Centre Énergie, Matériaux et Télécommunications

Blockchain-Empowered Crowdsourcing and Marketplace Design for the Metaverse

Par

Le Duy Hung

Thèse présentée pour l'obtention du grade de
Maître ès Sciences (M.Sc.)
en sciences de Télécommunications

Jury d'évaluation

Examinateur externe	Prof. Ping Wang York University
Examinateur interne	Prof. Zakaria Abou El Houda INRS-ÉMT
Directeur de recherche	Prof. Long Bao Le INRS-ÉMT

Acknowledgments

First and foremost, I am deeply grateful to my supervisor, Professor Long Bao Le, for his invaluable support and motivation throughout my MSc studies at INRS-EMT, University of Québec. Above all, I would like to express my sincere appreciation for his consistent guidance and encouragement. Professor Le has consistently exhibited patience in guiding my academic progress and has shown remarkable empathy toward my individual circumstances. His profound expertise and passion for research have served as an enduring source of inspiration to me.

I am also grateful to Professor Zakaria Abou El Houda of INRS-ÉMT, who provided valuable feedback and reviewed my dissertation as a member of my MSc committee. I extend my appreciation to Professor Ping Wang from York University for serving as an external examiner and providing constructive feedback to enhance the technical quality of my dissertation.

I would like to acknowledge my colleagues at the Networks and Cyber-Physical Systems Lab (NECPHY-Lab), INRS-EMT, University of Québec, and friends: Dat Nguyen, Tung Phan, Vu Truong, Duc Hoang, Vu Nguyen, Luan Dang, Harry Bui, and Toan Do, for their recommendations in research and for the memorable life moments during my time in Canada.

Last but not least, I am deeply grateful to my family members, including my parents, and sister, for their unwavering support and encouragement in all aspects of my life. Their belief in me and their constant presence has been instrumental in my achievements.

Abstract

The metaverse is envisioned as a fully immersive, hyper-spatial, and self-sustaining virtual space merging physical, human, and digital realms with an interconnected economic structure. This burgeoning paradigm shift in internet technology is facilitated by advancements in digital twins, VR/AR, 5G wireless communications, artificial intelligence (AI), and blockchain technologies. Among these, AI is pivotal, enabling the virtual realm through technologies like computer vision (CV) and machine learning (ML). These technologies are employed by metaverse service providers (MSPs) to deliver sophisticated virtual services to metaverse users (MUs). However, challenges in data collection and privacy concerns necessitate a decentralized platform for exchanging metaverse data and ML models in a trustless environment.

To address these challenges, this thesis proposes MetaAICM, a blockchain-empowered framework that facilitates AI crowdsourcing and provides a decentralized marketplace for intelligent virtual services in the metaverse. MetaAICM enables MUs to proactively collect metaverse data or train ML models for sale. If the desired data or models are unavailable, MSPs can use the crowdsourcing mode to source ML models from machine learning workers (MLWs) or request metaverse data from data workers (DWs).

Unlike existing frameworks, MetaAICM eliminates the need for task requesters to provide evaluation functions for worker contributions, thus removing trust assumptions. A concrete incentive mechanism motivates MUs to contribute computational resources, while an integrated reputation system filters out malicious entities. MetaAICM ensures payment is finalized only when the traded product is verified, enhancing trust and security.

The framework incorporates a reputation-based Raft consensus protocol to address scalability, transaction speed, and sustainability. Extensive evaluations demonstrate that MetaAICM can resist security threats such as denial of service (DoS), single point of failure (SPoF), data leakage, and Sybil attacks. It offers high performance and automation with low operational costs, significantly advancing the integration of blockchain technology in the metaverse.

Contents

List of Figures

List of Algorithms

List of Abbreviations

AI	Artificial Intelligence
AP	Access Points
AR	Augmented Reality
ABAC	Attribute-Based Access Control
BCI	Brain-Computer Interface
BFT	Byzantine Fault Tolerant
CV	Computer Vision
DAG	Directed Acyclic Graph
DApp	Decentralized Application
DC	Data Contract
DP	Differential Privacy
DoS	Denial of Service
DPaaS	Data Processing-as-a-Service
DSs	Data Sellers
DT	Digital Twin
DW	Data Worker
EVM	Ethereum Virtual Machine
FL	Federated Learning
IBE	Identity-Based Encryption
IoT	Internet of Things
IPFS	InterPlanetary File System
ML	Machine Learning
MLC	Machine Learning Contract

MLW	Machine Learning Worker
MR	Mixed Reality
MSP	Metaverse Service Provider
MSs	Model Sellers
MU	Metaverse User
MPC	Marketplace Contract
MTBF	Mean Time Between Failures
NFT	Non-Fungible Token
P2P	Peer-to-Peer
PBFT	Practical Byzantine Fault Tolerance
PoA	Proof of Authority
PoD	Proof of Delivery
PoS	Proof of Stake
PoW	Proof of Work
RC	Reputation Contract
SGX	Software Guard Extensions
SMC	Secure Multi-party Computation
SPoF	Single Point of Failure
TPS	Transactions Per Second
TSNs	Trusted Seed Nodes
UAV	Unmanned Aerial Vehicle
VSPs	Virtual Service Providers
VR	Virtual Reality
VRF	Verifiable Random Function
ZKP	Zero-Knowledge Proofs

Chapter 1

Résumé Long

Ce chapitre est le résumée en français de la thèse intitulée:

"Crowdsourcing et Marché Renforcés par la Blockchain pour le Métavers"

1.1 Contexte et Motivations

Le Métavers, un terme dérivé du préfixe "méta" (signifiant au-delà) et "vers" (une forme abrégée de l'univers), représente un royaume numérique parallèle qui reflète un environnement du monde réel avec une structure économique indépendante, interconnectée avec le monde physique. Depuis sa première apparition, le concept de métavers a évolué sous diverses formes, telles qu'une seconde vie [1], des mondes virtuels 3D [2] et le lifelogging [3]. Il est souvent envisagé comme un espace partagé virtuel entièrement immersif, hyper-spatial et auto-suffisant, fusionnant les domaines physique, humain et numérique [4]. Il est reconnu comme un changement de paradigme émergent dans la technologie Internet, succédant aux révolutions du Web et de l'internet mobile [5], permettant aux individus de s'immerger en tant que natifs numériques et d'explorer une existence alternative dans un cadre virtuel.

Le métavers incorpore une pléthore de technologies émergentes [6], [7], [8]. Notamment, les jumeaux numériques répliquent les environnements du monde réel ; la réalité virtuelle (VR) et la réalité augmentée (AR) facilitent des expériences 3D immersives ; les communications avancées

comme la 5G et au-delà des technologies sans fil assurent des connexions ultra-fiables et à faible latence pour une multitude de dispositifs au sein du métavers ; les capteurs portables et les interfaces cerveau-ordinateur (BCI) améliorent les interactions entre les utilisateurs et les avatars ; l'intelligence artificielle (IA) soutient la création et le rendu à grande échelle des environnements du métavers ; et la technologie blockchain avec les jetons non fongibles (NFTs) sécurisent les droits de propriété vérifiables sur les actifs du métavers [9]. Avec l'adoption généralisée des appareils intelligents et la maturation de ces technologies habilitantes, le métavers passe d'un stade naissant à une réalisation dans un futur proche. De plus, des innovations technologiques significatives et des avancées favorisent une nouvelle écologie de l'information et créent de nouvelles demandes pour des applications dans divers domaines tels que l'éducation, le jeu, le travail et l'interaction sociale, positionnant le métavers comme une plate-forme fondamentale pour cet écosystème émergent [8].

Parmi les diverses technologies avancées, l'IA joue un rôle crucial dans la facilitation du royaume virtuel du métavers [7,10]. Par exemple, les algorithmes de vision par ordinateur (CV) exploitent les données du monde réel capturées par des caméras déployées dans les rues de la ville, les bâtiments et sur les véhicules aériens sans pilote (UAVs) pour construire l'environnement virtuel du métavers. De plus, les modèles d'apprentissage automatique (ML) déployés sur les dispositifs portables analysent les données des utilisateurs, telles que l'apparence, les gestes et les expressions faciales, pour refléter les comportements des utilisateurs dans les actions des avatars [11]. Bien que les éditeurs de métavers gèrent généralement ces tâches, les fournisseurs de services de métavers (MSP) utilisent habituellement le ML pour fournir des services virtuels aux utilisateurs du métavers (MUs). Par exemple, les MSP peuvent utiliser des modèles d'IA générative pour créer du contenu généré par IA, comme des objets virtuels, des vêtements et des décorations pour le métavers [12]. De plus, les organisateurs d'événements peuvent employer des modèles de détection pour identifier les MUs non autorisés accédant aux événements virtuels en analysant leurs profils, tels que la réputation, la localisation et l'historique opérationnel.

Les MSP seraient intéressés à exploiter l'expertise en IA/ML et les ressources computationnelles de la communauté du métavers pour construire et entraîner leurs modèles de ML souhaités. Inversement, les professionnels du ML capables de concevoir et d'entraîner des modèles de ML peuvent rencontrer des difficultés à collecter les données nécessaires en raison des préoccupations de confidentialité et de la nature à grande échelle et hétérogène du métavers. Il est donc essentiel de fournir aux MUs et MSP une plate-forme décentralisée soutenant l'échange et le commerce de données de

métavers et de modèles de ML dans un environnement sans confiance, distribué, avec des frais de transaction faibles, une haute performance et la préservation de la confidentialité.

Les systèmes de commerce et de crowdsourcing traditionnels offerts par des autorités tierces souffrent souvent de points de défaillance uniques (SPoF) et de problèmes de confiance, y compris les attaques de passagers clandestins et de faux rapports parmi les participants. De plus, ces systèmes manquent généralement de transparence, de confidentialité et de mécanismes d'incitation [13]. Bien que ces limitations puissent être acceptables pour certaines plates-formes centralisées conventionnelles, elles sont inadéquates pour l'environnement décentralisé du métavers, où les escroqueries et les fraudes sont exacerbées par diverses nouvelles attaques d'ingénierie sociale [14]. La technologie blockchain offre une solution potentielle pour la gestion de la confiance dans de tels systèmes de commerce de métavers avec ses propriétés immuables, transparentes et auditées [15]. Spécifiquement, la blockchain fournit un environnement de stockage fiable pour enregistrer les transactions commerciales, tandis que les contrats intelligents et les comités de la blockchain peuvent remplacer les autorités tierces dans la prise de décision, l'application des politiques, la gestion de la réputation et des incitations [16].

Motivé par ces besoins urgents, cette thèse propose un cadre renforcé par la blockchain nommé MetaAICM qui facilite le crowdsourcing de l'IA et fournit un marché pour les services virtuels intelligents dans le métavers. Dans MetaAICM, les MUs peuvent exploiter les capacités de leurs appareils et les ressources disponibles pour collecter des données ou entraîner des modèles de ML à des fins spécifiques, puis les mettre en vente sur le marché de MetaAICM. Si les MSP ne peuvent pas trouver les données ou les modèles requis sur le marché, ils peuvent initier une tâche de crowdsourcing sur les applications décentralisées de MetaAICM pour obtenir les produits désirés d'autres MUs. MetaAICM assure que le paiement n'est finalisé que lorsque le produit échangé est vérifié, éliminant ainsi le besoin de confiance mutuelle entre les parties. Pour aborder la scalabilité, la vitesse des transactions et la durabilité, MetaAICM intègre un protocole de consensus Raft basé sur la réputation. Une description détaillée de nos contributions à la recherche est donnée ci-après.

1.2 Contributions de la Recherche

Cette thèse aborde plusieurs lacunes critiques dans la littérature concernant l'évaluation et le commerce des données de métavers et des modèles de ML dans des environnements décentralisés. Les principales lacunes incluent l'absence de mécanismes d'évaluation pratiques pour les données et les modèles de ML contribués, la dépendance à des hypothèses de confiance, et les vulnérabilités aux menaces de sécurité dans les cadres existants basés sur la blockchain. Pour surmonter ces défis, cette thèse propose MetaAICM, un cadre renforcé par la blockchain conçu pour faciliter le crowdsourcing de ML et un marché décentralisé pour les services intelligents dans le métavers. Les contributions novatrices de MetaAICM peuvent être résumées comme suit:

- **Marché Décentralisé et Crowdsourcing** : MetaAICM établit un marché décentralisé basé sur la blockchain qui encourage les utilisateurs du métavers (MUs) à collecter de manière proactive des données de métavers ou à entraîner des modèles de ML pour les vendre. Si les données et les modèles souhaités ne sont pas disponibles sur le marché, le mode de crowdsourcing permet aux fournisseurs de services de métavers (MSPs) de sourcer des modèles de ML auprès des travailleurs de l'apprentissage automatique (MLWs), tandis que les MLWs peuvent demander des données de métavers aux travailleurs de données (DWs).

- **Élimination des Hypothèses de Confiance** : Contrairement aux cadres existants habilités par la blockchain qui dépendent des demandeurs de tâches pour fournir des fonctions d'évaluation pour évaluer les contributions des travailleurs, MetaAICM ne nécessite pas de telles hypothèses. La conception décentralisée élimine le besoin de confiance entre les participants, garantissant qu'aucune partie unique ne détient un pouvoir ou une responsabilité excessifs, répondant ainsi au besoin d'éviter les mécanismes d'évaluation basés sur la confiance.

- **Mécanismes d'Incitation et de Réputation** : Un mécanisme d'incitation concret est conçu pour motiver les MUs à contribuer des ressources computationnelles, permettant ainsi de renforcer les services de ML dans le métavers. De plus, un système de réputation intégré aide à filtrer les entités malveillantes et à assurer une participation honnête.

- **Sécurité et Performance** : MetaAICM est démontré résister à diverses menaces de sécurité, telles que les attaques par déni de service (DoS), les points de défaillance uniques (SPoF), la fuite de données et les attaques Sybil. Des résultats numériques étendus confirment que

MetaAICM offre une haute performance et une automatisation avec des coûts opérationnels faibles.

Les détails de notre conception proposée sont décrits ci-après.

1.3 Modèle de Système et Vue d'Ensemble du Design

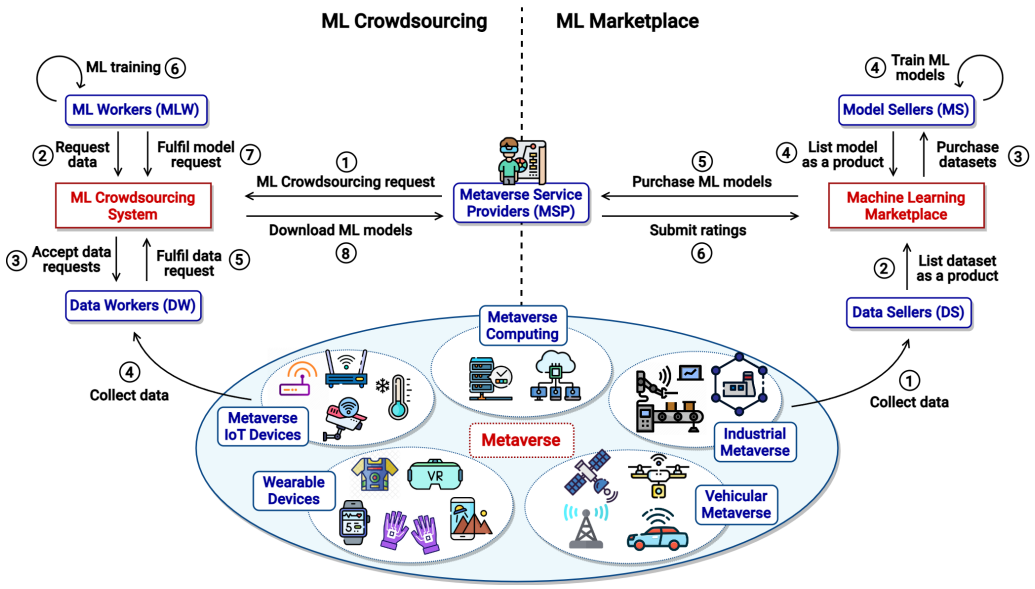

Figure 1.1: Aperçu du métavers et de l'architecture de MetaAICM avec deux modes de fonctionnement, à savoir le système de crowdsourcing et le marché décentralisé.

1.3.1 Modèle de Système et Objectifs du Design

Comme illustré à la Fig.1.1, le métavers est envisagé pour tirer parti de nombreux dispositifs (par exemple, des capteurs IoT, des véhicules intelligents, des dispositifs industriels, des dispositifs portables AR/VR et haptics) et de l'infrastructure informatique (par exemple, des serveurs edge/cloud et des centres de données) pour son fonctionnement. Plus précisément, ces dispositifs collectent et génèrent en continu une quantité massive de données, ce qui est particulièrement précieux pour les outils et services activés par l'IA/ML [17]. Cependant, ces dispositifs et les données collectées sont souvent possédés par de nombreux MUs distribués au lieu d'un éditeur centralisé comme les plateformes médiatiques traditionnelles. Par conséquent, cela pose des défis significatifs en matière de collecte de données, y compris des problèmes de confidentialité [18], de gestion de la confiance

et d'incitations pour les collecteurs de données. Par exemple, les MUs peuvent ne pas vouloir divulguer leurs données largement au public (c'est-à-dire, problème de confidentialité), tandis que des participants malveillants pourraient essayer d'obtenir les données sans payer de jetons (c'est-à-dire, problème de confiance).

Notre conception du système de crowdsourcing et du marché vise à atteindre plusieurs objectifs. Premièrement, nous visons à concevoir des cadres décentralisés qui permettent le commerce ou le crowdsourcing des données de métavers et des modèles de ML pour des services virtuels intelligents dans le monde numérique. Deuxièmement, notre objectif est de traiter efficacement les problèmes de sécurité, de confidentialité et de gestion de la confiance qui sont inhérents à un environnement sans confiance, et qui n'ont pas été suffisamment traités dans les solutions existantes. Enfin, nous nous efforçons de garantir que les conceptions proposées atteignent les niveaux souhaités de scalabilité, de décentralisation et de sécurité tout en restant durables, en évitant la consommation d'énergie significative associée à des opérations comme celles de la blockchain Bitcoin.

À cette fin, nos conceptions proposées reposent sur une blockchain de consortium utilisant un mécanisme de consensus basé sur Raft, comme nous l'expliquerons plus en détail dans la section suivante. Dans ce qui suit, le terme "comité blockchain" est utilisé pour désigner un groupe de nœuds de consensus blockchain.

1.3.2 Vue d'Ensemble du Design

Nous proposons des conceptions de crowdsourcing et de marché pour différents scénarios d'application.

1.3.2.1 Aperçu de la Conception de Crowdsourcing

Dans les systèmes de crowdsourcing, les collecteurs de données sont appelés travailleurs de données (DWs), tandis que les professionnels du ML qui offrent des services de formation en ML sont des travailleurs de ML (MLWs). Il existe trois principaux contrats intelligents régulant le processus de crowdsourcing dans notre cadre proposé. Le contrat de Machine Learning (MLC) permet aux MSPs de faire du crowdsourcing pour les modèles de ML auprès des MLWs. Le contrat de données (DC) permet aux MLWs de faire du crowdsourcing pour les données de métavers auprès des DWs.

Le contrat de réputation (RC) gère les profils de réputation des participants et peut être considéré comme une référence indiquant la fiabilité des individus.

Le fonctionnement du mode de crowdsourcing est illustré sur le côté gauche de la Fig. 1.1 avec un total de 8 étapes. Plus précisément, pour faire du crowdsourcing d'un modèle de ML auprès des MUs, un MSP_i soumet d'abord une demande de crowdsourcing de ML au contrat intelligent MLC (étape 1). Le MSP_i dépose également auprès de MLC un certain nombre de jetons de métavers comme récompense pour les travailleurs. Ensuite, les MLWs intéressés peuvent accepter la demande de MLC et commencer à entraîner le modèle demandé. Pendant le processus de formation, les MLWs peuvent également demander plus de données de métavers aux DWs si nécessaire. Pour ce faire, un MLW_j doit soumettre une demande de données au contrat intelligent DC (étape 2), ce qui déclenche une nouvelle tâche de crowdsourcing de données. Après que certains DWs aient collecté des données et répondu à la demande de données (étapes 4 et 5), le MLW_j peut utiliser les données obtenues pour entraîner son modèle (étape 6). Enfin, une fois la tâche de crowdsourcing de ML terminée (étape 7), le MSP_i téléchargera le modèle de ML final à partir de MLC, et les jetons déposés seront automatiquement distribués aux travailleurs honnêtes.

1.3.2.2 Aperçu du Marché

Au lieu d'attendre des demandes spécifiques de crowdsourcing, les MUs peuvent également collecter de manière proactive des données de métavers pour constituer leurs propres ensembles de données et les mettre en vente sur le marché. Ces MUs sont appelés vendeurs de données (DSs). De même, certains vendeurs de modèles (MSs) peuvent concevoir/entraîner des modèles de ML qui répondent aux besoins du métavers, puis les vendre sur le marché avec un prix et des descriptions prédéfinis. L'arrière-plan du marché est construit sur un contrat intelligent appelé contrat de marché (MPC). Le fonctionnement général du marché de MetaAICM est décrit sur le côté droit de la Fig. 1.1.

Pour préserver la confidentialité, les modèles de ML/données sont stockés sur le système de fichiers interplanétaire (IPFS) [19] au lieu d'utiliser le stockage sur la chaîne. De plus, l'URL de l'IPFS est cryptée par la clé publique de l'acheteur. Dans le cas où un acheteur n'est pas satisfait du produit reçu (par exemple, il affirme que le vendeur a envoyé des données incorrectes ou que les données ont été modifiées), un mécanisme de comparaison en deux étapes est activé pour valider les rapports. En conséquence, les acheteurs malveillants perdront leurs jetons en raison des rapports

de contestation malhonnêtes, tandis que la transaction est annulée automatiquement si le rapport est vérifié comme étant honnête. Lorsque le paiement est finalisé, l'acheteur peut soumettre une évaluation pour le produit avec une note de 0 à 10, ce qui reflète la qualité du produit et impacte également la réputation du vendeur.

1.3.3 Modèle de Menace

Dans MetaAICM, toute entité, y compris les DWs, MLWs, DSs, et MSs, peut agir de manière malveillante, cherchant à maximiser leur bénéfice par des activités malhonnêtes ou la collusion. Sans une autorité de confiance, les opérations peuvent être soudainement corrompues, conduisant à diverses attaques potentielles.

Attaque par Faux Rapport: Dans le crowdsourcing, un demandeur peut prétendre faussement ne pas avoir reçu un ensemble de données ou un modèle de haute qualité pour éviter de payer les frais associés. De même, sur le marché, les acheteurs pourraient faussement signaler des produits de mauvaise qualité ou inaccessibles, nuisant aux vendeurs et gagnant illégalement.

Attaque de Passager Clandestin: Dans le système de crowdsourcing, un travailleur pourrait soumettre des données/modèles arbitraires ou de mauvaise qualité pour obtenir des récompenses sans réel effort, affectant les bénéfices du demandeur. Sur le marché, les vendeurs peuvent tromper les acheteurs en vendant des produits de mauvaise qualité, sapant la confiance dans le système.

Attaque de Sybil et DoS: Une attaque de Sybil consiste à créer plusieurs fausses identités pour manipuler le système, comme soumettre plusieurs évaluations malhonnêtes ou submerger la blockchain de demandes, ce qui peut conduire à une attaque par déni de service (DoS).

Fuite de Confidentialité: Dans le crowdsourcing, les ensembles de données soumis peuvent être consultés par d'autres entités en raison de la transparence de la blockchain, conduisant à une utilisation abusive potentielle d'informations sensibles. De plus, des hackers pourraient attaquer l'environnement de stockage du marché pour voler des données/modèles sans payer, compromettant la confidentialité et les bénéfices des travailleurs.

1.4 Cadre Proposé

Nous décrivons dans cette section le système de crowdsourcing et la conception du marché du métavers proposés.

1.4.1 Système de Crowdsourcing de ML pour le Métavers

Le système de crowdsourcing proposé est illustré à la Fig. 1.2 avec deux composants principaux, à savoir le crowdsourcing de modèles de ML et le crowdsourcing de données, comme présenté ci-dessous.

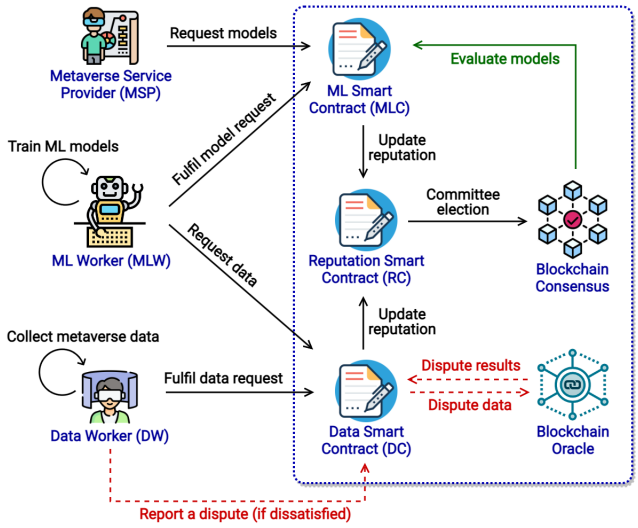

Figure 1.2: Le système de crowdsourcing de MetaAICM.

1.4.1.1 Crowdsourcing de Modèles de ML

La conception de crowdsourcing proposée pour les modèles de ML est résumée dans l'Algorithme 1.1, dans lequel un MSP_i fait appel à des modèles de ML de n différents MLWs. Premièrement, MSP_i engage une transaction de "crowdsourcing de modèle" pour activer le contrat intelligent MLC, où la structure de la transaction est la suivante:

$$T_{mc} = \{T_{\text{token}} = \Theta | T_{\text{desc}} | t_{\text{deadline}} | R_{\min} | Sig_{sk}(T_{mc})\}, \tag{1.1}$$

où Θ est la récompense de la tâche (résultant en certains jetons de métavers), T_{desc} est la description de la tâche (c'est-à-dire les exigences pour le modèle en crowdsourcing), t_{deadline} est la date limite de la tâche, $R_{\min}$ est la réputation minimale requise pour rejoindre la tâche, et $Sig_{sk}(T_{mc})$ est la signature numérique signée par la clé secrète de MSP_i.

Les MLWs intéressés par la tâche de crowdsourcing peuvent entraîner un modèle de ML qui satisfait les exigences de la description de la tâche. Une fois l'entraînement terminé, les MLWs participants téléchargent leurs solutions (c'est-à-dire les modèles de ML entraînés) sur IPFS, puis soumettent les URLs d'IPFS au contrat intelligent MLC. Lorsque la date limite de la tâche est atteinte, le demandeur de la tâche MSP_i doit publier les données de validation D_{val} au comité de la blockchain via IPFS. Ensuite, les nœuds de consensus du comité téléchargent tous les modèles de ML en crowdsourcing et les données de validation depuis IPFS, puis utilisent cet ensemble de données pour évaluer les modèles. Basé sur les résultats d'évaluation, le modèle ayant la meilleure performance ($\text{Model}_{\text{best}}$) est sélectionné comme la solution finale pour la tâche. L'URL d'IPFS du modèle sélectionné est envoyée au demandeur MSP_i par le contrat intelligent MLC, terminant ainsi la tâche de crowdsourcing de modèle.

