1] bidValidbid Let *bidTask* be the task in *bid*. isUpdateBid(bid) Require *bid* type (buy or sell) to be consistent to the bid to be updated. Require *bid* amount to be higher than the bid to be updated. Require *bidTask* to be in *auctionTasks*.

Nachdem festgestellt wurde, dass das Gebot gültig ist, kann es in den Auktionszustand und in die Blockchain eingebaut werden. [H] [1] processBidbid Let *bidTask* be the task in *bid*. isUpdateBid(bid) Update old bid to new bid. *bid* is a sale bid Increase sale offer count for *bidTask* auction. *bid* is first bid of this bidder Add bidder to bidder list for the *bidTask* auction. Increase count of instances bid for by the same bidder. Sort *bid* into *bidList* or move it around in *bidList* if it is an update bid.

auctionReady ist eine Bedingung, die erfüllt ist, wenn die Auktion aus mindestens einem Kauf- und einem Verkaufsgebot besteht. Es muss nämlich möglich sein, Kauf- und Verkaufsgebote zu matchen. Vorher brauchen keine Preisangebote zurückgesendet werden.

priceQuoteHandling ist die Teilfunktion einer Auktion, die berechnet, welche Bieter zu welchem Preis, wie viele Aufgaben gewinnen würden, also die Berechnung und Versendung der Preisangebote. Die höchsten Kaufgebote werden mit den niedrigsten Verkaufgeboten gepaart.

[H] [1] priceQuoteHandlingbid Let *wins* be a structure managing what bidder would win what amount of task instances of the current auction. Let $mIdx$ be the index of the $(m+1)$ highest bid in *bidList* where m is the number of task instances up for auction. Let $m1Price$ be the the value of the bid at $mIdx$ in *bidList*. Let *aboveBids* be the number of bids above the $mIdx$ index in *bidList*. $i \leftarrow 0$ to $mIdx$ while $aboveBids > 0$ `bid[i]` is selling **and** `bid[i].value` $\leq m1Price$ $j \leftarrow aboveBids$ **downto** 1 $aboveBids \leftarrow aboveBids - 1$ `bid[j + mIdx]` is buying Increment the number of task instances the matched bidders would each win in *wins*. *bidder* in *bidderList* Let *pq* be a message containing task, the amount of task instances *bidder* would win according to *wins* with the $m1Price$ and a bid id for the last bid received from *bidder* for task. Send *pq* to *bidder*.

3.3.3 Wahl der Technologien

Als simulierte Blockchain wurde die populäre Ethereum-Plattform [2] gewählt. Sie ist die erste und älteste Smart-Contract-Plattform und daher das reifeste Blockchain-Tooling-Ökosystem. Ethereum bietet eine Turing-vollständige abstrakte Programmiersprache (Solidity [10]) über die Ethereum Virtual Machine (EVM) an, in der Smart Contracts definiert werden können. Es gibt verschiedene Bibliotheken, über die mit Ethereum interagiert werden kann, wie web3.py für Python [14] oder web3.js [13] für JavaScript. Aufgrund persönlicher Fähigkeiten ist ein Anschluss via Python-Bibliothek von Interesse. Agenten, die mit Blockchain oder zentralem Auktionsserver kommunizieren, wurden somit in Python geschrieben. Sowohl Blockchain als auch Auktionsserver werden lokal simuliert, um Echtgeldkosten zu meiden und Vergleichbarkeit durch die Gemeinsamkeit ihrer Lokalität herzustellen. Aufgrund der Lokalität kann der Auktionsserver ebenfalls mit Python realisiert werden.

Blockchain	Launch	SC-Support	TPS	Sicherheit
Bitcoin [3]	2008	Limitiert	7	Extreme Sicherheit aufgrund Alter und Größe
Ethereum [2]	2015	Ausgereift	15-30	großes Blockchain-Netzwerk mit extensivem Testing
Solana [9]	2020	Vorhanden	Tausenderbereich	Erbt Ethereum-Sicherheit
Tron [12]	2017	Vorhanden	Hunderterbereich	stärkere Zentralisierung (27 Super Representatives)
Cardano [4]	2017	Vorhanden	Tausenderbereich	academically designed, peer-reviewed, formally verified security and correctness Proof-of-Stake, definiert Entwicklungs-Ären, aktuell in der letzten Ära

Tabelle 3.1: Blockchain-Auswahl

Tabelle 3.1 zeigt den Vergleich einer limitierten Auswahl von Blockchains anhand einiger Kriterien. Angegeben werden hier das Startdatum (Launch) der Blockchain, Smart-Contract-Unterstützung (SC-Support), Verarbeitung von Transaktionen pro Sekunde (TPS) und Sicherheit. Das Kriterium TPS spielt hier keine so große Rolle, da TPS einer lokalen Simulation nur an Systemhardware gebunden ist. Jedoch lässt sich feststellen, dass die ältesten Blockchains die geringsten TPS aufweisen. Neuere Blockchains ermöglichen TPS im Hunderter- bzw. Tausenderbereich. Sogenannte Layer-2-Blockchains (L2), wie hier Solana, bauen auf Ethereum (Layer 1) auf, um weiterhin von der Sicherheit und Größe der Ethereum-Blockchain zu profitieren, während Mechanismen eingeführt werden, die extrem hohe TPS zulassen. Die Wahl von Ethereum kann somit trotz geringer TPS getroffen werden, um von einem ausgereiften Ökosystem und Popularität profitieren zu können. Ethereum-Messungen können als eine Obergrenze angesehen werden, denn Anwendungen auf Ethereum sind kompatibel mit L2 und daher lässt sich geringe TPS leicht lösen.

