Comment Gagner de l'Argent sans Travailler

Jean-Philippe Abegg, Quentin Bramas, Thomas Noël

Comment Gagner de l'Argent sans Travailler

Jean-Philippe Abegg[1,2] et Quentin Bramas[1] et Thomas Noël[1]

[1] *ICUBE, Université de Strasbourg, CNRS*
[2] *Transchain, Strasbourg*

Le protocole Bitcoin utilise la *preuve de calcul* pour élire un nœud aléatoire du réseau pouvant ajouter un bloc à la chaîne de blocs. Ce processus d'élection d'un nœud est au coeur de la sécurité du protocole mais est responsable d'une consommation importante d'énergie. Les nœuds sont encouragés à participer par le fait qu'un nœud élu remporte une récompense, qui a une valeur monétaire.

Notre objectif est d'obtenir le même résultat sans travail. Dans cet article, nous présentons un nouvel algorithme, appelé *preuve d'interaction*, permettant l'élection aléatoire d'un nœud du réseau. En utilisant cet algorithme, nous construisons une Blockchain, similaire à Bitcoin, mais où les nœuds n'ont pas besoin d'une grande puissance de calcul.

Mots-clefs : Blockchain, Preuve-de-travail, Système distribué

1 Introduction

La Blockchain est un protocole permettant de maintenir un livre de comptes de manière distribuée entre plusieurs participants. Les transactions sont ajoutées sous la forme de *blocs* liés entre eux pour former une chaîne. Comme tous les nœuds du réseau ont le même rôle, un système d'élection de leader est utilisé pour élire le nœud responsable de l'ajout du prochain bloc. Le protocole Blockchain le plus connu, Bitcoin, utilise la *preuve de travail* (PoW) comme système d'élection [Nak08]. Avec la PoW, la probabilité qu'un nœud soit élu est proportionnelle à sa puissance de calcul. Cela incite donc les participants à dépenser beaucoup d'énergie afin d'augmenter leur chance d'être élu.

Dans cet article nous proposons une alternative à la preuve de travail appelée preuve d'interaction. Puis nous montrons comment utiliser ce système de preuve pour construire un protocole Blockchain, similaire à Bitcoin, mais qui ne nécessite presque aucun travail. De plus, notre Blockchain offre une meilleure résistance aux attaques par minage égoïste. Un rapport technique contenant plus de détails est disponible en ligne [ABN20].

Travaux connexes La *preuve de travail* (PoW) est à l'origine une technique anti-spam proposée en 1999 et adaptée en 2008 par le protocole Bitcoin [Nak08]. La PoW peut être vue comme un système d'élection de leader, pour sélectionner le nœud responsable de l'ajout du prochain bloc dans la chaine de blocs. La PoW permet de se protéger contre les attaques Sybil, les dénis de service et a un faible coût en communication, mais elle est énergivore.

Plusieurs alternatives à la PoW existent, par exemple la preuve d'enjeu (PoS) et la preuve de temps écoulé (PoET). Ces alternatives posent cependant des contraintes sur le réseau sous-jacent, ou introduisent de nouvelles failles de sécurité.

M. Abliz et T. Znati [AZ09] ont présenté un système de protection anti-spam qui n'a pas, à notre connaissance, été utilisé dans le contexte des Blockchains. L'idée de ce système est la suivante : quand un serveur de ressources reçoit un grand nombre de requêtes, il demande au client de faire un tour dans le réseau *i.e.*, visiter un ensemble de serveurs ayant le même propriétaire que le serveur de ressources. Une fois ce tour réalisé, le client peut prouver au serveur de ressources la réalisation de ce tour pour obtenir l'accès à la ressource. La preuve d'interaction étend ce schéma pour être utilisé au sein d'une Blockchain.

Modèle On considère un réseau N constitué de n nœuds, dont le graphe de communication est complet. Les nœuds sont identifiés de manière unique par leur clé publique (l'association clé publique – nœud est