Algorithm 1.1. Crowdsourcing de Modèle de ML

Entrée : MSP_i, T_{desc}, t_{deadline}, $R_{\min}$, D_{val}, Θ, $\mathcal{W} = \{\text{MLW}_i\}_1^n$.
Sortie : Le modèle ayant la meilleure performance $\text{Model}_{\text{best}}$.

MSP_i initialise une demande de crowdsourcing de modèle via une transaction : $T_{mc} = \{T_{\text{token}} = \Theta \mid T_{\text{desc}} \mid t_{\text{deadline}} \mid R_{\min} \mid \text{Sig}_{sk}(Tx)\}$;
for $\text{MLW}_j \in \mathcal{W}$ **do**
    if la réputation de $\text{MLW}_j < R_{\min}$ **then**
        MLW_j est refusé;
    else
        MLW_j entraîne Model_j selon la description de la tâche T_{desc};
        MLW_j stocke Model_j sur IPFS pour obtenir le lien URL_j;
        MLW_j soumet URL_j au contrat intelligent MLC;
    end if
end for
if $t_{\text{current}} == t_{\text{deadline}}$ **then**
    MSP_i publie les données de validation D_{val} au comité;
end if
Le comité télécharge tous les modèles soumis depuis IPFS;
Le comité évalue chaque modèle sur les données de validation D_{val} pour obtenir $\text{Model}_{\text{best}}$ avec la meilleure performance;
return $\text{Model}_{\text{best}}$;

Le MLW dont le modèle a été choisi comme solution finale reçoit 50% des jetons déposés Θ, tandis que les jetons restants sont distribués aux autres MLWs dont la performance du modèle est supérieure à un seuil prédéfini (par exemple, une précision supérieure à 85%). Cependant, si le demandeur MSP_i ne parvient pas à publier les données de validation lorsque la date limite est atteinte, les jetons déposés sont distribués également à tous les MLWs ayant soumis un modèle, quelle que soit la performance de leur modèle. Dans ce cas, bien que le $\text{Model}_{\text{best}}$ ne soit pas déterminé, le demandeur obtient toujours tous les modèles soumis.

1.4.1.2 Crowdsourcing de Données

Si les MLWs ne possèdent pas les données nécessaires pour entraîner leurs modèles, ils peuvent initier une tâche de crowdsourcing de données pour collecter les données de métavers souhaitées auprès d'autres MUs. Dans ce cas, les MLWs sont également appelés demandeurs de données. Comparé au processus de crowdsourcing de modèle présenté ci-dessus, la procédure de crowdsourcing de données pose d'autres défis fondamentaux:

Problème de Confidentialité: Contrairement aux modèles de ML, la fuite des données de crowdsourcing peut menacer la confidentialité des DWs. Par conséquent, les données de crowdsourcing doivent être cryptées avant d'être soumises pour garantir la préservation de la confidentialité.

Évaluation des Données: Alors que la qualité des modèles de ML peut être évaluée en fonction d'une métrique commune telle que la précision ou la perte, il n'existe pas de métrique standard similaire pour l'évaluation des données. Par conséquent, MetaAICM permet aux demandeurs de données d'évaluer les ensembles de données en crowdsourcing et de décider eux-mêmes de la distribution des récompenses. Les attaques par faux rapports sont découragées en exigeant que les demandeurs de données déposent les frais de crowdsourcing dans le contrat intelligent DC à l'avance, tandis que l'oracle décentralisé de la blockchain peut résoudre tout différend entre les demandeurs de données et les DWs.

Le processus de crowdsourcing de données proposé est résumé dans l'Algorithme 1.2, dans lequel un demandeur de données MLW_i doit engager une transaction de "crowdsourcing de données" pour initier la tâche où le contenu de la transaction est:

$$T_{dc} = \{T_{\text{token}} = \Theta + \alpha | f_{\text{data}} | pk | t_{\text{deadline}} | R_{\min} | Sig_{sk}(T_{dc})\}, \tag{1.2}$$

Algorithm 1.2. Crowdsourcing de Données

Entrée : MLW_i, f_{data}, pk, t_{deadline}, $R_{\min}$, Θ, α, $\mathcal{W} = \{\text{DW}_i\}_1^n$.
Sortie : ensembles de données qualifiés D.

$T_{dc} = \{T_{\text{token}} = \Theta + \alpha | f_{\text{data}} | pk | t_{\text{deadline}} | R_{\min} | Sig_{sk}(T_{dc})\}$;
for $\text{DW}_j \in \mathcal{W}$ **do**
    if la réputation de $\text{DW}_j < R_{\min}$ **then**
        DW_j est refusé;
    else
        DW_j collecte Dataset_j selon f_{data};
        DW_j stocke Dataset_j sur IPFS, obtenant URL_j;
        DW_j crypte URL_j, obtenant une chaîne chiffrée eURL_j;
        DW_j soumet eURL_j à DC via la transaction Sub_j;
    end if
end for
if $t_{\text{current}} == t_{\text{deadline}}$ **then**
    MSP_i publie les résultats d'évaluation au contrat intelligent DC;
    DC distribue les récompenses aux ensembles de données qualifiés;
end if
if DW_j envoie une demande de contestation et URL_j à DC **then**
    DC vérifie que la demande de contestation correspond à Sub_j;
    DC déclenche le réseau Oracle pour réévaluer Dataset_j;
    if Dataset_j est qualifié **then**
        DC envoie une compensation à DW_j;
    end if
end if
return D;

où α est une participation à la compensation qui sera récompensée aux DWs dont l'ensemble de données est honnête mais non reconnu par le demandeur, f_{data} est une liste de caractéristiques souhaitées pour l'ensemble de données en crowdsourcing, pk est la clé publique de MLW_i, et les éléments restants sont similaires à ceux présentés dans (1.1).

Les DWs intéressés peuvent collecter des données pour répondre à la demande de crowdsourcing. Les ensembles de données collectés doivent comprendre toutes les caractéristiques de données listées dans f_{data}. À l'atteinte de la date limite désignée, les DWs téléchargent leurs ensembles de données sur IPFS, recevant une URL unique pour chaque soumission. Ils cryptent ensuite ces URLs en utilisant la clé publique de MLW_i pk_i, et chaque chaîne chiffrée résultante est soumise au contrat intelligent DC via une transaction Sub_j. Bien que accessibles publiquement, ces chaînes chiffrées ne peuvent être décryptées que par MLW_i utilisant sa clé privée, garantissant la confidentialité des données soumises.

En conséquence, MLW_i peut récupérer tous les ensembles de données en crowdsourcing à partir du contrat intelligent DC. MLW_i doit évaluer ces ensembles de données et soumettre les résultats

d'évaluation à DC pour la distribution des récompenses. Les résultats d'évaluation indiquent quels ensembles de données sont honnêtes (par exemple, des ensembles de données propres et de haute qualité) ou malhonnêtes en fonction de la décision de MLW_i. En conséquence, les jetons déposés Θ sont distribués aux DWs dont l'ensemble de données est déclaré honnête, tandis que le reste des DWs ne recevra aucune récompense.

Dans les cas où les DWs contestent les résultats d'évaluation, ils ont le droit d'initier un différend. Cela se fait en déclenchant la fonction de contestation de DC et en soumettant l'URL de leurs ensembles de données hébergés sur l'IPFS. Le contrat intelligent DC vérifie alors la légitimité de chaque demande de contestation en s'assurant qu'elle correspond à la transaction correspondante Sub_j, où un DW_j a précédemment soumis une URL chiffrée. Lors de la confirmation de cette correspondance, le différend est escaladé au réseau oracle décentralisé de la blockchain. Ce réseau comprend plusieurs nœuds professionnels qui évaluent indépendamment si les ensembles de données contestés répondent aux normes de qualité requises. Chaque nœud émet un vote pour déterminer si un ensemble de données est qualifié, et la décision finale est basée sur le résultat composé de la majorité des votes. Cette décision est ensuite relayée à DC, qui applique le résultat. Si les DWs réussissent leur contestation, ils reçoivent collectivement la participation à la compensation α, partagée également entre eux. En revanche, s'il n'y a pas de demande de contestation ou si aucun DW ne remporte sa contestation, la participation à la compensation α est restituée à MLW_i après certains temps de blocage t_{lock}.

1.4.2 Marché de ML pour le Métavers

Selon la demande du marché, les MUs peuvent collecter proactivement des données ou entraîner des modèles de ML pour les vendre sur le marché de MetaAICM, sans attendre des demandes spécifiques de crowdsourcing. Pour simplifier la discussion, les ensembles de données et les modèles de ML sont appelés "produits" dans ce contexte, avec les entités impliquées étant globalement catégorisées comme acheteurs et vendeurs.

Pour répertorier un nouveau produit sur le marché, un vendeur télécharge d'abord les données du produit sur IPFS et obtient une URL IPFS correspondante. L'URL peut être considérée comme une représentation du produit, car tout le monde peut télécharger le produit s'il possède son URL. Le vendeur initie ensuite le processus de mise en vente sur le marché en soumettant une transaction

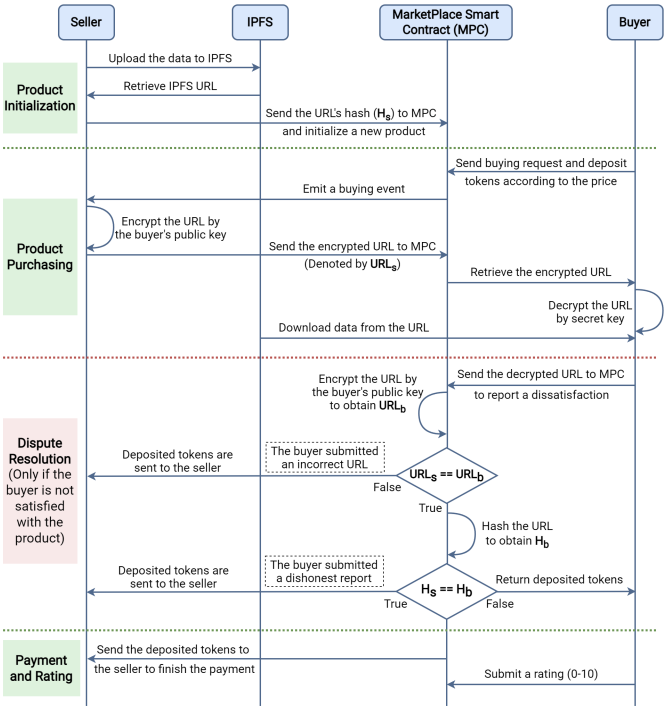

Figure 1.3: Le flux de travail du marché de MetaAICM.

à MPC, structurée comme suit:

$$T_{\text{list}} = \{P_{\text{price}}|P_{\text{desc}}|\text{H}_\text{s}|Sig_{sk}(T_{\text{list}})\}, \tag{1.3}$$

où P_{price} est le prix du produit, P_{desc} est la description du produit (par exemple, liste des caractéristiques d'un ensemble de données, ou type de modèle de ML tel que régression/classification), H_s est le hash de l'URL IPFS, et $Sig_{sk}(T_{\text{list}})$ est la signature numérique de la transaction.

Un acheteur intéressé par l'achat d'un produit initie le processus en soumettant une demande d'achat et en déposant les jetons requis, selon le prix du produit, dans le MPC. Lors de la réception de cette demande, le MPC émet automatiquement un événement pour notifier le vendeur. Par la suite, le vendeur chiffre l'URL IPFS du produit en utilisant la clé publique de l'acheteur, obtenant ainsi une URL chiffrée notée URL_s, qui est ensuite envoyée au MPC. L'acheteur récupère URL_s depuis le MPC et la décrypte en utilisant sa clé secrète pour accéder à l'URL originale et télécharger les données sources du produit. Bien que l'URL chiffrée URL_s soit accessible publiquement comme information sur la chaîne, seule l'acheteur peut la déchiffrer grâce à sa clé secrète.

Dans les cas où l'acheteur n'est pas satisfait du produit, suspectant des problèmes tels que la modification des données ou une mauvaise conduite du vendeur, il dispose d'une période définie t_{chal} pour initier un différend. Cela se fait en invoquant la fonction "contestation" du MPC, accompagnée de l'URL déchiffrée. Le MPC rechiffre cette URL avec la clé publique de l'acheteur pour générer $\mathrm{URL_b}$ et la compare ensuite avec $\mathrm{URL_s}$ précédemment stockée. En cas de divergence entre $\mathrm{URL_s}$ et $\mathrm{URL_b}$, cela indique que l'acheteur a intentionnellement fourni une URL incorrecte, entraînant le transfert des jetons déposés de l'acheteur au vendeur, finalisant ainsi l'achat. À l'inverse, si les URLs correspondent, le MPC hache $\mathrm{URL_b}$ pour obtenir $\mathrm{H_b}$ et la compare avec $\mathrm{H_s}$ de la transaction de mise en vente (1.3). Une différence à ce stade implique une modification de l'URL IPFS pendant la transaction, entraînant un remboursement des jetons déposés à l'acheteur. Si aucune divergence n'est trouvée, l'acheteur pourrait être considéré comme un potentiel attaquant DoS, et son dépôt est transféré au vendeur. Si aucun différend n'émerge dans le délai t_{chal}, les jetons déposés sont automatiquement transférés au vendeur, concluant ainsi le processus de paiement. De plus, l'acheteur a la possibilité d'évaluer le produit acheté, sur une échelle de 0 à 10, en utilisant la fonction "évaluation" du MPC. Cette évaluation contribue à la note moyenne du produit, qui est affichée sur le marché à titre de référence. Le flux de travail du marché de MetaAICM est présenté à la Fig. 1.3.

1.4.3 Système d'Incitation et de Réputation

Le cadre d'incitation de MetaAICM comprend des récompenses en jetons et un mécanisme de réputation. Les récompenses en jetons sont attribuées aux entités garantissant le fonctionnement du système : les nœuds oracle de la blockchain et les nœuds de consensus. Les nœuds oracle dont la décision correspond à celle de la majorité du réseau oracle sont récompensés par des jetons de métavers, tandis que ceux dont la décision est opposée subiront des déductions. Les nœuds de consensus du comité reçoivent également des jetons pour chaque nouveau bloc ajouté à la blockchain après le processus de consensus. Au-delà des incitations en jetons, MetaAICM intègre des scores de réputation pour inciter les contributeurs à être honnêtes et à apporter des contributions précieuses. Les scores de réputation jouent un rôle crucial dans la détermination de l'éligibilité des participants à être sélectionnés comme nœuds de consensus dans le comité de la blockchain pour les cycles suivants. La gestion de la réputation est prise en charge de manière autonome par le contrat intelligent RC. Ce contrat intelligent suit et met à jour le score de réputation de chaque avatar en fonction de ses contributions au système. Il est important de noter que toutes les modifications des

scores de réputation résultent de transactions, qui sont exécutées et vérifiées par le comité de la blockchain, garantissant qu'aucun individu ne peut modifier unilatéralement son score de réputation; les tentatives de le faire par des transactions non autorisées seront rejetées. Pour chaque participant i, le score de réputation r_i est mis à jour en réponse à ses comportements dans des contextes définis. Dans le crowdsourcing de modèles, les MLWs dont la performance du modèle est supérieure au seuil requis recevront une récompense. Les autres MLWs subissent une pénalité de réputation en raison de leurs modèles de mauvaise qualité. Dans le crowdsourcing de données, les DWs dont les ensembles de données sont acceptés par le demandeur de données, ou qui remportent des litiges, recevront des récompenses de score de réputation. Les autres DWs voient leur score de réputation diminuer. En cas de litiges réussis, les demandeurs de données subissent également des réductions de leurs scores de réputation. Sur le marché, les vendeurs gagnent des récompenses de réputation pour avoir reçu des évaluations élevées (par exemple, des évaluations ≥ 8) et perdent de la réputation pour des évaluations basses (par exemple, des évaluations ≤ 3). De plus, chaque décision négative de litige entraîne une sanction pour le score de réputation du vendeur/acheteur.

Soit α la variation du score de réputation après un événement, avec $\alpha = R$ représentant le cas de récompense et $\alpha = P$ représentant le cas de pénalité. Ainsi, la réputation r_i peut être calculée comme suit :

$$r_i = \begin{cases} r_i + \frac{A_i}{P_i}, & \text{si } \alpha = R \\ r_i - 1, & \text{si } \alpha = P \end{cases}$$

où A_i représente le nombre d'avatars uniques avec lesquels l'utilisateur i a interagi à travers les activités de crowdsourcing et de marché, et P_i représente le nombre total d'instances de participation de l'utilisateur i, incluant toutes les soumissions de tâches de crowdsourcing et les activités d'achat/vente sur le marché. Ce mécanisme d'incitation garantit que les récompenses sont directement proportionnelles à la diversité et à la fréquence des contributions positives, renforçant ainsi la résilience du système contre les attaques de collusion. Dans le Métavers, le nombre d'avatars uniques est significativement inférieur au nombre d'activités, ce qui influence le taux d'augmentation de la réputation. En conséquence, les utilisateurs ayant de nombreuses activités connaissent une augmentation de réputation plus lente par rapport aux nouveaux utilisateurs. Cette conception est efficace pour prévenir la domination du système par des utilisateurs de longue date et encourage les nouveaux utilisateurs à contribuer activement au Métavers.

1.4.4 Consensus Basé sur la Réputation de Raft

Le mécanisme de consensus pour MetaAICM doit garantir l'intégrité des transactions, la sécurité et l'efficacité dans la blockchain du consortium. Notre sélection se concentre sur un débit de transaction élevé, l'alignement avec le système de réputation, la tolérance aux pannes et la durabilité. Le protocole choisi doit gérer des données de métavers à grande échelle, s'intégrer à un système de réputation pour la sélection des nœuds, et fonctionner sans problème malgré les défaillances des nœuds, tout en étant économe en énergie. Les mécanismes de consensus courants incluent PoW, PoS, PBFT, BFT-Smart et Raft. PoW et PoS, utilisés dans les blockchains publiques, ont des problèmes de scalabilité. PBFT et BFT-Smart font également face à des limitations de scalabilité en raison de la surcharge de communication. Raft, connu pour sa simplicité et sa tolérance aux pannes, offre de meilleures performances et scalabilité mais manque de protection contre les comportements malveillants. Pour améliorer la sécurité et la fiabilité, le protocole de consensus de MetaAICM intègre Raft avec un système de réputation pour l'élection des nœuds. Le comité blockchain comprend une *partition de comité autorisé* pour les organisations de métavers et une *partition de comité dynamique* pour les nœuds normaux élus en fonction de la réputation. Cette structure assure la stabilité, la gouvernance continue et l'adaptabilité aux besoins évolutifs, avec des réélections tous les k tours, comme illustré à la Fig. 1.4.

1.4.4.1 Processus de Consensus

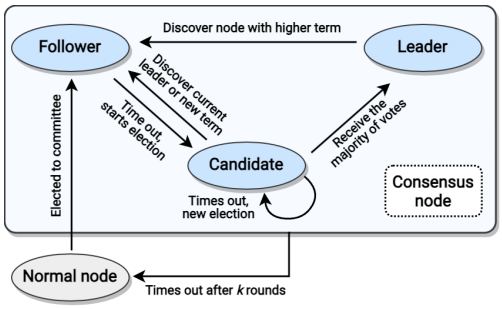

Figure 1.4: Transition d'état d'un nœud de consensus.

Le mécanisme de consensus dans MetaAICM, basé sur le modèle Raft [20], est essentiel pour maintenir un registre blockchain cohérent et fiable. Ce processus est illustré à la Fig. 1.4 et implique trois états distincts pour un nœud de consensus : follower, candidat et leader. L'objectif principal

de ce mécanisme est de réaliser un système robuste, démocratique et résistant aux pannes pour la proposition de blocs et la cohérence du registre.

Opération de Consensus: Le modèle Raft atteint le consensus grâce à une approche simplifiée basée sur le leader. Dans ce système, le leader désigné est responsable de la gestion des journaux et de la réplication des données aux nœuds followers. Le nœud leader propose de nouvelles entrées de journal (blocs) et s'assure que les followers répliquent ces entrées de manière cohérente. Les followers, à leur tour, acceptent et appliquent ces entrées à leur état local. Ce processus minimise le risque d'entrées conflictuelles et garantit une séquence unique et acceptée de journaux à travers le réseau. L'efficacité du modèle Raft réside dans sa simplicité – avec un seul leader coordonnant la réplication des journaux, le processus devient plus prévisible et gérable, réduisant les surcharges typiquement associées aux modèles de consensus plus complexes.

Élection du Leader: L'élection du leader dans le modèle est un processus démocratique, faisant passer les nœuds par les rôles de follower, candidat et leader. Tous les nœuds agissent initialement comme des followers. Si un follower ne reçoit pas de message de pulsation du leader actuel dans un délai spécifié, il perçoit cela comme un signal de défaillance du leader et passe au rôle de candidat. Le candidat cherche alors des votes des autres nœuds pour devenir le nouveau leader. Obtenir une majorité de votes est crucial pour que le candidat assume le rôle de leader, empêchant tout nœud unique de dominer le processus et garantissant un large soutien du réseau pour le leader élu. Une fois élu, le leader reprend l'envoi de messages de pulsation pour maintenir son autorité et garder l'état du réseau synchronisé.

Ce processus d'élection basé sur Raft a montré une capacité de tolérance aux pannes de 50% [20]. En d'autres termes, même si jusqu'à 50% des nœuds sont compromis, MetaAICM fonctionne toujours correctement et sans interruption. De plus, comme il n'y a qu'un seul leader vérifiant les transactions à la fois, cela réduit significativement la consommation d'énergie tout en améliorant le débit des transactions. En conséquence, MetaAICM offre une durabilité supérieure par rapport à d'autres protocoles de consensus gourmands en énergie comme PoW.

1.4.4.2 Sélection du Comité Basée sur la Réputation

Dans MetaAICM, la composition de la partition de comité dynamique est périodiquement renouvelée tous les k tours. Ce processus est orchestré par le leader actuel, qui exécute l'Algorithme 1.3 pour sélectionner de nouveaux nœuds pour la période à venir, où M désigne le nombre de nœuds à élire. Le leader commence le processus d'élection en générant un nombre aléatoire et sa preuve correspondante, $\langle \phi_1, \pi \rangle$, en utilisant une fonction de hachage aléatoire vérifiable (VRF) [21]. La graine pour la VRF est dérivée du hachage du bloc précédent, garantissant ainsi une résistance à la manipulation. Le leader hache ensuite itérativement ϕ_1 $M-1$ fois pour produire un ensemble de M nombres aléatoires, noté $\Phi = \{\phi_1, ..., \phi_M\}$. Chaque ϕ_i dans Φ est instrumental dans la sélection des nœuds. La probabilité qu'un nœud soit élu est proportionnelle à son score de réputation. Les scores de réputation de tous les nœuds sont représentés par K unités de réputation indexées, $R_{\text{spread}} = \{r_1, ..., r_K\}$. Pour chaque ϕ_i, le leader sélectionne une unité de réputation, r_{Idx}, de R_{spread}. Le nœud possédant r_{Idx} est ensuite élu au comité, garantissant que les nœuds ayant une réputation élevée sont plus susceptibles d'être choisis. En intégrant cette élection de comité basée sur la réputation, nous assurons que les nœuds hautement réputés sont sélectionnés pour réguler le processus de consensus, tandis que ceux avec une faible réputation ne peuvent pas dominer le système. Cela améliore encore la fiabilité du cadre. De plus, le processus d'élection peut être vérifié par tout participant via la validation de la preuve π, empêchant ainsi toute entité unique de manipuler le système.

Algorithm 1.3. Élection du Comité Basée sur la Réputation

Entrée : Unités de réputation indexées $R_{\text{spread}} = \{r_1, ..., r_K\}$, nombre de nœuds M, bloc précédent $\mathcal{B}$.
Sortie : Liste des nœuds pour la prochaine partition dynamique $\mathcal{N} = \{N_1, ..., N_M\}$, preuve de l'aléa π.

1: Générer la graine à partir du bloc : seed $\leftarrow$ hash($\mathcal{B}$);
2: Calculer le nombre aléatoire VRF et la preuve : $\langle \phi_1, \pi \rangle \leftarrow \text{VRF}_{sk}(\text{seed})$;
3: Générer $M-1$ nombres aléatoires supplémentaires en hachant ϕ_1 $M-1$ fois, obtenant $\Phi = \{\phi_1, ..., \phi_M\}$;
4: **for** chaque $\phi_i \in \Phi$ **do**
5: Calculer l'index : $\text{Idx} = K \cdot \frac{\phi_i}{2^{||\phi_i||}-1}$, où $||\phi_i||$ est la longueur de ϕ_i en bits;
6: Ajouter le nœud N_i avec l'unité de réputation à Idx à $\mathcal{N}$;
7: **end for**
8: **return** $\mathcal{N}, \pi$

1.4.4.3 Processus de Bootstrap du Consensus

Au stade de démarrage du système, tous les nœuds sont initialisés avec un score de réputation égal à zéro. Pour atténuer le risque d'infiltration d'entités malveillantes dans le comité blockchain pendant

cette phase initiale (c'est-à-dire la période de démarrage à froid), les scores de réputation ne sont pas utilisés pour la sélection du comité pendant les premiers W tours. Au lieu de cela, l'éditeur de la plate-forme MetaAICM nomme un ensemble de Nœuds de Graine de Confiance (TSNs) pour former le comité blockchain. Après la période de démarrage à froid, une fois que W tours se sont écoulés et que les nœuds ont été attribués des scores de réputation reflétant leur comportement, le mécanisme d'élection du comité (voir Algorithme 1.3) s'active, élisant les nœuds de consensus sur la base de ces scores. Les avatars du métavers peuvent postuler pour devenir des TSNs pendant la phase d'initialisation de MetaAICM, avec des sélections effectuées en fonction des profils des avatars. En opérant dans le cadre d'une blockchain de consortium, MetaAICM garantit que les identités de ces avatars sont authentifiées avant qu'ils puissent assumer des rôles au sein du système.

1.4.5 Analyse de Sécurité

La conception de MetaAICM traite efficacement les différentes menaces de sécurité comme discuté ci-dessous.

Attaque par Faux Rapport: Dans le mode de crowdsourcing de MetaAICM, les demandeurs déposent la récompense de la tâche lors de la soumission d'une transaction. Les contrats intelligents distribuent automatiquement les récompenses aux travailleurs avec des contributions valides, empêchant les attaques par faux rapport. Sur le marché, un contrat intelligent vérifie les divergences dans les URLs signalées, garantissant que les jetons sont attribués aux vendeurs honnêtes.

Attaque de Passager Clandestin: Les modèles et ensembles de données en crowdsourcing sont validés à l'aide des données de validation du publieur de la tâche et d'un oracle décentralisé. Les travailleurs malveillants soumettant des résultats de faible qualité subissent des pénalités, y compris la réduction des jetons et la diminution des scores de réputation. Sur le marché, un système de notation expose les passagers clandestins en reflétant la qualité des produits et la réputation des vendeurs, aidant les acheteurs à prendre des décisions éclairées.

Fuite de Confidentialité: MetaAICM garantit la confidentialité des données par le chiffrement. Les URLs des ensembles de données en crowdsourcing sont chiffrées avec la clé publique du demandeur, les rendant inaccessibles aux parties non autorisées. De même, les ensembles de

données et modèles du marché sont protégés par la clé publique de l'acheteur, garantissant que seul le propriétaire légitime peut déchiffrer et accéder aux produits.