Für die Auswahl des Testing-Frameworks zur Simulation einer lokalen Ethereum-Blockchain wurden die Einträge der Tabelle 3.2 in Betracht gezogen. Aufgrund der Instabilität und schlechten Skalierbarkeit wurde auf die Nutzung von Ganache-UI verzichtet. Hardhat

Simulationstool	Stabilität	Benutzerfreundlichkeit	Skalierbarkeit
Ganache-UI [7]	instabil	leichter, visuell	schlecht
Ganache-CLI [7]	stabil	moderat	hoch
Hardhat [8]	stabil	fortgeschritten	hoch

Tabelle 3.2: lokale Ethereum-Simulationstools

ist am fortgeschrittensten, jedoch mit hoher Stabilität und Skalierbarkeit, wie Ganache-CLI. Ganache-CLI bildet nach dieser Einschätzung einen Kompromiss und wurde hier gewählt. Da leicht zwischen Ganache-UI und -CLI gewechselt werden kann, lässt sich zusätzlich bei der Entwicklung auch die GUI von Ganache-UI nutzen.

4 Evaluierung der Blockchain-Anwendung

4.1 Bestimmung der Metriken

Zum Vergleich einer Blockchain-integrierten Variante müssen Vergleichspunkte identifiziert werden. Dafür können folgende Metriken in Betracht gezogen werden:

- Auftrag-Lebenszeit: Zum Vergleich des Original-Algorithmus mit seiner Blockchain-Variante kann die Lebenszeit der gestellten Aufgaben eines Konsumenten betrachtet werden. Damit ist die Zeit gemeint ab Ankündigung der Aufgaben bis hin zur Findung einer validen Aufgaben-Zuweisung. Womöglich könnten sich nämlich Unterschiede in der Laufzeit feststellen lassen, aufgrund der Einbindung von Smart-Contract-Code.
- Verfahrenskosten, wie beispielsweise Gaskosten: Ethereum-Gas ist ein Maß für die Berechnungskosten, die u.a. bei der Ausführung von Smart-Contract-Code auftreten.
- Skalierbarkeit: Die Skalierbarkeit der Verfahren hinsichtlich weiterer Aufgaben und Agenten kann auf die Probe gestellt werden.

Smart-Contract-Umgebung Um Messungen der Blockchain-Applikation anstellen zu können, muss der Smart Contract auf eine Blockchain deployt werden. Dafür können mehrere Optionen in Betracht gezogen werden. Es gibt eine Reihe Blockchains, die Smart Contracts unterstützen, allerdings ist das Einsetzen eines Smart Contracts relativ kostspielig. Daher gibt es kostenfreie Alternativen durch lokale Blockchains oder öffentliche Testnets. Ein Mainnet ist das öffentliche, offizielle Blockchain-Netzwerk und würde realistische Messungen ermöglichen, da aktuelle Netzwerklast und Konsenszeiten in die Messungen mit eingehen würden. Allerdings würden dafür Echtgeldkosten anfallen, die im Rahmen dieser Arbeit gemieden werden. Eine lokale Blockchain ist ein

Blockchain-Netzwerk, das vollständig auf dem eigenen Computer läuft. Simulation einer lokalen Blockchain ist somit eine kostenfreie Alternative. Transaktionen werden sofort validiert und es gibt keine Konsensdelays, daher sind lokale Blockchains sehr gut geeignet zum Ausprobieren und Debuggen des eigenen Smart Contracts. Jedoch fehlen dadurch realistischere Transaktionsverarbeitungszeiten. Ein öffentliches Testnet ist ein separates Blockchain-Netzwerk parallel zum Mainnet, das echte Bedingungen simuliert. Mit einem öffentlichen Testnet können realistische Transaktionsverarbeitungszeiten, Netzwerklasten und Interaktionen mit anderen Smart Contracts simuliert werden. Zudem wird in Testnets wertlose Kryptowährung anstelle der wahren Kryptowährung eingesetzt. Dies hat zum Nachteil, dass Verhalten mit nur wertloser Kryptowährung im Spiel deutlich anders bzw. „abgeschwächter“ ist, als in einem Mainnet. Dennoch ist dies eine gute Umgebung, um seine Smart-Contract-Applikation weiter zu messen und zu optimieren, und eine weitere kostenfreie Alternative. Aus zeitlichen Gründen und da die Teilnahme an Testnets ein Wallet erfordert, wobei scheinbar echte Kryptowährung nachgewiesen werden muss, wird sich zunächst auf die Simulation einer lokalen Blockchain mit Fokus auf Kosten beschränkt. Da die traditionelle Implementierung ebenfalls lokal ist, sind beide Versionen in der Hinsicht auch ebenbürtig für die Messungen.

