UFR de mathématique et d'informatique
Université de Strasbourg

MASTER 2 CALCUL SCIENTIFIQUE ET MATHÉMATIQUES DE L'INOVATION

Mémoire de stage présenté par

Thomas LELIÈVRE
thomas.lelievre@etu.unistra.fr

REDUCING FORK PROBABILITY IN PROOF-OF-INTERACTION BLOCKCHAIN

28 août 2024

Stage encadré par

Quentin BRAMAS
bramas@unistra.fr

Au sein du laboratoire

ICUBE

Table des matières

Chapitre 1

Présentation

J'ai effectuer mon stage au sein du laboratoire ICube du 15 février au 31 août. Ce laboratoire est spécialisé dans la recherche dans différents domaines de l'ingénieurie. L'équipe Résaux (CNRS) au sein de laquelle je me trouvais effectue des recherches sur les sytèmes d'exploitation, les réseaux informatiques et l'algorithmique distribuée. J'ai travaillé durant mon stage sur les blockchains que j'ai étudié sous l'angle des systèmes distribuées.

Mon stage s'inscrit dans le cadre du projet ANR *Base Block*, porté par Quentin Brama qui vise à concevoir des blockchains moins consomatrices d'énergie et nécessitant moins d'espace stockage [1].

Le but de mon stage était de modélisé et d'optimiser les probalités de fork dans une blockchain utilisant la preuve d'intéraction. Un algorithme de consensus, la preuve d'intéraction (POI) [ABN21], dévelopé par Jean-Philippe Abbeg, Quentin Bramas et Thomas Noël dans le cadre de la thèse de Jean-Philippe Abbeg conduite au sein de l'équipe.

Mon travail a consisté en une étude théorique des forks dans les blockchains qui a donné lieu à un article coécrit avec Quentin Bramas. J'ai également posé les bases d'un simlateur de blockchain permettant de simuler le comportement des forks dans une blockchain utilisant la POI.

Dans ce rapport, nous allons détailler les différentes étapes de mon travail.

D'abord nous définirons les concepts de Blockchains, d'algorithme de concensus et de fork.

Puis, nous motiverons le fait de s'intéresser aux forks dans les blockchains avant de d'explorer plusieurs models de propagation.

Enfin, nous détaillerons la conception du simulateur.

1. Une courte vidéo de présentation du projet est disponible à l'adresse suivante : https://savoirs.unistra.fr/innovation/base-bloc-une-solution-pour-des-blockchains-plus-vertueuses

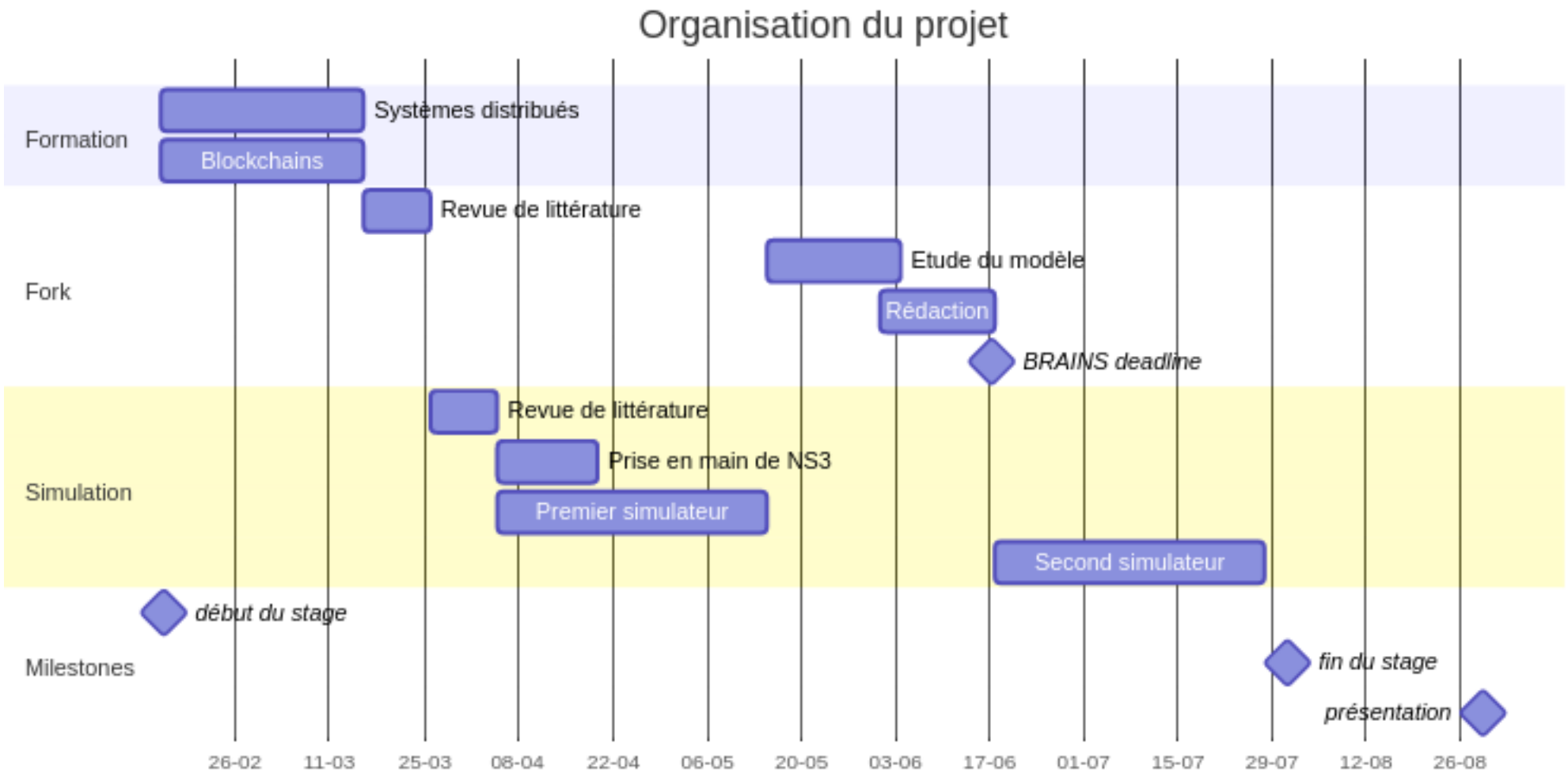


FIGURE 1.1 – Gantt

Chapitre 2

Blockchain

De 2017 à 2022, la *blockchain*, et en particulier son application phare le *Bitcoin*, ont connu un engoument médiatique considérable, comparable à celui que connait aujourd'hui *ChatGPT* (2.1). Cette attention médiatique c'est essentiellement portée sur les cryptomonaies, des monaies numériques indépendantes d'une entité centrale (banque, État) pour garentir leur valeur et à laquelle elles substituent des moyens cryptographiques : les blockchains. Si les cryptomonaies constituent la principale application des blockchains, ces dernières ont beaucoup évolué depuis la parution du *Bitcoin* et leurs applications ne se limite désormais plus aux seules cryptomonaies. Avant d'expliquer le fonctionnement des algorithmes de consensus, intéressons à leur évolution qui permet d'éclairer les enjeux techniques quelles soulèvent.

