POLYTECH SORBONNE

Projet
Blockchain IOTA

KALLEL ABDELAZIZ
RODRIGUEZ AMAURY
COMBARET LÉO
AMEZIANE SABRI

Encadrante :
POTOP-BUTUCARU MARIA
MAIN3
2021/2022

Table des matières

1 Introduction

1.1 Mise en contexte

Une blockchain, ou chaîne de blocs, est une technologie de stockage et de transmission d'informations sans autorité centrale. Les informations transmises sont uniques et ont une valeur. Ces dernières ne peuvent être ni copiées, ni falsifiées. Les blockchains les plus connus sont celles de Bitcoin et d'Etherum. Cependant les blockchains qui existent à l'heure actuel ne sont pas parfaites. En effet, trois critères sont importants dans une blockchain.

— **La sécurité :** En sachant que les informations transmises sont uniques et ont une valeur, il est très problématique que la blockchain se fasse facilement pénétrée.

— **La décentralisation :** Le concept d'une blockchain est le fait que personne n'ait la main mise dessus et que personne ne peut ni altérer ni contrôler son fonctionnement.

— **La scalabilité :** C'est le fait de pouvoir effectuer un grand nombre de transcations aux plus petits coûts.

Ainsi le défi est de pouvoir construire une blockchain qui est la plus décentralisée, scalable et sécurisée possible. Cela s'appelle le trilemme des blockchains.

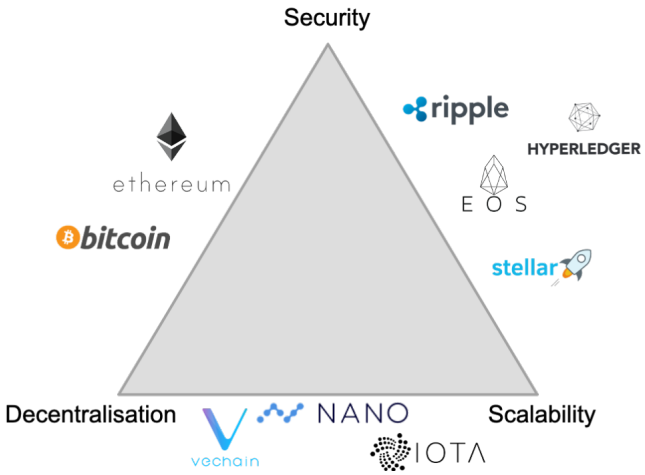


FIGURE 1 – Trilemme des blockchain

Notre projet porte sur la blockchain Iota. C'est un projet open-source qui a pour objectif de fournir un moyen de paiement sécurisé pour la monétisation de données entre les appareils dits de l'Internet des objets "IOT". L'Internet des objets est l'ensemble des objets connectés, de la voiture intelligente (comme les Tesla par exemple) au smartphone en passant par la domotique, pouvant communiquer entre eux. C'est une notion assez complexe et évolutive mais qui est aujourd'hui omniprésente et qui sera, dans le futur, encore plus utilisée.
Comme on peut le constater sur la figure 1, IOTA est une blockchain très scalable mais peu

décentralisée et sécurisée par rapport à celles de Bitcoin et d'Etherum. IOTA est basée sur une architecture de type DAG (Directed Acyclic Graph) appelée "Tangle" dont nous allons discuter dans la section 2.2.1.

1.2 Sujet

Les dispositifs de l'Internet des objets (IoT) sont utilisés dans un large éventail d'applications et l'environnement de l'IoT devrait continuer à se développer exponentiellement notamment grâce à la 5G. Cependant, c'est un environnement très vulnérable de par la potentielle fragilité des objets connectés et des interfaces, mais aussi du grand nombre de connections réalisées par tous les objets de l'IOT. Cette vulnérabilité se caractérise par la faible sécurité face aux attaques. Dans le cadre de ce projet, nous allons donc étudier et comprendre le fontionnement de la blockchain IOTA en se basant sur les documents fournis par M. Potop-Butucaru ainsi que les travaux effectués par nos ainés en Main 4.
Nous serons confrontés à beaucoup de vocabulaires et de notions complexes propre à ce domaine (tous ces termes sont expliqués dans l'annexe [7]), l'objectif premier est donc de se familiariser avec le fonctionnement de la blockchain IOTA, ainsi que tous les concepts qu'elle introduit afin de pouvoir par nous même implémenter et tester son fonctionnement.
Ensuite, nous chercherons les limites de leurs protocoles et nous proposerons notre idée pour pallier ces limites.
Enfin, nous implémenterons notre solution en C afin de pouvoir la tester face aux problèmes que nous cherchions à résoudre. Nous avons choisi le C car c'est un langage que nous maitrisons et qui semble être assez approprié à ce genre de modélisation, par exemple grâce à la possibilité d'initialiser des structures et de pouvoir visualiser nos graphes avec GraphViz.

2 IOTA 1.0

2.1 Le ledger

En raison des vulnérabilités de l'IOT aux différentes attaques et des conséquences potentiellements très néfastes, l'approche actuellement dominante dans la gestion des objets connectés est la centralisation des opérations de contrôle. Cependant cette centralisation ne semble pas une option viable lorsqu'on envisage des réseaux de centaines de milliers de dispositifs par km^2 en particulier lorsque ces dispositifs peuvent être limités en taille et en alimentation électrique.

L'utilisation de la technologie des Ledger [7] distribués peut répondre à la fois aux besoins de sécurité et de décentralisation dans la gestion des dispositifs IoT. La technologie des Ledger distribués (DLT) tel que les blockchain fournissent un moyen sécurisé pour partager des informations entre un grand nombre de noeuds indépendants fonctionnant sous différentes autorités, tout en assurant une haute disponibilité.

La DLT, inaugurée par la technologie Bitcoin, a créé une nouvelle philosophie de conception pour exécuter et stocker les transactions de manière décentralisée et sécurisée. Une blockchain est un ledger distribué qui imite le fonctionnement d'un registre traditionnel classique (c'est-à-dire la transparence et preuve de falsification de la documentation) dans un environnement non fiable. Les systèmes de blockchains traditionnels tel que Bitcoin ou Etherum maintiennent une liste continuellement croissante des blocs vérifiés par les membres du réseau appelés mineurs [7]. Les blocs de transactions sont liés entre-eux grâce à la cryptographie et leur ordre dans la blockchain est le résultat d'un accord (consensus) entre les participants du réseau. La technologie Bitcoin et les propositions similaires présentent plusieurs inconvénients qui les empêchent d'être utilisés comme normes pour l'industrie des objets connectés notamment le temps d'attente pour la validation d'une transaction et les frais engendrés. Par exemple il n'est pas possible d'acheter sa baguette en payant avec son smartphone, la validation serait trop longue.