Messwertabhängigkeiten zum Aufgabennetzwerk Die Messwerte werden in Abhängigkeit vom zugrundeliegenden Aufgabenabhängigkeitsnetzwerk ermittelt. Im Zentrum der Analyse stehen dabei Struktur und Größe des Netzwerks. Es setzt sich zusammen aus Lieferanten, Produzenten, Konsumenten und Aufgaben/Auktionen. Um Trends in den Messwerten zu erkennen, muss das Netzwerk gezielt variiert werden. Relevante Netzwerkvariablen umfassen Agentenzahl, Aufgabenzahl, Teilaufgabenzahl pro Produzent, Tiefe des Netzwerks, Zahl konkurrierender Agenten pro Produktionsaufgabe oder Teilaufgabe (resource contention) sowie Parameter der einzelnen Agenten, wie Preise der Lieferanten, Konsumenten und Delta-, Lambda-Werte der Produzenten. Da die gleichzeitige Variation aller Parameter unhandlich wäre, ist es entscheidend, den Fokus auf einflussreiche Variablen zu legen. Eine Auktion umfasst u.a. einen Preisbestimmungsmechanismus, in unserem Falle die Bestimmung des $(M + 1)$-Preises. Somit sollten größere Auktionen entsprechend direkte Auswirkungen auf die Preisberechnungskosten haben. Stoßen Agenten zu einem existierenden Netzwerk, so können sie an unterschiedlichen Stellen hinzugefügt werden. Da das Netzwerk hierarchisch ist, können neue Agenten entweder eine Breite oder Tiefe des Netzwerks verstärken. Dabei lässt sich unterscheiden, ob der Agent um neue Aufgaben bietet oder für Aufgaben, für die bereits eine Auktion

existiert. Damit kann ein Agent die Größe einer Auktion oder die Anzahl von Auktionen erhöhen. Da Produzenten nach und nach ihre Gebote erhöhen und gleichzeitig Auktionen still werden, wenn ein Gewinner erkennbar wird, kann man annehmen, dass Auktionen weiter oben auf Signale der Auktionen weiter „unten“ (nahe den Lieferanten) warten müssen. Die Idee ist somit, Kosten und Zeit primär in Abhängigkeit von Auktionsgrößen und Netzwerktiefe zu messen.

4.2 Generation von Aufgabenabhängigkeitsnetzwerken

Netzwerkgeneration Nun muss definiert werden, wie sich solche Netzwerke generieren lassen. Es wurden zwei Eigenschaften identifiziert, die hauptsächlich variiert werden sollen: Auktionsgröße, Netzwerktiefe. Erhöhung beider Eigenschaften ist nur möglich durch Erhöhung der Agentenzahl. Die Agentenzahl kann somit als skalierbarer Parameter genutzt werden, um Netzwerke zu generieren bzw. Messwerte in Abhängigkeit zu bestimmen. Ein genauer Generationsalgorithmus stellt jedoch trotz Einschränkung der zu variierenden Eigenschaften eine Herausforderung dar. Soll Agentenzahl hauptsächlich Auktionsgröße beeinflussen, so bestünde das „optimal“ generierte Netzwerk, das auch Produzenten enthält, aus genau zwei riesigen Auktionen. Jedoch könnten solche Grenzfall-Netzwerke zu wenig repräsentativen Messungen führen. Auch wenn der Hauptfokus auf einer Eigenschaft des Netzwerks liegt, sollten andere Eigenschaften weiterhin eine Rolle spielen in der Generation. Eine Generationsmethode für möglichst große Auktionen in Abhängigkeit von der Agentenzahl wurde hierfür implementiert.

Das Startnetzwerk für den Algorithmus besteht aus genau einem Konsumenten, drei Produzenten und drei Lieferanten, sowie 5 Aufgaben (siehe Abbildung 4.1). Dieses Netzwerk hat eine konstante Tiefe von zwei. Eine Tiefe von 1 wäre zu sehr ein Grenzfall, und da für die Messung der Kosten hauptsächlich die Variation der Auktionsgröße als relevant angesehen wird, lässt sich die Generationsmethode durch die Einschränkung auf eine konstante Tiefe somit vereinfachen. Bei einer Tiefe von zwei ist der Effekt durch Bottom-up-Aufgabenzuweisung sozusagen in der Rechnung mit drin.

Hinzukommende Agenten werden alternativ einer der beiden Produzenten-Schichten zugewiesen, damit unterschiedliche Auktionen möglichst gleich groß bleiben. Aus der vorigen Bestimmung, dass es mehrere Auktionen innerhalb einer Ebene geben soll (Breite), können neu generierte Agenten, neue Aufgaben, als Teilaufgaben einführen. Dafür wählt der Agent zufällig aus der Menge Teilaufgaben seiner Schicht plus einer neuen Aufgabe.