2.1 Perspective Historique

2.1.1 Le *Bitcoin* au origine des blockchains

Si l'idée d'une cryptomonaies remonte au moins à 1982 [Cha83], c'est en 2007 que Satoshi Nakamoto rend public le *Bitcoin* qui sera à la fois la première blockchain et la première cryptomonaie. Dans son livre blanc [Nak07], il propose le *Bitcoin* comme une alternative aux établissements bancaires qui imposent alors des frais de transaction élevés randant le payement en ligne impossible pour des transactions au montant peu élevé.

Les pièces de cette cryptomonaie sont définies par une chaine de possession. Lors d'une transaction, le nouveau propriétaire applique une fonction de hachage à la concaténation de la transactions précédente.

Ces transactions sont regroupées dans des blocs qui sont lié entre eux car ils contiennent le hash du bloc précédent, formant ainsi la blockchain. Ceci permet de garantir que les transactions sont ordonnées ce qui permet de résoudre le problème du double dépense.

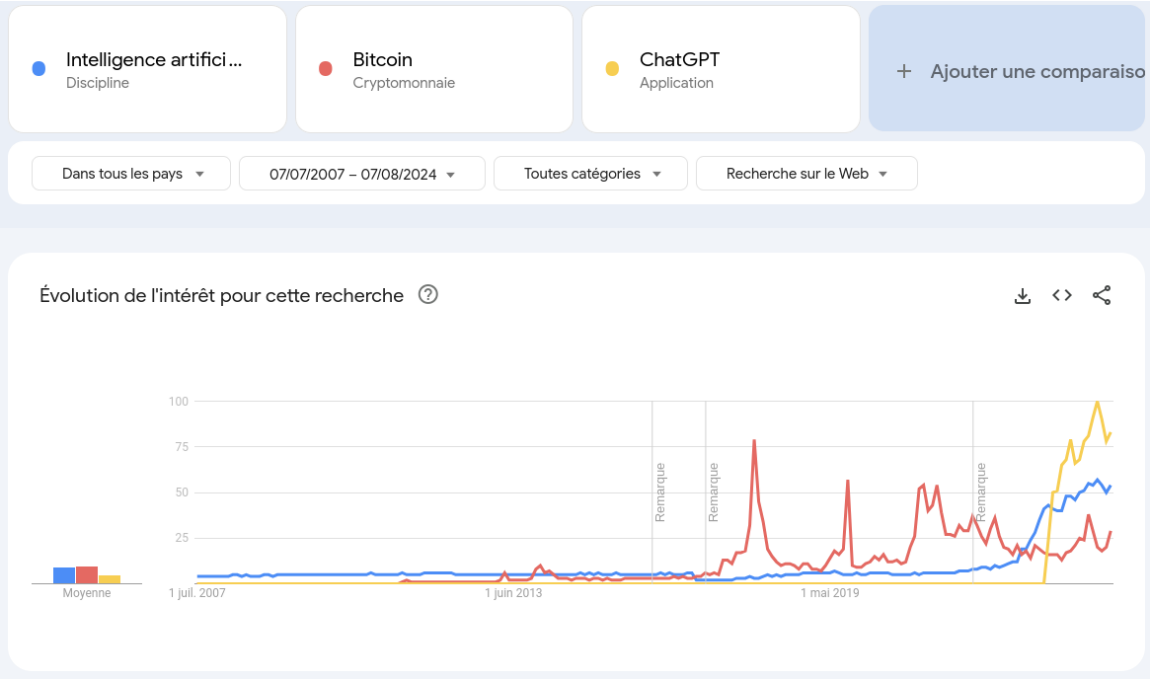


FIGURE 2.1 – petit texte explicatif Source: Google Trends

la preuve de travail

La preuve de travail est un mécanisme qui permet de garantir la sécurité de la blockchain. Et en particulier de la protéger contre les attaques Sybil.

Une attaque Sybil est une attaque dans laquelle un attaquant crée un grand nombre de blocs pour prendre le contrôle de la blockchain.

La preuve de travail vise à rendre cette attaque coûteuse en demandant aux mineurs d'ajuster le nonce, une valeur contenue dans l'en-tête du bloc, de sorte que le hash soit inférieur à une certaine valeur. Cette valeur est ajustée de sorte que le temps de minage moyen soit de 10 minutes. Tous les 2016 blocs, les mineurs calcul le temps écoulé pour miner ces blocs et augmente la difficulté si ce temps est inférieur à 10 minutes en proportionellement à l'écart.

La preuve de travail est un mécanisme d'élection qui permet de déterminer quel mineur peut ajouter un bloc à la chaine. Les mineurs sont incités à valider les transactions en recevant une récompense quand ils trouvent un bloc.

L'arbre de Merkle

Un arbres de Merkle est une structure de données qui permet de vérifier l'intégrité de la chaine de blocs tout en permettant de ne pas stocker l'ensemble des transactions. Ce qui est utile d'une part pour réduire la taille de la blockchain et d'autre part pour permettre à des client léger (ne stockant pas l'ensemble de la blockchain) de vérifier l'intégrité des transactions.

Chaque feuille de l'arbre est un hachage d'une transactions. Chaque nœud interne est

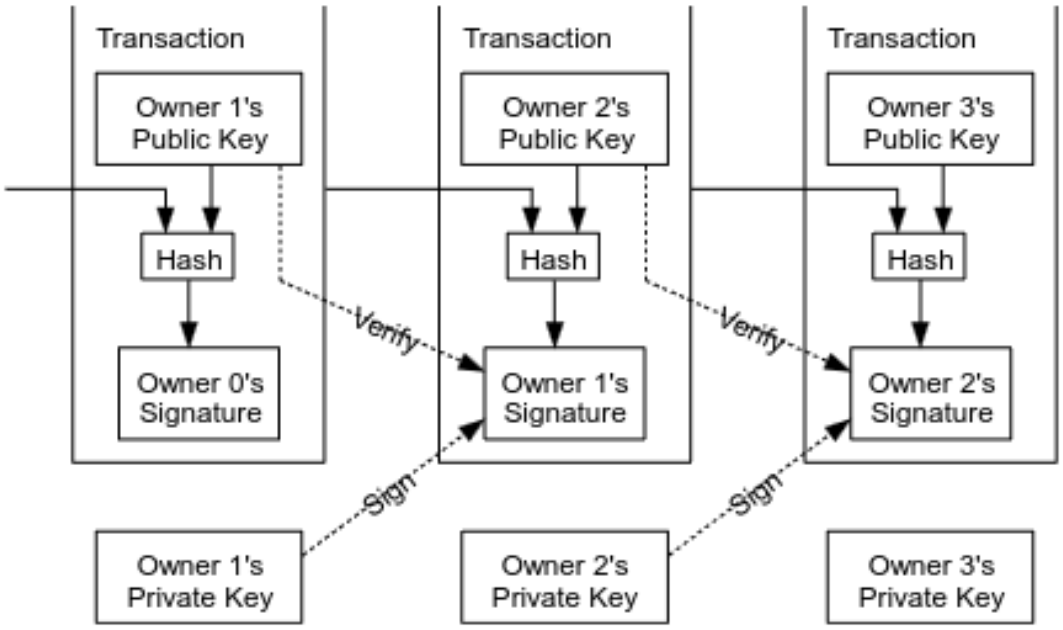


FIGURE 2.2 – Transactions. Source : [Nak07]