2.2 L'approche d'IOTA

2.2.1 Le Tangle

Le tangle [7] est une alternative à la blockchain. Ce concept a été expliqué par S. Popov dans The Tangle [10]. Il va faire office de Ledger à la technologie Iota. C'est un registre qui stocke les transactions sous la forme d'un graphe orienté dans une seule direction et non circulaire, comme représenté sur la figure 2, on appelle cela un graphe acyclique dirigé (DAG). Pour être validée, une transaction doit approuver deux transactions précédentes. Cette alternative vise à surmonter les limites des ledgers de type Bitcoin lorsqu'ils sont utilisés dans un environnement IoT tout en préservant des niveaux de sécurité équivalents. En effet, le but est de permettre d'effectuer des transactions très rapidement et sans frais tout en étant sécurisées. Ces transactions sont continuellement ajoutées au Tangle.
Comme nous pouvons le voir sur la figure 2, le Tangle est composé d'un Genesis [7], de Tips [7] et de Sites. Les Tips (en jaune) sont les transactions (noeuds) qui ne sont pas encore approuvées, les Sites (en bleu) sont les transactions qui ont déjà été approuvées et qui font partie du Tangle. Enfin, le genesis est la genèse du Tangle, c'est à dire la première transaction effectuée au sein du Tangle.

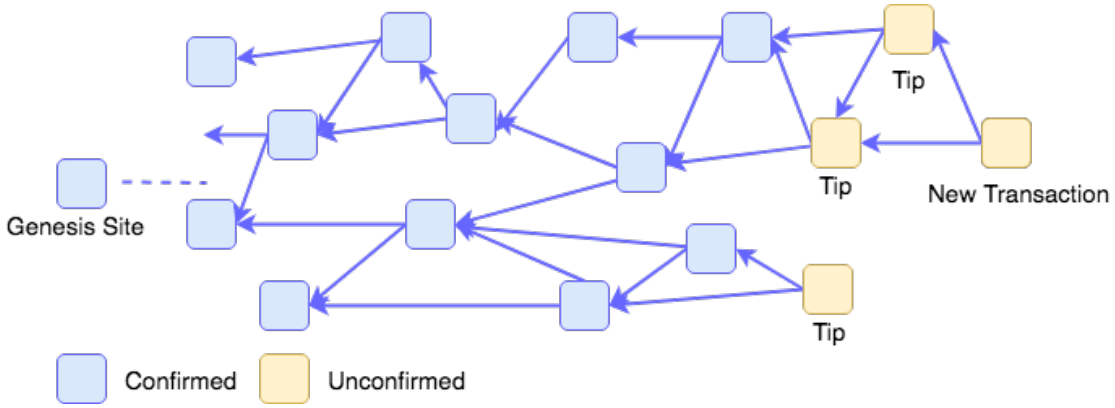


FIGURE 2 – Représentation d'un Tangle simplifié

2.2.2 Le Coordinateur

Défini par Iota comme une entité de confiance dans le papier [1], le coordinateur est utilisé comme protection contre les transactions malveillantes. Le Tangle n'est pas encore un produit final, il est encore en version bêta donc le réseau repose actuellement sur une sorte de bouclier appelé coordinateur. Le coordinateur agit comme un mécanisme de consensus [7] alternatif centralisé, volontaire et temporaire pour le Tangle afin de déterminer si une transaction est frauduleuse ou non. Pour ce faire, le coordinateur envoie des transactions honnêtes aux nœuds complets à intervalles réguliers. Ces paquets contiennent un message signé sans valeur, appelé jalon. Les nœuds complets du Tangle considèrent qu'une transaction est confirmée uniquement si elle est approuvée par un jalon. Le coordinateur peut seulement confirmer les transactions, mais il ne peut pas contourner les règles de consensus. Il ne lui est pas possible de créer, geler ou voler des token. Cette règle fixe et l'adresse du coordinateur sont codées en dur sur chaque nœud complet, de sorte que l'influence du coordinateur sur le Tangle est très limitée, puisque le Tangle est aussi constamment surveillé par tous les autres nœuds complets. Le coordinateur sera désactivé avec la mise à jour de IOTA 2.0.

2.3 Les limites du protocole Iota 1.0

2.3.1 La centralisation

Le coordinateur du réseau IOTA a été lancé pour éviter les dépenses superflues des utilisateurs et la scission du réseau en palliant au faible nombre de noeuds présents dans le tangle. Le coordinateur expose le réseau à un risque important en cas d'arrêt du fonctionnement ou de prise en charge par un malfaiteur. De plus, cela va totalement à l'encontre du principe de blockchain qui se veut sécurisé, scalable et décentralisé. Cette décentralisation est très importante pour avoir un système autonome totalement protégé des acteurs malveillants.

2.3.2 Le Double Spending

Le Double spending [7] est un acte frauduleux dans lequel le même token est dépensé plusieurs fois comme nous pouvons le voir sur la figure 3 tirée de l'article de Hans Moog [8].

L'adresse C possède 550 Iota qu'elle va dépenser avec l'adresse K et avec l'adresse D, il va y avoir une double utilisation de ces token Iota alors qu'il n'en existe pas autant. Un token va donc être utilisé pour deux transactions différentes, chaque blockchain utilise un système de sécurité pour contrer ces attaques très fréquentes. Nous allons voir dans la section suivante les différents moyens mis en place par Iota pour combattre ce fléau. C'est l'attaque la plus répandue contre les cryptomonnaies, c'est donc un des points central des systèmes de sécurités mis en place par Iota.