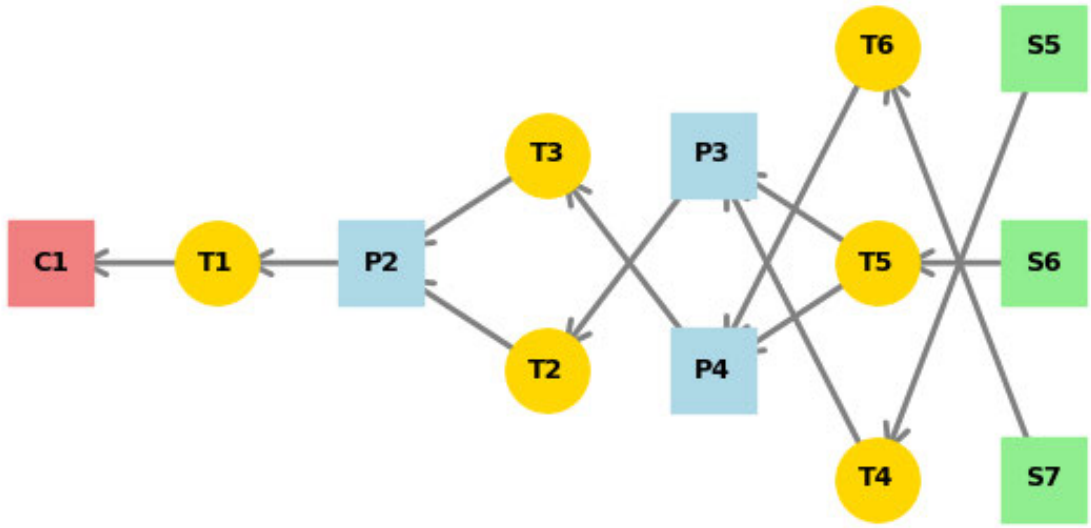

Abbildung 4.1: 3-Produzenten-Gen-Netzwerk

Je mehr Aufgaben in dieser Menge sind, desto unwahrscheinlicher ist es, dass eine neue Aufgabe eingeführt wird. Somit wachsen hauptsächlich die bereits existierenden Auktionen, und das im gleichen Maße. Weiterhin zu klären ist allerdings auch die Anzahl der Teilaufgaben pro Produzent. Da ein Produzent mit mehr Teilaufgaben generell stärker im Nachteil gegenüber Produzenten mit weniger Teilaufgaben ist, vor allem in einem Netzwerk konstanter Tiefe und viel Wettbewerb um dieselben Aufgaben, soll hier die Zahl konstant pro Agent sein. Um wieder möglichst einen Grenzfall zu vermeiden, wird ein konstanter Wert von zwei gewählt. Da es nicht viele Auktionen/Aufgaben gibt, könnte eine zu hohe Teilaufgabenzahl zu geringer Lösungsdiversität führen, da die benötigten Teilaufgaben unterschiedlicher Produzenten, größtenteils gleich wären. Bisher wurden nur Produzenten betrachtet, da diese nach dem angenommenen Bietverhalten aus dem gegebenen Marktprotokoll ihre Gebote steigern. Nach Erstellung beider Produzenten-Schichten werden hinterher Lieferanten pro Aufgabe hinzugefügt. Abbildung 4.2 zeigt ein Beispielnetzwerk, das für 12 Produzenten generiert wurde. C1 repräsentiert den Konsumenten, PX einen Produzenten, TX eine Aufgabe (task), SX einen Lieferanten.

Zuguterletzt werden die Preiswerte hinzugefügt. Der Konsument sollte einen hinreichend großen Reservepreis bieten, damit zumindest eine Aufgabenzuweisung gefunden werden kann. Die Lieferanten haben zufällig generierte kleine Werte über 1. Produzenten inkrementieren ihr Eingabegebot (Delta = 1) und erhalten einen Sicherheitsfaktor in Abhän-