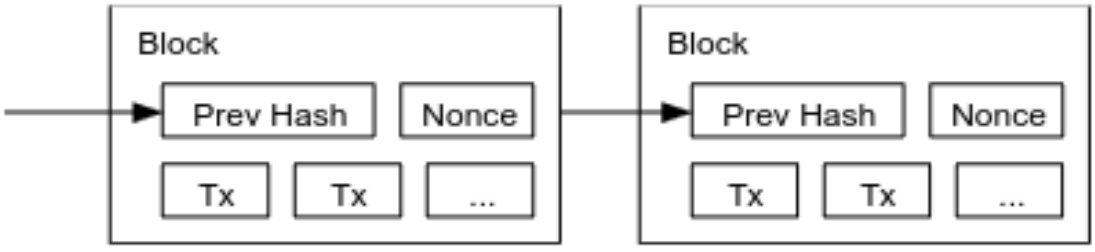


FIGURE 2.3 – Preuve de Travail. Source : [Nak07]

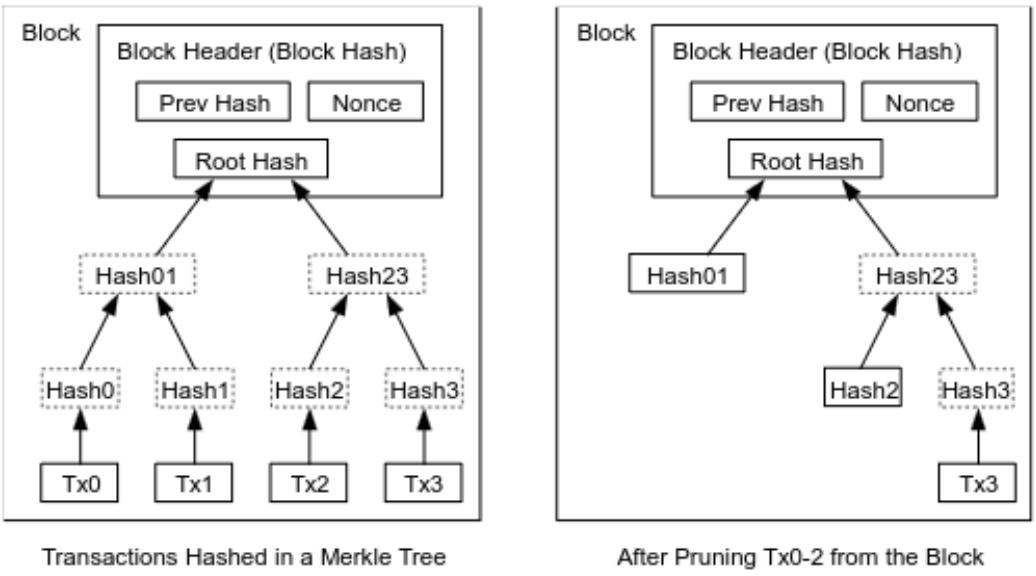


FIGURE 2.4 – Arbre de Merkle. Source : [Nak07]

un hachage de la concaténation des hachages de ses deux nœuds enfants. La racine de l'arbre est un hachage unique qui représente l'intégrité de toutes les données sous-jacentes.

C'est un parfait exemple de scalabilité on-chain.

Le *Bitcoin Script*

Non mentionné dans le livre blanc mais présent dès 2009 dans la version 0.1.5 du client *Bitcoin* [Nak09], le *Bitcoin Script* est un language de script composé d'instructions (Optcodes) qui permettent entre autre l'exécution conditionnelle des transactions. Il peut être vu comme un ancêtre des *smart contracts*.

2.1.2 *The Block Size War*

Bien que les premières versions du *Bitcoin* portaient déjà en germe les principaux concepts des blockchains modernes, elles n'ont pas réussit à créer une cryptomonaie pouvant concurrancer les établissements bancaires en matière de frais de transactions.

Dès 2010, le Bitcoin éprouve des difficultés à s'adapter à l'augmentation des transactions sur le réseau. La cause est due à des choix d'implémentation. D'un côté, pour faciliter le consensus et limiter les attaques Sibil, le temps interbloque a été fixée à en moyenne 10 minutes. De plus, la limite a été fixée à 1 Mbps. Face à un nombre un nombre de transactions journalières toujours croissants, le coût des frais de transaction a explosé.

Dans la communauté, deux blocs se font face. D'un côté des petits bloqueurs qui tiennent à garder le block à 1 Mo. Pour eux l'important est de garantir la décentralisation du Bitcoin en limitant la taille des blocs. Ils proposent pour améliorer la scalabilité des