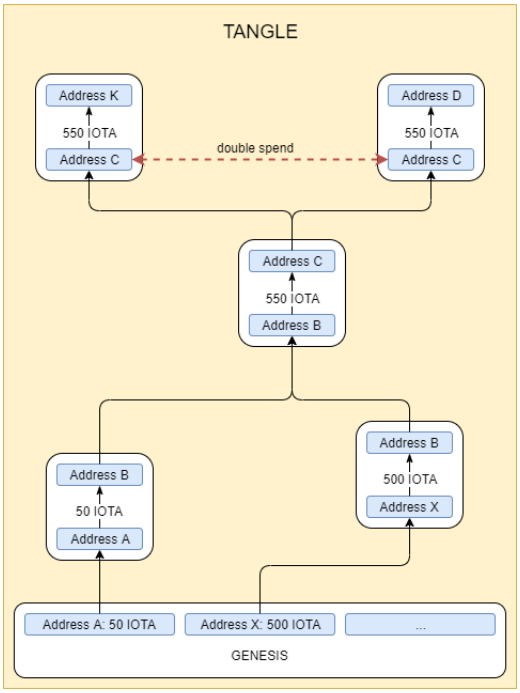


FIGURE 3 – Représentation d'un double spending

3 IOTA 2.0

3.1 Coordicide

Suite à des failles de sécurité, le protocole IOTA n'est plus décentralisé et un coordinateur a été mis en place. Celui-ci assure le bon fonctionnement du réseau, sa sécurité et un consensus [7] sur le Tangle. Il est donc essentiel pour le bon fonctionnement de la blockchain, néanmoins la centralisation de l'autorité et de la vérification la rend vulnérable. En effet, la blockchain parfaite est un compromis de trois facteurs : la sécurité, la scalabilité et la décentralisation. C'est le projet d'Iota avec son nouveau protocole Iota 2.0 et le concept de Coordicide (tuer le coordinateur)[9]. La suppression du coordinateur permettrait une grande avancée dans la décentralisation d'Iota. Pour y arriver il faut que les fonctions effectuées par le coordinateur puissent être transférées aux acteurs du réseau sans compromettre sa sécurité. Le rôle du coordinateur étant assez important il est difficile de s'en débarasser. La sélection du point de départ, le calcul de l'état du Ledger et la synchronisation des noeuds sont des actions initialement réalisées par le coordinateur qui devront être réaliséees en trouvant une alternative. Néanmoins, les principaux défis sont le consensus et le Rate Control [7] de par leur importance primordial dans le fonctionnement et la sécurité d'Iota. Dans la sous section suivante, nous allons analyser le concept de Consensus proposé.

3.2 Consensus

Le consensus est le principe selon lequel le réseau se met d'accord sur la validité d'une transaction (c'est à dire si la transaction est frauduleuse ou non, par exemple dans le cas d'un double spending) lorsque plusieurs transactions sont contradictoires. Différents acteurs de la blockchain vont donc avoir des opinions différentes sur la validité d'une transaction. Le consensus permet de tous les mettre d'accord pour conserver un unique Tangle. L'idée de base est que tous les nœuds honnêtes du système doivent dynamiquement s'accorder vers une seule et même opinion, de sorte que cette opinion ne puisse pas être modifiée facilement par d'autres [8].

Considérons un réseau connecté composé de N nœuds, enumérés comme tel : $\{1 \cdots i \cdots N\}$. Des liens sans pertes relient les nœuds du réseau entre eux. Les nœuds connectés directement sont des voisins. Nous supposons que le temps est discret et divisé en tours. Chaque nœud a un statut d'opinion, $O_i(r) \in 0, 1$ au tour r. Le consensus est atteint, si $\forall i, j \in N, O_i(r_{end}) = O_j(r_{end})$, où r_{end} est le dernier tour de simulation. Une opinion détenue par la plupart des nœuds est une opinion majeure, cette opinion majeure va influencer l'opinion des noeuds voisins au tour suivant. Si un nœud a l'opinion 0 (transaction non valide) au tour 0 mais que ses voisins ont majoritairement l'opinion 1 (transaction valide), il aura une certaine probabilité (dépendant de l'opinion majeure) de prendre l'opinion 1 au tour 1. Le taux de convergence est le pourcentage de passages menant à un stade de consensus, plus le consensus sera atteint rapidement plus la transaction sera validée ou réfutée rapidement, on parle de convergence du consensus. Ainsi, avec cette méthode, les noeuds sont censés pouvoir décider de la validité d'une transaction de manière autonome, sans l'aide d'un coordinateur.

3.3 Utxo

Le modèle de sortie de transaction non dépensée (UTXO) définit un état du grand livre (ledger) où les soldes ne sont pas directement associés aux adresses mais aux sorties de transactions. Dans ce modèle, les transactions spécifient les sorties des transactions précédentes comme entrées, qui sont consommées afin de créer de nouvelles sorties comme on peut le voir sur la figure 4. Une transaction doit consommer l'intégralité des entrées spécifiées. La section qui déverrouille les entrées est appelée bloc de déverrouillage. Un bloc de déverrouillage peut contenir une signature prouvant la propriété de l'adresse d'une entrée donnée et/ou d'autres critères de déverrouillage.

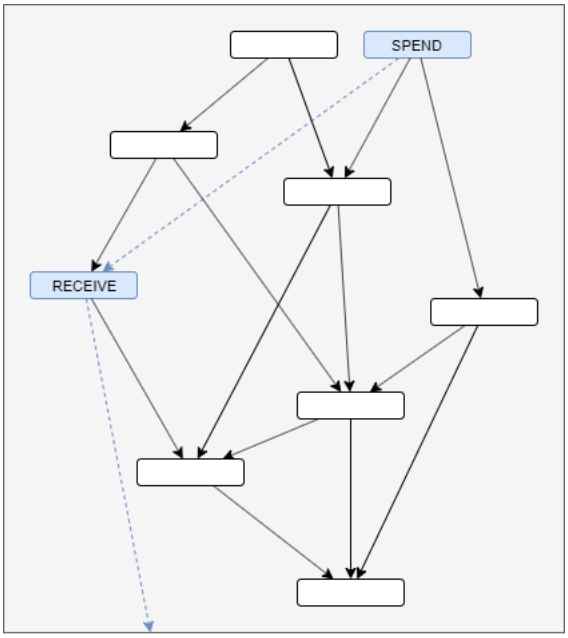


FIGURE 4 – Suivi d'une transaction

Le DAG UTXO modélise la relation entre les transactions, en suivant quelles sorties ont été dépensées par quelles transactions. Comme les sorties ne peuvent être dépensées qu'une seule fois, nous utilisons cette propriété pour détecter les dépenses doubles.

Nous permettons à différentes versions du Ledger de coexister temporairement tel sur la figure 5 et 6. Ceci est possible en étendant le DAG UTXO par l'introduction de branches. Nous pouvons alors déterminer quelles versions conflictuelles de l'état du grand livre existent en présence de conflits (grâce à notre consensus Grands électeurs). Ainsi, nous permettons à différentes versions du Ledger de coexister temporairement.