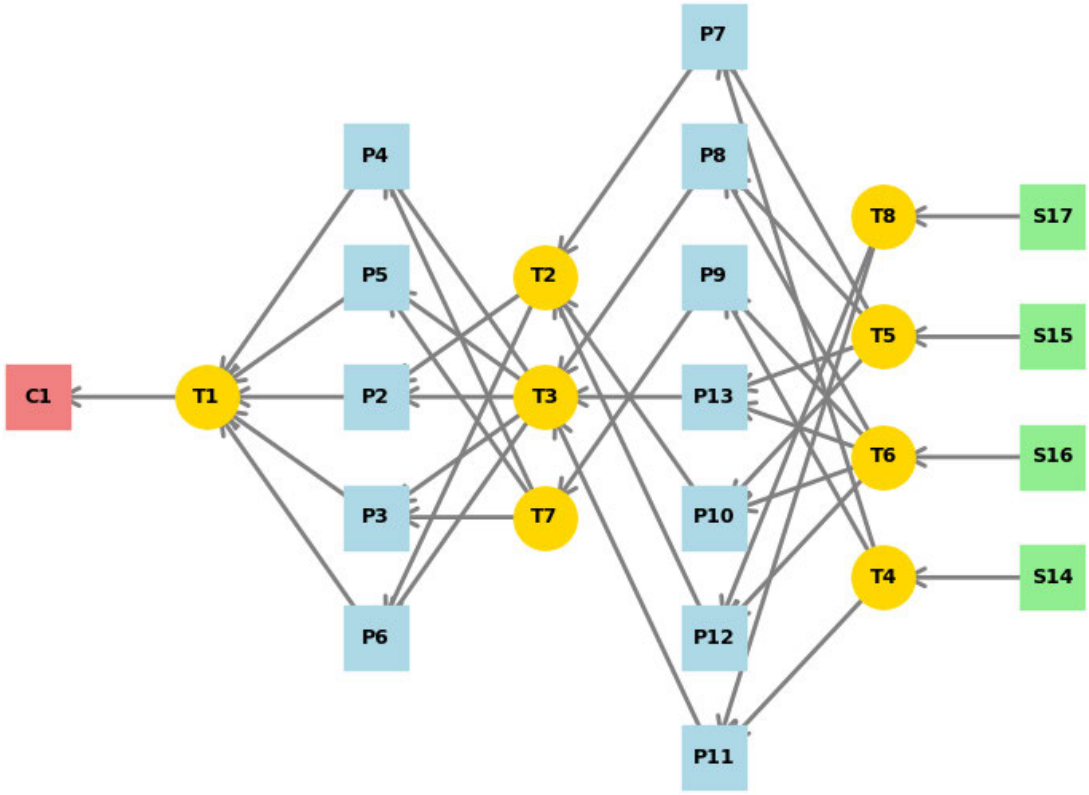

Abbildung 4.2: 12-Agenten-Gen-Netzwerk

gigkeit von Delta, um keine Minusgeschäfte einzugehen, d.h. Lambda = Delta * kleiner Wert über 1.

4.3 Durchführung von Messungen

Messung der Verfahrens-Kosten Im Folgenden wird klargestellt, wie die Verfahrenskosten erhoben werden. Transaktionen verursachen jeweils Gaskosten. In Ethereum, tragen bestimmte ausgeführte Operationen jeweils einen konkreten Gaswert, z. B. Minimum Gas-Einheits-Kosten von 5 für eine MUL-Operation [6]. web3.py ermöglicht die Einsicht der verursachten Gaskosten einer Transaktion im Transaktions-Beleg [14]. Die Kosten einer Aufgabenzuweisung bestimmen sich somit aus der Summe aller Transaktionskosten. Die Gaskosten können in Ether und anschließend in USD zum Vergleich umgerechnet werden. In der Auktionsserver-Variante müssen Kosten anders erhoben werden.

Es lassen sich Ressourcennutzung durch den Auktionsserver messen und diese anhand eines Cloud-Preismodells in vergleichbare Kosten umrechnen. Für die Ressourcennutzung werden CPU-Zeit, benötigter Arbeitsspeicher pro Stunde/Sekunde und übertragene Datenmengen berücksichtigt. Bei der Wahl eines Cloud-Modells gibt es eine große Auswahl. Die Wahl kann eingeschränkt werden durch Betrachtung verschiedener Service-Arten: Infrastructure-as-a-Service (IaaS), Container-as-a-Service (CaaS) etc. Es stellt sich somit die Frage, welche Service-Art passend wäre für den nichtlokalen Auktionsserver. Eine Rolle dabei spielt die Zeit, die der Server verfügbar sein muss. In diesem Falle wartet ein Server auf einen Konsumentenauftrag, sodass Bildung eines Aufgabenabhängigkeitsnetzwerks losgetreten wird. Der Server ist somit ständig verfügbar, was für eine IaaS-Lösung spricht. AWS EC2 ist ein IaaS und der de facto Industriestandard, weshalb für die Kostenberechnung dieses Modell verwendet wird. Aktuelle Preise können aus dem AWS Pricing Calculator bezogen werden [1]. Der betrachtete Instanztyp ist ‚t3.medium‘, welcher für generelle Web-Applikationen gedacht ist, siehe Tabelle 4.1.

CPU-Kosten in $/h	0.048$
GiB-Kosten in $/h	0.012$
Datentransferkosten in $/GB	0.09$

Tabelle 4.1: AWS Preise am 27.09.2024

Ergebnisse Abbildung 4.3 setzt die berechneten Kosten ins Verhältnis für genau eine Aufgabenzuweisung. Alle Kosten sind in USD angegeben. Hierbei werden zusätzlich zu ETH-Kosten und geschätzten Cloudkosten auch noch Solana-Kosten angegeben. Damit der Trend der Cloudkosten im Graph erkennbar bleibt, wurden sie mit 10^3 multipliziert, genauso wie die ETH-Kosten durch 100 geteilt, d.h., alle Abstände zu Cloudkosten und zu ETH-Kosten wären visuell deutlich höher. Somit zeigt sich, dass die ETH-Kosten die geschätzten Cloudkosten sehr stark übersteigen. Da L2-Ethereum-Blockchains, wie Solana, Eigenschaften von Ethereum erben und gleichzeitig Kosten reduzieren, wurden Solana-Kosten ebenfalls berechnet und abgebildet (Informationen zu Solana-Preisen [9]; für die Komplexität der Solana-Transaktionen wird hier von komplexen dApp-Interaktionen ausgegangen).