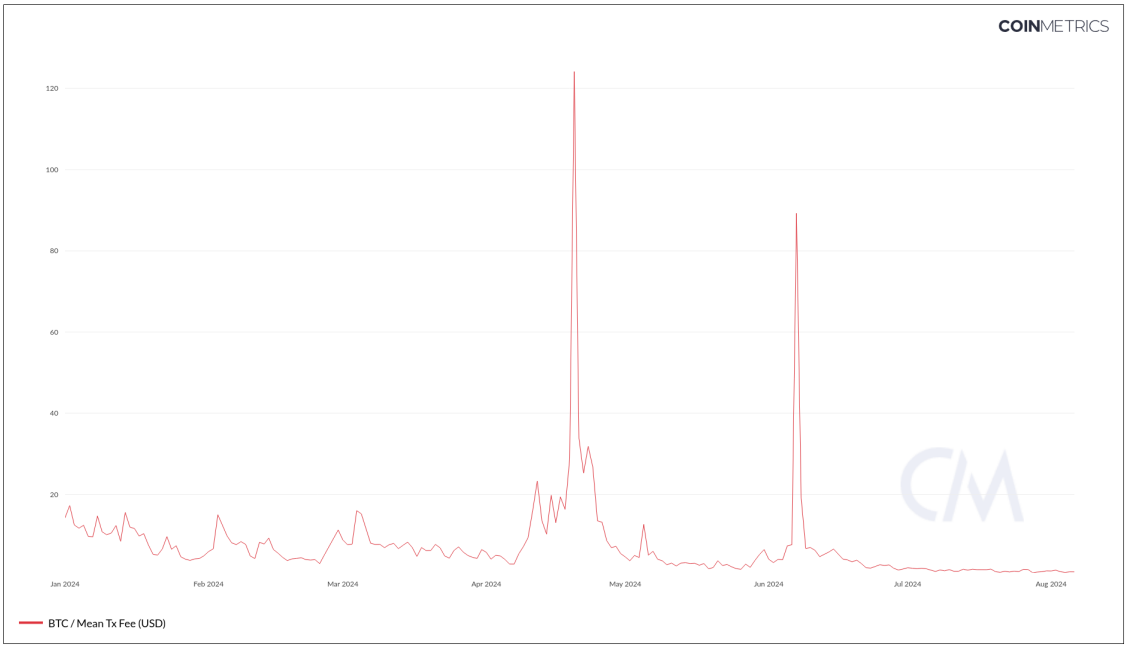


FIGURE 2.5 – Source : https://charts.coinmetrics.io/crypto-data/

solutions de second layer.

Les layer 2 sont des protocoles qui permettent de réaliser des transactions en dehors de la blockchain principale. Elle permettent de grouper un grands nombre de transactions régulière en une seule transaction sur la blockchain principale.

Et des gros bloqueurs qui mettent en avant, l'opérabilité du réseau et qui par conséquent demande à augmenter la taille des blocs pour s'adapter à la quantité des transactions et garder un coût de transaction bas.

L'introduction de SegWit, un protocole du Bitcoin préparant la création du Lighting Network, le layer 2 du Bitcoin, a été le déclencheur d'un fork qui a conduit à la création de Bitcoin Cash en 2017.

Cet événement loins d'être anecdotique permet de mettre en lumière une des principales limitation des blockchain, la scalabilité. Il permet aussi de montrer l'importance des forks de protocole (voire chapitre 3) dans la gouvernance de la blockchain. Enfin, il introduit la notion de layer 2 dans les blockchains qui sont un moyen de contourner les limitations de la blockchain. Le problème de la scalabilité reste aujourd'hui encore un des principaux enjeux de la recherche sur les blockchains. Cette annéee Bitcoin a encore démontré cette faiblesse. En effet, alors que la moyenne des frais de transaction sur l'année était de 7,32 (9 août 2023 et le 8 août 2024) dollars pour transaction, les frais de transaction en moyenne journalière ont atteint des records, 124,17 $ le 20 avril 2024 et 89,25$ le 7 juin 2024, voire figure 2,5.

2.1.3 *Ethereum* et *smart contracts*

Comme nous l'avons evoqué, le Bitcoin Script offrait certaines fonctionnalités dans la création de transactions. Avec la création d'*Ethereum*, on assiste à la naissance des *smart contracts*. Ethereum a, en effet, développé une couche d'exécution complète, permettant non seulement de gérer des transactions, mais aussi de créer des programmes autonomes. Ces *smart contracts* sont des morceaux de code déployés sur la blockchain qui peuvent gérer automatiquement les accords, contrats, sans nécessiter d'intervention humaine. Ils ont ainsi ouvert la voie à de nouvelles applications décentralisées dans divers domaines, allant des finances décentralisées, aux systèmes de vote, en passant par les jeux et la gestion des identités.

Cependant, cette innovation n'est pas sans risques. En 2016, le célèbre piratage de The DAO (Decentralized Autonomous Organization), un projet basé sur un contrat intelligent Ethereum, a révélé la vulnérabilité des smart contracts. The DAO avait pour but de créer un fonds d'investissement décentralisé, géré automatiquement par des smart contracts. Mais plusieurs failles de sécurité dans le code du smart contract ont été exploitées. Ce qui a permis aux attaquants de détourner environ 50 millions de dollars en Ether.

Cet incident a mis en lumière les risques inhérents au développement de smart contract. Il a provoqué un débat au sein de la communauté Ethereum sur la manière de réagir, menant à un hard fork controversé de la blockchain Ethereum. Ce fork a conduit à la création de deux versions distinctes de la blockchain : Ethereum (qui a annulé les transactions du piratage) et Ethereum Classic (qui a maintenu la chaîne d'origine sans modification).

2.1.4 The Merge

Avec The Merge, Ethereum est passé d'un consensus basé sur la preuve de travail (PoW) à la preuve d'enjeu (PoS).

Le passage à la preuve d'enjeu permet de réduire considérablement la consommation énergétique de la chaine. Contrairement à PoW, où les mineurs doivent calculer un grand nombre de hash pour valider un bloc, la PoS sélectionne aléatoirement des validateurs en fonction de la quantité d'Ether qu'ils possèdent et qu'ils sont prêts à "staker" (mettre en jeu) comme garantie. Cela élimine le besoin de puissants équipements informatiques pour miner, rendant les nœuds du réseau beaucoup moins coûteux à faire fonctionner.

Selon les estimations, cette transition a réduit la consommation d'énergie d'Ethereum de 99,95 %.

Le passage de la preuve de travail à la preuve de enjeu montre l'impacte de l'algorithme de consensus sur la consomation énergétique de la blockchain.

Par cette approche historique nous avons pu explorer les principaux concepts des blockchains. Attardons maintenant plus précisement sur la notions d'algorithme de consensus.

2.2 Le consensus

La couche consensus est la couche qui permet aux noeuds de se mettre d'accord sur l'état de la blockchain. Historiquement pensé en dehors du cadre de l'algorithmique distribué, il repose néanmoins sur des concepts développés dans ce domaine.

Le problème du consensus est un problème classique de l'algorithmique distribuée.

Definition 2.2.1 (Le problème du consensus)**.** On considère un ensemble de processus qui communiquent par envoi de messages. Chaque processus propose une valeur et doient vérifier les trois propriétés suivantes :

- **Accord (Agreement)** : Tous les processus non-défaillant doivent s'accorder sur une même valeurs.
- **Validité (Validity)** : Toute valeur renvoyer doit avoir été proposée par un des processus.
- **Finalisation (Termination)** : Tous les processus non-défaillant doivent renvoyer une valeurs en un temps fini.

Dans le cas d'une communication assynchrone avec panne, le consensus est impossible comme l'ont montré Fisher, Lynch et Paterson [FLP85].

Par contre le consensus est possible dans le cas d'une communication synchrone avec panne.

La couche de consensus de la blockchain doit satisfaire plusieurs propriétés :

1. Accord : choix des forks
2. Sureté (safety) : deux blocs ne peuvent pas entrer en conflit (double dépense, ...)
3. Résistance aux attaques Sybil : il est improbable (ou très coûteux) qu'un attaquant puisse créer une chaine plus grande que la chaine légitime
4. Liveness : La chaine doit pouvoir continuer de grandir

Intérerssons nous maintenant à un algorithme de consensus particulier, la preuve d'intéraction (PoI).