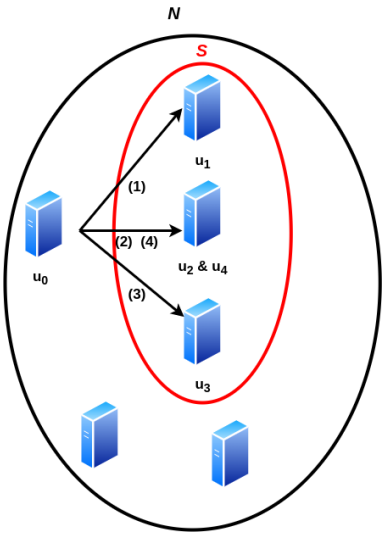

FIGURE 1: u_0 interagit avec un sous ensemble S de N

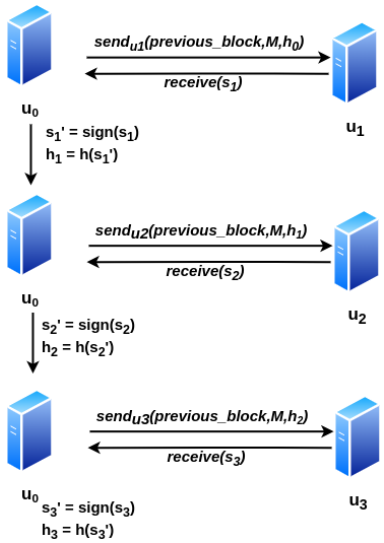

FIGURE 2: u_0 interagit de manière séquentielle avec les différents nœuds.

connue pour l'ensemble du réseau). Chaque message envoyé est signé par l'émetteur et ne peut pas être modifié ou falsifié par un autre nœud. Les communications sont supposées partiellement synchrones, comme dans le protocole Bitcoin. Le temps maximum d'attente avant la réception d'un message est borné par une valeur inconnue.

L'opérateur $\cdot$ désigne la concaténation. La fonction H est une fonction cryptographique de hachage. Nous écrirons $\texttt{sign}_u(m)$ la signature par le nœud u du message m. Nous supposons que la fonction de signature est déterministe et ne dépend que du message et de la clé privée de u. Cette hypothèse peut sembler forte car avec les schémas de signatures habituels, un message peut avoir plusieurs signatures différentes valides. Cependant, nous pensons qu'elle peut être levée en pratique, et nous traiterons ce problème dans des travaux futurs.

Dans la Section 2, nous présentons les algorithmes permettant la génération et la vérification d'une preuve d'interaction. La Section 3 montre comment ces algorithmes sont utilisés pour construire un protocole Blockchain.

2 La Preuve d'Interaction (PoI)

On note d la *dépendance* de la preuve (dans notre cas, elle correspond au hash du bloc précédent) et m le *message* prouvé (dans notre cas, il correspond à la racine de l'arbre de Merkle qui stocke les transactions du bloc en cours). On note D la distribution de probabilité utilisée pour tirer au hasard la longueur des tours dans le réseau de chacun des nœuds. D représente la difficulté de la preuve.

Calculs Préliminaires Un nœud u_0 qui veut générer une PoI doit connaître *(i)* le sous-ensemble S de nœuds du réseau avec qui il va interagir et *(ii)* la taille du tour L qu'il doit réaliser. Ces deux informations sont calculées de manière pseudo-aléatoire avec comme graine s_0, la signature par u_0 de la dépendance d ($s_0 = \texttt{sign}_{u_0}(d)$).

On a $S = \{S_0, S_2, \ldots, S_{n_S-1}\}$, un sous-ensemble de N pseudo-aléatoire de taille $n_S = \min(20, n/2)$, généré de manière déterministe à partir de s_0. De même, L est un nombre pseudo-aléatoire qui suit la distribution D, générée de manière déterministe à partir de s_0. Le nœud u_0 n'a donc aucun moyen de modifier S et L, et chaque nœud du réseau qui génère une PoI a son propre ensemble S et sa propre longueur L, supposés aléatoires. On voit sur la Figure 1 que le nœud u_0 interagit avec une suite de nœuds $u_1, u_2, \ldots$ dans le sous-ensemble S.

Génération de la Preuve Pour un nœud u_0, le tour dans le réseau se réalise en L phases séquentielles. Chaque phase $j \in [1,L]$ consiste en la création d'un hash h_j, à partir du hash h_{j-1} de la phase précédente, avec $h_0 = H(s_0 \cdot m)$. A chaque phase $j \in [1,L]$, u_0 réalise les étapes suivantes : (i) calculer le nœud u_j de S avec qui il va interagir. On a $u_j = S_i$ où l'indice i est le hash précédent modulo n_S, $i \equiv h_{j-1} \, [n_S]$; (ii) envoyer à u_j le tuple (d,m,h_{j-1}). Le nœud u_j répond en envoyant $s_j = \mathtt{sign}_{u_j}(d \cdot m \cdot h_{j-1})$; (iii) signer s_j pour obtenir $s'_j = \mathtt{sign}_{u_0}(s_j)$; (iv) calculer $h_j = H(s'_j)$.