Attaques de Sybil et DoS: Un système de réputation complet empêche les attaques de Sybil en traitant chaque avatar comme une entité indépendante, limitant l'influence des multiples fausses identités. Les attaques DoS sont découragées par un mécanisme de punition qui épuise rapidement les soldes des attaquants, rendant de telles attaques non rentables.

Problème de Confiance et SPoF: MetaAICM traite les problèmes de confiance et élimine les points de défaillance unique (SPoF) grâce à la décentralisation. La plateforme fonctionne sans autorité centralisée, distribuant la confiance à travers une blockchain de consortium. Les données et les modèles de ML sont stockés sur IPFS, garantissant des opérations transparentes et vérifiables. Cette approche décentralisée améliore la sécurité et la fiabilité.

1.5 Évaluation des Performances

1.5.1 Configuration Expérimentale

MetaAICM est déployé en tant que blockchain autorisée basée sur Hyperledger Fabric [22], une plateforme de développement blockchain open-source. Le code source de l'implémentation est publié sur GitHub[1] avec des instructions détaillées. Pour participer au système, chaque nœud blockchain dans MetaAICM maintient un conteneur Docker pour exécuter son opération (par exemple, entraîner des modèles de ML, effectuer des mécanismes de consensus, exécuter des contrats intelligents et soumettre des transactions). Les nœuds de consensus utilisent PyTorch pour effectuer l'évaluation des modèles dans le crowdsourcing de modèles. Un outil de benchmarking blockchain nommé Hyperledger Caliper est utilisé pour simuler les charges de transactions et surveiller les performances du réseau. IPFS est également ré-implémenté localement pour éviter la latence Internet, avec 100 nœuds peer-to-peer.

[1] https://github.com/duyhung2201/MetaAICM

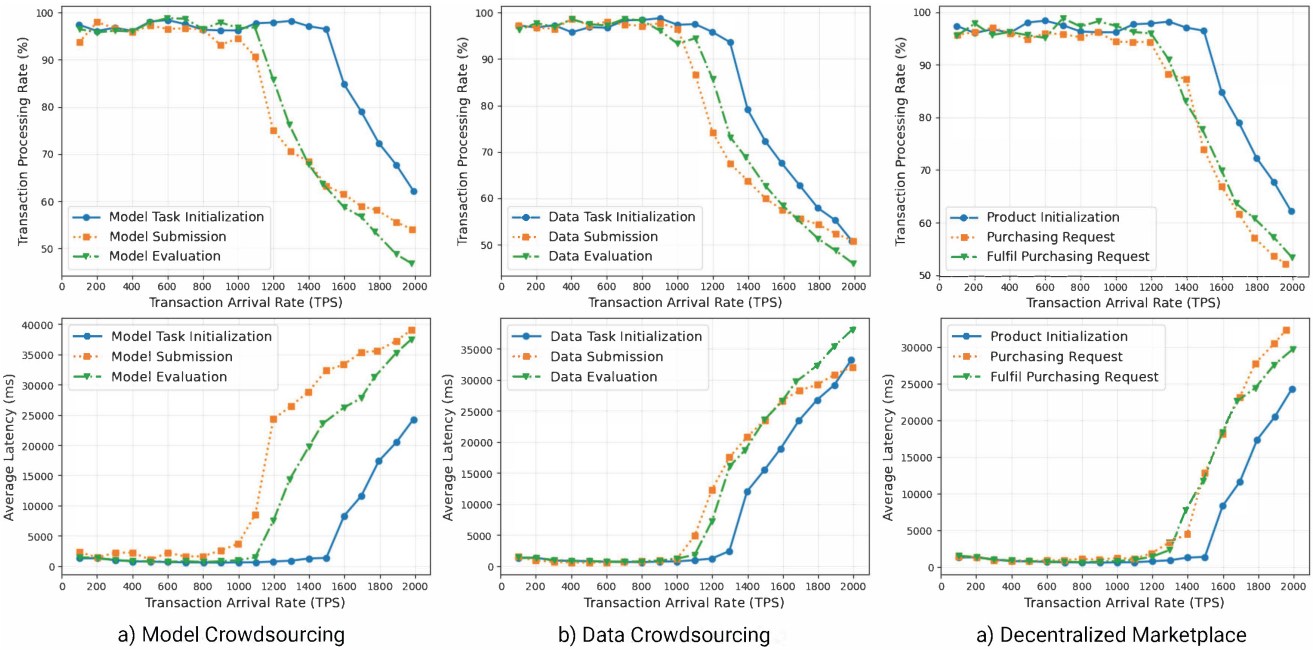

Figure 1.5: Performances des fonctions de contrats intelligents sous différentes charges de travail, allant de 100 à 2000 TPS.

1.5.2 Évaluation des Performances

1.5.2.1 Performances de la Blockchain sous Différentes Charges de Travail

Dans l'expérience illustrée à la Fig. 1.5, 100 clients sont configurés pour soumettre simultanément des transactions invoquant les fonctions de contrats intelligents des trois principaux systèmes, à savoir le crowdsourcing de données, le crowdsourcing de modèles de ML et le marché décentralisé. La charge de travail des transactions augmente de 100 transactions par seconde (TPS) à 2000 TPS. Selon la Fig. 1.5, lorsque la charge de travail est inférieure à 1000 TPS, le système peut traiter la plupart des transactions soumises avec un taux de traitement des transactions de plus de 92%. Cependant, lorsque la charge de travail dépasse 1200 TPS, le taux de traitement des transactions diminue fortement. À 2000 TPS, seules 50% des transactions sont traitées à chaque tour. Cela indique que la capacité de traitement du système est limitée à environ 1000–1200 TPS. De même, la latence moyenne du réseau (c'est-à-dire le temps moyen nécessaire pour qu'une transaction soit traitée) est négligeable jusqu'à atteindre le point de saturation mentionné de 1000 TPS.

1.5.2.2 Performances de la Blockchain avec des Tailles de Bloc Variables

La Fig. 1.6 présente les résultats d'une autre expérience dans laquelle les performances de la blockchain sont surveillées avec différentes tailles de blocs (de 100 à 1000 transactions par bloc) et sous différentes charges de travail (de 700 à 2000 TPS). Il est montré que le système avec la plus petite taille de bloc de 100 transactions peut gérer efficacement la charge de travail de 1200 TPS. Avec une charge de travail transactionnelle plus élevée, ses performances commencent à diminuer rapidement. Intuitivement, si la taille du bloc est petite, il faut plus de blocs pour traiter/stocker le même nombre de transactions. Cependant, la Fig. 1.6 montre également qu'une taille de bloc excessivement grande de 1000 transactions par bloc atteint des performances inférieures à un autre cas de test avec 500 transactions par bloc. Cela est dû au fait que le temps de consensus est souvent plus élevé avec des tailles de blocs plus grandes. Lorsque la taille du bloc devient supérieure à la demande de traitement réelle, cela peut entraîner une redondance et conduire à des performances inférieures.

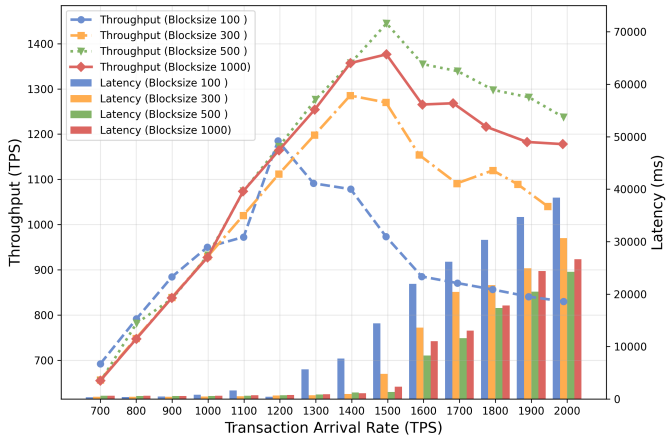

Figure 1.6: Performances de la blockchain avec des tailles de bloc variant de 100 à 1000 transactions par bloc.

1.5.2.3 Évaluation de l'Environnement de Stockage

Table 1.1 montre à la fois le temps de téléversement et de téléchargement du système en fonction de plusieurs ensembles de données et modèles de ML de référence. En général, un ensemble de données ou un modèle plus grand entraîne souvent un temps de téléversement et de téléchargement plus élevé. D'autre part, on observe que le temps de téléversement est significativement plus élevé que le temps de téléchargement. La principale raison de ce délai supplémentaire est qu'un fichier

Table 1.1: Temps de téléchargement et de téléversement du système en fonction de différents ensembles de données et modèles de ML de référence.

Ensemble de données	**Taille (KB)**	**Téléversement (ms)**	**Téléchargement (ms)**
Reuters	3931	79.08	7.80
Ratings1m	24,017	223.84	42.74
Flowers-102	24,567	234.55	43.98
IMDB	27,712	256.51	45.88
MNIST	53,594	374.25	66.41
LFW	176,334	1312.62	214.46
CIFAR-10	180,000	1335.29	264.69
SVHN	297,965	2341.91	368.98
Modèle	**Taille (KB)**	**Téléversement (ms)**	**Téléchargement (ms)**
MobileNet	16,871	169.88	24.39
NASNetMobile	22,884	192.57	41.88
DenseNet21	32,635	265.78	44.36
Xception	89,889	694.78	141.49
InceptionV3	94,016	607.33	131.78
ResNet50	100,651	726.15	138.78
ResNet152V2	237,353	1698.98	357.59
ConvNeXtBase	346,879	2446.19	506.08
VGG16	540,532	3788.18	612.51

doit être divisé en petits morceaux lors de son téléversement sur IPFS, chaque morceau nécessitant également un certain temps pour générer une valeur de hachage correspondante. En revanche, le temps de téléchargement est relativement plus court puisque l'IPFS est déployé localement, ce qui facilite la recherche des morceaux de données souhaités en fonction des hachages fournis.

1.5.2.4 Expérience sur le Score de Réputation

Cette analyse expérimentale examine les modèles de comportement de cinq groupes d'utilisateurs distincts au cours de 100 tâches de crowdsourcing pour évaluer les performances du mécanisme de réputation, comme illustré à la Fig. 1.7. L'axe des x représente le nombre cumulé de tâches de crowdsourcing, tandis que l'axe des y mesure le score de réputation moyen pour chaque groupe. Plus précisément, le Groupe 1 soumet constamment un travail honnête, tandis que le Groupe 4 ne s'engage que dans des activités malveillantes. Le Groupe 2 commence par des contributions

honnêtes pour les 40 premières tâches avant de passer à un comportement malveillant pour le reste de l'expérience. Le Groupe 3 présente une combinaison de soumissions honnêtes et malveillantes tout au long de l'expérience.

Le Groupe 1 montre une augmentation constante de la réputation, bien qu'à un rythme décroissant, en raison d'une diminution de la diversité des interactions (c'est-à-dire des avatars uniques rencontrés) à mesure que le nombre de tâches augmente. Le Groupe 2 connaît une augmentation initiale de la réputation grâce à des contributions positives, suivie d'un déclin significatif après la tâche 40, coïncidant avec leur passage à des activités malveillantes. Ce déclin de réputation est initialement abrupt, se stabilisant à mesure que leur score diminue, ce qui est une conséquence de la réduction de l'éligibilité aux tâches en raison de soumissions malveillantes antérieures. Il convient de noter que les comportements passés sont transparents dans la blockchain, ce qui affecte la capacité de participation des entités malveillantes dans le système. La réputation du Groupe 3 fluctue, reflétant leur modèle de comportement mixte, avec des périodes d'augmentation reflétant des contributions honnêtes et des diminutions correspondant à des actions malveillantes. En revanche, le Groupe 4 subit un déclin rapide de la réputation en raison d'un comportement malveillant persistant, le taux de déclin se stabilisant vers la fin de l'expérience. Cette stabilisation se produit à mesure que leur réputation diminue, restreignant ainsi progressivement leur accès aux tâches. Finalement, les utilisateurs du groupe 4 sont exclus de toute participation ultérieure au système.

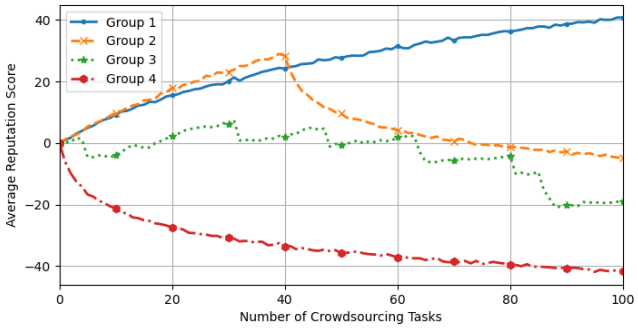

Figure 1.7: Évolution des scores de réputation moyens.

1.6 Conclusion

Dans cette dissertation de master, nous avons proposé MetaAICM, un cadre basé sur la blockchain conçu pour renforcer le métavers. MetaAICM intègre un système de crowdsourcing distribué, per-

mettant aux MSP de collecter des données de métavers et des modèles de ML auprès des MUs, ainsi qu'un marché décentralisé, permettant aux MUs de collecter et de vendre des données et d'entraîner des modèles de ML en utilisant leurs ressources. En tirant parti de la blockchain et des contrats intelligents, MetaAICM garantit la sécurité et la confidentialité grâce à la décentralisation, éliminant le besoin d'une autorité tierce de confiance ou d'hypothèses de confiance supplémentaires entre les participants.

Chapter 2

Introduction

2.1 Background and Motivations

The Metaverse, a term derived from the prefix "meta" (meaning beyond) and "verse" (a short form of the universe), represents a parallel digital realm that mirrors a real-world environment with an independent economic structure interconnected with the physical world. Since its first appearance, the concept of metaverse is still evolving with various forms, such as a second life [1], 3D virtual worlds [2], and life-logging [3]. It is often envisioned as a fully immersive, hyper-spatial, and self-sustaining virtual shared space that merges the physical, human, and digital realms [4]. It is acknowledged as a burgeoning paradigm shift in internet technology, succeeding the Web and mobile internet revolutions [5], where individuals can immerse themselves as digital natives and explore an alternative existence in a virtual setting.

The metaverse incorporates a plethora of emerging technologies [6], [7], [8]. Notably, digital twins replicate real-world environments; virtual reality (VR) and augmented reality (AR) facilitate immersive 3D experiences; advanced communications like 5G and beyond wireless technologies ensure ultra-reliable and low-latency connections for a multitude of devices within the metaverse; wearable sensors and brain-computer interfaces (BCI) enhance interactions between users and avatars; artificial intelligence (AI) supports large-scale creation and rendering of metaverse environments; and blockchain technology along with non-fungible tokens (NFTs) secure verifiable ownership rights over metaverse assets [9]. With the widespread adoption of smart devices and the maturation of these

enabling technologies, the metaverse is transitioning from a nascent stage to a realization in the near future. Moreover, significant technological innovations and advancements have been fostering a new information ecology and creating new demands for applications in various fields such as education, gaming, work, and social interaction, positioning the metaverse as a foundational platform for this emerging ecosystem [8].

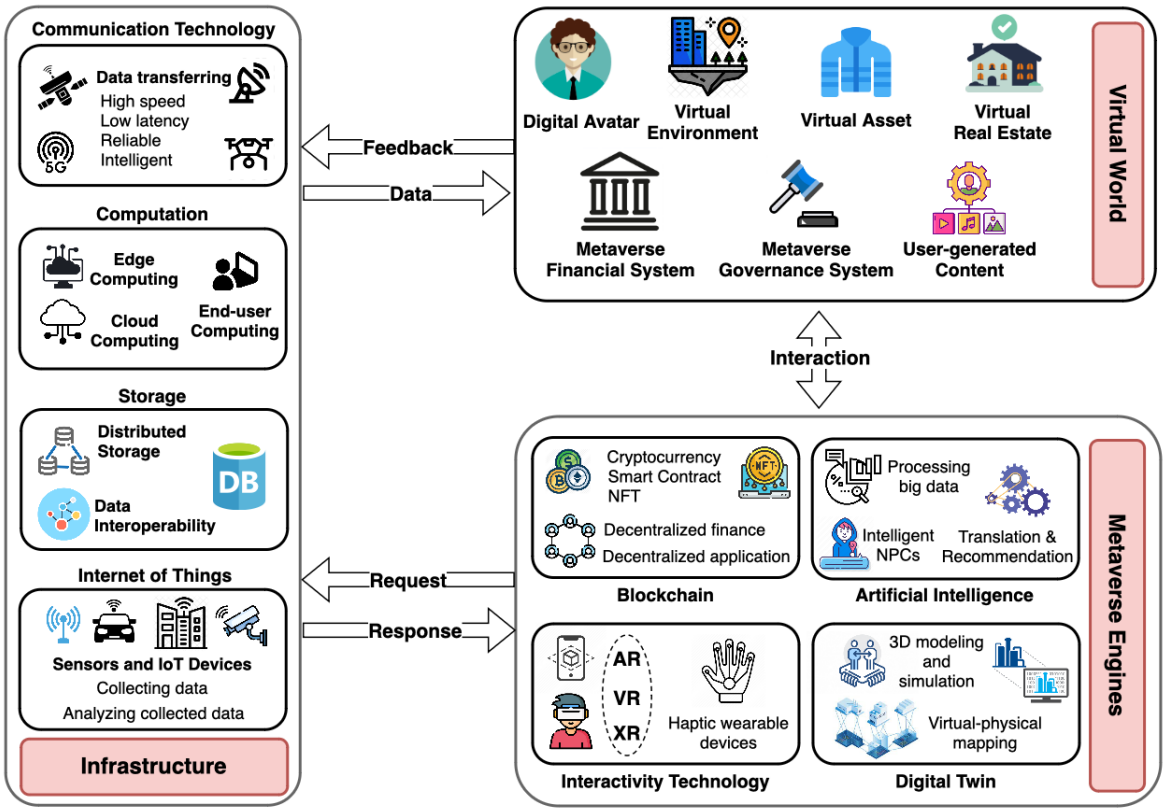

Figure 2.1: The architecture of Metaverse and core technologies.

Among various advanced technologies, AI plays a pivotal role in enabling the virtual realm of the metaverse [7, 10]. For example, computer vision (CV) algorithms leverage real-world data captured by cameras deployed in city streets, buildings, and on unmanned aerial vehicles (UAVs) to construct the metaverse's virtual environment. Additionally, machine learning (ML) models deployed on wearable devices analyze user data, such as appearance, gestures, and facial expressions, to reflect user behaviors into avatar actions [11]. While metaverse publishers typically manage these tasks, metaverse service providers (MSPs) usually use ML to deliver virtual services to metaverse users (MUs). For instance, MSPs can utilize generative AI models to create AI-generated content like virtual items, clothing, and decorations for the metaverse [12]. Additionally, event organizers can employ detection models to identify unauthorized MUs accessing virtual events by analyzing

their profiles, such as reputation, location, and operational history. The metaverse and its enabling technologies are illustrated in Fig. 2.1.

MSPs would be interested in leveraging the expertise in AI/ML and computational resources from the metaverse community to construct and train their desired ML models. Conversely, ML professionals capable of designing and training ML models may face challenges in collecting the required data due to privacy concerns and the large-scale, heterogeneous nature of the metaverse. Thus, it is essential to provide MUs and MSPs with a decentralized platform supporting the exchange and trading of metaverse data and ML models in a trustless, distributed environment with low trading fees, high performance, and privacy preservation.

Traditional trading and crowdsourcing systems offered by third-party authorities often suffer from the single points of failure (SPoF) and trust issues, including free-riding and false-reporting attacks among participants. Moreover, these systems usually lack transparency, privacy, and incentive mechanisms [13]. Although these limitations might be acceptable for certain conventional centralized platforms, they are inadequate for the decentralized metaverse environment, where scams and frauds are exacerbated by various novel social engineering attacks [14]. Blockchain technology offers a potential solution for trust management in such metaverse trading systems with its immutable, transparent, and auditable properties [15]. Specifically, blockchain provides a reliable storage environment to record trading transactions, while smart contracts and blockchain committees can replace third-party authorities in decision-making, policy enforcement, reputation, and incentive management [16].

Motivated by these urgent needs, this thesis proposes a blockchain-empowered framework named MetaAICM that facilitates AI crowdsourcing and provides a marketplace for intelligent virtual services in the metaverse. In MetaAICM, MUs can leverage their devices' capabilities and available resources to collect data or train ML models for specific purposes, then list them on MetaAICM's marketplace for sale. If MSPs cannot find the required data or models on the marketplace, they can initiate a crowdsourcing task on MetaAICM's decentralized applications to obtain the desired products from other MUs. MetaAICM ensures that payment is only finalized when the traded product is verified, eliminating the need for mutual trust between parties. To address scalability, transaction speed, and sustainability, MetaAICM incorporates a reputation-based Raft consensus protocol.

2.2 Literature Review

We provide a literature survey in this section where existing research efforts and works on blockchain-based crowdsourcing and trading systems suitable for the decentralized Metaverse environment are discussed.

2.2.1 Blockchain-Based Data Marketplace

Previous research has explored the use of blockchain for data trading within data marketplaces. In particular, the framework presented in [23] outlines a blockchain-based market for IoT data trading, featuring three modes of operation: general trading, selling on demand, and buying on demand. These transactions are governed by smart contracts, which remove the single point of failure (SPoF) and negate the need for third-party intermediaries. Nonetheless, this model does not address potential dishonesty of sellers, such as submitting incorrect data, which could result in buyers receiving poor-quality data and losing funds. Additionally, the inherent transparency of blockchain can lead to data leaks if no appropriate encryption method is implemented. To address the privacy concerns, the approach in [24] proposes a novel data trading system where sellers do not transmit original data but rather process analytical results, facilitated by trusted nodes using Software Guard Extensions (SGX) for smart contract execution. However, the complexity and resource constraints of smart contracts limit this system to relatively simple tasks. Further developing the concept, the work [25] implemented a decentralized IoT data marketplace based on blockchain to enhance fairness and trust management. This system incorporates a security manager layer with multiple storage operators responsible for data encryption and storage. Nevertheless, post-decryption disputes over data quality present challenges in determining the culpable party—whether the buyer, seller, or storage operator. The framework in [26] introduces a data-trading center capable of retaining data using blockchain and machine learning, employing similarity-learning techniques to verify data authenticity and an "arbitration institution" to mediate disputes. However, reliance on this arbitration institution could lead to a new SPoF if compromised or attacked.

The work in [27] proposes a proof of delivery (PoD) scheme where smart contracts replace intermediaries in regulating trading operations. Although this setup ensures transaction transparency and verifiability, it still rely on the existence of a trustworthy, always-available arbitrator, which may

not be realistic in a trustless environment. Similarly, the blockchain-based digital asset management (DAM) framework in [28] allows customer to report data mismatches but lacks a mechanism to verify the authenticity of such claims, potentially enabling fraudulent attempts to avoid payment.

Exploring privacy preservation, SPChain in [29] proposes encrypting traded data using the user's public key to prevent leaks. However, this method's high computational demands make it non-scalable for large datasets. An alternative DAM approach in [30] uses an attribute-based access control (ABAC) model within smart contracts to manage digital asset access based on predefined client conditions. This method, however, relies on the assumption that data remains unaltered in the cloud or during transit, which may not always be the case.

In summary, the integration of blockchain technology into data marketplaces has ushered in notable advancements in terms of security, transparency, and the decentralization of trading mechanisms. The studies reviewed highlight various approaches that employ the blockchain for IoT data trading, each addressing specific challenges like privacy, data quality assurance, and dispute resolution. However, these frameworks still reveal inherent limitations, such as the capability to handle large amounts of transactions, dependency on third-party arbitrators, and the scalability of encryption techniques. The research work presented in this thesis aims to address the critical shortcomings in existing frameworks by introducing robust mechanisms for ensuring data quality and integrity among buyers-sellers and implementing an encryption strategy that effectively balances complexity and privacy, offering enhanced scalability.

2.2.2 Blockchain for Crowdsourcing

Several studies have explored how the blockchain technology can be used to develop various crowdsourcing frameworks. In particular, the CrowdBC framework presented in [31] employs smart contracts to manage tasks, reducing human intervention and trust issues, and provide resistance against malicious actors. However, a significant limitation of CrowdBC comes from its reliance on requesters to develop complex evaluation functions within smart contracts, a challenging task for those without technical expertise, particularly for AI/ML tasks due to their complexity and resource demands. Another noteworthy approach, named zkCrowd proposed in [32], emphasizes user privacy using zero-knowledge proofs and combines public and private blockchains to tailor the privacy levels so as to meet the user requirements. Nonetheless, zkCrowd also hinges on sophisticated

reward distribution functions, which can be impractical in many scenarios due to the complexity of accurately assessing the contributions. Further enhancing privacy, the system described in [33] utilizes dual blockchain layers to offer varied privacy protections, while incorporating noise into data transmissions to provide participant privacy protection. Meanwhile, systems like ZebraLancer [34] and BPCM [35] address privacy by anonymizing participant identities but they still struggle with the effective evaluation of the contributions.

In the spatial crowdsourcing, the framework [36] facilitates decentralized task matching in the Metaverse context using a novel ciphertext-based scheme that leverages geographic transformations and bilinear mapping to restrict task location access to eligible workers. Similarly, TBSCrowd, introduced in [37], employs threshold blind signatures to ensure privacy, unforgeability, and robustness in mobile crowdsourcing.

Considering device heterogeneity and privacy in mobile crowdsourcing, the design in [38] integrates pairwise additive masking with the Chinese Remainder Theorem in a blockchain-based federated learning system, ensuring fair reward distribution and result verifiability. In the realm of 3D modeling, Web3DP proposed in [39], is a Web3.0-based crowdsourcing platform for 3D models. Additionally, the method proposed in [40] optimizes crowdsourcing tasks for scanning 3D models, aiming to reduce costs and enhance realism in the Metaverse. Addressing crowdsensing challenges, the authors in [41] introduce MetaCS, a multi-agent deep reinforcement learning framework with a transformer-based mobility prediction module and a relational graph learning mechanism to optimize partner selection for UAVs. Lastly, the work [42] tackles the budgeted pricing problem under uncertainty by considering blocking constraints and varying attributes of virtual service providers (VSPs).

In conclusion, the blockchain technology has great potential in crowdsourcing design which allows to enhance the transparency, trust, and privacy in decentralized environments. Various frameworks, such as CrowdBC, zkCrowd, and TBSCrowd, demonstrate the blockchain's capacity to mitigate traditional crowdsourcing challenges, including those related to task management, privacy preservation, and equitable reward distribution. However, these advancements also encounter limitations such as the need for sophisticated contract programming, scalability issues, and the complexity of accurately evaluating contributions. Our research work addresses these issues by

introducing a robust mechanism for evaluating crowdsourcing contributions, enhancing scalability and privacy, and providing resilience against various types of attacks.

2.2.3 Blockchain-Based Metaverse Applications

Blockchain technology has been instrumental in developing various metaverse applications [15, 43, 44]. The authors in [45] propose using smart contracts to efficiently manage and automate interactions between MSPs and MUs. This blockchain-based design allows MSPs to optimize resource allocation when offering virtual services and applications, while incentivizing MUs to contribute resources. The design is based on the Stackelberg game theory. In another approach, the authors in [46] develop a blockchain-based federated learning (FL) framework for the industrial metaverse. By leveraging blockchain, the designed system provides two privacy options for FL in both physical and virtual spaces, utilizing various sub-blockchains that communicate via cross-chain techniques. Jiang et al. [47] present an FL-enabled digital twin (DT) edge network where access points (APs) act as edge nodes to assist end-user devices in building DT models. This framework employs a directed acyclic graph (DAG) based blockchain to securely record both local and global model updates in FL as well as resource transactions between APs and users.