2.2.1 Proof of Interaction

La preuve d'intéraction développé par Abbeg et al. [ABN21] est un algorithme de consensus qui repose sur les interactions entre les noeuds du réseau pour créer des blocs.

Il suppose que chaque noeuds du réseau connait l'ensemble des autres noeuds et qu'il peuvent communiquer. Cette hypothèse correspond à un réseau de blockchain dit "permissionned" qui est utilisé de le cas de blockchain d'entreprise.

Dans ce modèle chaque noeud, pour créer un bloc, doit suivre les étapes suivantes :

1. Le noeud sélectionne aléatoirement un sous-ensemble de noeuds.
2. Il détermine en suite le noeud la longueur de son tour est déterminé par la difficulté

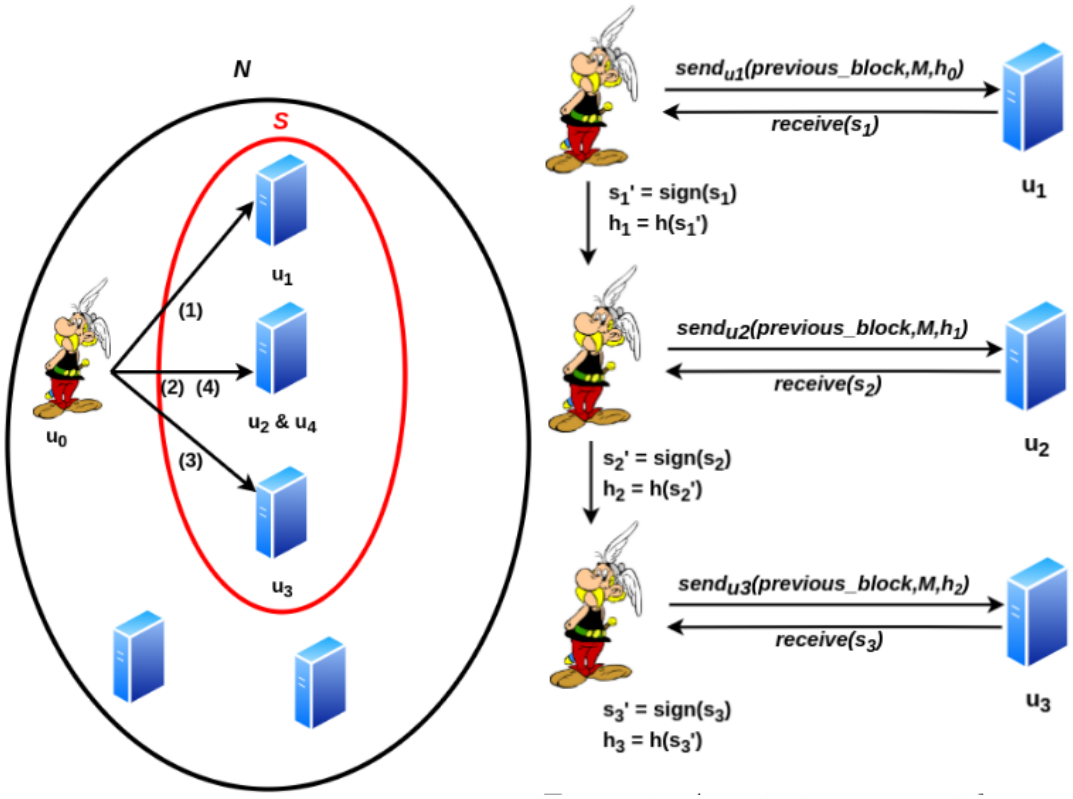


FIGURE 2.6 – Illustration de la preuve d'intéraction. Source : [ABN21]

3. Il détermine aléatoirement l'ordre dans lequel il va interagir avec les autres noeuds du sous-ensemble.
4. Il envoie alors le bloc au premier noeud de la liste qui le signe et le renvoie. Le noeud envoie alors le bloc et le hash de la signature au deuxième noeud de la liste et ainsi de suite.
5. Le premier noeud qui finit son tour gagne le droit de créer un bloc.

Pour garentir que l'ordre des interactions à bien été choisi aléatoirement, chaque noeud doit initialiser sont générateur de nombre aléatoire avec une graine correspondant au hash du dernier bloc.

Chapitre 3

Fork

Les *forks* sont des événements qui surviennent lorsqu'une blockchain se divise en deux branches distinctes. Ces événements sont généralement causés par des divergences dans les règles de consensus de la blockchain. Il existe plusieurs types de fork, chacun ayant des conséquences différentes sur la blockchain. Les forks sont important, car ils peuvent avoir un impact significatif sur la sécurité et la fiabilité d'une blockchain. Dans ce chapitre, nous allons discuter des différents types de fork, des modèles théoriques existants pour les étudier, et des méthodologies employées pour les analyser.

3.1 Typologies des forks

Les origines des *forks* sont diverses. Dans [Sch20], Schär propose quatre types de fork, illustrés dans la figure 3.1 : les forks de protocole et les forks de processus, chacun d'entre eux pouvant être intentionnel ou non. Nous présenterons ici les forks de protocol intentionnels et les forks de processus inintentionnels.

	Process-based $(A = B = S)$	**Protocol-based** $(A \neq B)$
Unintentional	Probabilistic Block Race	Client Incompatibility • Soft Fork • Hard Fork • Forced Fork
Deliberate	Block Withholding & Forced Block Race	Rule Change • Soft Fork • Hard Fork • Forced Fork

FIGURE 3.1 – Les quatre types de fork. Source : [Sch20]

3.1.1 Forks de protocole

Les *forks* de protocole involontaire surviennent généralement à la suite d'un bug dans le code des clients de minages ou de validation. Un exemple de ce type de fork est le fork du *Bitcoin* de 2013 décrit par Gavin Anderson, le cocréateur du *Bitcoin* dans la *BIP 50* [And13]. En mars 2013, suite à la parution de la version 0.8 du client *Bitcoin*, les noeuds qui avaient mis à jour leur client on commencé à rejetter les noeuds sur la base de leur taille et plus sur la base de la règle implicite du nombre de locks necessaire pour ajouter le bloc dans la base de donnée *Berkeley DB*. Cette modification bien à entrainer un rejet des nouveaux blocs par les noeuds n'ayant pas mit à jour leur client car des blocs pouvais necessiter des nombres de blocks dépassant la lime du nombre de locks. Ces rejets ont entrainé un fork de la blockchain qui bien que limité par la réponse rapide de la communautre n'a été entièrement résolu qu'en août 2013 et a permit une attaque de *double dépense*.