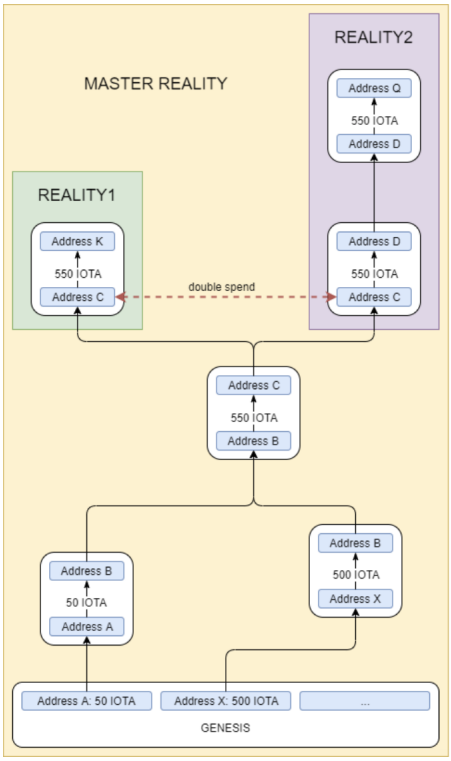


FIGURE 5 – Creation d'une double réalité au moment ou on a détecté un double spending

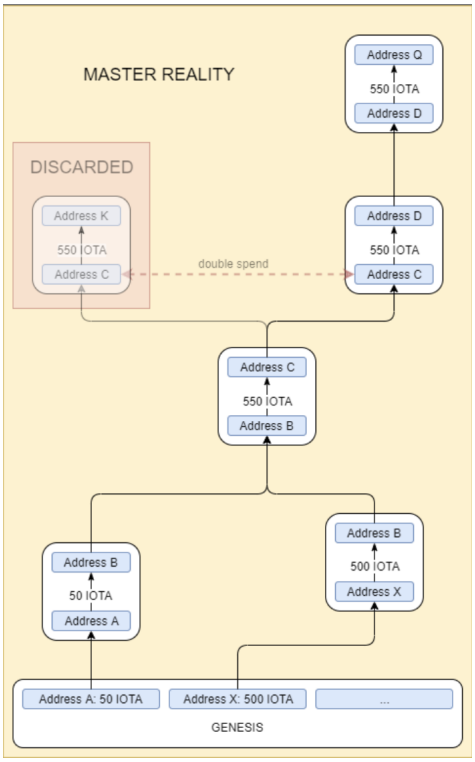


FIGURE 6 – Suppression de la version frauduleuse

3.4 Les limites du protocole Iota 2.0

Tout comme le protocole Iota 1.0, Iota 2.0 a des limites. Ces limites sont principalement liés au système de consensus donc au système de validation des transactions.

3.4.1 Les Adversaires Byzantin

Le travail de nos collègues de l'année précédente [6] ont démontré la faiblesse du consensus face à différents noeuds perturbateurs appelés adversaires Byzantins. Afin de simuler les attaques possibles et prédire les cas critiques de sécurité, ces trois adversaires ont été implémenter et sont intervenu lors du processus de consensus.

- Adversaires prudents :

Ces noeuds sont capables de mentir à chaque tour du processus avec une probabilité donnée. Cependant, l'opinion donnée durant un seul et même tour est tout le temps identique même si les requêtes proviennent de différents noeuds. Ces noeuds vont très souvent avoir un impact néfaste sur la convergence du consensus (cela dépend des topologies [7] et du type de consensus).

- Adversaires semi-prudents :

Ces noeuds vont toujours mentir. Cependant, ils peuvent ne pas répondre à une requête avec une probabilité donnée. Ainsi, ils retardent le processus de convergence et reduisent le

nombre de noeuds accessibles dans le réseau. Ces noeuds vont avoir un faible impact sur la convergence du consensus.

- **Adversaires Berserk :**

Cet adversaire est plus fort que les deux précédents. Il se comporte similairement aux adversaires prudents, excepté le fait qu'il soit capable de fournir des réponses différentes pour chaques requêtes qu'il recoit dans le même tour. Ainsi, pendant le même tour, il peut envoyer sa vraie opinion puis mentir et répondre avec une fausse opinion. Comme pour les adversaires prudents, ces noeuds vont très souvent avoir un impact néfaste sur la convergence du consensus (cela dépend des topologies [7] et du type de consensus).

En outre, nous remarquons avec ces simulations d'attaque que le consensus a ses limites. Il se peut qu'il n'y ait pas convergence. Pour résoudre ce problème nous avons eu l'idée de créer le système des grands électeurs qui, selon nous, peut résoudre le problème initiale de double spending mais aussi celui de non convergence du consensus.

4 Notre Idée - Le système des Grands Électeurs

Pour contrer les attaques de double spending, nous avons réfléchis à un système de vérification qui pourrait décider de la validité d'une transaction sans craindre le sabotage de noeuds malveillants (comme expliqué précédemment 3.4.1). Pour pallier tous les problèmes évoqués précédemment nous avons pensé à notre propre solution entre le coordicide et le coordinateur : les grands électeurs. Cette solution répond au problème de convergence de consensus sur lequel nos camarades de MAIN ont travaillé l'année dernière[6]. En effet avec un nombre d'acteurs variant il y aura aussi des variations dans la vitesse de convergence causant des ralentissements et des erreurs. Les problèmes étaient surtout liés à la vitesse de convergence et à la manipulation d'opinion en fonction du nombre d'attaquant.

4.1 Sélection des noeuds Grands Électeurs

Les Grands Électeurs doivent être des noeuds fiables et fidèles mais aussi des noeuds choisis aléatoirement pour éviter tout risque d'attaque. Nous avons donc pensé à un système permettant d'allier ces deux conditions.

4.1.1 Système de fidelité - Fiabilité

Pour palier à tout les problème évoqués précédemment nous avons pensé à notre propre solution entre le coordicide et le coordinateur : les grands électeurs. Cette solution répond au problème de convergence de consensus sur lequel nos camarades de Main ont travaillé l'année dernière. Les problèmes étaient surtout liés à la vitesse de convergence et à la manipulation d'opinion en fonction du nombre d'attaquant .
Notre idée est qu'à partir du réseau de noeuds chaque acteur du réseau gagne, à partir de son activité, ancienneté, nombre de transactions, un score de confiance apellé mana 7.

$$NouveauManaEffectif = \alpha(ManaDeLaTransaction) + (1-\alpha)(AncienManaEffectif) \quad (1)$$

Quand une transaction est effectuée, le mana est attaché à l'identité du noeud en plus de son nombre de token. Cette quantité est proportionnelle au montant de la transaction. Le seul moyen de gagner du mana est donc de convaincre un acteur de vous faire confiance. Cela permet une défense contre les attaques Sybil [7] car il est difficile d'en obtenir sans être un membre actif du réseau. Mais surtout une défense face aux adversaires Byzantin qui ne devront plus être de simple noeuds participant aux votes mais des noeuds fiables et fidèles au réseau.

4.1.2 Choix des Grands Électeurs

A partir de ce score est alors tiré aléatoirement, parmi les meilleurs acteurs du réseau, un groupe de noeuds qui vont devenir des coordinateurs temporaires : les grands électeurs. Ils sont tirés au sort assez fréquemment et le tangle est transmis de grands électeurs en grands électeurs. Les premiers noeuds choisis comme Grands Électeurs seront tirés au sort par le coordinateur afin de laisser le système de fidélité se mettre en place. Le coordinateur sera ensuite remplacé par la marche aléatoire.