Les phases sont exécutées de manière séquentielle jusqu'à trouver s_L et $s'_L = \mathtt{sign}_{u_0}(s_L)$. La Figure 2 montre un exemple de tour complet pour $L = 3$, avec pour d le hash du bloc précédent.

Définition 1 *En utilisant les notations précédentes, la preuve d'interaction de dépendance d, du message m par le nœud u_0, avec la difficulté D est la suite* $(s_0, s_1, s'_1, s_2, s'_2, \ldots, s_L, s'_L)$.

Vérification de la Preuve Pour qu'un nœud u vérifie une preuve $(s_0, s_1, s'_1, s_2, s'_2, \ldots, s_L, s'_L)$ venant de u_0 de dépendance d, message m et difficulté D, il doit d'abord vérifier que s_0 est bien la signature par u_0 de d, puis recalculer S, L et h_0 à partir de s_0, D et m. Par ailleurs, u peut calculer la suite des hashs $h_j = H(s'_j)$, $1 \leq j \leq L$.

Puis u doit vérifier que chaque s'_j $(1 \leq j \leq L)$ est bien une signature valide par u_0 de s_j. Pour finir, u doit vérifier que chaque s_j $(1 \leq j \leq L)$ est bien une signature valide de $d \cdot m \cdot h_{j-1}$ par S_i, où $i \equiv h_{j-1} \, [n_S]$.

3 Protocole Blockchain Utilisant la Preuve d'Interaction

Dans cette partie, nous présentons le protocole Blockchain qui utilise notre Preuve d'interaction et en listons plusieurs propriétés intéressantes.

Génération de blocs On construit une Blockchain similaire à la Blockchain Bitcoin. Un bloc contient la liste des transactions (stockées dans un arbre de Merkle), le hash du bloc précédent, la difficulté (on suppose que la distribution D est caractérisée par un nombre, qui peut être la moyenne par exemple) et d'autres méta-données (*eg.*, version, horodatage). Dans Bitcoin, la preuve de travail est stockée sous forme de nonce. Dans notre cas, chaque bloc contient la preuve d'interaction (Définition 1) dont la dépendance est le hash du bloc précédent et le message est la racine de l'arbre de Merkle contenant les transactions du bloc.

Quand un bloc, de hash d, vient juste d'être ajouté à la Blockchain, chaque nœud du réseau regroupe les transactions valides non encore présentes dans un bloc, les stocke dans un arbre de Merkle de racine m, et commence à générer une PoI de dépendance d et message m. Le premier nœud qui termine sa PoI ajoute son bloc à sa chaîne de blocs et diffuse son bloc dans tout le réseau. Quand un nœud reçoit un bloc contenant une preuve valide, il ajoute ce bloc à sa chaîne de blocs. Si ce bloc augmente une branche qui devient la branche la plus longue, alors ce nœud arrête sa PoI en cours et en commence une nouvelle en utilisant ce nouveau bloc comme dépendance.

Ajustement de la difficulté La difficulté peut être ajustée comme dans le protocole Bitcoin, c'est-à-dire en utilisant les horodatages inscrits dans les blocs, de manière à ce que la durée entre deux blocs soit en moyenne une constante B. Ici, cette difficulté peut être paramétrée précisément en utilisant la distribution D. Par exemple, supposons que la durée moyenne d'une communication dans le réseau soit notée Com. Si chaque tour a une longueur choisie aléatoirement de manière uniforme entre 1 et $\lceil B/Com \rceil (n+1) - 1$, alors on peut montrer que la longueur du plus petit tour, parmi les PoI de tous les nœuds, sera en moyenne $\lceil B/Com \rceil$ (soit une durée $\lceil B/Com \rceil \times Com$ entre deux blocs).

Récompenses et Pénalités Chaque bloc rapporte une récompense à tous les nœuds ayant participé à la PoI. Cette récompense peut être définie comme dans le protocole Bitcoin *i.e.*, le nombre de *tokens* offerts aux nœuds récompensés contient les frais de transaction plus une certaine quantité, qui décroît après un certain nombre de blocs.

On suppose aussi que tous les noeuds du réseau ont verrouillé une certaine quantité de *tokens* dans la Blockchain. Ils peuvent ainsi être pénalisés s'ils ont certains comportements. Par exemple, si un nœud crée deux blocs différents, ayant la même dépendance d, alors il peut être pénalisé, comme le détaille le paragraphe suivant.