On a different front, the authors in [8] develop a campus-oriented metaverse prototype using blockchain to underpin the platform's economic system. Moreover, blockchain-based non-fungible tokens (NFTs) represent virtual items and user-generated content. However, these studies do not explore the use of blockchain to enable AI/ML-based services within the metaverse, such as ML crowdsourcing or decentralized marketplaces for ML models and metaverse data. Ryskeldiev et al. [48] propose a peer-to-peer distribution model based on blockchain for mixed reality (MR) applications.

Bai et al. [49] propose a blockchain-based decentralized framework for digital city governance, encouraging user engagement in administrative processes. Their framework dynamically selects a verifier group from digital citizens for transaction verification in a hybrid blockchain, using a private-prior peer prediction mechanism to prevent collusion among verifiers and a Stackelberg game theory-based model to motivate participation. Li et al. [50] design a decentralized forensics method utilizing blockchain, where customized cryptography allows for fine-grained access control over forensics data and smart contracts ensure auditable execution of forensic procedures. This

method can automate forensic processes across multiple entities and platforms although it requires further research to address practical implementation challenges.

Several existing studies propose to use blockchain in virtual economy design. Rehman et al. [51] examine cryptocurrency ecosystem design principles, including centrality, privacy, price manipulation, and governance. De Biase et al. [52] propose a swarm economy model for digital resource sharing among heterogeneous smart devices, incorporating blockchain for transparent and immutable transaction auditing. However, this model faces issues with non-automatic transaction settlement, high computational overhead, and lack of supervision. To address these, Liu et al. [53] develop a blockchain-based automatic transaction settlement framework with a three-layer sharding architecture for enhanced scalability.

In industrial IoT, Shen et al. [54] employ blockchain to design a decentralized and transparent cross-domain authentication scheme for devices in different domains, employing a consortium blockchain for trust establishment and identity-based encryption (IBE) for device authentication. They also propose an anonymous authentication protocol to handle IBE's identity revocation limitations. Chen et al. [55] build on this by creating XAuth, an efficient cross-domain authentication scheme using optimized blockchain and privacy preservation functions, along with a zero-knowledge proof-based anonymous authentication protocol. Wang et al. [56] introduce a data processing-as-a-service (DPaaS) model, leveraging blockchain for fine-grained data usage policy-making, policy execution via smart contracts, and policy auditing on transparent ledgers.

2.3 Research Objectives and Contributions

To address the research gaps in the literature, this thesis proposes MetaAICM, a blockchain-empowered framework that enables ML crowdsourcing and a marketplace to facilitate intelligent services in the metaverse. The novel contributions of MetaAICM can be summarized as follows:

- **Decentralized Marketplace and Crowdsourcing**: MetaAICM establishes a blockchain-based decentralized marketplace that encourages MUs to proactively collect metaverse data or train ML models for sale. If the desired data and models are not available on the marketplace, the crowdsourcing mode allows MSPs to source ML models from MLWs, while MLWs can request metaverse data from DWs.

- **Elimination of Trust Assumptions**: Unlike existing blockchain-enabled frameworks, MetaAICM does not require task requesters to provide an evaluation function to automatically assess the contribution of workers. Thanks to our decentralized design, other trust assumptions are also eliminated, and the involved participants do not have to trust each other.
- **Incentive and Reputation Mechanisms**: A concrete incentive mechanism is designed to motivate MUs to contribute computational resources to empower metaverse ML services. Additionally, a reputation system is integrated to filter out malicious entities and ensure honest participation.
- **Security and Performance**: We demonstrate that MetaAICM can resist various security threats such as DoS, SPoF, data leakage, and Sybil attacks. Extensive numerical results confirm that MetaAICM offers high performance and automation with low operational costs.

2.4 Dissertation Outline

The remainder of this thesis is organized as follows:

- Chapter 3 presents the background knowledge essential for technical development in the subsequent chapters.
- In Chapter 4, we present our framework, named MetaAICM, enabling decentralized crowdsourcing and marketplace for trading machine learning data and models.
- Finally, Chapter 5 concludes the thesis and outlines future research directions.

Chapter 3

Background

This chapter provides a comprehensive overview of the fundamental concepts and components of blockchain technology. We start by examining the architecture of blockchain, detailing its core elements and their operations. Next, we explore the taxonomy of blockchain, differentiating between permissionless and permissioned blockchains. We then discuss various consensus mechanisms, with a focus on the scalability trilemma among decentralization, security, and scalability. Following this, we introduce smart contracts and their role in automating secure agreements on the blockchain. We delve into Hyperledger Fabric, an enterprise-grade blockchain framework, highlighting its architecture and unique chaincode smart contracts. Finally, we explore the Raft consensus algorithm used in Hyperledger Fabric, emphasizing its importance in achieving reliable and fault-tolerant consensus. This chapter lays the groundwork for understanding the technical intricacies and applications of blockchain technology.

3.1 Blockchain

Blockchain is a decentralized digital ledger that records transactions in such a way that the registered transactions cannot be altered retroactively. This technology, thus, ensures the integrity and transparency of data without the need for a central authority or intermediary. At its core, a blockchain consists of multiple blocks linked together as a chain, where each block contains a certain number of transactions. These transactions are verified by participants in the network, known as

nodes, through a consensus mechanism, ensuring security and mutual agreement on the state of the ledger at any given time. The immutable nature of blockchain technology not only enhances security but also builds trust among users, making it a foundational technology for various applications in the metaverse [44], such as security [43], digital asset management [57, 58], networking [59], and distributed learning [60].

Blockchains are typically managed by a peer-to-peer (P2P) computer network for use as a public distributed ledger, where nodes collectively adhere to a consensus algorithm to add and validate new transaction blocks. Although blockchain records are not unalterable, since blockchain forks are possible, blockchains may be considered secure by design and exemplify a distributed computing system with high Byzantine fault tolerance.

3.1.1 Blockchain Architecture

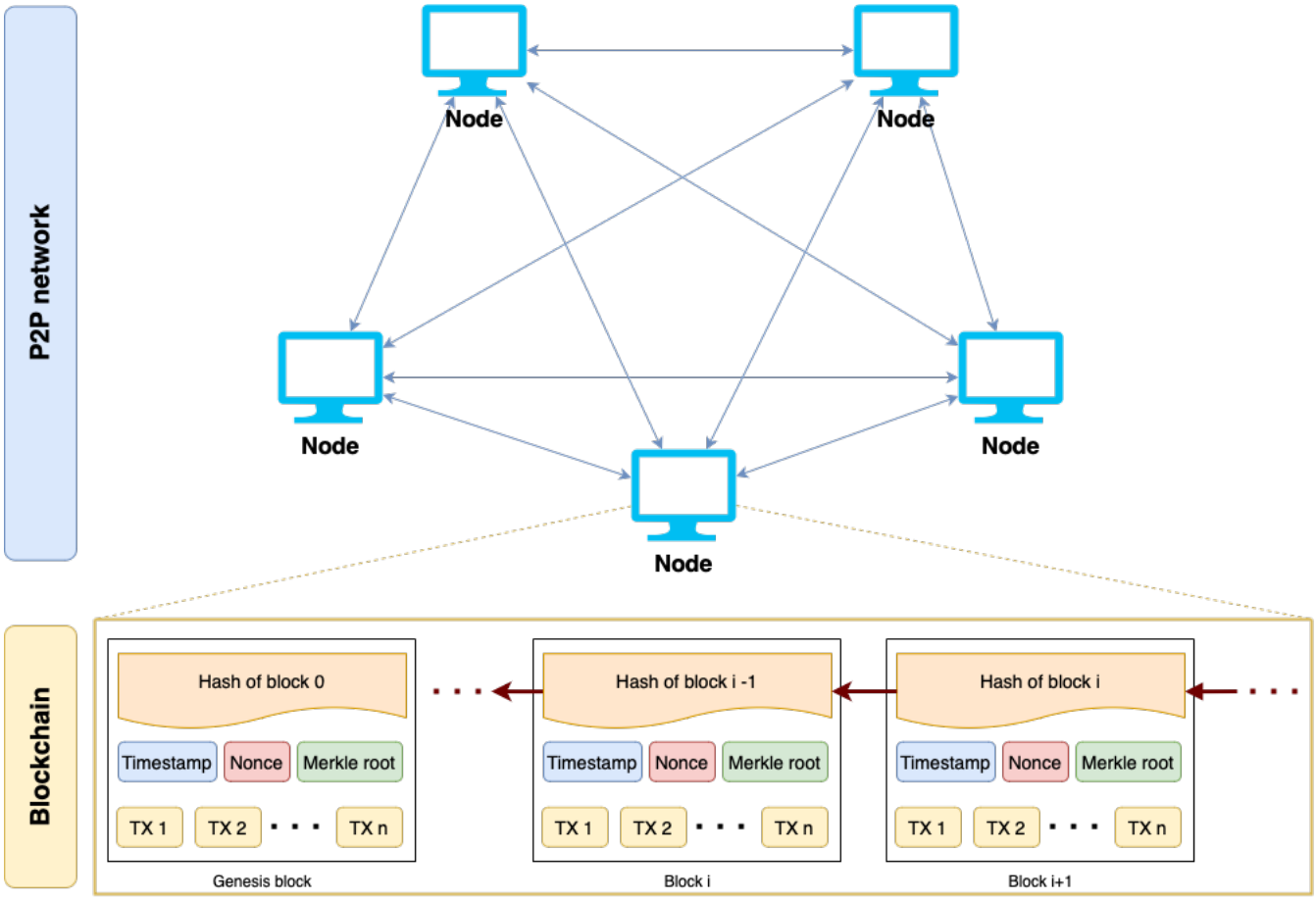

Figure 3.1: Architecture of a blockchain, consisting of a continuous sequence of blocks.

A typical blockchain architecture consists of the following main components:

Blocks: Blocks hold batches of valid transactions that are hashed and encoded into a Merkle tree [61]. Each block includes the cryptographic hash of the prior block in the blockchain, linking the two. The linked blocks form a chain. This iterative process confirms the integrity of the previous block, all the way back to the initial block, which is called the genesis block (Block 0). To assure the integrity of a block and the data contained in it, the block is usually digitally signed. Each block consists of the block header and the block body, which contains the list of transactions. The block header includes:

- *Block version*: Indicates which set of block validation rules to follow.
- *Previous block hash*: A 256-bit hash value that points to the previous block.
- *Merkle tree root*: Contains the hash value representing all the transactions in the block, ensuring data integrity by allowing efficient and secure verification.
- *Timestamp*: The current timestamp as seconds since 1970-01-01T00:00 UTC, indicating when the block was created.
- *Nonce*: A variable that starts at 0 and is incremented for every hash calculation during the block-creating process.

The block time is the average time it takes for the network to generate one extra block in the blockchain. By the time of block completion, the included data becomes verifiable. For the cryptocurrency, shorter block time means faster transaction processing. The block time for Ethereum is set to be between 14 and 15 seconds, while for Bitcoin, it is, on average, 10 minutes.

Chain: The chain is a sequence of blocks linked together in chronological order. Each block references the hash of the previous block, forming an unbroken chain that ensures the immutability of the data. This linkage of blocks creates a secure and verifiable history of all transactions within the blockchain network. However, the blockchain can experience a phenomenon known as a fork. A fork occurs when there is a divergence in the chain, leading to the creation of two or more separate chains. Forks can be classified into two main types: soft forks and hard forks. A *soft fork* is a backward-compatible upgrade to the blockchain protocol, where only previously valid transactions or blocks are made invalid. Since soft forks are backward-compatible, nodes that do not upgrade

to the new rules will still recognize the new blocks as valid, maintaining compatibility within the network. In contrast, a *hard fork* is not backward-compatible. It occurs when there is a fundamental change in the blockchain protocol, resulting in the creation of a new version of the blockchain that is incompatible with the old one. This creates a permanent divergence in the blockchain, where nodes following the new rules will reject blocks from nodes following the old rules. Notable examples of hard forks include the split of Bitcoin and Bitcoin Cash, as well as Ethereum and Ethereum Classic. Forks are significant because they can impact the consensus process, network stability, and overall trust in the blockchain system. They often arise from differences in opinion within the community regarding the implementation of protocol updates or from attempts to resolve security vulnerabilities. Proper management of forks is crucial to maintaining the integrity and continuity of the blockchain.

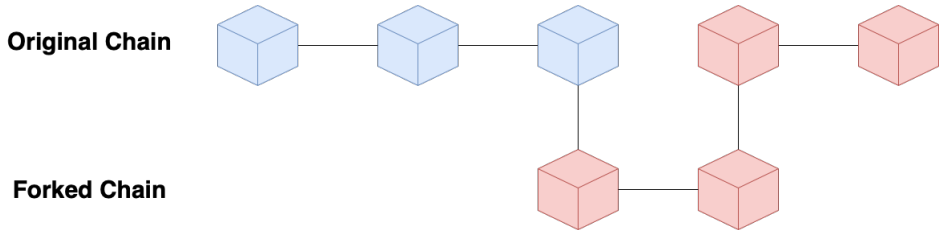

Figure 3.2: Blockchain soft forking.

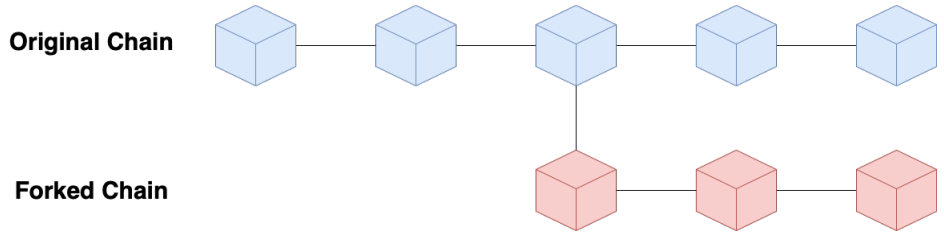

Figure 3.3: Blockchain hard forking.

Nodes: Nodes are computers that participate in the blockchain network. They maintain copies of the blockchain, validate new transactions, and ensure the overall security and integrity of the blockchain.

Transactions: Transactions are the basic units of data recorded on the blockchain. Each transaction represents a transfer of value or information between participants, and once validated, they are permanently stored in the blockchain.

Miners: Miners are specialized nodes that validate and add new transactions to the blockchain. This process, known as mining, results in the creation of new blocks and is rewarded with cryptocurrency incentives.

3.1.2 Blockchain Taxonomy

In terms of accessibility and control, blockchains can be categorized into two main types: permissionless and permissioned blockchains [62].

Permissionless Blockchains: Also known as public blockchains, these are open to the public, allowing anyone to participate, read, write, or audit the blockchain without requiring explicit permission. Prominent examples of public blockchains include Bitcoin [63] and Ethereum [64]. These blockchains are characterized by their transparency, security, and immutability, as all transactions are publicly verifiable and tamper-proof due to the decentralized consensus mechanisms they employ.

Permissioned Blockchains: In contrast, permissioned blockchains restrict access and require participants to obtain explicit permission to join the network. This permission can be granted by a central authority or by a consortium of existing participants. Permissioned blockchains are further divided into two subcategories:

- *Private Blockchains*: Controlled by a single organization, private blockchains restrict access to selected participants. They offer greater privacy and control over data at the expense of reduced transparency compared to public blockchains.
- *Consortium Blockchains*: Governed by a group of organizations, consortium blockchains strike a balance between decentralization and control. They are often used in industries where multiple organizations need to collaborate and share data securely, such as in supply chain management or finance. Examples include R3 Corda and Hyperledger Fabric [22].

Table 3.1 compares these systems using different perspectives, as described below.

Consensus determination: Refers to how the blockchain network reaches agreement on the state of the ledger.

Read permission: Indicates who can read the data on the blockchain.

Immutability: Refers to the ability to alter data after it has been added to the blockchain.

Efficiency: Concerns the speed and resource consumption of processing transactions.

Centralization: Measures the distribution of control within the blockchain.

Property	Public blockchain	Consortium blockchain	Private blockchain
Consensus process	Permissionless	Permissioned	Permissioned
Consensus determination	All miners	Selected set of nodes	One organization
Read permission	Public	Could be public or restricted	Could be public or restricted
Immutability	Nearly impossible to tamper	Could be tampered	Could be tampered
Efficiency	Low	High	High
Centralization	No	Partial	Yes

Table 3.1: Comparison of blockchain types based on various properties.

3.1.3 Consensus Mechanism

Blockchain consensus refers to the mechanism by which all participants in a network reach an agreement on the state of the blockchain without the need for a central authority. Various consensus algorithms have been developed and used in different blockchain networks, each with its own set of advantages and trade-offs. Some of the most common consensus mechanisms include Proof of Work (PoW) [65, 66], Proof of Stake (PoS) [67], Raft [20], Practical Byzantine Fault Tolerance (pBFT) [68, 69], and Proof of Authority (PoA) [70].

Proof of Work (PoW): PoW is one of the earliest and most well-known consensus mechanisms, primarily used by Bitcoin. In PoW, miners compete to solve complex cryptographic puzzles, and the first one to solve the puzzle gets to add the next block to the blockchain and receive a reward. This process requires substantial computational power, making it secure against attacks but also energy-intensive and slow in terms of transaction throughput.

Proof of Stake (PoS): PoS is designed to be more energy-efficient than PoW. Instead of miners, PoS uses validators who lock up a certain amount of cryptocurrency as a stake. Validators are then selected to create new blocks based on their stake and other factors like randomization. This approach reduces the need for massive computational power and incentivizes honest behavior, as validators risk losing their stake if they act maliciously.

Raft: Raft is a consensus algorithm designed for distributed systems to achieve crash fault tolerance. It simplifies the consensus process by electing a leader who manages the replication of logs across followers. Raft ensures consistency and reliability in a network but does not provide Byzantine fault tolerance, making it suitable for trusted environments.

Practical Byzantine Fault Tolerance (pBFT): pBFT is designed to handle Byzantine faults, where nodes can act maliciously or arbitrarily. It operates through a three-phase protocol involving pre-preparation, preparation, and commit stages. pBFT is efficient in smaller networks and provides high throughput and low latency, but its complexity increases with the number of nodes, making it less scalable for larger networks.

Proof of Authority (PoA): PoA is a consensus mechanism where transactions and blocks are validated by approved accounts known as validators. These validators are pre-approved and trusted entities. PoA is efficient and provides high throughput, making it suitable for private or consortium blockchains where participants are known and trusted.

3.1.4 Blockchain Scalability Trilemma

In the design of any blockchain network, there is an inherent trade-off known as the blockchain scalability trilemma, which posits that it is challenging to simultaneously achieve all three of the following properties: decentralization, scalability, and security. Consensus algorithms play a key role in determining the characteristics of a blockchain network according to these three properties.

Decentralization and Scalability: A highly decentralized network, like Bitcoin, often suffers from low processing speed because it takes more time to reach consensus among numerous nodes due to network delay and synchronization issues. In contrast, semi-decentralized blockchain systems can offer higher scalability as they reach consensus more quickly, but they may be prone to manipulation and single points of failure (SPoF).

Decentralization and Security: Decentralization enhances security by distributing control across many nodes, making it difficult for a single entity to manipulate the system. However, achieving higher decentralization often involves increased complexity in maintaining network synchronization and consensus, which can inadvertently introduce security vulnerabilities. Conversely, reducing decentralization might simplify consensus processes and reduce latency, but it increases

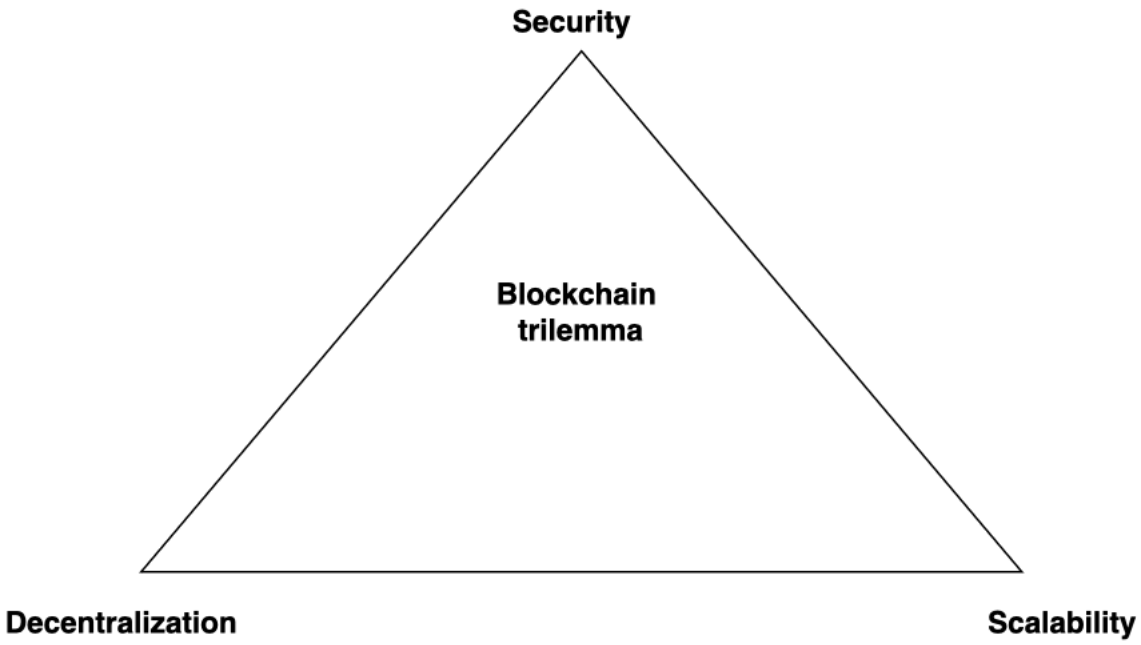

Figure 3.4: The blockchain scalability trilemma.

the risk of centralization, where control by a few entities can lead to manipulation and security breaches.

Scalability and Security: Enhancing scalability often involves reducing the number of nodes involved in the consensus process, which can compromise security. For instance, having fewer nodes can make the network more vulnerable to attacks and reduce the robustness of the consensus mechanism.

Consensus	**Decentralization**	**Scalability**	**Security**	**Examples**
PoW	High	Low	High	Bitcoin, Ethereum (pre-2.0)
PoS	Medium to High	Medium to High	Medium to High	Ethereum 2.0, Cardano
Raft	Low to Medium	High	Medium	Hyperledger Fabric
pBFT	Medium	High	Medium to High	Hyperledger Sawtooth, Tendermint
PoA	Low to Medium	High	Medium	VeChain, POA Network

Table 3.2: Comparison of consensus mechanisms with respect to the scalability trilemma.

Several existing blockchains attempt to balance these three aspects by adopting a semi-decentralized approach. Specifically, they select a limited number of consensus nodes to form a committee. A consensus protocol (e.g., pBFT) is then carried out among the committee members instead of the

entire network, as seen in Hyperledger Fabric. This results in significantly higher throughput compared to the fully decentralized approach. When electing nodes for the committee, reliability can be assessed based on factors such as the amount of stake (PoS) or reputation score.

Various research studies have explored methods to improve scalability without sacrificing decentralization or security [71]. Some potential techniques include:

- **Sharding**: This involves splitting the blockchain into smaller, more manageable pieces called shards, each capable of processing transactions independently [72]. However, sharding can reduce the decentralization and security of each shard, as smaller groups of nodes manage each shard.
- **Layer-2 Solutions**: These include various approaches such as rollups, sidechains, and state channels, which process transactions off the main blockchain to reduce congestion and enhance scalability.
  - *Rollups*: Bundling multiple transactions into a single transaction off-chain and then posting the compressed data back to the main chain.
  - *Sidechains*: Independent blockchains running in parallel to the main chain, with their own consensus mechanisms and security models.
  - *State Channels*: Enabling multiple transactions between parties off-chain, with only the initial and final states recorded on the main chain.

  These methods can face challenges, such as security vulnerabilities in the bridges connecting layer-2 solutions to the main chain and potential attacks targeting these connections.

In conclusion, achieving a balance between decentralization, scalability, and security remains a central challenge in blockchain design. While semi-decentralized approaches and advanced techniques like sharding and layer-2 solutions offer promising pathways, each comes with its own set of trade-offs and challenges that need to be carefully managed.

3.2 Smart Contracts

A smart contract functions as a trusted distributed application that gains its security and trust from the blockchain and the underlying consensus among its peers. It embodies the business logic of a blockchain application. Three key points apply to smart contracts, especially when deployed on a platform:

- Many smart contracts run concurrently in the network,
- They may be deployed dynamically, often by anyone, and
- Application code should be treated as untrusted and potentially even malicious.

Most existing smart-contract capable blockchain platforms follow an order-execute architecture, where the consensus protocol:

- Validates and orders transactions, then propagates them to all peer nodes,
- Each peer then executes the transactions sequentially.

This order-execute architecture is prevalent in virtually all existing blockchain systems, ranging from public/permissionless platforms such as Ethereum (with PoW-based consensus) to permissioned platforms such as Tendermint and Quorum.

Smart contract execution in a blockchain operating with the order-execute architecture must be deterministic; otherwise, consensus might never be reached. To address non-determinism, many platforms require smart contracts to be written in non-standard or domain-specific languages (such as Solidity), which can eliminate non-deterministic operations. This requirement can hinder widespread adoption as it necessitates developers to learn new languages and can lead to programming errors. Moreover, since all transactions are executed sequentially by all nodes, performance and scalability are limited. The fact that smart contract code executes on every node in the system necessitates complex measures to protect the overall system from potentially malicious contracts, ensuring system resiliency.

3.2.1 How Smart Contracts Work

Smart contracts operate on blockchain networks, where their code is stored and executed. They are triggered by specific events or conditions predefined within the contract. The blockchain ensures that once deployed, the contract code cannot be altered, providing immutability and transparency. The smart contract lifecycle consists of four steps:

- *Creation*: The smart contract is coded and deployed onto the blockchain. This involves writing the contract terms in a programming language like Solidity (for Ethereum) and deploying it via a transaction.
- *Execution*: When predefined conditions are met, the smart contract automatically executes the corresponding actions, such as making a payment when a specific task is completed and verified.
- *Verification*: The decentralized nature of the blockchain ensures that all nodes can independently verify the contract's execution, preventing the single point of failure and ensuring trust.
- *Immutability*: Once deployed, the smart contract code is immutable, ensuring that the contract terms cannot be tampered with, providing security and trust.

3.2.2 Notable Smart Contract Platforms

Several blockchain platforms facilitate the deployment and execution of smart contracts. Some of the most notable platforms include:

Ethereum: Ethereum is the most widely used platform for smart contracts. It provides a robust and flexible environment for developing dApps through its Ethereum Virtual Machine (EVM) [64]. Smart contracts on Ethereum are written in Solidity, a high-level programming language designed specifically for this purpose.

Cardano: Cardano is a blockchain platform that focuses on security and scalability. It uses a unique proof-of-stake consensus mechanism called Ouroboros. Smart contracts on Cardano are

written in Plutus, a Haskell-based language that enables formal verification to ensure the correctness of contract code.

Polkadot: Polkadot enables interoperability between different blockchains, allowing smart contracts on one blockchain to interact with those on another. It uses a multi-chain framework that enhances the functionality and utility of smart contracts.