Eine interessante Eigenschaft von Smart Contracts ist, dass allein ihr Bestehen auf der Blockchain keine Kosten für Smart-Contract-Besitzer oder -Nutzer verursacht. Serverzeit wird hingegen in Cloud-Preisen miteinberechnet. Daraus folgt, dass länger andauernde Auktionen in der Blockchain-Lösung gleichbleibende Kosten mit sich tragen. In der lo-

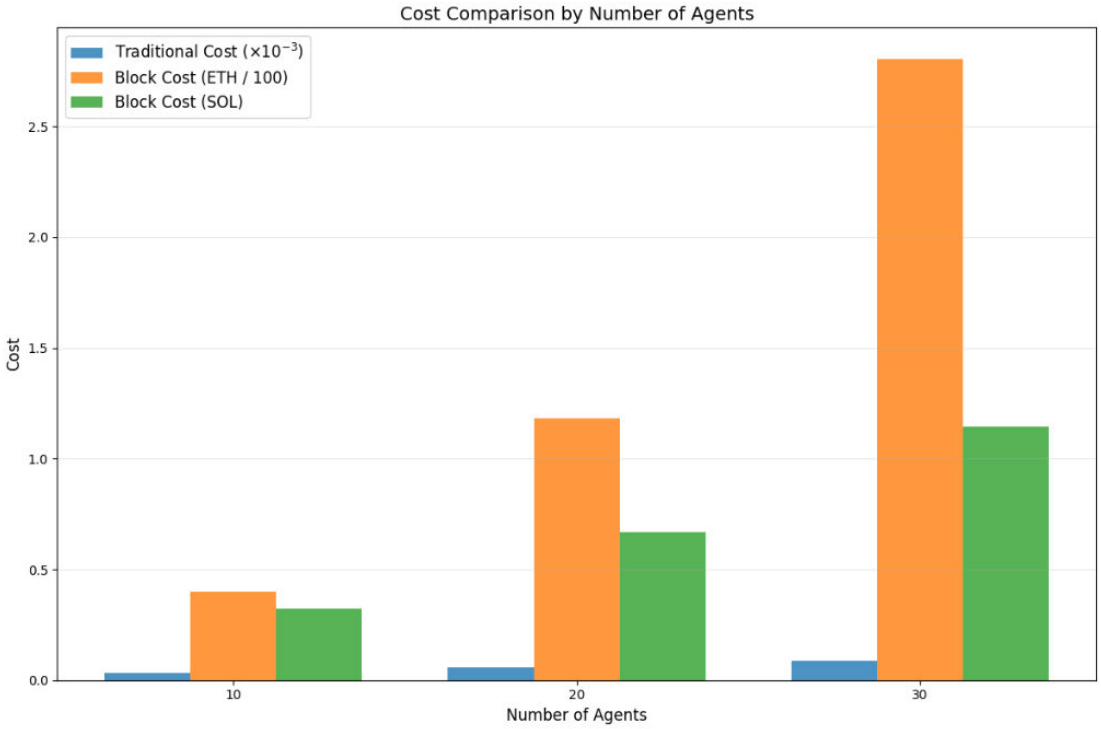

Abbildung 4.3: Kosten abhängig von Agentenzahl für eine Echtzeit-Aufgabenzuweisung

kalen Simulation antworten Agenten und Auktionen in Echtzeit aufeinander, in realen Applikationen, gibt es jedoch Verzögerungen und die Zeit, die Agenten individuell zur Bestimmung ihrer Gebote, ihrer gewollten Aufgabe bzw. Unterteilung in Teilaufgaben benötigen, kann deutlich höher sein. Für die folgende Abbildung 4.4 wird daher angenommen, dass die Auktionsdauer statt weniger Sekunden, genau eine Stunde beträgt. Dadurch ändern sich nur die Schätzungen der Cloudkosten. Die Abbildungen sehen zwar relativ gleich aus, jedoch ist die Angabe der Cloudkosten nun ohne Lesbarkeitsfaktor 1000, und der visuelle Abstand zu Solana-Kosten akkurat.