Ces forks étant rare et imprévisible, nous nous intéresseront plutôt aux *forks* de protocole volontaire qui surviennent à la suite d'une modification du protocole de consensus de la blockchain. Ces modifications sont importantes à plusieurs égars. D'une part, elles participent de la gouvernance de la blockchain comme l'a mis en lumière la guerre des blocs. Cette forme de gouvernance pensée des l'article fondateur de Sakamoto [Nak07], consacre une gestion démocratique de la blockchain donnant à chaque participant une voix par processeur. D'autre part, elles permettent de corriger des failles de sécurité comme ce fut le cas lors de l'attaque de *The DAO* sur *Ethereum*. Ces modifications de protocole peuvent être de deux types : les *hard forks*, les *soft forks*

Hard Forks

Un hard fork est une mise à jour majeure du protocole qui entraîne une divergence permanente de la chaîne en deux versions incompatibles.

Si une partie de la communauté continue d'utiliser l'ancienne version du protocol, cela entraîne une division de la blockchain en deux branches distinctes.

Après un hard fork, les nœuds qui continuent d'utiliser l'ancienne version du logiciel ne peuvent plus valider les blocs ou les transactions. Ce fut le cas après la guerre des blocs qui a conduit à la création de Bitcoin Cash ou après l'attaque de The DAO d'Ethereum qui a conduit à la création d'Ethereum Classic.

Les deux branches étant issue d'une même chaîne, les utilisateurs détenant des jetons sur la chaîne originale avant le hard fork se retrouvent généralement avec des jetons sur les deux chaînes après la scission, correspondant à leur solde avant le fork.

Les hard forks peuvent être planifiés pour apporter des améliorations, corriger des bugs ou introduire de nouvelles fonctionnalités, mais ils peuvent aussi être causés par des désaccords dans la communauté concernant l'évolution de la blockchain.

Soft Forks

Contrairement aux hard forks, les soft forks sont des mises à jour du protocole qui résulte en une division temporaire de la chaîne. La mise à jour est rétrocompatible et

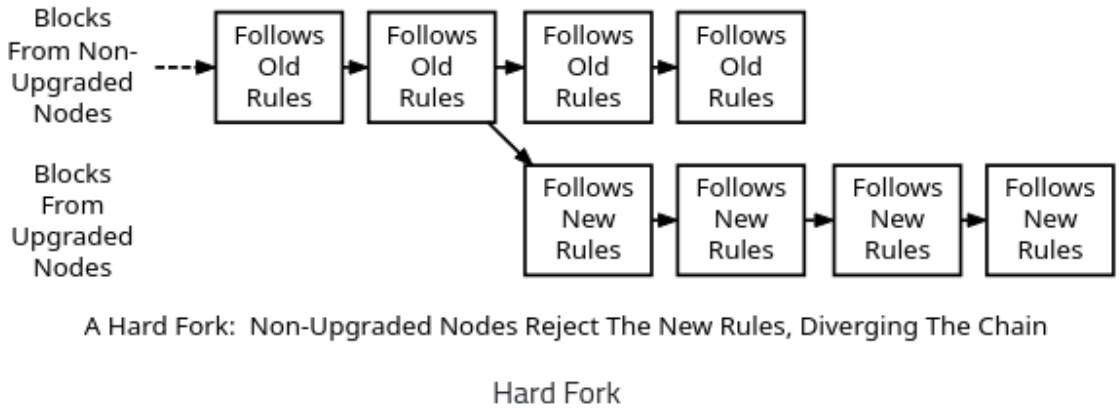


FIGURE 3.2 – *Hard Fork*. Source : developer.bitcoin.org

la chaîne se réunifie une fois que la majorité des mineurs a adopté la nouvelle version.

Le soft fork modifie les règles de consensus de manière plus stricte, limitant les types de transactions ou de blocs acceptés par les nœuds qui ont adopté la mise à jour. Les nœuds non mis à jour continuent d'accepter les blocs et transactions conformes à l'ancienne et à la nouvelle version des règles.

Un exemple de soft fork est la mise à jour est l'adoption de *Segregated Witness* par le réseau Bitcoin en 2017.

3.1.2 Forks de processus

Un fork de processus intervient lorsque deux mineurs trouvent un bloc en même temps. Les deux blocs sont alors propagés dans deux parties distinctes du réseau et les mineurs. Les nœuds reçoivent, alors, ces blocs et les ajoutent à leur copie locale de la blockchain. Le réseau résout ensuite cette situation en continuant à miner. Les blocs s'ajoutent à l'une ou l'autre des branches, et la branche la plus longue est considérée comme valide. On remarque que les forks de processus sont couteux car la branche n'ayant pas été retenue aura été miné en vain.

L'impact de la vitesse de propagation sur la probabilité de fork a été étudié par plusieurs auteurs. Dans [Cro+16], Croman et al. ont montré que le délai de propagation était un facteur important dans la probabilité de fork. Ce résultat théorique, vient confirmer la corrélation observer entre vitesse de transmission et taux de fork [DW13].

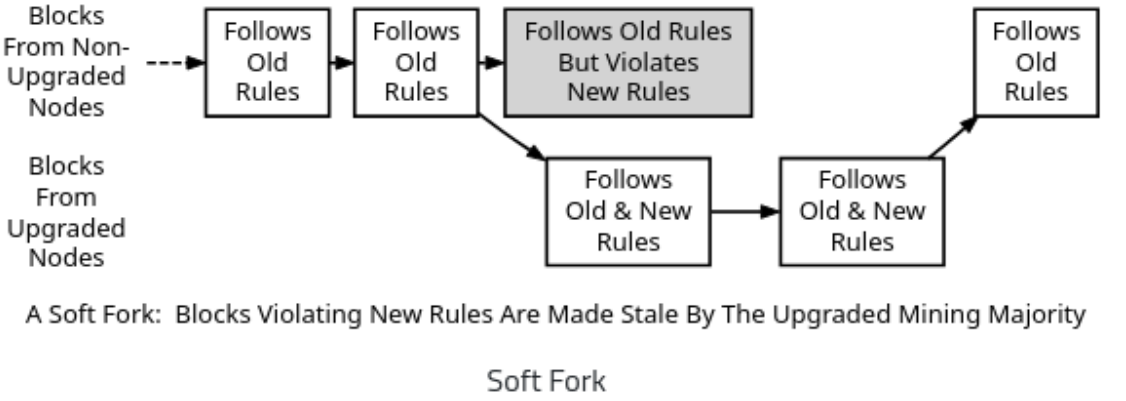
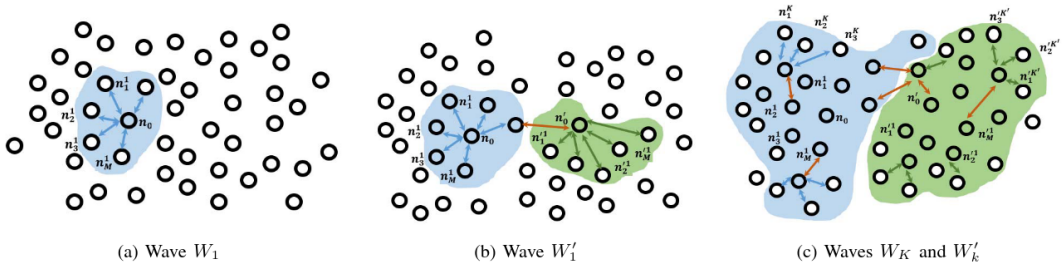


FIGURE 3.3 – *Hard Fork*. Source : developer.bitcoin.org



(a) Wave W_1 (b) Wave W_1' (c) Waves W_K and W_k'

FIGURE 3.4 – Fork de processus créé par la propagation de deux blocs différents. Source : [DW13]

3.2 *The impact of block mining time distribution on the probability of forks.*

Une des contribution de mon stage a été la rédaction d'un article coécrit avec Quentin Bramas, intitulé *The impact of block mining time distribution on the probability of forks* [LB24]. Cet article a été accepté à la conférence BRAINS et sera présenté en octobre.