Tezos: Tezos is a platform designed for deploying smart contracts and decentralized applications, with a focus on formal verification to ensure the correctness of contract code. Smart contracts on Tezos are written in Michelson, a stack-based programming language.

3.3 Hyperledger Fabric blockchain and smart contract

3.3.1 Architecture

Hyperledger Fabric is an open-source enterprise-grade blockchain framework designed to support the development of permissioned blockchain networks. It provides a modular architecture that allows for high flexibility and customization, making it suitable for various industrial applications. This section details the key components of Hyperledger Fabric and their roles in the overall architecture.

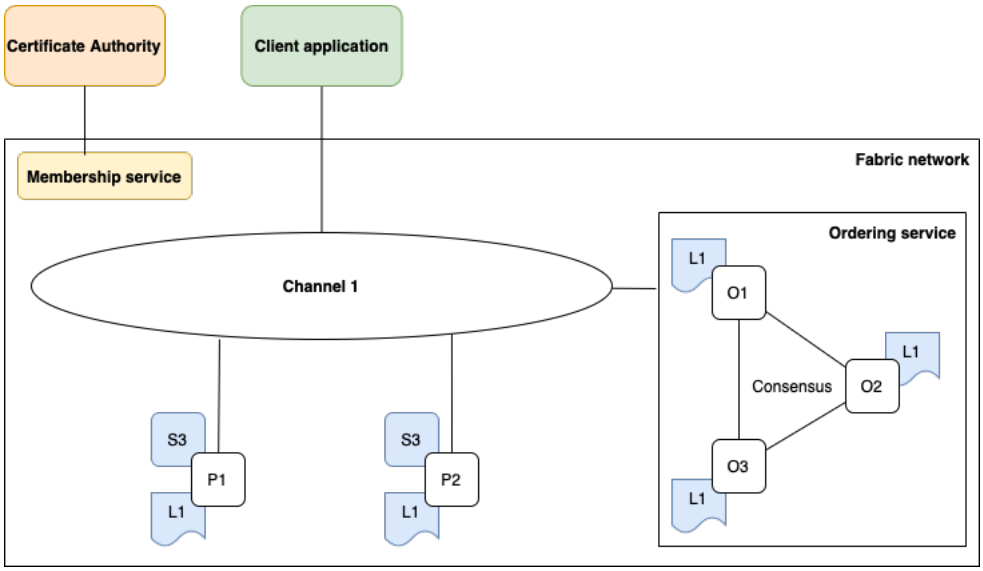

Figure 3.5: Hyperledger Fabric architecture featuring a channel (Channel 1) that includes smart contract (S3) and ledger (L1). The architecture shows five participants: peer nodes (P1, P2) and orderer nodes (O1, O2, O3).

3.3.1.1 Peer Nodes

Peer nodes are the fundamental building blocks of the Hyperledger Fabric network. They maintain the ledger, execute smart contracts (chaincode), and validate transactions. There are different types of peers in Hyperledger Fabric:

- **Endorser Peers**: These peers execute chaincode to simulate transaction proposals. They endorse transactions by signing the simulation results, which are then sent back to the client application.
- **Committer Peers**: These peers maintain the blockchain ledger. They validate and commit endorsed transactions into the ledger.
- **Anchor Peers**: These are designated peers within an organization that communicate with peers from other organizations. They help in the discovery process and ensure network connectivity.

3.3.1.2 Ordering Service

The ordering service is a crucial component responsible for ordering transactions and creating blocks. It ensures that all transactions are processed in the correct order. The ordering service can be implemented using various consensus mechanisms, such as Kafka, Raft, or a Byzantine Fault Tolerant (BFT) protocol. It operates independently from the peer nodes and does not execute, validate, or endorse transactions.

3.3.1.3 Channels

Channels are private sub-networks within the Hyperledger Fabric network that allow a specific group of participants to create a separate ledger. Each channel has its own ledger and chaincode, ensuring data privacy and isolation among different business entities. Transactions within a channel are only visible to the participants of that channel.

3.3.1.4 Membership Service Provider

The Membership Service Provider is responsible for managing identities in the Hyperledger Fabric network. It issues and validates certificates for participants, enabling secure and authenticated communication within the network. Membership Service Provider ensures that only authorized entities can participate in the network operations.

3.3.1.5 Certificate Authority

The Certificate Authority issues digital certificates that serve as credentials for participants in the Hyperledger Fabric network. These certificates are used to authenticate identities and secure communications. Each organization in the network typically operates its own Certificate Authority to manage the issuance, renewal, and revocation of certificates.

3.3.1.6 Ledger

The ledger in Hyperledger Fabric consists of two parts: the blockchain and the world state.

- **Blockchain**: an immutable log of all transactions that have been validated and committed by the network. Each block contains a list of transactions and is linked to the previous block through a cryptographic hash.
- **World State**: a database that holds the current value of the ledger states. It represents the latest state of all assets as modified by the transactions recorded in the blockchain. The world state can be implemented using different databases, such as LevelDB or CouchDB.

3.3.1.7 Hyperledger Fabric's Smart Contracts

In Hyperledger Fabric, smart contracts are referred to as chaincode. Chaincode is written in general-purpose programming languages such as Go, Java, and Node.js. This flexibility allows developers to leverage existing programming skills to create robust and secure smart contracts.

The lifecycle of chaincode in Hyperledger Fabric involves several key steps:

1. *Development*: Writing the chaincode in a supported programming language, defining the contract's logic and functions.
2. *Packaging*: Packaging the chaincode into a deployable format.
3. *Installation*: Installing the packaged chaincode on the peer nodes of the blockchain network.
4. *Approval*: Gaining approval for the chaincode definition from the required number of organizations in the network.
5. *Commitment*: Committing the chaincode definition to the network, making it available for execution.
6. *Invocation*: Invoking the chaincode functions to execute the smart contract logic based on predefined conditions.

Chaincode operates distinctly within the Hyperledger Fabric network, offering a more controlled and secure environment compared to smart contracts on public blockchains. It executes transactions within a secure and permissioned environment, providing businesses with the high levels of data privacy and security they require. Additionally, chaincode supports versioning and upgrades, allowing business logic to evolve over time without disrupting the network, thus adding flexibility and longevity to blockchain applications to accommodate changing business needs. Furthermore, chaincode execution is governed by an endorsement policy, which requires specific peers on the network to validate a transaction before it is committed to the ledger. This endorsement mechanism enhances security and consensus, ensuring that all transactions adhere to the network's agreed-upon rules.

3.3.1.8 Client

Clients represent the end user or client applications that trigger blockchain events. Clients are software programs that propose transactions on a network on behalf of a person. Clients communicate with peers and ordering services. The Client interacts with the network using a Fabric SDK. When reading or writing data from a Fabric blockchain the clients interact with the SDK. To ensure that a legitimate client has started the network transaction, the CA authority issues a certificate to the client as well.

3.3.2 Transaction Flow

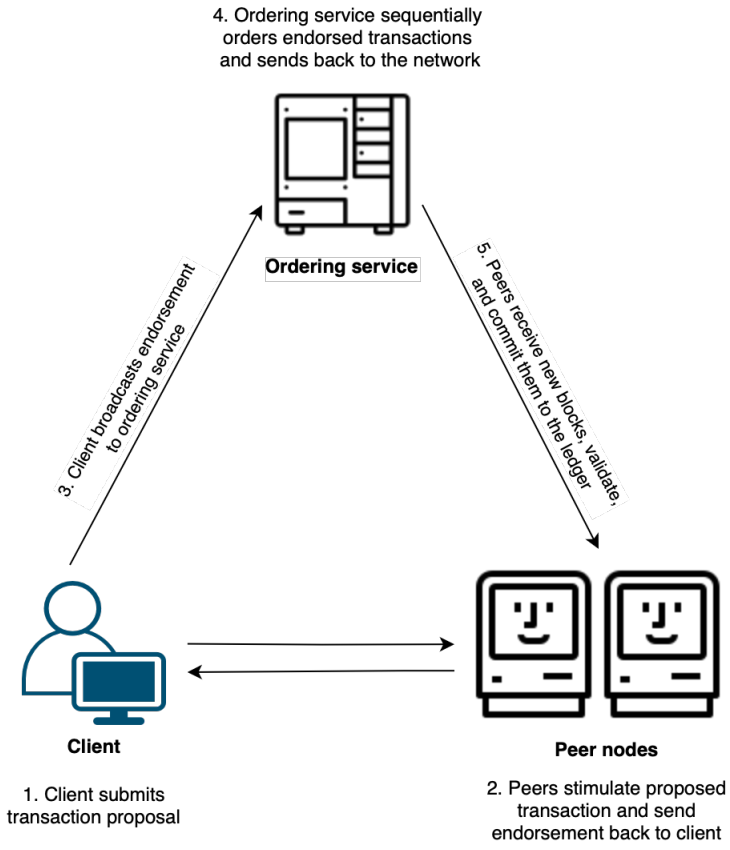

Figure 3.6: Hyperledger Fabric Transaction Flow

Figure 3.6 illustrates the transaction flow in Hyperledger Fabric that is described in the following.

1. The transaction flow begins when a client application sends a transaction proposal to peers in each organization for endorsement.

2. The peers verify the submitting client's identity and authority to submit the transaction. Next, they simulate the outcome of the proposed transaction and, if it matches what was expected, they send an endorsement signature back to the client.

3. The client collects endorsements from peers, and once it receives the proper number of endorsements defined in the endorsement policy, it sends the transaction to the ordering service.

4. The ordering service checks if the transaction has the proper number of endorsements to satisfy the endorsement policy. It then chronologically orders and packages the approved transactions into blocks and sends these blocks to peer nodes in each organization.

5. Peer nodes receive new blocks of transactions from the ordering service, and then perform a final validation for transactions in that block. Once this is complete, the new block is added to the ledger and the state of the ledger is updated. The new transactions are now committed.

3.4 Raft consensus

The Raft consensus algorithm is a popular protocol designed to manage a replicated log in a fault-tolerant manner. It is particularly well-suited for use in distributed systems and is known for its simplicity and understandability compared to other consensus algorithms like Paxos. The Raft algorithm ensures that multiple nodes in a distributed system can agree on a single sequence of actions to be executed, even in the presence of failures.

Raft achieves consensus through an elected leader. A server in a Raft cluster can be a leader, follower, or candidate (during an election). The leader manages log replication to followers and sends heartbeat messages to confirm its presence. Followers expect these heartbeats within a specified timeout (typically 150-300 ms). If no heartbeat is received, the follower becomes a candidate and initiates a leader election. Raft decomposes the consensus problem into two sub-problems: Leader election and log replication.

3.4.1 Leader Election

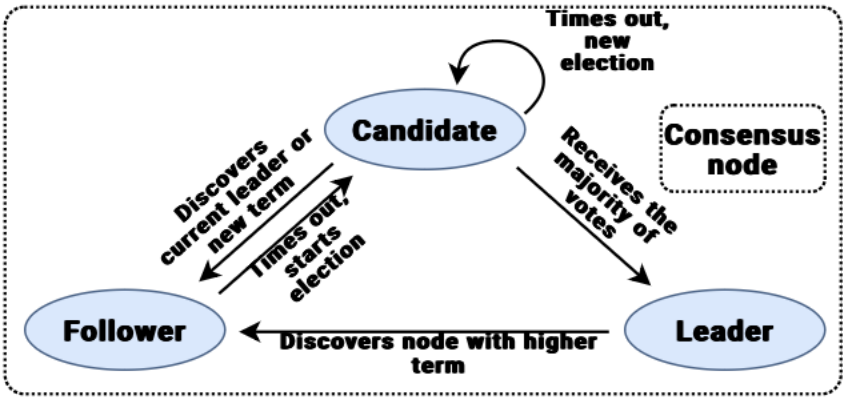

Figure 3.7: Raft leader election

When a leader fails or the algorithm initializes, a new leader is elected. Each term starts with an election. If an election succeeds, the term continues with the new leader. If it fails, a new term begins with another election. A server becomes a candidate if it doesn't receive a heartbeat within the election timeout. It increments its term number, votes for itself, and requests votes from other servers. A server votes only once per term. If a candidate receives a majority of votes, it becomes the leader. If there's a split vote, a new term and election begin. Raft uses a randomized election timeout to minimize split vote issues, allowing one server to timeout, win the election, and send heartbeats before others become candidates.

3.4.2 Log Replication

The leader handles log replication by accepting client requests and appending them as new log entries. These entries are sent to followers as AppendEntries messages. If followers are unavailable, the leader retries indefinitely until the entry is stored. Once a majority of followers confirm replication, the leader commits the entry, applies it to its state machine, and notifies followers to do the same. This ensures log consistency and respects the Log Matching safety rule. In case of a leader crash, the new leader handles log inconsistencies by comparing its log with followers, identifying the last matching entry, and synchronizing logs to restore consistency.

3.4.3 Safety Mechanisms

Raft guarantees these five safety properties to ensure reliable operation:

- Election Safety: Only one leader is elected per term.
- Leader Append-Only: A leader can only append new entries to its logs.
- Log Matching: If two logs contain an entry with the same index and term, they are identical up to that index.
- Leader Completeness: A committed log entry in a term is present in logs of leaders for all subsequent terms.
- State Machine Safety: No server applies different commands for the same log entry. This rule is upheld by requiring candidates to have all committed entries in their logs to win an election.

Raft determines which log is more up-to-date by comparing the index term of the last entries in the logs, ensuring the most recent and complete log is propagated.

Raft is designed to handle follower crashes and ensure availability. If a follower crashes, other servers continue to send AppendEntries and vote requests until the follower recovers. Upon recovery, the follower processes any pending requests, ensuring the system remains consistent and operational. Timing is critical in Raft to maintain a stable leader and cluster availability. Stability is maintained if the broadcast time is significantly less than the election timeout, which in turn must be significantly less than the mean time between failures (MTBF). Typical values for broadcast time range from 0.5 to 20 ms, and election timeout is set between 10 to 500 ms, providing sufficient stability for clusters to operate reliably for weeks or months between failures.

3.5 Summary

The chapter provided a comprehensive overview of several fundamental blockchain technologies and concepts that form the backbone of modern decentralized systems. We explored the architecture of blockchain technology, highlighting its key components, taxonomy, consensus mechanisms, and the inherent trade-offs encapsulated in the blockchain scalability trilemma. By understanding the intricate balance between decentralization, scalability, and security, we gain insights into the challenges and solutions that shape the design of robust blockchain networks.

Additionally, the concept of smart contracts was introduced, showcasing their role in automating and securing agreements on the blockchain. Our exploration included a detailed look at Hyperledger Fabric, an enterprise-grade blockchain framework, where we discussed its architecture and the unique implementation of smart contracts, known as chaincode. Finally, we covered the Raft consensus algorithm, emphasizing its significance in ensuring reliable and fault-tolerant consensus within the network.

This chapter has laid a solid foundation for understanding the technical intricacies and practical applications of blockchain technology, setting the stage for further exploration in the subsequent chapter.

Chapter 4

MetaAICM: Decentralized Crowdsourcing and Marketplace for Machine Learning

The content of this chapter was published in the following paper [73]:

H. D. Le, V. T. Truong, and L. B. Le, "Blockchain-empowered metaverse: Decentralized crowdsourcing and marketplace for trading machine learning data and models," *IEEE Access*, vol. 12, pp. 68556–68572, May 2024.

4.1 Abstract

The Metaverse relies on advanced machine learning (ML) techniques to facilitate the seamless mapping between the virtual and physical realms. ML-based technologies also enable metaverse service providers (MSPs) to offer a diverse range of intelligent virtual services to metaverse users (MUs). However, it can be challenging for MSPs to collect sufficient metaverse data to train ML models by themselves. As a result, MSPs can be interested in seeking contributions from MUs in both ML data and models. To address these challenges, we propose MetaAICM, a blockchain-based framework that empowers the metaverse through two key components. Firstly, it incorporates a distributed

crowdsourcing system that allows MSPs to gather metaverse data and ML models from MUs. Secondly, it features a decentralized marketplace, enabling MUs to proactively collect data and train ML models for sale using their metaverse devices and computing resources. MetaAICM leverages blockchain and smart contracts to achieve decentralization, ensuring security and privacy without relying on a trusted third-party authority or additional trust assumptions between MUs and MSPs. Numerical studies show that MetaAICM offers high processing performance and cost efficiency, while the framework is implemented on top of a consortium blockchain to show its feasibility.

4.2 Introduction

The metaverse, envisioned as the next-generation Internet, is an immersive digital representation of the physical world where users can participate via augmented/virtual reality (AR/VR) devices [74–76] and wearable haptic devices [77]. It enables individuals to engage in a wide range of virtual activities such as learning, gaming, working, and socializing [78]. Among various advanced technologies, artificial intelligence (AI) plays a pivotal role in enabling this virtual realm [7, 10]. For example, computer vision (CV) algorithms leverage real-world data captured by Internet-of-Things (IoT) devices and unmanned aerial vehicles (UAVs) to construct the metaverse's virtual environment. Additionally, machine learning (ML) models deployed on wearable devices analyze user data, such as appearance, gestures, and facial expressions, to reflect user behaviors into avatar actions [11]. While metaverse publishers typically handle these tasks, metaverse service providers (MSPs) also require ML models to deliver virtual services to metaverse users (MUs). For example, MSPs can utilize generative models to create AI-generated content like metaverse virtual items, clothing, and decorations for the virtual environment [12]. Another example is related to virtual events where event organizers can employ detection models to identify unauthorized MUs accessing the virtual space by analyzing their profile such as reputation, location, and operational history.

MSPs could be interested in seeking and leveraging the expertise in AI/ML and computational resources from the metaverse community to construct and train their desired ML models. On the other hand, ML professionals capable of designing and training ML models may face challenges in collecting the required data due to privacy concerns and the large-scale and heterogeneous nature of the metaverse. As a result, it is mandatory to provide MUs and MSPs with a decentralized

platform that allows them to exchange both metaverse data and ML models in a trustless distributed environment with low trading fees, high performance, and privacy preservation.

Traditional trading and crowdsourcing systems offered by third-party authorities are often vulnerable to a single point of failure (SPoF) and trust issues, free-riding and false-reporting attacks among participants, in addition to the lack of transparency, privacy, and incentive mechanisms [13]. Although these limitations might be acceptable for certain conventional centralized platforms, they should be eliminated to fit with the decentralized metaverse where scams and frauds are exacerbated by various novel social engineering attacks [14]. To this end, blockchain is a potential solution for trust management in such metaverse trading systems with immutable, transparent, and auditable properties [15]. In particular, blockchain provides a reliable storage environment to record trading transactions, while smart contracts and blockchain's committee can replace third-party authorities in decision-making, policy enforcement, reputation, and incentive management [16].

Motivated by the aforementioned urgent needs, this work proposes MetaAICM, a blockchain-empowered framework that enables AI crowdsourcing and provides the marketplace to facilitate intelligent virtual services in the metaverse. In MetaAICM, MUs can leverage their devices' capability and available resources to collect data or train ML models for specific purposes, then list them on MetaAICM's marketplace for sale. On the other hand, if MSPs cannot find the expected data/models on the marketplace, they can initialize a crowdsourcing task on MetaAICM's decentralized applications to obtain the desired products from other MUs. MetaAICM ensures that payment is only finalized when the traded product is valid, while both parties do not need to trust each other. To address scalability, transaction speed, and sustainability, MetaAICM incorporates a reputation-based Raft consensus protocol.

4.2.1 Related Work and Research Gaps

In any crowdsourcing and trading systems, it is important to evaluate the contribution of each worker, thereby distributing the award fairly to honest workers and punishing malicious workers who submitted low-quality results. Most existing frameworks assume that there exists an evaluation function that can, somehow, assess the quality of the submitted data/models automatically. Unfortunately, assessing the quality of data is not a simple task. Therefore, the novel contribution of MetaAICM compared to existing works is that it enables a practical solution for evaluating the

submitted data/models. With a novel decentralized design, MetaAICM does not require the assumption of such an evaluation function, while also resolves various security risks such as privacy leakage, false-reporting, and free-riding attacks.

4.2.1.1 Blockchain-Based Data Marketplace

Prior to our work, several studies explored the applications of blockchain for data trading and data marketplace. For instance, the authors in [23] designed a blockchain-based market for IoT data trading with three different trading modes, namely general trading, selling on demand, and buying on demand. All of these functions are regulated by smart contracts, thereby eliminating SPoF and the intervention of the third-party middleman. However, this framework does not take into account the situation in which the sellers act dishonestly by committing incorrect data. In such cases, the buyers will lose their funds while only receiving low-quality data. Furthermore, without proper encryption techniques, the traded data can be leaked widely due to the transparent property of blockchain. To address the privacy issue, the authors in [24] take a novel approach to data trading, in which the sellers are not required to send the original data to the buyers. Instead, a buyer only purchases the processed analysis results of the seller's dataset. The analysis process is carried out by certain trusted nodes that use the trusted hardware called Software Guard Extensions (SGX) to execute smart contract functions. However, the smart contract is only suitable for relatively simple tasks due to its resource limitation and complexity, thus limiting the framework's capacity.

Regarding the marketplace, the authors in [25] implemented a decentralized IoT data marketplace, which is based on blockchain to improve fairness and trust management. This framework supplements a security manager layer consisting of multiple storage operators who take responsibility for data storage. In addition, the traded data are encrypted before being sent to the storage operators to ensure privacy. However, if the buyers claim that they obtained a low-quality dataset after the decryption, it is almost infeasible to determine which party (i.e., the buyer, seller, or storage operator) was malicious. The authors in [26], proposed a solution to data-trading center, which has the ability to retain data by taking advantage of blockchain and ML. In addition, it employs a similarity-learning technique to verify data availability, while introducing a new role called "arbitration institution" to resolve any controversy between the buyers and sellers. However, relying on

such the trusted third-party may cause the single-point-of-failure (SPoF) if it acts maliciously or being under attacks.

4.2.1.2 Blockchain for Crowdsourcing

There have been several existing works that employ blockchain technology to engineer different crowdsourcing frameworks. In particular, the authors in [31] proposed a blockchain-based crowdsourcing design named CrowdBC. In CrowdBC, different smart contracts are deployed to regulate various tasks, thereby limiting human interactions and trust issues. The framework can resist malicious requesters, workers, and miners. However, CrowdBC requires an assumption that the requesters must be able to provide an evaluation function for the requested task on smart contract codes to assess the contribution of workers. This assumption would be difficult to realize in practice since the requesters are often just unprofessional clients who cannot write smart contracts by themselves, while sophisticated tasks such as AI/ML are even more challenging to be implemented on smart contracts due to their complexity and resource demand. The authors in [32] proposed zkCrowd, a blockchain-enabled crowdsourcing platform emphasizing user privacy with the integration of zero-knowledge proof [79]. In zkCrowd, both public and private blockchains are leveraged to enable flexible adjustment of the privacy degree corresponding to the user's demand. However, zkCrowd also requires a reward distribution function that is capable of evaluating the contribution of workers and correspondingly distribute rewards to the workers, which would be difficult to realize in most practical use cases. On the other hand, several studies such as ZebraLancer [34] and BPCM [35] focus on resolving privacy issues in crowdsourcing systems. Although the frameworks can enable privacy preservation by hiding the true identities of participants, the problem regarding contribution evaluation remains unsolved.

4.2.1.3 Blockchain-Based Metaverse Applications

Blockchain technology has been leveraged to develop various metaverse applications [15,43,44]. The authors in [45] proposed using smart contracts to efficiently manage and automate the interaction between MSPs and MUs. This blockchain-based design allows MSPs to optimize resource allocation when offering virtual services and applications to MUs and encourages MUs to contribute their resource to support the operation of the metaverse thanks to an incentive mechanism based on

Table 4.1: Comparison between MetaAICM and existing frameworks.

Feature	**[23]**	**[24]**	**[25]**	**[26]**	**[31]**	**[34]**	**[32]**	**[35]**	**MetaAICM**
Practical contribution evaluation									✓
SPoF & manipulation resistance	✓	✓	✓		✓			✓	✓
False-reporting attack		✓		✓	✓	✓		✓	✓
Free-riding attack		✓	✓	✓	✓	✓	✓		✓
Sybil attack	✓	✓	✓		✓	✓	✓	✓	✓
Privacy leakage		✓	✓			✓	✓	✓	✓
Incentive mechanism		✓	✓	✓	✓	✓		✓	✓
Reputation system			✓		✓			✓	✓

Stackelberg game theory. The authors in [46] proposed a blockchain-based federated learning (FL) framework to enable the industrial metaverse. By leveraging blockchain, the system can offer two privacy options for FL in the physical and virtual spaces using various sub-blockchains, which communicate with each other by using cross-chain communication techniques. On the other hand, the authors in [8] constructed a campus-oriented prototype of the metaverse, in which blockchain is leveraged to enable the platform's economic system. For instance, the blockchain-based non-fungible token is used to represent virtual items and user-generated content. Nonetheless, the use of blockchain to enable metaverse AI/ML based services such as ML crowdsourcing or decentralized marketplace for ML models and metaverse data has not been investigated in these studies.

Comparison of our proposed design and existing frameworks is given in Table. 4.1. It can be seen that the proposed MetaAICM framework addresses the contribution evaluation issue and it tackles many important security risks and attacks, which have not been handled adequately by other related works.

The preliminary results of this work are presented in a conference paper where we described a decentralized ML crowdsourcing system for the metaverse, which solved the problem of contribution verification and reward distribution by using committee-based model validation and a blockchain oracle network. This work is an extension of the mentioned work, in which a decentralized marketplace is integrated to enable both selling and buying on demand. Furthermore, intensive experiments are supplemented to illustrate the full potential of MetaAICM.

4.2.2 Key Contributions and Paper Structure

To fill the research gaps presented above, this work proposes MetaAICM, a blockchain-empowered framework that enables ML crowdsourcing and marketplace to facilitate intelligent services in the metaverse. The novel contributions of MetaAICM can be summarized as follows:

- Our MetaAICM framework enables a blockchain-based decentralized marketplace that encourages MUs to proactively collect metaverse data or train ML models for sale. If the desired data and models are not available on the marketplace, the crowdsourcing mode allows MSPs to crowdsource ML models from machine learning workers (MLWs), while MLWs can also request metaverse data from data workers (DWs).
- Unlike existing blockchain-enabled frameworks, MetaAICM does not require task requesters to provide an evaluation function that can automatically assess the contribution of workers. Thanks to our decentralized design, other trust assumptions are also eliminated and the involved participants do not have to trust each other. A concrete incentive mechanism is designed to motivate MUs to contribute computational resources to empower metaverse ML services, while a reputation system is integrated to filter out malicious entities who act dishonestly.
- We show that MetaAICM can resist various security threats such as denial of service (DoS), SPoF, data leakage, and Sybil attacks. In addition, we present extensive numerical results to confirm that MetaAICM offers high performance and automation with low operation costs.

The rest of this work is structured as follows. Section 4.4 presents the system model and design overview of MetaAICM. Section 4.5 proposes the detailed architecture and operation of MetaAICM with security analysis. Section 4.6 describe the numerical results for MetaAICM and Section 4.7 concludes the paper.