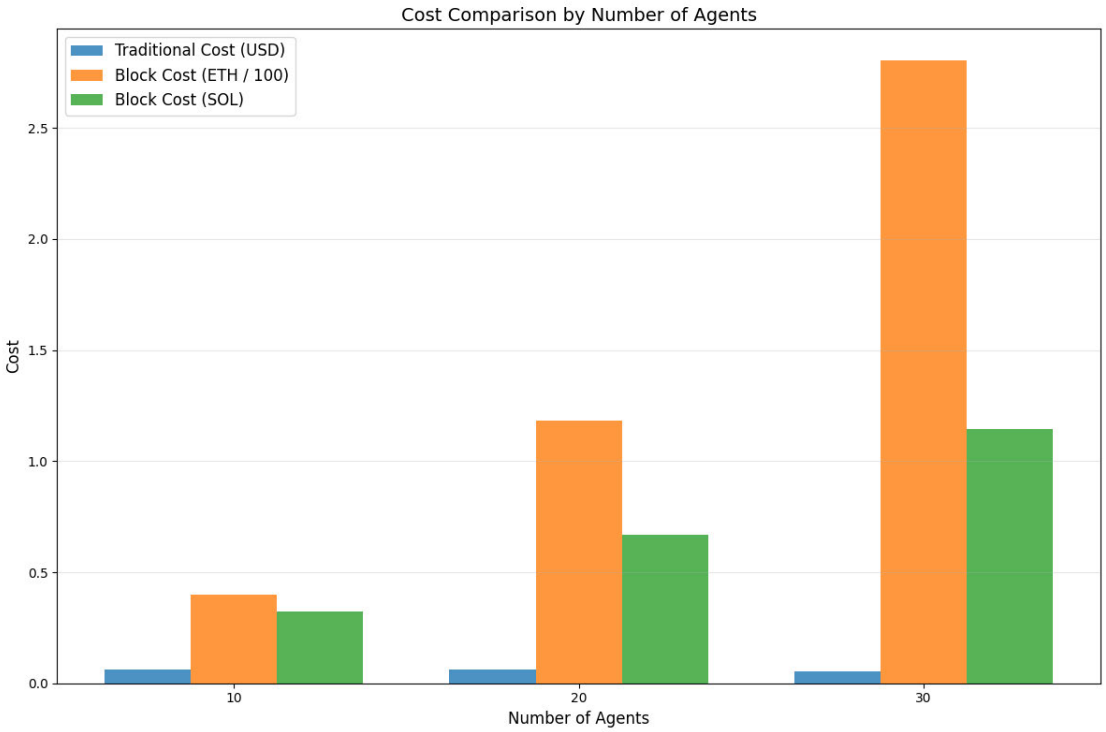

Abbildung 4.4: Kosten abhängig von Agentenzahl für eine 1-stündige Aufgabenzuweisung

5 Fazit & Ausblick

Es wurde gezeigt, dass eine Transformation eines traditionellen, zentralisierten Ansatzes in eine dezentrale Blockchain-Lösung technisch machbar ist. Dafür wurden der Prozess zur Auswahl eines geeigneten traditionellen Ansatzes erläutert und durchgeführt, dieser Ansatz von Grund auf in Python implementiert, sowie in eine Blockchain-Variante überführt, die ebenfalls von Grund auf für Ethereum in Solidity implementiert wurde. Zudem bietet diese Arbeit eine Gegenüberstellung der Architekturen anhand der entscheidenden Kriterien, Kosten und Skalierbarkeit. Dabei wurden Echtgeldkosten vermieden und erklärt, welche Möglichkeiten trotz dieser Limitation bestehen. In der Kostenbetrachtung erwies sich die Blockchain-Lösung durchweg als teurere Alternative, auch bei Betrachtung neuester Blockchain-Technologie. Dies führt zu der Schlussfolgerung, dass der Einsatz einer Blockchain durch einen konkreten Mehrwert, beispielsweise durch erhöhte Sicherheit, gerechtfertigt werden muss.

Für die zukünftige Forschung bietet sich die Untersuchung weiterer und fortgeschrittenerer Blockchain-Technologien an, wie Non-fungible Token (NFT), Layer-2-Lösungen und Kryptowährung. Es könnte untersucht werden, inwiefern diese sich in Aufgabenzuweisungsprobleme integrieren lassen oder wie sie Skalierungs- und Kostenprobleme adressieren können. Eine Kostenabwägung, die Blockchain-Vorteile wie Sicherheit geldlich bewertet, könnte zu konkreteren Abschätzungen der praktischen Nutzbarkeit von Blockchain führen. Dabei kann sich auf konkrete Anwendungsgebiete bezogen werden, die unterschiedliche Anforderungen fokussieren, um konkretere Fazite zur Nutzung von Blockchain zu ermöglichen. Eine weitere Möglichkeit ist es, speziell Algorithmen zu untersuchen, die nicht nur in eine Blockchain speichern, sondern auch selbst von ihr lesen, wie womöglich mit lernbasierten Systemen. Die Rolle von Blockchain-Query-Zeiten kann untersucht werden, vor allem bei länger werdender Blockchain und gesuchter Informationen, die innerhalb der Blockchain unsortiert und verteilt sind. Zu guter Letzt wären eine Untersuchung und ein Vergleich andersartiger Blockchains interessant.

Literaturverzeichnis

[1] : *AWS pricing calculator*. – URL https://calculator.aws/#/

[2] : *Beigepaper: An Ethereum Technical Specification*. – URL https://github.com/chronaeon/beigepaper/

[3] : *bitcoin paper*. – URL https://bitcoin.org/en/bitcoin-paper

[4] : *Cardano*. – URL https://docs.cardano.org/about-cardano/explore-more/relevant-research-papers

[5] : *Everledger Concept Paper*. – URL https://everledger.io/insights/concept-papers/

[6] : *EVM codes*. – URL https://www.evm.codes/

[7] : *Ganache*. – URL https://archive.trufflesuite.com/docs/ganache/

[8] : *Hardhat*. – URL https://hardhat.org/

[9] : *Solana*. – URL https://solana.com/

[10] : *Solidity*. – URL https://soliditylang.org/

[11] : *Storj Whitepaper*. – URL https://github.com/Storj/whitepaper

[12] : *Tron*. – URL https://tron.network/

[13] : *web3.js*. – URL https://docs.web3js.org/

[14] : *web3.py*. – URL https://web3py.readthedocs.io/en/latest/

[15] Abou Jaoude, Joe ; George Saade, Raafat: Blockchain Applications – Usage in Different Domains. In: *IEEE Access* 7 (2019), S. 45360–45381

[16] An, Bo ; Lesser, Victor ; Irwin, David ; Zink, Michael: Automated negotiation with decommitment for dynamic resource allocation in cloud computing. In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1 - Volume 1*. Richland, SC : International Foundation for Autonomous Agents and Multiagent Systems, 2010 (AAMAS '10), S. 981–988. – ISBN 9780982657119