Dans cet article nous nous somme intéresser à chercher une distribution des temps de minage des blocs qui minimise l'écart en le premier et le second bloc miné.

Soit N le nombre de mineurs d'une blockchain. On considère la suite de variable aléatoire $(X_i)_{\{1,\dots,N\}}$, ou X_i mesure le temps de minage du i-ème bloc. On suppose que les X_i sont des variables aléatoires discretes. En ordonnant les X_i par ordre croissant on obtient les statistiques d'ordres $X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(N)}$. On cherche la distribution de probabilité qui maximise l'écart entre le premier et le second bloc.

$$\text{argmax}_{p\in[0,1]^{T+1}} E(X_{(2)}) - E(X_{(1)}) \tag{3.1}$$

$$\text{tel que} \quad \sum_{i=0}^{T} p_i = 1 \tag{3.2}$$

$$E(X_{(1)}) = \sum_{k=0}^{T-1} \left(1 - \sum_{i=0}^{k} p_i\right)^n = \mathfrak{m} \tag{3.3}$$

Nous avons ensuite montré la proposition suivante.

Proposition 3.2.1. *L'unique solution du problème de maximisation 3.1 est la loi de probabilité définit par*

$$\mathbb{P}(X = 0) = 1 - \left(\frac{\mathfrak{m}}{T}\right)^{1/n}$$
$$\mathbb{P}(X = T) = \left(\frac{\mathfrak{m}}{T}\right)^{1/n}$$

Nous avons ensuite comparé cette distribution la distribution géométrique utilisée par le bitcoin pour montrer que la distribution minimisant l'écarts de minage permettait de réduire l'apparition de fork.

De cette étude, nous avons conclu que la distribution des temps de minage avait un impact sur la probabilité de fork.

Nous avons vu dans ce chapitre les différents types de forks et leur importance. La plupart des travaux existants utilisent pour appuyer leur modèles théoriques de les comparer avec des simulations. C'est pourquoi nous allons dans le chapitre suivant nous nous intéresseront au simulateurs existants ainsi qu'à la démarche qui nous a mené à développer notre propre simulateur.

Chapitre 4

Simulating Blockchain

4.1 Simulateurs existants

Bitcoin-Simulator [Ger+16] est l'un des premiers simulateurs de blockchain. Il a été développé par Gervais et al. pour étudier la sécurité de Bitcoin. Le simulateur, est basé sur NS3, un simulateur de réseau très utilisé dans la communauté scientifique. Le bitcoin-simulator permet de simuler différent type de comportement des mineurs. Cependant, il ne permet pas de simler d'autre type de blockchain que le Bitcoin et bien qu'il utiliser NS3, il utilise une topologie simplifié de réseau.

BlockSim [AV20], un simlateur de blockchain qui se veut générique, extensible et simple. BlockSim utilise un modèle de base organisé en trois couches d'abstraction : réseau, consensus et incitations. La couche réseau est basé ne prend en compte que deux paramètres le nombre de noeuds et le délai de transmission.

BlockPerf [Pol+21] est un simulateur hybride qui étend BlockSim en y ajoutant améliorant la modélisation des couches consensus et incitations. Pour la couche réseau, BlockPerf fait le choix de en pas la simuler mais de déployer le simulateur sur des machines réelles qui sont connecté sur un réseau de pair à pair semblable à celui d'un réseau de blockchain. Il implémente même un protocole de découverte des noeuds.

Pour des simulateurs étudiant la preuve de travail, la simulation de la topologie du réseau n'est pas critique. En effet, elle n'a pas d'impact sur la création de blocs et la propagation est de l'ordre de la seconde alors que la création de blocs est de l'ordre de la minute.

4.2 POISimulator

La particulité de la Preuve d'Intéraction étant que l'algorithme de consensus dépend entièrement du réseau dans la création des blocs, la couche réseau de la bockchain doit être simuler plus présisement que ne le permet les simulateurs existants.

En effet, si BlockPerf parait semble le plus prometteur la simulation du réseau, il ne permet de reproduire la topologie du réseau d'une blockchain que dans la mesure ou il est installer sur une topologie de réseau proche de celle du d'une blockchain avec

ses pools de mineur fortements interconnecter dans des data-center proche du coeur du réseau et des noeuds de particulier en bordure. Et, comme le souligne les auteurs, installer un tel simulateurs en des endroit géographiquement éloigner n'est pas aisé.

On définit une API pour modéliser une blockchain du niveau réseau au niveau application et on utilise NS3 pour la simuler.

4.2.1 Réseau

Pour la définition du réseau on fournit une interface de haut niveau qui permet definir le réseau comme un graphe de sous-graphes.

C'est aussi à ce niveau que sont géré les bindings avec NS3.

Le réseau fournit les services suivants à la couche Overlay :
- listes des connections tcp ouvertes
- envoye/reception de message TCP
- protocole applicatif pour la découverte de noeuds (DNS)

Dans l'état actuel, l'intégration de NS3 n'est pas encore finalisé.

4.2.2 Overlay

Pour la définition de l'overlay on le definit également comment un graphe et on fournit des algorithmes de détection des pairs et de routage dans l'overlay

L'overlay fournit à la couche consensus les services suivants :
- Listes des pairs connus/connectés
- Découverte de pairs
- Envoye d'un message à un pair (broadcast)
- Reception du message d'un pair

Dans l'état actuel de la PoI, on suppose tous les noeuds connus et connectés la découverte de pairs n'est donc pas necessaire.

4.2.3 Consensus

La couche consensus permet la validation des blocks et définit un choix de fork. Elle permet également de récompenser les pairs pour leurs efforts lors de la validation.

C'est à ce niveau que sont récupérées les donnés suivantes :

Consomation (temps/energie) nombre de fork

4.2.4 Data

La couche définit les block de données et stocke l'état de la blockchain

Les couches exécution et application ne sont pas encore définies.