4.3 Preliminaries

4.3.1 Blockchain

Blockchain is a decentralized digital ledger that records transactions across many computers in such a manner that the registered transactions cannot be altered retroactively. This technology ensures the integrity and transparency of data without the need for a central authority or intermediary. At its core, a blockchain consists of multiple blocks linked together as a chain, where each block contains a certain number of transactions. These transactions are verified by participants in the network, known as nodes, through a consensus mechanism, ensuring security and mutual agreement on the state of the ledger at any given time. The immutable nature of blockchain technology not only enhances security but also builds trust among users, making it a foundational technology for various applications in the metaverse [44], such as security [43], digital asset management [57,58], networking [59], and distributed learning [60].

4.3.1.1 Blockchain Types

In terms of accessibility, there are two main types of blockchain, including permissionless and permissioned blockchains [62]. Permissionless blockchain is sometimes referred to as public blockchain, which is open to the public and anyone can participate, read, write, or audit the blockchain without requiring explicit permission. Bitcoin [63] and Ethereum [64] are prominent examples of public blockchains, characterized by their transparency, security, and immutability. On the other hand, if a blockchain belongs to the permissioned category, its participants have to obtain explicit permission to join the network, which can be granted by a central authority or by existing participants. Permissioned blockchain includes private and consortium blockchains. In a private blockchain, there is a single authority controlling the network, while a consortium blockchain is often regulated by multiple authorities.

Permissionless blockchains provide the highest extent of transparency and openness, as their data are available on the Internet and anyone can download and audit the information. On the other hand, permissioned blockchains prefer privacy and controllability over transparency. Depending on specific use cases and system requirements, a suitable type of blockchain could be chosen accordingly.

4.3.1.2 Consensus Mechanism

Blockchain consensus refers to the mechanism by which all participants reach an agreement on the state of the network without the need for a central authority. Various consensus algorithms have been developed and used in different blockchain networks, each with its own set of advantages and trade-offs. Some of the most common consensus mechanisms include Proof of Work (PoW) [65,66], Proof of Stake (PoS) [67], Raft [20], practical Byzantine Fault Tolerance (pBFT) [68,69], and Proof of Authority (PoA) [70].

Consensus algorithms play a key role in deciding the characteristics of a blockchain network, including scalability, decentralization, and security. Furthermore, each consensus mechanism takes a different approach to offer fault tolerance capability. For example, pBFT tackles the byzantine problem via a three-stage voting mechanism, while PoA filters out malicious nodes by only selecting reputable nodes to conduct the consensus process. On the other hand, certain algorithms like Raft and Paxos offer crash fault tolerance instead of byzantine fault tolerance [20]. It is also possible to improve the security of Raft and Paxos by integrating additional techniques for selecting consensus nodes (e.g., techniques that are based on the amount of stake or reputation scores). Our design uses the reputation score for eliminating malicious nodes from the consensus, while the consensus process is crash fault-tolerant, which is the key characteristic of the Raft-based protocol.

4.3.1.3 Decentralization and Scalability

In any blockchain network, there is an inevitable trade-off between decentralization and scalability. A highly decentralized network like Bitcoin often suffers from low processing speed because it takes more time to reach consensus among numerous nodes due to networking delay and synchronization problems. In contrast, semi-decentralized blockchain systems usually offers higher scalability as they allow to reach consensus more quickly, but they are usually prone to manipulation and single point of failure (SPoF).

Several existing blockchains attempt to achieve a balance between these two aspects by taking a semi-decentralized approach. Specifically, they first select a limited number of consensus nodes to form a committee. Then, a consensus protocol (e.g., pBFT) is carried out among the committee members instead of the entire network like Bitcoin's PoW, resulting in a significantly higher

throughput compared to the fully decentralized approach. In terms of electing nodes for the committee, we can assess the reliability of each node based on different factors such as their amount of stake (i.e., PoS) or reputation score. Our work is also a semi-decentralized design, which offers a balance between decentralization and scalability.

We would like to note that various research studies have investigated different methods to improve scalability without sacrificing decentralization [71]. Some potential techniques are sharding [72] and various variants of layer-2 solutions such as rollups, sidechains, and state channels. However, each of the methods has its own limitations and design challenges. For example, sharding might decrease the decentralization and security of each shard, while the bridges from layer-2 blockchains to their main chains are prone to various attacks.

4.3.2 Smart contracts

Smart contracts have significantly expanded the utility of blockchain technology, extending its use beyond mere transactional functions. Essentially, a smart contract functions like a computer program embedded with predefined rules and logic. Its code is executed across multiple nodes within the blockchain network through a consensus mechanism. This execution process ensures that outcomes are determined by the majority consensus, safeguarding against manipulation as long as most nodes operate honestly. The transparency of smart contracts, with both their code and outcomes verifiable on-chain, bolsters trust and verification. However, this same transparency poses challenges, particularly for processing sensitive information like private keys or credentials, which become visible on the blockchain and accessible to all nodes. Several blockchain platforms now facilitate smart contract deployment, notably Ethereum [64] and Hyperledger [22]. These platforms are engineered to execute smart contracts reliably, without the risks of fraud, downtime, or interference from third parties.

4.3.3 IPFS data storage

The InterPlanetary File System (IPFS) [19] is a peer-to-peer distributed file system. By dispersing file storage across a network of nodes, IPFS enhances data redundancy and resilience, making it robust against loss. It uniquely identifies files and their constituent blocks through cryptographic

hashes, which not only secures data integrity but also streamlines data retrieval by fetching content from the nearest node rather than a central server. This decentralized approach significantly accelerates access speeds and decreases bandwidth demands. IPFS finds critical use in applications requiring data integrity and availability, notably in blockchain-based decentralized applications (DApps). Here, while the blockchain stores only the references to data in the form of IPFS URLs or hashes, the actual data resides on IPFS. This methodology drastically cuts storage costs and amplifies data accessibility.

4.4 System Model and Design Overview

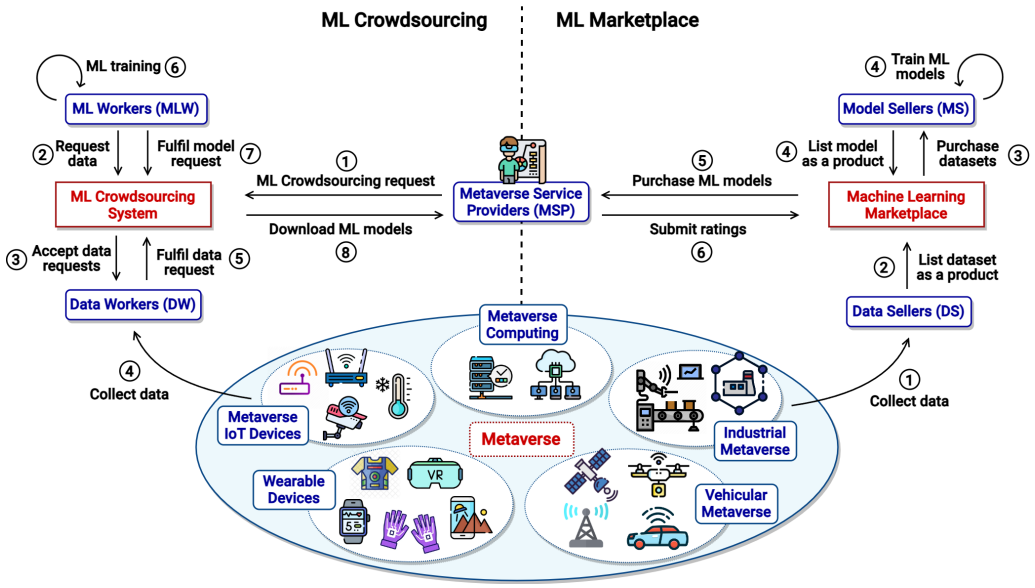

Figure 4.1: Overview of the metaverse and the architecture of MetaAICM with two operation modes, namely crowdsourcing system and decentralized marketplace.

4.4.1 System Model and Design Goals

As illustrated in Fig. 4.1, the metaverse is envisioned to leverage numerous devices (e.g., IoT sensors, smart vehicles, industrial devices, wearable AR/VR and haptic devices) and computational infrastructure (e.g., edge/cloud servers and data centers) for its operation. Specifically, these devices continuously collect and generate a massive amount of data, which is especially valuable for AI/ML-enabled tools and services [17]. However, these devices and the collected data are often owned by numerous distributed MUs instead of a centralized publisher like traditional media plat-

forms. Therefore, it poses significant challenges in data collection including privacy issues [18], trust management, and incentives for the data collectors. For instance, MUs may not want to disclose their data widely to the public (i.e., privacy issue), while malicious participants might try to obtain the data without paying tokens (i.e., trust issue).

Our design of the crowdsourcing system and marketplace aims to achieve the following goals:

- To engineer decentralized frameworks allowing metaverse data and ML models to be traded or crowdsourced for intelligent virtual services in the digital world.
- To effectively tackle the prevalent issues of security, privacy, and trust management that are inherent in a trustless environment and remain unaddressed thoroughly.
- To make sure that the proposed designs can achieve desired levels of scalability, decentralization, and security while being sustainable (i.e., it does not require a significant amount of energy for its operations as in the Bitcoin blockchain).

To this end, our proposed designs rely on a consortium blockchain using a Raft-based consensus mechanism as we will explain in more detail in the next section. In the following, the term "blockchain committee" is used to refer to a group of blockchain consensus nodes.

4.4.2 Design Overview

In this work, we propose both crowdsourcing and marketplace designs for different application scenarios as we will highlight in the following.

4.4.2.1 Overview of Crowdsourcing Design

In crowdsourcing systems, data collectors are called data workers (DWs), while ML professionals who offer ML training services are ML workers (MLWs). There are three main smart contracts regulating the crowdsourcing process in our proposed framework as follows.

- Machine Learning contract (MLC): It allows MSPs to crowdsource ML models from MLWs.

- Data contract (DC): This smart contract enables MLWs to crowdsource metaverse data from DWs.
- Reputation contract (RC): It manages the reputation profiles of participants and can be considered as a reference indicating the reliability of individuals.

The crowdsourcing mode's operation is illustrated in the left side of Fig. 4.1 with a total of 8 steps. Specifically, to crowdsource a ML model from MUs, a MSP_i first submits a ML crowdsourcing request to the smart contract MLC (step 1). The MSP_i also deposit to MLC a certain number of metaverse tokens as a reward for the workers. Then, interested MLWs can accept the request from MLC and start training the requested model. During the training process, the MLWs can further request more metaverse data from DWs if needed. To do so, a MLW_j must commit a data request to the smart contract DC (step 2), which triggers a new data crowdsourcing task. After some DWs collected data and fulfilled the data request (steps 4 and 5), the MLW_j can use the obtained data to train its model (step 6). Finally, once finishing the ML crowdsourcing task (step 7), the MSP_i will download the final ML model from MLC, and the deposited tokens are distributed automatically to honest workers.

4.4.2.2 Overview of Marketplace

Instead of waiting for specific crowdsourcing requests, MUs can also proactively collect metaverse data to build their own datasets and list them on the marketplace for sale. These MUs are called data sellers (DSs). Similarly, certain model sellers (MSs) can design/train ML models which meet the metaverse's demand, then sell them on the marketplace with a predefined price and descriptions. The marketplace's backend is constructed on top of a smart contract called marketplace contract (MPC). The general operation of the MetaAICM's marketplace is described in the right side of Fig. 4.1.

For privacy preservation, the ML models/data are stored on InterPlanetary File System (IPFS) [19] instead of using the on-chain storage. Moreover, the IPFS URL is encrypted by the buyer's public key. In case a buyer is not satisfied with the received product (e.g., she claims that the seller sent incorrect data, or the data has been modified), a two-stage comparison mechanism is activated to validate the reports. Consequently, malicious buyers will lose their tokens due to the dishonest

disputation reports, while the transaction is canceled automatically if the report is verified to be honest. When the payment is finalized, the buyer can submit a rating for the product with a score from 0 to 10, which reflects the product's quality and also impacts the seller's reputation.

4.4.3 Threat Model

The threat model is described in the following. In MetaAICM, it is assumed that any entities, including DWs, MLWs, DSs, and MSs, can be malicious. These involved participants do not trust each other and always want to maximize their benefit by committing malicious activities or colluding together to conduct attacks. Furthermore, we assume that there is no trusted authority in our framework, and the operation of any entity can be corrupted suddenly. Under these assumptions, there can be a wide range of potential attacks that MetaAICM aims to cope with as described in the following:

4.4.3.1 False-Reporting Attack

In the crowdsourcing mode, after receiving a high-quality dataset $\mathcal{D}_i$ from the worker DW_i, the data requester MLW_j may intentionally misreport that it has not received $\mathcal{D}_i$ from DW_i, or the received $\mathcal{D}_i$ is of low quality. The adversary goal is to avoid paying the fee $\mathcal{F}_i$ associated with $\mathcal{D}_i$. Similarly, a model requester MSP_j might also deny a high-quality model $\mathcal{M}_i$ committed by the worker MLW_i to repudiate the payment $\mathcal{F}_i$. As a result, honest workers would not be rewarded for their contribution and effort, while the malicious requesters could obtain high-quality products (i.e., $\mathcal{D}_i$ and $\mathcal{M}_i$) without paying the associated fee $\mathcal{F}_i$.

In the ML marketplace, a buyer MSP_i/MS_i could also conduct false-reporting attack by claiming that the purchased model/dataset is low-quality or even inaccessible, although that product is valid and of high quality. Consequently, it impacts directly on the benefits of the data/model sellers and allows illegal financial gains from the buyer side.

4.4.3.2 Free-Riding Attack

Regarding the crowdsourcing system, a worker MLW_i could accept a model request from a MSP_j, then only commit an arbitrary model $\mathcal{M}_i$ instead of making real effort to train the model. The purpose of this malicious MLW_i might be either attacking the requester MSP_j with a harmful model, or just for gaining metaverse tokens without real contribution. Similarly, a malicious DW_i could also submit an arbitrarily low-quality dataset to poison the MLWs and obtain the free-riding rewards. In contrast to false-reporting attack, this free-riding attack poses a threat to the benefits of the requesters instead of the workers.

In the marketplace of MetaAICM, a similar form of free-riding attack can manifest when a seller sells a product that is substantially lower in quality than described, or not as advertised. Such a deceitful act leads buyers to pay for products of negligible value, while enabling the seller to earn tokens with minimal or no genuine effort in data collection or model training. This form of attack directly impacts the marketplace's reliability and can erode the trust of buyers in the system.

4.4.3.3 Sybil and DoS Attack

Sybil attack [80] refers to the circumstance in which an attacker generates a large number of fake identities to attain superior impact, thus manipulating the system. For instance, in the data crowdsourcing system, a Sybil worker DW_i may use its k fake identities to submit k invalid datasets for a crowdsourcing task $\mathcal{T}_j$, increasing the possibility that one of its datasets is chosen and rewarded by the requester MLW_j. In terms of the marketplace, the Sybil identities can dominate the rating/reputation system by submitting multiple dishonest ratings to the seller.

Using a similar approach, the attacker can also conduct a DoS attack by committing numerous buying/selling requests on the marketplace, surpassing the processing capacity of the blockchain. As a result, this might corrupt the entire system instead of each individual.

4.4.3.4 Privacy Leakage

In the crowdsourcing system, when a data worker DW_i submit its dataset $\mathcal{D}_i$ to fulfil a data request from the requester MLW_j, it is possible that $\mathcal{D}_i$ is also accessible by other entities in the system

as on-chain information is transparent. As a result, some malicious entities may obtain sensitive information from $\mathcal{D}_i$, or even use such this dataset to fulfil the corresponding data request $\mathcal{T}_i$ and compete with the original worker DW_i. This not only raises privacy issues, but also threatens the workers' benefit.

On the other hand, hackers can attack the marketplace's storage environment to access the datasets and ML models listed on the marketplace. Consequently, the attackers could steal sensitive information from the data/models without paying the products' fees.

4.5 Proposed Framework

We describe the proposed metaverse crowdsourcing system and marketplace design in this section.

4.5.1 Metaverse ML Crowdsourcing System

The proposed crowdsourcing system is depicted in Fig. 4.2 with two main components, which are ML model crowdsourcing and data crowdsourcing as presented in the following.

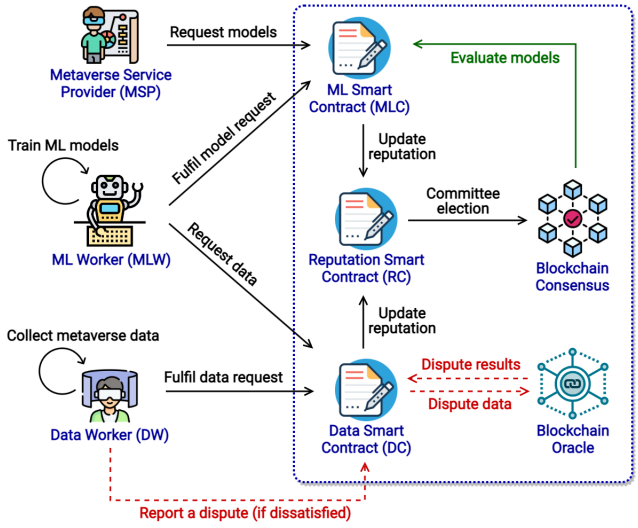

Figure 4.2: The crowdsourcing system of MetaAICM.

4.5.1.1 ML Model Crowdsourcing

The proposed crowdsourcing design for ML models is summarized in Algorithm 4.1, in which a MSP_i crowdsources ML models from n different MLWs. Firstly, MSP_i commits a "model crowdsourcing" transaction to activate the MLC smart contract, where the transaction structure is as follows:

$$T_{mc} = \{T_{\text{token}} = \Theta | T_{\text{desc}} | t_{\text{deadline}} | R_{\min} | Sig_{sk}(T_{mc})\}, \tag{4.1}$$

where Θ is the task reward (resulting in certain metaverse tokens), T_{desc} is the task description (i.e., requirements for the crowdsourced model), t_{deadline} is the task deadline, $R_{\min}$ is the minimum reputation required to join the task, and $Sig_{sk}(T_{mc})$ is the digital signature signed by the MSP_i's secret key.

MLWs who are interested in the crowdsourcing task can train an ML model that satisfies the requirements in the task description. Once finishing training, the participated MLWs upload their solutions (i.e., the trained ML models) to IPFS, then submit the IPFS URLs to the smart contract MLC. When the task deadline is reached, the task requester MSP_i must publish the validation data D_{val} to the blockchain committee via IPFS. Next, consensus nodes of the committee download all crowdsourced ML models and the validation data from IPFS, then use this dataset to evaluate the models. Based on the evaluation results, the highest-performance model ($\text{Model}_{\text{best}}$) is selected as the final solution for the task. The IPFS URL of the selected model is emitted to the requester MSP_i by the smart contract MLC, thus finishing the model crowdsourcing task.

The MLW whose model was chosen as the final solution is rewarded 50% of the deposited tokens Θ, while the remaining tokens are distributed to other MLWs whose model's performance is higher than a predefined threshold (e.g., the accuracy being greater than 85%). However, if the requester MSP_i fails to publish the validation data when the deadline is reached, the deposited tokens are distributed equally to all MLWs who submitted a model, regardless of their model's performance. In that case, although the $\text{Model}_{\text{best}}$ is not determined, the requester still obtains all the submitted models.

Algorithm 4.1. ML Model Crowdsourcing

Input: MSP_i, T_{desc}, t_{deadline}, $R_{\min}$, D_{val}, Θ, $\mathcal{W} = \{\text{MLW}_i\}_1^n$.
Output: The highest-performance model $\text{Model}_{\text{best}}$.

MSP_i initializes a model crowdsourcing request via a transaction: $T_{mc} = \{T_{\text{token}} = \Theta \mid T_{\text{desc}} \mid t_{\text{deadline}} \mid R_{\min} \mid \text{Sig}_{sk}(Tx)\}$;
for $\text{MLW}_j \in \mathcal{W}$ **do**
    if reputation of $\text{MLW}_j < R_{\min}$ **then**
        MLW_j is refused;
    else
        MLW_j trains Model_j according to task description T_{desc};
        MLW_j stores Model_j on IPFS to obtain the link URL_j;
        MLW_j submits URL_j to the smart contract MLC;
    end if
end for
if $t_{\text{current}} == t_{\text{deadline}}$ **then**
    MSP_i publishes the validation data D_{val} to the committee;
end if
The committee downloads all submitted models from IPFS;
The committee evaluates every model on the validation data D_{val} to obtain $\text{Model}_{\text{best}}$ with the highest performance;
return $\text{Model}_{\text{best}}$;

4.5.1.2 Data Crowdsourcing

If MLWs do not possess the necessary data to train their models, they can initialize a data crowdsourcing task to collect the desired metaverse data from other MUs. In this case, the MLWs are also called data requesters. In comparison to the model crowdsourcing process presented above, the data crowdsourcing procedure poses other fundamental challenges:

Privacy Issue: Unlike ML models, the leakage of crowdsourcing data may threaten the privacy of DWs. Therefore, the crowdsourcing data must be encrypted before being submitted to ensure privacy preservation.

Data Evaluation: While the quality of ML models can be evaluated based on a common metric such as accuracy or loss, there is no similar standard metric for data evaluation. Therefore, MetaAICM allows data requesters to evaluate the crowdsourced datasets and decide the reward distribution by themselves. False-reporting attacks are discouraged by requiring data requesters to deposit crowdsourcing fees into the smart contract DC in advance, while the decentralized blockchain oracle can resolve any disputation between data requesters and DWs.

The proposed data crowdsourcing process is summarized in Algorithm 4.2, in which a data requester MLW_i must commit a "data crowdsourcing" transaction to initialize the task where the transaction content is:

$$T_{dc} = \{T_{\text{token}} = \Theta + \alpha | f_{\text{data}} | pk | t_{\text{deadline}} | R_{\min} | Sig_{sk}(T_{dc})\}, \tag{4.2}$$

where α is a compensation stake that will be recompensed to the DWs whose dataset is honest but not recognized by the requester, f_{data} is a list of desired features for the crowdsourcing dataset, pk is MLW_i's public key, and the remaining elements are similar to those presented in (4.1).

Interested DWs can collect data to fulfill the crowdsourcing request. The collected datasets must comprise all data features listed in f_{data}. Upon reaching the designated deadline, DWs upload their datasets to IPFS, receiving a unique URL for each submission. They then encrypt these URLs using the MLW_i's public key pk_i, and each resulting encrypted string is submitted to the smart contract DC via a transaction Sub_j. Although accessible publicly, these encrypted strings can only be decrypted by MLW_i using their private key, ensuring the privacy of the submitted data.

Consequently, MLW_i can retrieve all the crowdsourced datasets from the smart contract DC. MLW_i must evaluate these datasets and submit the evaluation results to the DC for reward distribution. The evaluation results indicate which datasets are honest (e.g., clean and high-quality datasets) or dishonest based on MLW_i's decision. As a result, Θ deposited tokens are distributed to DWs whose dataset is stated to be honest, while the rest of the DWs will not receive any reward.

In cases where DWs contest the evaluation results, they are entitled to initiate a dispute. This is done by triggering the DC's disputation function and submitting the URL of their datasets hosted on the IPFS. The smart contract DC then verifies the legitimacy of each disputation request by ensuring it aligns with the corresponding transaction Sub_j, where a DW_j previously submitted an encrypted URL. Upon confirmation of this alignment, the dispute is escalated to the decentralized blockchain oracle network. This network comprises multiple professional nodes that independently assess whether the disputed datasets meet the required quality standards. Each node casts a vote to determine if a dataset is qualified, and the final decision is based on the compounded result of the majority votes. This decision is then relayed back to DC, which enforces the outcome. If the DWs succeed in their dispute, they are collectively awarded the compensation stake α, shared

equally among them. In contrast, if there is no disputation request or no DWs won their dispute, the compensation stake α is given back to MLW_i after certain block times t_{lock}.

Algorithm 4.2. Data Crowdsourcing

Input: MLW_i, f_{data}, pk, t_{deadline}, $R_{\min}$, Θ, α, $\mathcal{W} = \{\text{DW}_i\}_1^n$.
Output: qualified datasets D.

$T_{dc} = \{T_{\text{token}} = \Theta + \alpha | f_{\text{data}} | pk | t_{\text{deadline}} | R_{\min} | Sig_{sk}(T_{dc})\}$;
for $\text{DW}_j \in \mathcal{W}$ **do**
    if reputation of $\text{DW}_j < R_{\min}$ **then**
        DW_j is refused;
    else
        DW_j collect Dataset_j according to f_{data};
        DW_j stores Dataset_j on IPFS, obtaining URL_j;
        DW_j encrypts URL_j, obtaining an encrypted string eURL_j;
        DW_j submits eURL_j to DC via transaction Sub_j;
    end if
end for
if $t_{\text{current}} == t_{\text{deadline}}$ **then**
    MSP_i publishes evaluation results to smart contract DC;
    DC distributes rewards to qualified datasets;
end if
if DW_j send dispute request and URL_j to DC **then**
    DC verifies the dispute request aligns with Sub_j;
    DC triggers the Oracle network to re-evaluate Dataset_j;
    if Dataset_j is qualified **then**
        DC sends compensation to DW_j;
    end if
end if
return D;

4.5.2 Metaverse ML Marketplace

According to the market's demand, MUs can proactively gather data or train ML models for sale on MetaAICM's marketplace, bypassing the need to wait for specific crowdsourcing requests. To simplify the discussion, both datasets and ML models are referred to as "products" in this context, with the entities involved being broadly categorized as buyers and sellers.

To list a new product on the marketplace, a seller first uploads the product's data to IPFS and obtains a corresponding IPFS URL. The URL can be considered a representation of the product as everyone can download the product if they have its URL. The seller then initiates the marketplace listing process by submitting a transaction to MPC, structured as follows:

$$T_{\text{list}} = \{P_{\text{price}} | P_{\text{desc}} | \text{H}_\text{s} | Sig_{sk}(T_{\text{list}})\}, \tag{4.3}$$

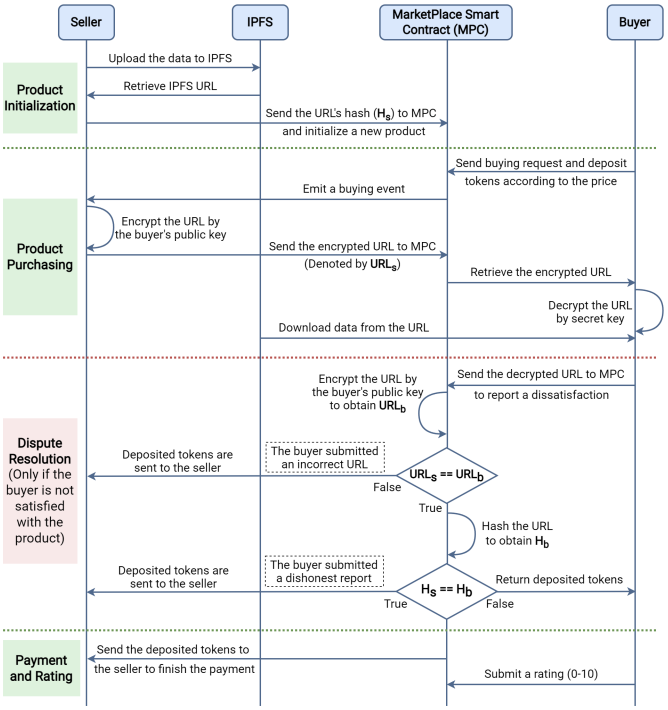

Figure 4.3: The workflow of MetaAICM's marketplace.

where P_{price} is the product's price, P_{desc} is the product's description (e.g., features list of a dataset, or ML model's type such as regression/classification), H_s is the hash of the IPFS URL, and $Sig_{sk}(T_{\text{list}})$ is the transaction's digital signature.