4.1.3 Nombre de Grands Électeurs

Le nombre de noeuds participant au consensus est un élément clé, en effet, il a été montré dans le travail de l'année antérieure [6] qu'un trop grand nombre de noeuds participant au consensus ralenti celui ci. Le système des grands électeurs va donc permettre un consensus convergeant très rapidement en utilisant un faible nombre de noeuds. Comme nous pouvons le voir sur la figure 7 tirée de [6], un grand nombre de noeuds empêchent une convergence optimale du consensus lors de la présence d'adversaires byzantin. En utilisant un faible nombre de noeuds (les grands électeurs), on pourrait se ramener au cas de la figure 8, aussi tirée de [6], c'est à dire avec un taux de convergence optimale. On voit que l'absence d'adversaires byzantin et l'utilisation de peu de noeuds amènent une convergence de quasiment 100%.
Ainsi, la sécurité du système de fidélité permet de filtrer les noeuds pour évacuer un maximum les adversaires byzantin, ensuite, la sélection des grands électeurs va permettre une nouvelle fois de réduire le nombre d'adversaires potentiel et enfin, le faible nombre de noeuds va réduire à néant l'impact sur la convergence du consensus de potentiel adversaires byzantins qui se serait faufilés jusqu'au rang de grand élécteur.

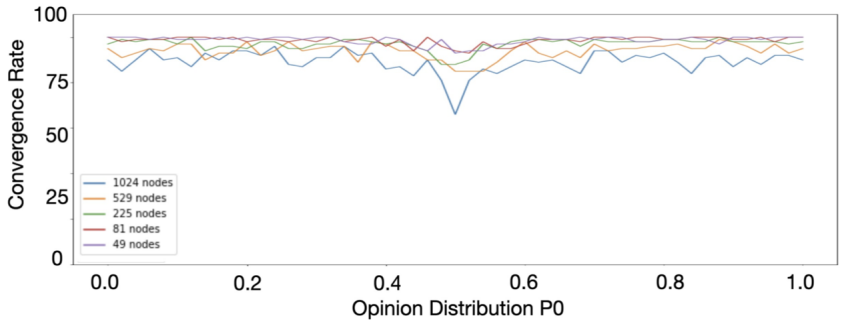


FIGURE 7 – Convergence du Consensus avec adversaires Byzantins

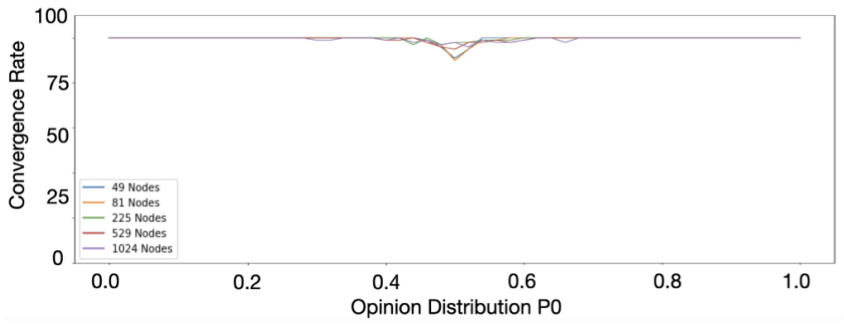


FIGURE 8 – Convergence du Consensus sans adversaires Byzantins

4.2 Rôle des Grands Électeurs

Si une transaction est douteuse et qu'il y a divergence d'opinion, les grands électeurs vont intervenir. Ils auront été selectionné au préalable comme nous l'avons expliqué au préalable 4.1. Ces noeuds vont donc participer au consensus des Grands Électeurs qui est un cellular consensus avec un faible nombre de noeuds. En effet, nous avons opté pour un cellular consensus qui offre une meilleure convergence mais qui est aussi plus simple à implémenter, nous reviendrons dessus dans la section 4.3.2. Une fois le consensus effectué et la transaction validée ou non, les grands électeurs sont déchus et remplacés par de nouveaux. Une fois qu'un noeud est remplacé en tant que grand électeur il redevient simple noeud et son score est baissé pour éviter la monopolisation du pouvoir de décision sur la blockchain et permettre une meilleure décentralisaion. Ainsi ce sysème pourrait éliminer le double spending grâce au consensus qui converge sans craindre d'attaques et avoir une version unique du tangle (sur laquelle tous les électeurs ont établi un consensus).

4.3 Implémentation

Afin de pouvoir implémenter notre idée de système des Grands Electeurs, nous avons d'abord choisi d'utiliser OMNET ++. Ce dernier est un logiciel basé sur le language de programmation C++ qui fournit des fonctionalités afin de pouvoir créer de manière optimale un réseau de noeud. Cependant, après avoir appris le language C++ ainsi que les différentes fonctinalités de OMNET++, la difficulté de prise en main du logiciel nous faisait perdre du temps pour des problèmes de logistique et non d'algorithmie. Nous avons donc décidé de coder notre simulation en C afin de pouvoir avoir un maximum de maniabilité et de contrôle lors de l'implémentation et des tests.

4.3.1 Topologie

Dans le papier des camarades de l'an passé sur la résilience du consensus d'Iota [6], une conlusion a pu être tirée, la topologie avec la meilleure résilience était Watts-Strogatz. En effet, cette topologie introduite dans le document [4] offrait le meilleur taux de convergence du consensus. Le modèle de Watts-Strogatz est un modèle de génération de graphes, qui possède la propriété du "small world" (le calcul du plus court chemin entre deux nœuds est logarithmique). Cette méthode prend en paramètre N le nombre de nœuds, K le degré moyen des nœuds du réseau et P la probabilité permettant de modifier les arêtes. Le but du processus est

de partir d'un graphe en anneau afin de traiter chaque arête d'un nœud. En effet, chaque arête peut changer de destinataire avec la probabilité P passée en paramètre.