Non parallélisable Un nœud u_0 ne peut pas calculer en parallèle une PoI pour un message m et une dépendance d donnés. En effet, pour contacter le i-ème nœud u_i, u_0 doit avoir demandé une signature, qu'il ne peut pas prévoir, au nœud u_{i-1}. Cela implique que le calcul de la preuve ne peut être réalisé que de manière séquentielle en commençant par le calcul de s_0.

Pour effectuer plusieurs preuves en parallèle, un nœud u_0 doit créer plusieurs blocs (avec des valeurs m différentes). Cela permet d'augmenter ses chances d'obtenir la récompense, mais il risque alors d'être fortement pénalisé si un autre nœud s'en rend compte. Or, l'ensemble S des participants et la longueur L ne dépendent pas de m. Donc, si le nœud u_0 crée deux blocs, avec deux valeurs m_1 et m_2, il y a une forte probabilité qu'un même nœud v participe aux deux PoI du nœud u_0. Ce nœud v va recevoir deux demandes de signature, une avec la valeur m_1 et l'autre avec la valeur m_2, émanant du même nœud u_0, pour la même dépendance d. Dans ce cas, v pourra donc lancer l'alerte et à son tour gagner une récompense.

Attaque par recherche d'un tour avantageux On suppose pour cette attaque qu'un sous-ensemble du réseau est malicieux, dont u_0. Ils partagent leur clé privée dans le but d'accélérer la génération de la PoI.

Comme la suite des nœuds de S avec qui u_0 doit interagir dépend du bloc en cours, u_0 pourrait, en théorie, créer plusieurs blocs afin de choisir celui dont tous des noeuds à contacter sont malicieux. S'il trouve un tel bloc, u_0 pourrait générer la PoI sans envoyer un seul message (il connaît les clés privées des autres noeuds malicieux). Cependant, comme l'ensemble S ne dépend pas du bloc en cours, u_0 ne peut pas le modifier. Comme S est supposé être un sous-ensemble aléatoire de N, il contient la même proportion de nœuds malicieux que dans le réseau tout entier. Supposons que cette fraction soit de 10% et que u_0 doive effectuer un tour de longueur 100, alors il lui faudra en moyenne générer 10^{100} blocs avant de trouver un tour qui ne contienne que des nœuds malicieux.

Minage égoïste Le minage égoïste est un comportement bien connu dans les Blockchains utilisant la preuve de travail. Elle consiste en un groupe de mineurs malicieux qui collaborent pour maximiser leurs gains. Quand un nœud du groupe trouve un bloc il le partage uniquement avec ses collaborateurs. Le groupe de mineurs commence alors à travailler secrètement sur le prochain bloc pendant que les nœuds honnêtes gaspillent leur énergie dans la recherche d'un bloc sur une branche obsolète.

Notre protocole permet de limiter ce genre de comportement. En effet, lorsqu'un nœud u_0 effectue sa PoI, chaque demande de signature envoyée à u_i contient la dépendance d utilisée par u_0. Si u_0 trouve un bloc, il est obligé de le révéler au moins au membre de S avec qui il interagit pour le bloc suivant (on a vu au paragraphe précédent, que la recherche d'un tour ne contenant que des noeuds malicieux n'était pas raisonnable en pratique). Un nœud honnête u_i refusera de signer un bloc qu'il ne connaît pas. Cette collaboration entre les différents nœuds pour générer les preuves impose une bonne propagation des blocs dans le réseau.

4 Conclusion

Nous avons présenté une alternative à la PoW demandant une faible puissance de calcul pour les participants. De plus, on peut montrer que le protocole Blockchain basé sur la PoI a une complexité en nombre de messages par unité de temps linéaire en fonction de la taille du réseau. Notre protocole fait l'hypothèse que les nœuds du réseau sont connus, mais nous pensons que les techniques existantes qui permettent de lever cette hypothèse sont particulièrement adaptées à la PoI.

Références

[ABN20] Jean-Philippe Abegg, Quentin Bramas, and Thomas Noël. Blockchain using proof-of-interaction, February 2020. `https://doi.org/10.5281/zenodo.3669309`.

[AZ09] Mehmud Abliz and Taieb Znati. A guided tour puzzle for denial of service prevention. In *2009 Annual Computer Security Applications Conference*, pages 279–288. IEEE, 2009.

[Nak08] Satoshi Nakamoto. Bitcoin : A peer-to-peer electronic cash system.(2008), 2008.