A buyer interested in purchasing a product initiates the process by submitting a buying request and depositing the requisite tokens, as per the product's price, into the MPC. Upon receiving this request, the MPC automatically emits an event to notify the seller. Subsequently, the seller encrypts the product's IPFS URL using the buyer's public key, resulting in an encrypted URL denoted as URL_s, which is then sent to the MPC. The buyer retrieves URL_s from the MPC and decrypts it using their secret key to access the original URL and download the product's source data. While the encrypted URL_s is publicly accessible as on-chain information, decryption is exclusive to the buyer due to the need for their secret key.

In cases where the buyer is not satisfied with the product, suspecting issues such as data modification or seller misconduct, they have a set period t_{chal} to initiate a dispute. This is done by invoking the "disputation" function of the MPC, accompanied by the decrypted URL. The MPC re-encrypts this URL with the buyer's public key to generate URL_b and then compares it with the previously stored URL_s. Should there be a discrepancy between URL_s and URL_b, it indicates the

buyer has intentionally provided an incorrect URL, leading to the transfer of the buyer's deposited tokens to the seller, thereby finalizing the purchase. Conversely, if the URLs match, the MPC hashes URL_b to get H_b, and compares it with H_s from the listing transaction (4.3). A mismatch at this stage implies a modification of the IPFS URL during the transaction, leading to a refund of the deposited tokens to the buyer. If no discrepancies are found, the buyer might be regarded as a potential DoS attacker, and their deposit is transferred to the seller.

If no disputes arise within the timeframe t_{chal}, the deposited tokens are automatically transferred to the seller, thus concluding the payment process. Additionally, the buyer has the option to rate the purchased product, on a scale from 0 to 10, using the "rating" function of the MPC. This rating contributes to the product's average score, which is displayed on the marketplace for reference. The workflow of the MetaAICM marketplace is presented in Fig. 4.3.

4.5.3 Incentive and Reputation System

MetaAICM's incentive framework includes token rewards and a reputation mechanism. Token rewards are allocated to the entities guaranteeing the operation of the system: blockchain oracle nodes and consensus nodes. Oracle nodes whose decision is the same as the majority of the oracle network are rewarded metaverse tokens, while those whose decision is opposite will face deductions. Consensus nodes in the committee also receive tokens for each new block added onto the blockchain after the consensus process.

Beyond token incentives, MetaAICM integrates reputation scores to incentivize contributors for their honesty and valuable contributions. The reputation scores play a crucial role in determining participants' eligibility for selection as consensus nodes in the blockchain committee for subsequent rounds. The reputation management is autonomously handled by the smart contract RC. This smart contract tracks and updates each avatar's reputation score based on contributions to the system. Importantly, all changes to reputation scores result from transactions, which are executed and verified by the blockchain committee, guaranteeing that no individual can unilaterally modify their reputation score; attempts to do so through unauthorized transactions will be rejected.

For each participant i, the reputation score r_i is updated in response to their behaviors in defined contexts:

- Model Crowdsourcing: MLWs whose model's performance is higher than the required threshold will receive a reward. The remaining MLWs incur a reputation penalty due to their low-quality models.
- Data Crowdsourcing: DWs whose datasets are accepted by the data requester, or win disputes, will receive reputation score rewards. The remaining DWs have their reputation scores slashed. In case of successful disputes, the data requesters also face reductions in their reputation scores.
- Marketplace: Sellers gain reputation reward for receiving high ratings (e.g., ratings ≥ 8) and lose reputation for low ratings (e.g., ratings ≤ 3). Moreover, each negative disputation decision results in punishment to the reputation score of the seller/buyer.

Let α refer to the change in reputation score following an event, with $\alpha = R$ standing for the rewarding case and $\alpha = P$ standing for the penalty case. Thus, the reputation r_i can be computed as follows:

$$r_i = \begin{cases} r_i + \frac{A_i}{P_i}, & \text{if } \alpha = R \\ r_i - 1, & \text{if } \alpha = P \end{cases}$$

where A_i represents the number of unique avatars interacted with by user i through crowdsourcing and marketplace activities, P_i represents the total participation instances of user i, including all submissions of crowdsourcing tasks and buying/selling activities in the marketplace. This incentive mechanism ensures that rewards are directly proportional to the diversity and frequency of positive contributions, thereby enhancing the system's resilience to collusion attacks. In the Metaverse, the number of unique avatars is significantly fewer than the number of activities, which influences the rate of reputation increase. Consequently, users with numerous activities experience a slower reputation increase compared to new users. This design is effective in preventing the domination of the system by longstanding users and encourages new users to actively contribute to the Metaverse.

4.5.4 Reputation-Based Raft Consensus

The developed smart contracts are deployed in a consortium blockchain that must be appropriately designed to meet our design goals. Note also that the consensus mechanism deployed to reach agreements on newly added blocks over time strongly impacts the resulting blockchain performance.

In this work, our selection of a consensus mechanism for MetaAICM aims at managing the blockchain network effectively while maintaining transaction integrity, security, and efficiency. Considering the demands of the metaverse environment, particularly the need for high transaction throughput and a decentralized, trust-based system, we identify the following requirements for our consensus mechanism:

- High Transaction Throughput: Our consensus protocol must efficiently handle large-scale metaverse data, supporting the network's high transaction rates.
- Alignment with Reputation Mechanism: The consensus algorithm must be well-suited for the integration of a reputation system for node selection, enhancing network trust and reliability.
- Fault-Tolerance Capacity: In a distributed system with multiple consensus nodes, some of the nodes might crash or behave maliciously during its operation, thus hindering the consensus process. Therefore, an important requirement of MetaAICM's consensus algorithm is to ensure its seamless operation even if a certain proportion of the consensus nodes are inoperative.
- Sustainability: Unlike energy-intensive algorithms like PoW [81], our consensus protocol must be efficient in both energy consumption and data storage, thereby contributing to the overall sustainability of the entire blockchain network.

In the MetaAICM, the consortium blockchain plays the key role in managing identity and ensuring the trustworthiness of the system. Common consensus mechanisms include PoW, PoS, PBFT, BFT-Smart [82], and Raft. PoW and PoS are predominant in public blockchain environments, where participants are anonymous. These consensuses, while central to common cryptocurrencies (e.g., Bitcoin, Ethereum) have limitations in scalability (i.e., limited transaction throughput), making them less ideal for the Metaverse. Conversely, PBFT, BFT-Smart, and Raft are frequently utilized in consortium blockchains. However, PBFT and BFT-Smart still have the scalability limitation due to the heavy communication overhead of their voting mechanisms, especially as the network size expands. On the other hand, Raft offers better performance and scalability even when the number of nodes is large. It is also known for its simplicity and understandability, simplifying the implementation and maintenance process [20]. It is designed primarily for crash-fault tolerance, allowing the system to sustain up to 50% node failures [83], yet lacks safeguards against malicious behaviors. Therefore, to address this limitation, we construct the MetaAICM's consensus protocol

partly based on Raft, with the integration of our reputation system for the election of consensus nodes. This integration aims to mitigate the vulnerability to malicious activities, hence, enhancing the security and reliability of consensus in a consortium blockchain setting.

In particular, MetaAICM features a blockchain committee comprising multiple consensus nodes. These nodes are responsible for transaction verification, block proposal, and ensuring a consistent ledger across the network. The committee's structure includes an *authorized committee partition*, which reserves u% of slots for metaverse organizations responsible for maintaining the infrastructure and operations of the virtual world. The rest of the slots are allocated to normal nodes through a reputation-based election mechanism, forming what we term the *dynamic committee partition*. This partition is subject to re-election every k rounds, as illustrated in Fig. 4.4, ensuring continual adaptability and responsiveness to the evolving needs of MetaAICM, while the authorized partition provides stability and ongoing governance.

4.5.4.1 Consensus Process

The consensus mechanism in MetaAICM, based on the Raft model [20], is integral to maintaining a consistent and reliable blockchain ledger. This process is depicted in Fig. 4.4 and involves three distinct states for a consensus node: follower, candidate, and leader. The primary goal of this mechanism is to achieve a robust, democratic, and failure-resistant system for block proposal and ledger consistency.

Consensus Operation: The Raft model achieves consensus through a streamlined leader-based approach. In this system, the designated leader is responsible for log management and data replication to the follower nodes. The leader node proposes new log entries (blocks) and ensures that followers replicate these entries consistently. Followers, in turn, accept and apply these entries to their local state. This process minimizes the risk of conflicting entries and ensures a single, agreed-upon sequence of logs across the network. The efficiency of the Raft model lies in its simplicity – with a single leader coordinating log replication, the process becomes more predictable and manageable, reducing the overheads typically associated with more complex consensus models.

Leader Election: The model's leader election is a democratic process, transitioning nodes through the roles of follower, candidate, and leader. All nodes initially act as followers. If a

follower fails to receive a heartbeat message from the current leader within a specified time frame, it perceives this as a signal of leader failure and shifts to a candidate role. The candidate then seeks votes from other nodes to become the new leader. Achieving a majority vote is crucial for the candidate to assume the leader role, preventing any single node from dominating the process and ensuring wide network support for the elected leader. The leader, once elected, resumes sending heartbeat messages to maintain its authority and keep the network's state synchronized.

This Raft-based election process has been shown to offer 50% fault-tolerance capacity [20]. In other words, even if up to 50% of the nodes are compromised, MetaAICM still operates correctly and seamlessly. Furthermore, as there is only one leader verifying transactions at a time, it reduces significantly energy consumption while improving the transaction throughput. As a result, MetaAICM provides a higher extent of sustainability compared to other energy-intensive consensus protocols like PoW.

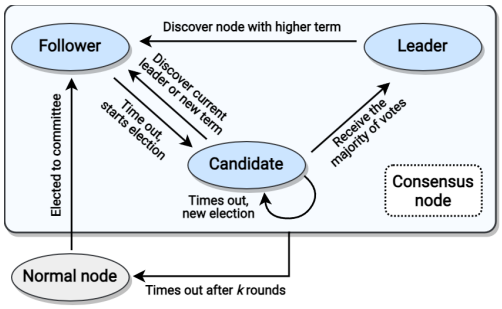

Figure 4.4: State transition of a consensus node.

4.5.4.2 Reputation-Based Committee Selection

In MetaAICM, the composition of the dynamic committee partition is periodically refreshed every k rounds. This process is orchestrated by the current leader, who executes Algorithm 4.3 for selecting new nodes for the upcoming period, where M denotes the number of nodes to be elected. The leader begins the election process by generating a random number and its corresponding proof, $\langle \phi_1, \pi \rangle$, using a Verifiable Random Function (VRF) algorithm [21]. The seed for the VRF is derived from the hash of the previous block, thereby ensuring resistance to manipulation. The leader then iteratively hashes ϕ_1 for $M-1$ times to produce a set of M random numbers, denoted as $\Phi = \{\phi_1, ..., \phi_M\}$. Each ϕ_i in Φ is instrumental in the node selection. The probability of a node being elected is proportional to its reputation score. The reputation scores of all nodes are represented as K indexed reputation

units, $R_{\text{spread}} = \{r_1, ..., r_K\}$. For each ϕ_i, the leader selects a reputation unit, r_{Idx}, from R_{spread}. The node owning r_{Idx} is then elected to the committee, ensuring that nodes with higher reputation are more likely to be chosen.

By integrating this reputation-based committee election, we ensure that highly reputable nodes are selected to regulate the consensus process, while those with low reputation are not able to dominate the system. This further improves the reliability of the framework. Moreover, the election process can be verified by any participant via validating the proof π, thereby preventing any single entity from manipulating the system.

Algorithm 4.3. Reputation-Based Committee Election

Input: Indexed reputation units $R_{\text{spread}} = \{r_1, ..., r_K\}$, number of nodes M, previous block $\mathcal{B}$.
Output: Nodes list for the next dynamic partition $\mathcal{N} = \{N_1, ..., N_M\}$, proof of randomness π.

1: Generate seed from block: seed $\leftarrow$ hash($\mathcal{B}$);
2: Compute VRF random number and proof: $\langle \phi_1, \pi \rangle \leftarrow \text{VRF}_{sk}(\text{seed})$;
3: Generate $M-1$ additional random numbers by hashing ϕ_1 $M-1$ times, obtaining $\Phi = \{\phi_1, ..., \phi_M\}$;
4: **for** each $\phi_i \in \Phi$ **do**
5: Compute index: $\text{Idx} = K \cdot \frac{\phi_i}{2^{||\phi_i||}-1}$, where $||\phi_i||$ is the length of ϕ_i in bits;
6: Append the node N_i with reputation unit at Idx to $\mathcal{N}$;
7: **end for**
8: **return** $\mathcal{N}, \pi$

4.5.4.3 Bootstrapping Consensus process

At the bootstrapping stage of the system, all nodes are initialized with an equal reputation score of zero. To mitigate the risk of malicious entities infiltrating the blockchain committee during this initial phase (i.e., the cold-start period), reputation scores are not utilized for committee selection for the first W rounds. Instead, the MetaAICM platform publisher appoints a set of Trusted Seed Nodes (TSNs) to form the blockchain committee. After the cold-start period, once W rounds have elapsed and nodes have been assigned reputation scores reflecting their behavior, the committee election mechanism (refer to Algorithm 4.3) activates, electing consensus nodes based on these scores.

Metaverse avatars can apply to become TSNs during MetaAICM's initialization phase, with selections made based on the avatars' profiles. Operating within a consortium blockchain framework, MetaAICM ensures that the identities of these avatars are authenticated before they can assume roles within the system.

4.5.5 Security Analysis

4.5.5.1 False-Reporting Attack

In MetaAICM's crowdsourcing mode, false-reporting attacks are prevented by requiring the task requesters to deposit the task reward Θ when submitting the crowdsourcing transaction T_{mc}/T_{dc}. Once the crowdsourced solutions are revealed, the crowdsourcing smart contracts automatically distribute Θ to every MLW_i and DW_j whose solutions are not considered low-quality or malicious. Therefore, requesters cannot repudiate the payment by false reporting.

Regarding false-reporting attacks in the marketplace, MetaAICM deployed the smart contract MPC to efficiently eliminate false-reporters. If a buyer tries to report a dispute indicating that the received data is not the same as the one listed on the marketplace, this request will be validated by MPC via a two-step verification process. Firstly, if the buyer intentionally committed an incorrect URL (i.e., $\text{URL}_\text{s} \neq \text{URL}_\text{b}$), the report is considered a false-reporting attack and the tokens Θ are sent to the seller. Then, if the URL's hashes are identical between the seller and buyer (i.e., $\text{H}_\text{s} = \text{H}_\text{b}$), this means that the seller submitted a correct product and the buyer is trying to conduct a false-reporting attack.

4.5.5.2 Free-Riding Attack

In terms of crowdsourcing, every Model_i from $\text{MLW}_i \in \mathcal{W}$ is validated with the validation data D_val of the task publisher. This helps filter out the free-riders who submitted low-quality results. Furthermore, these malicious workers are subject to certain punishments, resulting in the reduction of tokens and reputation scores. On the other hand, the crowdsourced datasets can be validated by the decentralized blockchain oracle via the disputation mechanism. This further prevents the existence of free-riding data workers.

In the MetaAICM's marketplace, free-riding attacks are prevented based on the rating system. If a Model_i or a Dataset_j is of low quality, it would receive low ratings from the buyers. These ratings not only deter the low-quality products, but also impact the overall reputation scores of the free-riding sellers. As a result, the buyers are aware of each product's quality based on its ratings, while the free-riders are exposed due to their low-reputation profile.

4.5.5.3 Privacy Leakage

In MetaAICM, data privacy is ensured based on different encryption techniques. Specifically, each URL_i associated with the crowdsourced Dataset_i is encrypted by the MLW's public key pk_j, ensuring that it is not accessible by any parties except the task requester MLW_j. Similarly, the datasets and models on the marketplace are also protected by the buyer's public key. Only the buyer with its private key can carry out decryption and obtain the products. Other blockchain entities, although being able to read any transaction information, cannot access the IPFS storage to download the data.

4.5.5.4 Sybil and DoS Attacks

MetaAICM employs a comprehensive reputation system designed to thwart Sybil attacks effectively. Within this framework, each avatar operates as an independent entity. Thus, the system prevents the aggregation of reputation scores from multiple avatars into a single, artificially inflated score. This system ensures that participants with multiple avatars, particularly those with low reputation scores, are restricted in their influence and capabilities within the network. By limiting the participation of such low-reputation entities, the integrity of the consensus process and other critical operations of the system is maintained. Therefore, our design significantly diminishes the potential impact of Sybil attackers, who typically rely on creating numerous fake identities to manipulate or disrupt network activities.

In the case of DoS attacks, MetaAICM's design inherently discourages such attempts. Once a group of attackers collude to conduct DoS, their balance would be drained quickly due to the punishment mechanism. Therefore, DoS attackers will gain nothing other than the loss of their tokens.

4.5.5.5 Trust Issue and SPoF

MetaAICM effectively addresses trust issues and eliminates SPoF by adopting a decentralized architecture across all its operations. The platform's approach involves a crowdsourcing system and marketplace that operate without any centralized authority, relying instead on a consortium blockchain to distribute trust across the network. This ensures that no single entity controls the system,

thereby enhancing security and reliability for all participants. In MetaAICM, both data and ML models are stored on the IPFS, a peer-to-peer storage solution. Moreover, the role of MSPs is designed to contribute to the metaverse's development without centralizing control or influence over the blockchain or the system. This decentralization ensures that MetaAICM's operations are transparent and verifiable, fostering a secure and trustworthy environment. By maintaining the platform through a robust network of decentralized nodes, we not only bolster its resilience against failures and disruptions but also guarantee uninterrupted service availability.

4.6 Performance Evaluation

4.6.1 Experimental Setup

MetaAICM is deployed as a permissioned blockchain based on Hyperledger Fabric [22], an open-source blockchain development platform. The implementation source code is published on GitHub[1] with detailed instructions. To take part in the system, each blockchain node in MetaAICM maintains a Docker container to execute its operation (e.g., training ML models, performing consensus mechanisms, executing smart contracts, and submitting transactions). Consensus nodes use PyTorch to perform model evaluation in model crowdsourcing. A blockchain benchmarking tool named Hyperledger Caplier is used to simulate transaction workloads and monitor the network's performance. IPFS is also re-implemented locally to prevent Internet latency, with 100 peer-to-peer nodes.

4.6.2 Performance Evaluation

4.6.2.1 Blockchain Performance under Different Workloads

In the experiment illustrated in Fig. 4.5, 100 clients are set up to simultaneously submit transactions invoking smart contract functions of three main systems, namely data crowdsourcing, ML model crowdsourcing, and decentralized marketplace. The transaction workload increases from 100 transactions per second (TPS) to 2000 TPS. According to Fig. 4.5, when the workload is less than 1000 TPS, the system can afford most of the submitted transactions with a transaction processing rate of more than 92%. However, when the workload increases beyond 1200 TPS, the transaction

[1] https://github.com/duyhung2201/MetaAICM

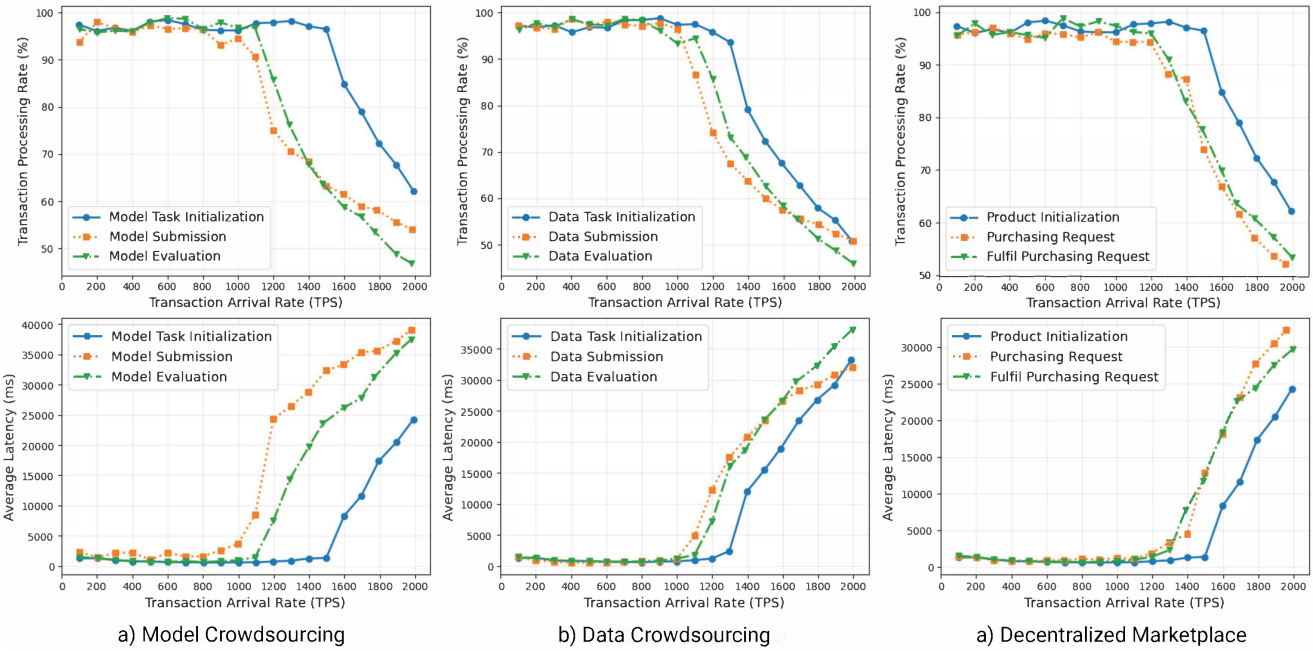

Figure 4.5: Performance of smart contract's functions under different workloads, ranging from 100 to 2000 TPS.

processing rate decreases sharply. At 2000 TPS, only 50% of transactions are processed in each round. This indicates that the system's processing capacity is limited to around 1000–1200 TPS. Similarly, the average latency of the network (i.e., the average time it costs for a transaction to be processed) is negligible until reaching the mentioned saturation point of 1000 TPS.

4.6.2.2 Blockchain Performance with Varying Block Size

Fig. 4.6 presents the results of another experiment in which the blockchain performance is monitored with different block sizes (from 100 to 1000 transactions per block) and under different workloads (from 700 to 2000 TPS). It is shown that the system with the smallest block size of 100 transactions can efficiently handle the workload of 1200 TPS. With a higher transaction workload, its performance starts decreasing rapidly. Intuitively, if the block size is small, it costs more blocks to process/store the same number of transactions. However, Fig. 4.6 also shows that an excessively large block size of 1000 transactions per block achieves lower performance than another test case with 500 transactions per block. This is because the consensus time is often higher with larger block sizes. When the block size becomes larger than the actual processing demand, it might cause redundancy and lead to lower performance.

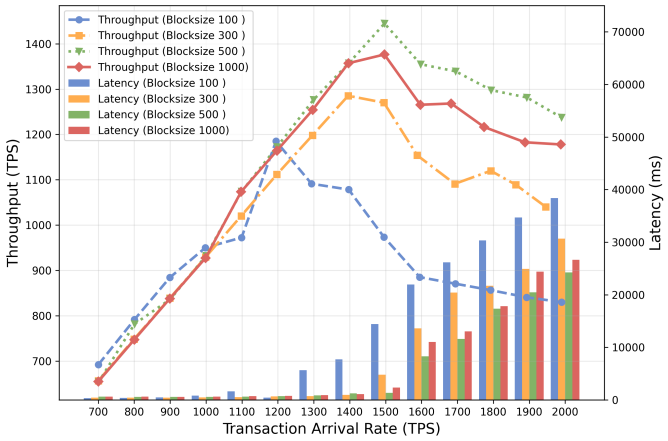

Figure 4.6: Blockchain performance as the block size varies from 100 to 1000 transactions per block.

4.6.2.3 Consensus Performance

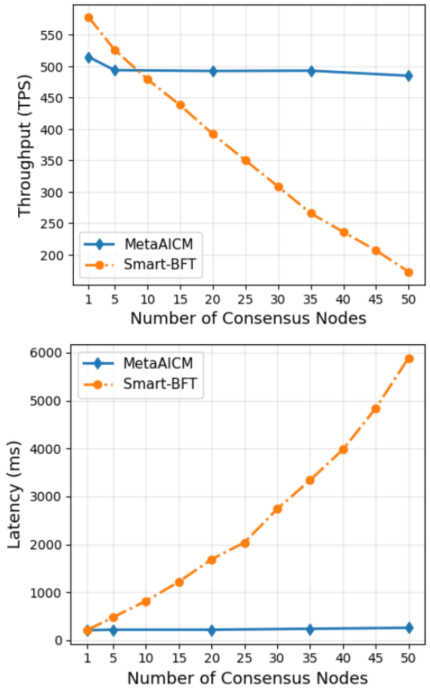

Figure 4.7: Performance of MetaAICM's consensus compared to BFT-Smart when the committee size varies from 1 to 50.

In terms of blockchain consensus, the performance of MetaAICM's consensus is compared to another baseline framework using BFT-Smart consensus algorithm [82]. The transaction arrival rate is set to 600 TPS in this experiment, and the committee size ranges from 1 to 50 consensus nodes. As shown in Fig. 4.7, MetaAICM maintains a high throughput and low latency regardless of the committee size, while the baseline system's performance drops rapidly when the committee

size increases. As a result, MetaAICM allows more consensus nodes to participate in the system, making it more decentralized without sacrificing throughput and latency.

4.6.2.4 Storage Environment Evaluation

Table 4.2 shows both the upload and download time of the system according to several benchmarking ML datasets and models. In general, a larger dataset/model often leads to a higher upload and download time. On the other hand, it is observed that the upload time is significantly higher than the download time. The main reason for this additional delay is that a file must be divided into smaller chunks when being uploaded to IPFS, while each chunk also costs a certain amount of time to generate a corresponding hash value. In contrast, the download time is relatively smaller since the IPFS is deployed locally, making it easier to search for the desired data chunks based on the provided hashes.

Table 4.2: Upload and download time of the system according to different benchmarking datasets and ML models.

Dataset	**Size (KB)**	**Upload (ms)**	**Download (ms)**
Reuteurs	3931	79.08	7.80
Ratings1m	24,017	223.84	42.74
Flowers-102	24,567	234.55	43.98
IMDB	27,712	256.51	45.88
MNIST	53,594	374.25	66.41
LFW	176,334	1312.62	214.46
CIFAR-10	180,000	1335.29	264.69
SVHN	297,965	2341.91	368.98
Model	**Size (KB)**	**Upload (ms)**	**Download (ms)**
MobileNet	16,871	169.88	24.39
NASNetMobile	22,884	192.57	41.88
DenseNet21	32,635	265.78	44.36
Xception	89,889	694.78	141.49
InceptionV3	94,016	607.33	131.78
ResNet50	100,651	726.15	138.78
ResNet152V2	237,353	1698.98	357.59
ConvNeXtBase	346,879	2446.19	506.08
VGG16	540,532	3788.18	612.51

4.6.2.5 Reputation Score Experiment

This experimental analysis examines the behavioral patterns of five distinct user groups over 100 crowdsourcing tasks to evaluate the performance of the reputation mechanism, as depicted in Figure 4.8. The x-axis represents the cumulative number of crowdsourcing tasks, while the y-axis measures the average reputation score for each group. Specifically, Group 1 consistently submits honest work, whereas Group 4 only engages in malicious activities. Group 2 starts with honest contributions for the first 40 tasks before switching to malicious behavior for the remainder of the experiment. Group 3 exhibits a combination of honest and malicious submissions throughout.