FIGURE 9 – Différents graphs en topologie Watts-Strogatz selon différentes valeurs de P

Ainsi, pour pouvoir implémenter cette topologie nous nous sommes basé sur un code déjà existant, disponible sur un répositoire Github public [3] qui nous a permi d'avoir une base sur laquelle commencer. Nous avons donc pu modifier le code afin de pouvoir l'utiliser à nos fins [7]. Ici, nous avons une structure nommée *graph* qui admet comme paramètres, une matrice de booléens afin de déterminer s'il existe une connexion entre deux noeuds, un entier *size* qui stocke le nombres de noeuds de la topologies, ainsi qu'un entier *edges* (N) dans lequel est stocké le nombre de liaisons initiales par noeuds (K).

```
struct graph_t* build_regular_graph(int num_vertices, int edges_per_vertex)
{
  struct graph_t* g = build_unconnected_graph(num_vertices);
  int i, j;

  #pragma omp parallel for private(i, j)
  for (i = 0; i < num_vertices; i++) {
    for (j = 1; j <= (edges_per_vertex >> 1); j++) {
      set_edge(g, i, (i + j) % g->size, true);
      set_edge(g, i, (g->size + i - j) % g->size, true);
    }
  }

  g->edges_per_vertex = edges_per_vertex;
  return g;
}
```

FIGURE 10 – Fonction *build-regular-graph* renvoyant un graph regulier

Afin de visualiser par la suite notre réseau de noeuds, nous enregistons la disposition des noeuds dans un fichier *.dot* pour utiliser l'extension graphviz. Tout d'abord, nous utilisons la fonction *build-regular-graph* qui prend en paramètres le nombres de noeuds (N) ainsi que le nombre de liaisons initiales (K). *build-regular-graph* (Figure 10) appelle une autre fonction *build-unconnected-graph* qui renvoie une structure graph sans aucune liaisons. Cela permet d'initia-

liser la matrice du graphe à la bonne taille avec des 0 avec l'utilisation de *calloc()*. Ensuite, la fonction parcours la matrice en définissant les connections de manière régulière à partir de la valeur de *edges* (K). Ainsi, la fonction renvoie une structure *graph* avec des liaisions régulières. Puis, afin de pouvoir définir des liaisons de manière aléatoire, nous faisons appelle à la fonction *randomise-graph* qui prend en paramètre la probabilité pour qu'une liaison soit déplacée (P). Cette fonction parcours la matrice et choisi une liaison entre deux noeuds au hasard, selon la probabilité P, la deconnecte en remplacant par 0 leur liaison dans la matrice et en choisissant un nouveau noeud en prenant soin de ne pas faire une boucle sur elle-même.

En utilisant ces deux fonctions, nous pouvons ainsi créer des graphs en choisissant nous même le comportement des liaisons grâces aux paramètres N, K et P comme sur la figure 11.

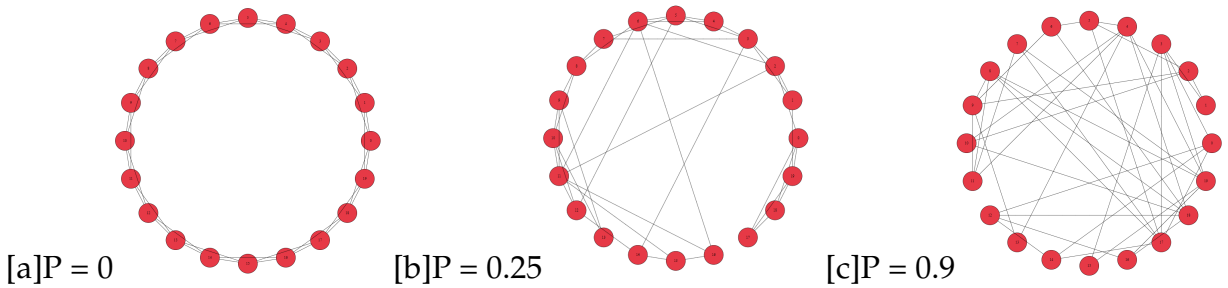


FIGURE 11 – Liaisons des noeuds selon la valeur que prend P avec N = 20, K = 4

4.3.2 Consensus des Grands Électeurs

Comme expliqué dans la partie 3.2, lorsque deux transactions sont en conflits, il faut que le système puisse se mettre d'accord afin de pouvoir choisir la transaction la plus pertinente. Pour cela, nous allons utiliser la méthode du *Cellular Automata Consensus* introduit dans le monde de la blockchain avec le papier [2].

Dans ce consensus, chaque nœud agit comme un agent individuel qui change d'opinion en suivant une règle locale en cas de conflit avec ses voisins. Chaque nœud n'a que quelques nœuds voisins. En se propageant à travers les interactions locales, les états locaux finiront par affecter le comportement global du consensus.

Au début de chaque tour, chaque nœud envoie un "heart-beat" de son opinion actuelle signée. Lorsqu'un nœud reçoit une opinion donnée par l'un de ses voisins, il évalue cette opinion par une "preuve" accompagnant l'opinion. Cette "preuve" est matérialisée par les opinions des voisins du voisin. Cela permettra aux nœuds de se surveiller mutuellement et de détecter si quelqu'un ment indépendamment de ses voisins. Si cette "preuve" montre que le voisin ment, il sera immédiatement mis sur liste noire par le nœud et aucun de ses avis ne sera pris en compte. Puisque les opinions précédentes des voisins ne peuvent pas être falsifiées, chaque nœud peut s'assurer que l'avis reçu est bien effectivement correcte. Nous avons choisi d'utiliser le Cellular Automata Consensus pour obtenir un système efficace, décentralisé, et dynamique de sorte que les informations et les données peuvent être transmises efficacement et dynamiquement sans connectivité centralisée.

Algorithm 5: Cellular consensus

foreach *node* i **do**
  Send initial opinion $X_0(i)$ to neighbors N_i

for $m \leftarrow 1$ **to** M **do**
  foreach *node* i **do**
    foreach *neighbor* $j \in N_i$ **do**
      if $X_{m-1}(j)$ *is inconsistent wrt* $X_{m'}(j')$ *for* $j' \in N_j, m' < m-1$ **then**
        `// drop neighbor` j
        $N_i \leftarrow N_i \setminus \{j\}$
    if *node* i *finalized* **then**
      $X_m(i) \leftarrow X_{m-1}(i)$
    else
      `// adopt majority opinion`
      $total \leftarrow \sum_{\{j \in N_i\}} p(|N_j|)$
      if $\sum_{\{j \in N_i | X_{m-1}(j)=0\}} p(|N_j|) > \frac{total}{2}$ **then**
        $X_m(i) \leftarrow 0$
      else if $\sum_{\{j \in N_i | X_{m-1}(j)=1\}} p(|N_j|) > \frac{total}{2}$ **then**
        $X_m(i) \leftarrow 1$
      else
        $X_m(i) \leftarrow \perp$ `// cancel all`
    `heartbeat(`i, m`)`
    if *opinion* $X(i)$ *did not change in the last* ℓ *rounds* **then**
      mark node i finalized