[17] Baségio, Túlio L. ; Bordini, Rafael H.: An Algorithm for Allocating Structured Tasks in Multi-Robot Scenarios. In: Jezic, Gordan (Hrsg.) ; Kusek, Mario (Hrsg.) ; Chen-Burger, Yun-Heh J. (Hrsg.) ; Howlett, Robert J. (Hrsg.) ; Jain, Lakhmi C. (Hrsg.): *Agent and Multi-Agent Systems: Technology and Applications, 11th KES International Conference, KES-AMSTA 2017, Vilamoura, Algarve, Portugal, June 21-23, 2017, Proceedings* Bd. 74, Springer, 2017, S. 99–109

[18] Baumann, Christian: In: *TeleTrust-Positionspapier „Blockchain". Handreichung zum Umgang mit der Blockchain. TeleTrusT - Bunderverband IT-Sicherheit e.V*, 4 2017

[19] Belotti, Marianna ; Bozic, Nikola ; Pujolle, Guy ; Secci, Stefano: A Vademecum on Blockchain Technologies: When, Which, and How. In: *IEEE Commun. Surv. Tutorials* 21 (2019), Nr. 4, S. 3796–3838

[20] Choi, Han-Lim ; Brunet, Luc ; How, Jonathan P.: Consensus-Based Decentralized Auctions for Robust Task Allocation. In: *IEEE Transactions on Robotics* 25 (2009), August, S. 912–926. – ISSN 1941-0468

[21] Gerkey, Brian P. ; Mataric, Maja J.: A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. In: *Int. J. Robotics Res.* 23 (2004), Nr. 9, S. 939–954

[22] Khanizad, Rahim ; Montazer, Gholamali: Optimal allocation of human resources based on operational performance of organizational units using fuzzy game theory. In: *Cogent Engineering* 5 (2018), Nr. 1, S. 1466382. – URL https://doi.org/10.1080/23311916.2018.1466382

[23] Klemperer, Paul: Auction Theory: A Guide to the Literature / University Library of Munich, Germany. URL https://ideas.repec.org/p/wpa/wuwpmi/9903002.html, Mar 1999 (9903002). – Microeconomics

[24] Kołodziej, Joanna ; Xhafa, Fatos: Modern approaches to modeling user requirements on resource and task allocation in hierarchical computational grids. In: *Int. J. Appl. Math. Comput. Sci.* 21 (2011), Juni, Nr. 2, S. 243–257. – ISSN 1641-876X

[25] Kong, Yan ; Zhang, Minjie ; Ye, Dayong: A belief propagation-based method for task allocation in open and dynamic cloud environments. In: *Knowl. Based Syst.* 115 (2017), S. 123–132

[26] Przytarski, Dennis ; Stach, Christoph ; Gritti, Clémentine ; Mitschang, Bernhard: Query Processing in Blockchain Systems: Current State and Future Challenges. In: *Future Internet* 14 (2022), Nr. 1, S. 1

[27] Shehory, Onn ; Kraus, Sarit: Methods for Task Allocation via Agent Coalition Formation. In: *Artif. Intell.* 101 (1998), Nr. 1-2, S. 165–200

[28] Skaltsis, George M. ; Shin, Hyo-Sang ; Tsourdos, Antonios: A survey of task allocation techniques in MAS. In: *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2021, S. 488–497

[29] Smith: The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. In: *IEEE Transactions on Computers* C-29 (1980), Nr. 12, S. 1104–1113

[30] Szabo, Nick: Formalizing and Securing Relationships on Public Networks. In: *First Monday* 2 (1997), Nr. 9

[31] Walsh, William ; Wellman, Michael ; Ygge, F.: Combinatorial Auctions for Supply Chain Formation. (2000), 10

[32] Walsh, William E. ; Wellman, Michael P.: Decentralized Supply Chain Formation: A Market Protocol and Competitive Equilibrium Analysis. In: *J. Artif. Intell. Res.* 19 (2003), S. 513–567

[33] Yaga, Dylan ; Mell, Peter ; Roby, Nik ; Scarfone, Karen: *Blockchain technology overview*. URL http://dx.doi.org/10.6028/NIST.IR.8202, Oktober 2018

[34] Zhao, Xinyi ; Zong, Qun ; Tian, Bailing ; Zhang, Boyuan ; You, Ming: Fast task allocation for heterogeneous unmanned aerial vehicles through reinforcement learning. In: *Aerospace Science and Technology* 92 (2019), S. 588–594. – URL https://www.sciencedirect.com/science/article/pii/S1270963818318704. – ISSN 1270-9638

A Anhang

A.1 Verwendete Hilfsmittel

In der Tabelle A.1 sind die im Rahmen der Bearbeitung des Themas der Bachelorarbeit verwendeten Werkzeuge und Hilfsmittel aufgelistet.

Tabelle A.1: Verwendete Hilfsmittel und Werkzeuge

Tool	Verwendung
LaTeX	Textsatz- und Layout-Werkzeug verwendet zur Erstellung dieses Dokuments
PyCharm	Entwicklungsumgebung verwendet zur Entwicklung in Python
Python	Programmiersprache verwendet zur Entwicklung von Systemkomponenten
Solidity	Programmiersprache verwendet zur Entwicklung eines Smart Contracts
Ganache-CLI	Framework verwendet zur Simulation einer lokalen Ethereum-Blockchain

Erklärung zur selbständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort Datum Unterschrift im Original