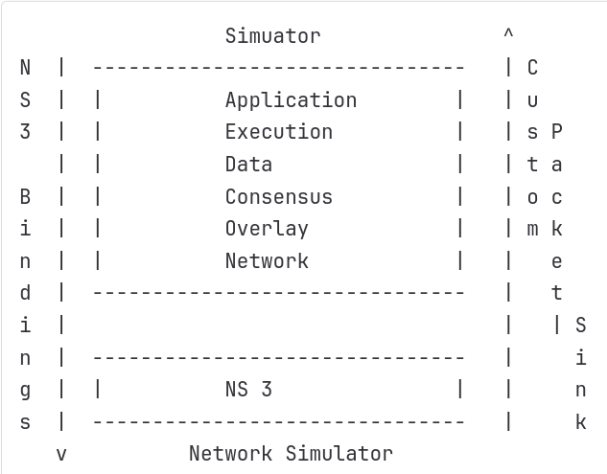


FIGURE 4.1 – Architecture du simulateur.

4.3 Travaux futur

Les travaux futurs seront de finaliser le simulateur réussissant à intégrer NS3. Une fois les couches réseau et overlay finalisées, il sera possible de comparer le simulateurs avec les simulateurs existants et les données empiriques.

Une autre étape de l'amélioration du simulateur sera de permettre de générer des topologies. Ce qui necessitera de mener une étude plus appronfondie des topologies des réseaux de blockchain existant.

Chapitre 5

Conclusion

Mon stage a été l'occasion de découvrir un domaine de recherche qui m'était inconnu. Il m'a permis d'acquérir des compétences en algorithmique distribuée et simulation de réseaux.

Les objectifs de mon stage étaient de modéliser et d'optimiser les probabilités de fork dans une blockchain utilisant la preuve d'intéraction.

Sur le plan théorique, j'ai étudié les blockchains et les algorithmes de consensus. et sur le plan pratique, débuter la conception d'un simulateur de blockchain.

Le simulateur que j'ai commencé à concevoir n'est pas encore abouti. Il faudra poursuivre le travail pour intégrer le simulateur NS3 et fournir une interface permetant de définir des topologies de réseau de blockchain.

Sur le plan théorique, notre résultat est un préliminaire qui vise à améliorer l'algorithme de consensus de Preuve d'Intéraction.

Bibliography

[ABN21] Jean-Philippe ABEGG, Quentin BRAMAS et Thomas NOËL. « Blockchain Using Proof-of-Interaction ». In : *Networked Systems*. Sous la dir. de Karima ECHIHABI et Roland MEYER. T. 12754. Springer International Publishing, 2021, p. 129-143. ISBN : 978-3-030-91013-6 978-3-030-91014-3. DOI : 10.1007/978-3-030-91014-3_9. URL : https://link.springer.com/10.1007/978-3-030-91014-3_9 (visité le 20/03/2024).

[AV20] Maher ALHARBY et Aad VAN MOORSEL. « BlockSim : An Extensible Simulation Tool for Blockchain Systems ». In : *Frontiers in Blockchain* 3 (9 juin 2020), p. 28. ISSN : 2624-7852. DOI : 10.3389/fbloc.2020.00028. URL : https://www.frontiersin.org/article/10.3389/fbloc.2020.00028/full (visité le 12/02/2024).

[And13] Gavin ANDRESEN. *BIP 50 - March 2013 Chain Fork Post-Mortem*. 20 mars 2013. URL : https://github.com/bitcoin/bips/blob/master/bip-0050.mediawiki.

[Cha83] D. CHAUM. « Blind Signature for Untraceable Payments, Advances in Cryptology ». In : *Proceedings of the Springer-Verlag Crypto'82*. T. 3. 1983, p. 199-203.

[Cro+16] Kyle CROMAN et al. « On Scaling Decentralized Blockchains : (A Position Paper) ». en. In : *Financial Cryptography and Data Security*. Sous la dir. de Jeremy CLARK et al. T. 9604. Series Title : Lecture Notes in Computer Science. Berlin, Heidelberg : Springer Berlin Heidelberg, 2016, p. 106-125. ISBN : 978-3-662-53356-7 978-3-662-53357-4. DOI : 10.1007/978-3-662-53357-4_8. URL : http://link.springer.com/10.1007/978-3-662-53357-4_8 (visité le 19/08/2024).

[DW13] Christian DECKER et Roger WATTENHOFER. « Information propagation in the Bitcoin network ». In : *IEEE P2P 2013 Proceedings*. Trento, Italy : IEEE, sept. 2013, p. 1-10. ISBN : 978-1-4799-0515-7. DOI : 10.1109/P2P.2013.6688704. URL : http://ieeexplore.ieee.org/document/6688704/ (visité le 19/08/2024).

[FLP85] Michael J. FISCHER, Nancy A. LYNCH et Michael S. PATERSON. « Impossibility of distributed consensus with one faulty process ». In : *Journal of the ACM* 32.2 (avr. 1985), p. 374-382. ISSN : 0004-5411, 1557-735X. DOI : 10.1145/3149.214121. URL : https://dl.acm.org/doi/10.1145/3149.214121 (visité le 18/04/2024).

[Ger+16] Arthur GERVAIS et al. « On the Security and Performance of Proof of Work Blockchains ». en. In : *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. Vienna Austria : ACM, oct. 2016, p. 3-16. ISBN : 978-1-4503-4139-4. DOI : 10.1145/2976749.2978341. URL : https://dl.acm.org/doi/10.1145/2976749.2978341 (visité le 12/02/2024).

[LB24] Thomas LELIÈVRE et Quentin BRAMAS. « The impact of block mining time distribution on the probability of forks ». In : *BRAINS ; 5th Conference on Blockchain Research & Applications for Innovative Networks and Services*. Berlin, Germany, oct. 2024. URL : https://hal.science/hal-04675227.

[Nak09] Satochi NAKAMOTO. *Bitcoin*. Version v0.1.5. 16 sept. 2009. URL : https://github.com/bitcoin/bitcoin.

[Nak07] Satoshi NAKAMOTO. « Bitcoin : A Peer-to-Peer Electronic Cash System ». In : (2007). URL : https://bitcoin.org/bitcoin.pdf (visité le 08/08/2024).

[Pol+21] Julien POLGE et al. « BlockPerf : A Hybrid Blockchain Emulator/Simulator Framework ». In : *IEEE Access* 9 (2021), p. 107858-107872. ISSN : 2169-3536. DOI : 10.1109/ACCESS.2021.3101044. URL : https://ieeexplore.ieee.org/document/9500200/ (visité le 16/05/2024).

[Sch20] Fabian SCHÄR. « BLOCKCHAIN FORKS : A FORMAL CLASSIFICATION FRAMEWORK AND PERSISTENCY ANALYSIS ». In : *The Singapore Economic Review* (2020), p. 1-11. URL : https://api.semanticscholar.org/CorpusID:225355815.

Table des figures