FIGURE 12 – Algorithme du *Cellular Automata Consensus*

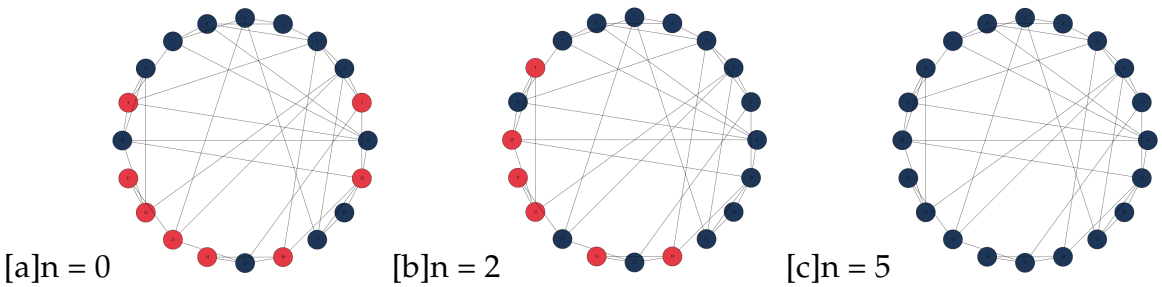
En détail, l'algorithme du consensus cellulaire fonctionne comme suit (Figure 12). A chaque étape de l'algorithme, chaque nœud détient une opinion, qui peut être 0, 1. Si l'opinion de plus de la moitié des voisins du noeud i au tour r est :

— 0, alors l'opinion du noeud i au tour r + 1 sera 0,

— 1, alors l'opinion du noeud i au tour r + 1 sera 1

Si autant d'opinions valant 1 que valant 0 :

— le noeud i garde son opinion initiale



[a]n = 0 [b]n = 2 [c]n = 5

FIGURE 13 – Consensus du réseau avec N = 20, K = 4, P = 0.5 avec p0 = 0.5

Ainsi, afin de pouvoir visualiser les opinions sur le graphe, nous avons changé la couleur de chaque noeud, selon sa propre opinion. De ce fait, si l'opinion du noeud est *0*, alors le noeud sera de couleur rouge. Si l'opinion du noeud est *1*, alors le noeud sera de couleur bleue. Nous pourrons donc facilement visualiser si la topologie a pu converger ou non en regardant l'uniformité des couleurs des noeuds composant le réseau.

Nous avons pu donc implémenter l'algorithme de *Cellular Automata Consensus* [7] en se basant sur le pseudo-code (Figure 12) en C dans notre code.(Figure 14)

Notre fonction *Cellular-consensus* prend en paramètre un graph de topologie *Watts-Strogatz*. La fonction va itérer pour chaque noeuds du graphe. Nous initialisons les variables *opinion-0* et *opinion-1* qui stockeront respectivement, le nombre de voisin avec l'opinion 0 et 1. La variable degree comporte la moitié du nombre de voisins pour le noeud k que l'on traite. Ce nombre sera un *int* car la topoligie *Watts-Strogatz* n'admet que des nombres paires.

Ainsi, nous parcourons la matrice du graphe sur la ligne k afin de nous arrêter lorsque un lien existe avec un autre i. Si lorsque l'opinion du noeud i vaut 0 i.e. la valeur de la matrice à l'indice (i,i) est 0, alors on incrémente la variable *opinion-0* (de même pour lorsque son opinion est 1). Après avoir parcouru toute la ligne de la matrice, nous décidons quelle opinion, le noeud k va prendre. Si le nombre des voisins qui ont une opinion 0 (*opinion-0*) est supérieur à plus de la moitié du nombre des noeuds *degree* alors l'opinion du noeud k au tour n+1 vaut 0.

De même, si le nombre de voisins qui ont une opinion 1 (*opinion-1*) est supérieur à plus de la moitié du nombre des noeuds *degree* alors l'opinion du noeud k au tour n+1 vaut 1. Si ces deux conditions ne sont pas respectées, alors le noeud k garde son opinion initiale au tour n+1.

```c
void cellular_consensus(struct graph_t* g){

    for(int k=0; k < g->size; k++){ //itere le nombre de noeuds
        int opinion_0 = 0; //somme des opinions valant 0
        int opinion_1 = 0; //somme des opinions valant 1
        int degree = get_degree(g,k) / 2; //nbr de voisins / 2

        if(tab_opinions[k] > DECISION || tab_opinions[k] < -DECISION) g->mat[k*g->size+k] = g->mat[k*g->size+k]; //cas ou decidé

        for(int i=0; i < g->size; i++){ //parcours ses voisins
            if(get_edge(g,i,k) && g->mat[i*g->size+i] == 0){ //si l'opinion du voisin est 0
                opinion_0 ++ ;
            }
            else if(get_edge(g,i,k) && g->mat[i*g->size+i] == 1){ //si l'opinion du voisin est 1
                opinion_1 ++ ;
            }
        }

        if(opinion_0 > degree){
            g->mat[k*g->size+k] = 0;
            tab_opinions[k] = tab_opinions[k] - 1;
        }
        else if(opinion_1 > degree){
            g->mat[k*g->size+k] = 1;
            tab_opinions[k] = tab_opinions[k] + 1;
        }
        else g->mat[k*g->size+k] = g->mat[k*g->size+k];
    }
}
```

FIGURE 14 – Implémentation du *Cellular Automata Consensus* en C

Dans le code, nous avons aussi implémenté un tableau nommé *tab-opinion*. Ce tableau est de taille *N* et est mis à jour après chaque tour. Chaque case représente l'opinion de chaque noeud. La valeur de la case diminue de 1 pour chaque tour où l'opinion du noeud vaut 0 et augmente de 1 pour chaque tour où l'opinion du noeud vaut 1. Ainsi, si dans le tableau, un noeud à une valeur supérieur à +/- 4, i.e. le noeud a gardé la même opinion depuis plus de 4 tours, alors le noeud ne changera plus d'opinion quoi qu'est l'opinion de ses voisins. (ligne 45 du code Figure 14)

Ce consensus est le consensus qui offre la meilleure convergence comme expliqué dans [8]. De plus, il est plus simple à implémenter que l'autre type de consensus (le fast probabilistic consensus). C'est pour cela que nous avons choisi de l'utiliser dans le système des Grands Électeurs.

4.4 Analyse

Après avoir testé différentes valeurs pour chaque paramètres, nous avons pu observer quelques comportements sur la convergence du réseau. La convergence est très rapide avec cette topologie (<10 itérations) cependant, pour que la rapidité de convergence soit constante et se fasse à chaque fois il faut que le réseau ait une taille critique de 20 et que son nombre de voisins soit au moins de 7. En effet avec un faible nombre de voisins du fait de la topologie du réseau peut entraîner une non convergence. Ce sont des premières observations qui pourront être approfondies avec l'utilisation de OMNET++.

4.5 L'algortihme de selection de Tip (TSA)

Pour finaliser notre système des grands électeurs, il nous reste à choisir un algortihme de selection de Tip [7] afin d'établir le Tangle [2.2.1]. Le choix de cet algorithme est plus que cruciale car selon celui qu'on choisi, la validation des transactions va se faire plus ou moins vite, plus ou moins sécurisée. Lorsqu'un utilisateur émet une transaction, elle est ajoutée au Tangle et doit approuver deux tips, c'est à dire deux transactions non approuvées par au moins deux autres transactions chacunes. Ainsi, la nouvelle transaction devient un Tip du Tangle et attend d'être choisie par deux nouvelles transactions pour être approuvée. Par conséquent, pour pouvoir émettre des transactions, un utilisateur est dans l'obligation de participer à la sécurité du Tangle, sinon il ne verra jamais ses transactions validées par les autres.