Group 1 demonstrates a consistent increase in reputation, though with a diminishing rate, attributable to a decrease in the diversity of interactions (i.e., unique avatars encountered) as the number of tasks increases. Group 2 experiences an initial reputation boost due to positive contributions, followed by a significant decline post-task 40, coinciding with their shift to malicious activities. This reputational decline is initially steep, stabilizing as their score lowers, which is a consequence of reduced task eligibility due to previously malicious submissions. It can be noted that past behaviors are transparent in the blockchain, which affects the participating ability of bad entities in the system. Group 3's reputation fluctuates, mirroring their mixed behavioral pattern, with periods of increase reflecting honest contributions and decreases corresponding to malicious actions. Conversely, Group 4 undergoes a sharp decline in reputation due to persistent malicious behavior, with the rate of decline stabilizing towards the experiment's end. This stabilization occurs as their reputation diminishes, hence progressively restricting their access to tasks. Eventually, group 4 users are excluded from further participation in the system.

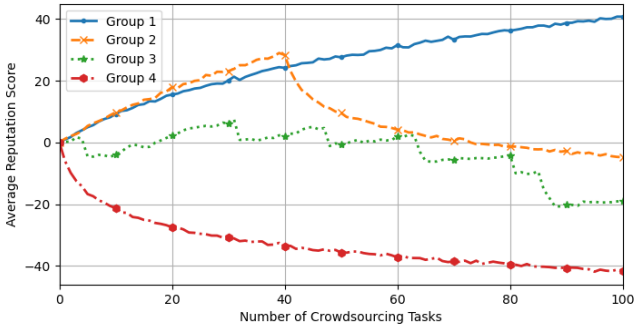

Figure 4.8: Average reputation scores change.

4.7 Conclusion

This work presented MetaAICM, our proposed blockchain-based framework aiming to maximize the potential of ML and intelligent services for the metaverse. In MetaAICM, we designed a distributed ML crowdsourcing system and a decentralized ML marketplace that enables both selling and buying on demand, serving both metaverse data and ML models. Instead of relying on a trusted authority, our design is completely decentralized and requires no trust assumption among participants, thereby being resistant to SPoF and trust issues. MetaAICM is robust against free-riding and false-reporting attacks thanks to the disputation resolution mechanism, while user privacy is guaranteed with the use of encryption techniques. Moreover, the concrete incentive and reputation mechanisms can efficiently eliminate malicious actors, while encouraging MUs to contribute their available recourse to the system. Experimental results showed that MetaAICM offers high performance with low processing latency and high throughput.

4.8 Open Challenges and Research Direction

4.8.1 Privacy Threats and Countermeasures

Although MetaAICM uses asymmetric encryption techniques to protect the trading data/models, other potential techniques could be useful to address privacy threats in more sophisticated metaverse application contexts such as homomorphic encryption, Zero-Knowledge Proofs (ZKP), Differential Privacy (DP), and Secure Multi-party Computation (SMC). However, it should be noted that each method comes with its own strength and disadvantages. For instance, homomorphic encryption and ZKP are restricted to a limited set of mathematical operations, while DP can reduce the quality of the data. Detailed studies of these techniques for different metaverse applications are outside the scope of our current work. Thus, we would like to leave them for our future work.

4.8.2 Scalability Issue

Based on the proposed Raft-based consensus algorithm, MetaAICM achieves the throughput around 1,500 TPS. As MetaAICM is only used for the crowdsourcing/marketplace purposes instead of other

applications like gaming, finance, healthcare, and education, this performance would be sufficient. However, a real-world metaverse environment might require even higher throughput as the number of metaverse users can be enormous. Therefore, development of efficient scalability techniques like sharding and layer-2 solutions is still necessary and worth further research. In our future work, we plan to apply scalability methods to scale up the throughput for the large-scale metaverse environment.

4.8.3 Data Evaluation Methods

While evaluating ML models is simply computing their performance on a given test data, evaluating the training data is much more challenging. In MetaAICM, we let requesters evaluate the data, and then allow workers to submit a dispute request in case of dissatisfaction, which will be verified by a blockchain oracle. However, the oracle is often less decentralized than the blockchain due to its small size. Therefore, design of an efficient method for automatically evaluating ML data is an open research direction.

Chapter 5

Conclusion and Future Work

5.1 Conclusion Remarks

This thesis presented MetaAICM, a blockchain-based framework designed to facilitate the deployment of machine learning (ML) and intelligent services for the metaverse. MetaAICM integrates a distributed ML crowdsourcing system with a decentralized ML marketplace, enabling both the selling and buying of metaverse data and ML models on demand. Unlike traditional systems that rely on a trusted authority, MetaAICM is fully decentralized, eliminating the need for trust among participants and making it resistant to the single point of failure (SPoF) and trust issues.

Our framework addresses common challenges in decentralized systems, such as free-riding and false-reporting attacks, through a robust disputation resolution mechanism. User privacy is safeguarded using asymmetric encryption techniques, and the incentive and reputation mechanisms efficiently filter out malicious actors while encouraging metaverse users to contribute their resources. Experimental results demonstrated that MetaAICM achieves high performance with low processing latency and high throughput, making it a viable solution to enable intelligent services in the metaverse.

5.2 Future Research and Extensions

While MetaAICM represents a significant advancement in leveraging blockchain technology for the metaverse, there are still several critical areas that require further exploration to fully realize its potential in a real-world setting. This section outlines key areas for future research and development, focusing on enhancing privacy, scalability, and data evaluation methods.

Privacy Threats and Countermeasures: Although MetaAICM employs asymmetric encryption techniques to protect trading data and models, further privacy enhancements are necessary for more sophisticated metaverse applications. Techniques such as homomorphic encryption, Zero-Knowledge Proofs (ZKP), Differential Privacy (DP), and Secure Multi-party Computation (SMC) offer promising avenues for enhancing privacy. However, each method has limitations: homomorphic encryption and ZKP support only a limited set of mathematical operations while DP can degrade data quality. Future research should explore these techniques to enhance privacy in trading systems without compromising scalability and performance.

Scalability Issues: MetaAICM, utilizing a Raft-based consensus algorithm, achieves a throughput of approximately 1,500 transactions per second (TPS). While this performance is sufficient for crowdsourcing and marketplace applications, a real-world metaverse environment with a vast number of users may demand higher throughput. Efficient scalability techniques, such as sharding and layer-2 solutions, are essential for meeting these demands. Future work will focus on implementing these scalability methods to enhance throughput and support large-scale metaverse environments.

Data Evaluation Methods: Evaluating ML models is straightforward, involving performance metrics on test data. However, evaluating the quality of training data poses a significant challenge. In MetaAICM, requesters assess the data quality, with workers having the option to dispute unsatisfactory evaluations. These disputes are verified by a blockchain oracle, which, due to its smaller size, is less decentralized than the blockchain itself. Developing an efficient and automated method for evaluating ML training data remains an open research direction. Future work will aim to design and implement such methods to ensure reliable data quality assessments in decentralized systems.

5.3 List of Publications

1. H. D. Le, V. T. Truong, and L. B. Le, "Blockchain-empowered metaverse: Decentralized crowdsourcing and marketplace for trading machine learning data and models," *IEEE Access*, vol. 12, pp. 68556–68572, May 2024.

2. H. D. Le, V. T. Truong, D. N. M. Hoang, and L. B. Le, "MetaCrowd: Blockchain-empowered metaverse via decentralized machine learning crowdsourcing," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2024, pp. 1–6, to appear.

3. V. T. Truong, H. D. Le, and L. B. Le, "Trust-free blockchain framework for AI-generated content trading and management in metaverse," *IEEE Access*, vol. 12, pp. 41815–41828, Mar. 2024.

4. D. N. M. Hoang, V. T. Truong, H. D. Le, and L. B. Le, "Delay and overhead efficient transmission scheduling for federated learning in UAV swarms," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2024, pp. 1–6, to appear.

References

[1] J. Sanchez, "Second life: An interactive qualitative analysis," in *Proc. Soc. Inf. Technol. Teach. Educ. Int. Conf.* Association for the Advancement of Computing in Education (AACE), Mar. 2007, pp. 1240–1243.

[2] J. D. N. Dionisio, W. G. Burns, and R. Gilbert, "3d virtual worlds and the metaverse: Current status and future possibilities," *ACM Comput. Surv.*, vol. 45, no. 3, pp. 1–38, Jul. 2013.

[3] A. Bruun and M. L. Stentoft, "Lifelogging in the wild: Participant experiences of using lifelogging as a research tool," in *Proc. Int. Conf. Hum.-Comput. Interact. (INTERACT)*. Springer, Aug. 2019, pp. 431–451.

[4] H. Ning, H. Wang, Y. Lin, W. Wang, S. Dhelim, F. Farha, J. Ding, and M. Daneshmand, "A survey on the metaverse: The state-of-the-art, technologies, applications, and challenges," *IEEE Internet Things J.*, Aug. 2023.

[5] C. Chen, L. Zhang, Y. Li, T. Liao, S. Zhao, Z. Zheng, H. Huang, and J. Wu, "When digital economy meets Web3.0: Applications and challenges," *IEEE Open J. Comput. Soc.*, vol. 3, pp. 233–245, Oct. 2022.

[6] L.-H. Lee, T. Braud, P. Zhou, L. Wang, D. Xu, Z. Lin, A. Kumar, C. Bermejo, and P. Hui, "All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda," *arXiv preprint arXiv:2110.05352*, Oct. 2021.

[7] Q. Yang, Y. Zhao, H. Huang, Z. Xiong, J. Kang, and Z. Zheng, "Fusing blockchain and AI with metaverse: A survey," *IEEE Open J. Comput. Soc.*, vol. 3, pp. 122–136, Jul. 2022.

[8] H. Duan, J. Li, S. Fan, Z. Lin, X. Wu, and W. Cai, "Metaverse for social good: A university campus prototype," in *Proc. ACM Int. Conf. Multimedia (MM)*, Oct. 2021, pp. 153–161.

[9] W. Y. B. Lim, Z. Xiong, D. Niyato, X. Cao, C. Miao, S. Sun, and Q. Yang, "Realizing the metaverse with edge intelligence: A match made in heaven," *IEEE Wirel. Commun.*, vol. 30, no. 4, pp. 64–71, Aug. 2023.

[10] V. T. Truong and L. B. Le, "Security for the metaverse: Blockchain and machine learning techniques for intrusion detection," *IEEE Netw.*, pp. 1–1, 2024, early access.

[11] Y. Zhou, H. Huang, S. Yuan, H. Zou, L. Xie, and J. Yang, "MetaFi++: Wifi-enabled transformer-based human pose estimation for metaverse avatar simulation," *IEEE Internet Things J.*, vol. 10, no. 16, pp. 14 128–14 136, Mar. 2023.

[12] Y. Lin, H. Du, D. Niyato, J. Nie, J. Zhang, Y. Cheng, and Z. Yang, "Blockchain-aided secure semantic communication for AI-generated content in metaverse," *IEEE Open J. Comput. Soc.*, vol. 4, pp. 72–83, Mar. 2023.

[13] M. Wang, T. Zhu, X. Zuo, M. Yang, S. Yu, and W. Zhou, "Differentially private crowdsourcing with the public and private blockchain," *IEEE Internet Things J.*, vol. 10, no. 10, pp. 8918–8930, May 2023.

[14] J. Wu, K. Lin, D. Lin, Z. Zheng, H. Huang, and Z. Zheng, "Financial crimes in Web3-empowered metaverse: Taxonomy, countermeasures, and opportunities," *IEEE Open J. Comput. Soc.*, vol. 4, pp. 37–49, Feb. 2023.

[15] V. T. Truong, L. B. Le, and D. Niyato, "Blockchain meets metaverse and digital asset management: A comprehensive survey," *IEEE Access*, vol. 11, pp. 26 258–26 288, Mar. 2023.

[16] H.-N. Dai, Z. Zheng, and Y. Zhang, "Blockchain for internet of things: A survey," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8076–8094, Oct. 2019.

[17] S. Zeng, Z. Li, H. Yu, Z. Zhang, L. Luo, B. Li, and D. Niyato, "HFedMS: Heterogeneous federated learning with memorable data semantics in industrial metaverse," *IEEE Trans. Cloud Comput.*, vol. 11, no. 3, pp. 3055–3069, Mar. 2023.

[18] Y. Wang, Z. Su, N. Zhang, R. Xing, D. Liu, T. H. Luan, and X. Shen, "A survey on metaverse: Fundamentals, security, and privacy," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 1, pp. 319–352, Sep. 2022.

[19] E. Daniel and F. Tschorsch, "IPFS and friends: A qualitative comparison of next generation peer-to-peer data networks," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 31–52, May 2022.

[20] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *Proc. USENIX Annu. Tech. Conf.*, Jun. 2014, pp. 305–319.

[21] S. Micali, M. Rabin, and S. Vadhan, "Verifiable random functions," in *Proc. IEEE Annu. Symp. Found. Comput. Sci.*, Oct. 1999, pp. 120–130.

[22] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yellick, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. ACM EuroSys Conf.*, Apr. 2018, pp. 1–15.

[23] L. D. Nguyen, I. Leyva-Mayorga, A. N. Lewis, and P. Popovski, "Modeling and analysis of data trading on blockchain-based market in IoT networks," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6487–6497, Apr. 2021.

[24] W. Dai, C. Dai, K.-K. R. Choo, C. Cui, D. Zou, and H. Jin, "SDTE: A secure blockchain-based data trading ecosystem," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 725–737, Jul. 2020.

[25] A. Dixit, A. Singh, Y. Rahulamathavan, and M. Rajarajan, "FAST DATA: A fair, secure, and trusted decentralized IIoT data marketplace enabled by blockchain," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 2934–2944, Feb. 2023.

[26] W. Xiong and L. Xiong, "Smart contract based data trading mode using blockchain and machine learning," *IEEE Access*, vol. 7, pp. 102 331–102 344, Jul. 2019.

[27] H. R. Hasan and K. Salah, "Proof of delivery of digital assets using blockchain and smart contracts," *IEEE Access*, vol. 6, pp. 65 439–65 448, Oct. 2018.

[28] A. Garba, A. D. Dwivedi, M. Kamal, G. Srivastava, M. Tariq, M. A. Hasan, and Z. Chen, "A digital rights management system based on a scalable blockchain," *Peer-to-Peer Netw. Appl.*, vol. 14, pp. 2665–2680, Sep. 2021.

[29] W.-S. Lee, A. John, H.-C. Hsu, and P.-A. Hsiung, "SPChain: A smart and private blockchain-enabled framework for combining gdpr-compliant digital assets management with ai models," *IEEE Access*, vol. 10, pp. 130 424–130 443, Dec. 2022.

[30] Y. Zhu, Y. Qin, Z. Zhou, X. Song, G. Liu, and W. C.-C. Chu, "Digital asset management with distributed permission over blockchain and attribute-based access control," in *Proc. IEEE Int. Conf. Serv. Comput. (SCC)*, Jul. 2018, pp. 193–200.

[31] M. Li, J. Weng, A. Yang, W. Lu, Y. Zhang, L. Hou, J.-N. Liu, Y. Xiang, and R. H. Deng, "CrowdBC: A blockchain-based decentralized framework for crowdsourcing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 6, pp. 1251–1266, Jun. 2019.

[32] S. Zhu, Z. Cai, H. Hu, Y. Li, and W. Li, "zkCrowd: A hybrid blockchain-based crowdsourcing platform," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4196–4205, Jun. 2020.

[33] M. Wang, T. Zhu, X. Zuo, M. Yang, S. Yu, and W. Zhou, "Differentially private crowdsourcing with the public and private blockchain," *IEEE Internet Things J.*, May 2023.

[34] Y. Lu, Q. Tang, and G. Wang, "Zebralancer: Private and anonymous crowdsourcing system atop open blockchain," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2018, pp. 853–865.

[35] X. Xu, Q. Liu, X. Zhang, J. Zhang, L. Qi, and W. Dou, "A blockchain-powered crowdsourcing method with privacy preservation in mobile environment," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 6, pp. 1407–1419, Dec. 2019.

[36] Y. Liu, Y. Zhang, S. Su, L. Zhang, X. Du, M. Guizani, and Z. Tian, "BlockSC: A blockchain empowered spatial crowdsourcing service in metaverse while preserving user location privacy," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 4, pp. 880–892, Apr. 2024.

[37] J. Liu, S. Dong, J. Wen, B. Tang, and Y. Yu, "TBSCrowd: A blockchain assisted privacy-preserving mobile crowdsourcing scheme," *IEEE Internet Things J.*, vol. 11, no. 11, pp. 19 344–19 354, Jun. 2024.

[38] H. Ma, S. Huang, J. Guo, K.-Y. Lam, and T. Yang, "Blockchain-based privacy-preserving federated learning for mobile crowdsourcing," *IEEE Internet Things J.*, vol. 11, no. 8, pp. 13 884–13 899, Apr. 2024.

[39] L. Lin, H. Duan, and W. Cai, "Web3DP: A crowdsourcing platform for 3d models based on web3 infrastructure," in *Proc. 14th ACM Multimedia Syst. Conf.*, Jun. 2023, pp. 397–402.

[40] S. Akiyama, Y. Matsuda, H. Suwa, and K. Yasumoto, "A method of crowdsourced task request optimization for 3D reconstruction in urban space," in *Proc. IEEE Glob. Conf. Consum. Electron. (GCCE)*, Oct. 2023, pp. 623–624.

[41] Y. Ye, H. Wang, C. H. Liu, Z. Dai, G. Li, G. Wang, and J. Tang, "QoI-aware mobile crowdsensing for metaverse by multi-agent deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, Mar. 2024.

[42] X. Liu, Y. Qin, W. Wu, C. Fu, Y. Lyu, F. Dong, and J. Luo, "B^2-bandit: Budgeted pricing with blocking constraints for metaverse crowdsensing under uncertainty," *IEEE J. Sel. Areas Commun.*, Mar. 2024.

[43] V. T. Truong and L. B. Le, "MetaCIDS: Privacy-preserving collaborative intrusion detection for metaverse based on blockchain and online federated learning," *IEEE Open J. Comput. Soc.*, vol. 4, pp. 253–266, Sep. 2023.

[44] Y. Wang, Z. Su, and M. Yan, "Social metaverse: Challenges and solutions," *IEEE Internet Things Mag.*, vol. 6, no. 3, pp. 144–150, Sep. 2023.

[45] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, and E. Dutkiewicz, "MetaChain: A novel blockchain-based framework for metaverse applications," in *Proc. IEEE Veh. Technol. Conf. (VTC)*, Jun. 2022, pp. 1–5.

[46] J. Kang, D. Ye, J. Nie, J. Xiao, X. Deng, S. Wang, Z. Xiong, R. Yu, and D. Niyato, "Blockchain-based federated learning for industrial metaverses: Incentive scheme with optimal AoI," in *Proc. IEEE Int. Conf. Blockchain*, Aug. 2022, pp. 71–78.

[47] L. Jiang, H. Zheng, H. Tian, S. Xie, and Y. Zhang, "Cooperative federated learning and model update verification in blockchain-empowered digital twin edge networks," *IEEE Internet Things J.*, vol. 9, no. 13, pp. 11 154–11 167, Jul. 2022.

[48] B. Ryskeldiev, Y. Ochiai, M. Cohen, and J. Herder, "Distributed metaverse: Creating decentralized blockchain-based model for peer-to-peer sharing of virtual spaces for mixed reality applications," in *Proc. Augmented Human Int. Conf.*, Feb. 2018, pp. 1–3.

[49] Y. Bai, Q. Hu, S.-H. Seo, K. Kang, and J. J. Lee, "Public participation consortium blockchain for smart city governance," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 2094–2108, Feb. 2022.

[50] M. Li, J. Weng, J.-N. Liu, X. Lin, and C. Obimbo, "Toward vehicular digital forensics from decentralized trust: An accountable, privacy-preserving, and secure realization," *IEEE Internet Things J.*, vol. 9, no. 9, pp. 7009–7024, May 2022.

[51] M. H. ur Rehman, K. Salah, E. Damiani, and D. Svetinovic, "Trust in blockchain cryptocurrency ecosystem," *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1196–1212, Nov. 2020.

[52] L. C. C. De Biase, P. C. Calcina-Ccori, G. Fedrecheski, G. M. Duarte, P. S. S. Rangel, and M. K. Zuffo, "Swarm economy: A model for transactions in a distributed and organic iot platform," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4561–4572, Jun. 2019.

[53] C. Liu, Y. Xiao, V. Javangula, Q. Hu, S. Wang, and X. Cheng, "NormaChain: A blockchain-based normalized autonomous transaction settlement system for IoT-based E-commerce," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4680–4693, Jun. 2019.

[54] M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, and M. Guizani, "Blockchain-assisted secure device authentication for cross-domain industrial IoT," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 942–954, May 2020.

[55] J. Chen, Z. Zhan, K. He, R. Du, D. Wang, and F. Liu, "XAuth: Efficient privacy-preserving cross-domain authentication," *IEEE Trans. Dependable Secur. Comput.*, vol. 19, no. 5, pp. 3301–3311, 01 Sept.-Oct. 2021.

[56] Y. Wang, Z. Su, N. Zhang, J. Chen, X. Sun, Z. Ye, and Z. Zhou, "SPDS: A secure and auditable private data sharing scheme for smart grid based on blockchain," *IEEE Trans. Ind. Inform.*, vol. 17, no. 11, pp. 7688–7699, Nov. 2021.

[57] V. T. Truong, H. D. Le, and L. B. Le, "Trust-free blockchain framework for AI-generated content trading and management in metaverse," *IEEE Access*, vol. 12, pp. 41 815–41 828, Mar. 2024.

[58] V. T. Truong and L. B. Le, "A blockchain-based framework for secure digital asset management," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2023, pp. 1911–1916.

[59] Y. Fu, C. Li, F. R. Yu, T. H. Luan, P. Zhao, and S. Liu, "A survey of blockchain and intelligent networking for the metaverse," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 3587–3610, Nov. 2023.

[60] V. T. Truong, D. N. M. Hoang, and L. B. Le, "BFLMeta: Blockchain-empowered metaverse with byzantine-robust federated learning," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2023, pp. 5537–5542.

[61] Wikipedia contributors, "Merkle tree," 2024, accessed: 2024-05-29. [Online]. Available: https://en.wikipedia.org/wiki/Merkle_tree

[62] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *Int. J. Web Grid Serv.*, vol. 14, no. 4, pp. 352–375, Oct. 2018.

[63] A. M. Antonopoulos and D. A. Harding, *Mastering Bitcoin.* O'Reilly Media, Inc., 2023.

[64] G. Wood *et al.*, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.

[65] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, Oct. 2016, pp. 3–16.

[66] Y. Wang, H. Peng, Z. Su, T. H. Luan, A. Benslimane, and Y. Wu, "A platform-free proof of federated learning consensus mechanism for sustainable blockchains," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 12, pp. 3305–3324, Dec. 2022.

[67] F. Saleh, "Blockchain without waste: Proof-of-stake," *Rev. Financ. Stud.*, vol. 34, no. 3, pp. 1156–1190, Mar. 2021.

[68] M. Castro, B. Liskov *et al.*, "Practical byzantine fault tolerance," in *Proc. Symp. Oper. Syst. Des. Implement. (OSDI)*, vol. 99, Feb. 1999, pp. 173–186.

[69] Z. Su, Y. Wang, Q. Xu, M. Fei, Y.-C. Tian, and N. Zhang, "A secure charging scheme for electric vehicles with smart communities in energy blockchain," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4601–4613, Jun. 2018.

[70] S. De Angelis, L. Aniello, R. Baldoni, F. Lombardi, A. Margheri, V. Sassone *et al.*, "PBFT vs proof-of-authority: Applying the CAP theorem to permissioned blockchain," in *CEUR Workshop Proc.*, vol. 2058. CEUR-WS, Feb. 2018.

[71] Q. Zhou, H. Huang, Z. Zheng, and J. Bian, "Solutions to scalability of blockchain: A survey," *IEEE Access*, vol. 8, pp. 16 440–16 455, Jan. 2020.

[72] H. Dang, T. T. A. Dinh, D. Loghin, E.-C. Chang, Q. Lin, and B. C. Ooi, "Towards scaling blockchain systems via sharding," in *Proc. Int. Conf. Manag. Data (SIGMOD)*, Jun. 2019, pp. 123–140.

[73] H. Duy Le, V. Tuan Truong, and L. B. Le, "Blockchain-empowered metaverse: Decentralized crowdsourcing and marketplace for trading machine learning data and models," *IEEE Access*, vol. 12, pp. 68 556–68 572, May 2024.

[74] X. Yang, G. Gong, and X. Wang, "Virtual sound modeling and real-time rendering in aircraft simulator," *Acta Aeronaut. Astronaut. Sin.*, vol. 30, no. 7, pp. 1305–1310, Jul. 2009.

[75] ——, "Real-time visual system of night-flying across the sea," *J. Beijing Univ. Aeronaut. Astronaut.*, vol. 34, no. 1, pp. 35–38, Jan. 2008.

[76] Y. Xinying, G. Guanghong, Z. Bo, and H. Zhanpeng, "Virtual modeling and rendering technologies of high-seas environment," *J. Beijing Univ. Aeronaut. Astronaut.*, vol. 35, no. 4, pp. 493–496, Apr. 2009.

[77] H. Ning, H. Wang, Y. Lin, W. Wang, S. Dhelim, F. Farha, J. Ding, and M. Daneshmand, "A survey on the metaverse: The state-of-the-art, technologies, applications, and challenges," *IEEE Internet Things J.*, vol. 10, no. 16, pp. 14 671–14 688, Aug. 2023.

[78] M. Xu, W. C. Ng, W. Y. B. Lim, J. Kang, Z. Xiong, D. Niyato, Q. Yang, X. Shen, and C. Miao, "A full dive into realizing the edge-enabled metaverse: Visions, enabling technologies, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 1, pp. 656–700, Mar. 2023.

[79] X. Sun, F. R. Yu, P. Zhang, Z. Sun, W. Xie, and X. Peng, "A survey on zero-knowledge proof in blockchain," *IEEE Netw.*, vol. 35, no. 4, pp. 198–205, Jul. 2021.

[80] J. R. Douceur, "The sybil attack," in *Proc. Int. Workshop Peer-to-Peer Syst.* Springer, Oct. 2002, pp. 251–260.

[81] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "SoK: Research perspectives and challenges for bitcoin and cryptocurrencies," in *Proc. IEEE Symp. Secur. Privacy*, Jul. 2015, pp. 104–121.

[82] A. Bessani, J. Sousa, and E. E. Alchieri, "State machine replication for the masses with BFT-SMART," in *Proc. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, Jun. 2014, pp. 355–362.

[83] D. Huang, X. Ma, and S. Zhang, "Performance analysis of the Raft consensus algorithm for private blockchains," *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 50, no. 1, pp. 172–181, Jan. 2020.