L'enjeu est donc de savoir choisir les tips à approuver de manière aléatoire tout en garantissant la sécurité du Tangle face à de potentielles attaques (discuté dans le papier [10]) en ayant un taux de transactions validées satisfaisant.

Ce papier *Implementation of different TSA for IOTA using OMNET++* [11] a étudié *Tips selection algorithm* en les implémentant et les comparant, nous allons nous en servir de base. Ces simulations se font sur OMNET++.

4.5.1 E-IOTA

Dans cette section, nous allons nous concentrer sur l'algorithme de selection de Tip nommé **E-IOTA** [5]. Cet algorithme utilise l'algorithme de marche aléatoire proposé par le papier [10]. Nous générons ici un nombre aléatoire r à chaque fois qu'une transaction est émise tel que $r \epsilon [0, 1)$ et poser *p1/p2* tels que $0 < p1 < p2 < 1$. L'algoritme de cette méthode est la suivante :

Algorithm : E-IOTA

Input : - N, le nombre de walkers
- W, un entier
- $r \in [0, 1]$, un réel généré aléatoirement
- $p1/p2 \in]0, 1[$

Output: Deux tips à approuver

Pseudo-code :

if $r < p1$ **then**
  on effectue une marche aléatoire uniforme, i.e, avec $\alpha = 0$ en utilisant les paramètres N et W.
else if $p1 \leq r < p2$ **then**
  on effectue une marche aléatoire avec un α faible en utilisant les paramètres N et W.
else if $p2 \leq r < 1$ **then**
  on effectue une marche aléatoire avec un α élevé en utilisant les paramètres N et W.

FIGURE 15 – Algorithme de la méthode E-IOTA

Grâce au Github [11], nous avons pu visualiser un Tangle construit avec le TSA **E-IOTA**

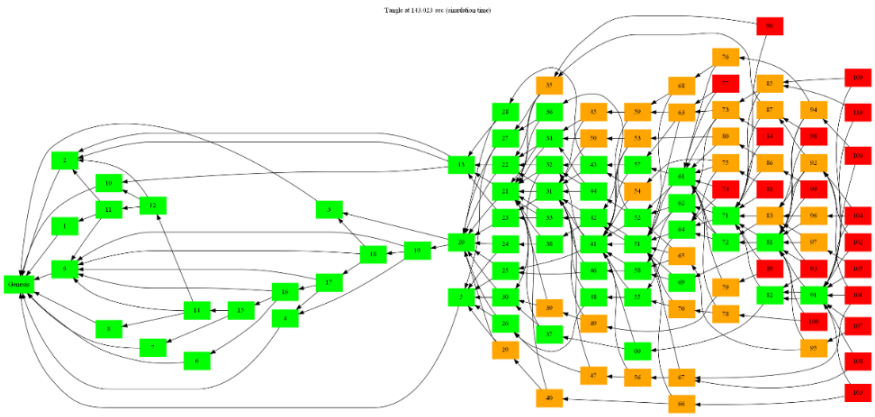


FIGURE 16 – Tangle construit avec E-IOTA

Ainsi, l'utilisation du TSA **E-IOTA** est celle avec le meilleur compromis, qui minimise le nombre de Tips tout en minimisant la durée de l'exécution du processus selon [11] comparé aux autres méthode de TSA tels que **IOTA** ou encore *G-IOTA*

5 Conclusion

En définitive, l'IoT est un domaine en pleine expansion mais qui est déjà très présent tout autour de nous. Cet envrionnement a besoin d'une décentralisation pour pouvoir continuer à s'étendre tout en restant sécurisé. C'est la solution que veut proposer Iota en franchissant les limites imposées par les blockchains tradtionnelles en terme d'adaptation à l'IoT. En effet, Iota propose un protocole qui permettra de réaliser des transactions rapides et sans frais tout en étant décentralisé, sécurisé et scalable (trilemme de la blockchain 1).
Cependant, avec son premier protocole, Iota n'était pas capable de fournir une solution décentralisée aux problèmes de l'IoT. Comme nous l'avons vu à la section 2.3, le coordinateur assure une certaine sécurité mais empêche la décentralisation d'Iota qui est un élément essentiel. De plus, le double spending est un risque très répandu donc sujet central dans les discussions de sécurité. Iota 2.0 apporte une solution decentralisée à ce fléau qu'est le double spending. Cette solution est le coordicide et la mise en place d'un consensus comme expliqué dans la section 3. Supposé apporter une issue viable, le consensus a des faiblesses qui ont été démontrées dans [6].
Ainsi, nous avons eut l'idée de créer le système des Grands Électeurs (énoncé dans la section 4) pour pallier ces limites . En reprenant le cellular consensus mais en y apportant des mesures de sécurité supplémentaire et un nombre de noeuds électeurs plus faible, ce concept permet de résoudre le problème de convergence qu'il y avait avec le consensus classique d'Iota. De plus, la sélection grâce au système de fidélité permet de limiter grandement l'intrusion d'adversaires Byzantins. Nous avons ensuite implémenté cette méthode en C et nous l'avons visualisé avec GraphViz
Finalement, c'est bien cette méthode qui nous semble la plus adaptée pour le moment en répondant aux plusieurs problématiques posées par les blockchains et l'IoT. Elle reste néanmoins un concept améliorable.

6 Bilan personnel du projet

6.1 Membres

6.1.1 RODRIGUEZ AMAURY

Ce projet a été très différent du premier, le challenge n'était pas sur l'implémentation mais sur la comprehension du sujet et des différents aspects théoriques de la blockchain. Le travail de recherche et de compréhension a été très intéressant car cette blockchain diffère sur de nombreux points des blockchain traditionnelles tels que bitcoin et ethereum sur la forme de la blockchain par exemple(TAG).

6.1.2 AMEZIANE SABRI

Ce projet m'a permis de découvrir le monde des blockchain qui est considéré comme une des technologies majeures de notre société future. J'étais totalement étranger à ce domaine, la phase d'apprentissage et de compréhension a été assez longue, difficile au vu de la complexité de ces notions. J'ai compris la majeure partie du fonctionnement d'Iota et les enjeux autour de blockchain en général, cependant, je reste hostile à toutes ces innovations qui, selon moi, peuvent avoir un effet extrêmement néfaste sur notre société. Ensuite, j'ai trouvé intéressant la recherche d'idées pour pallier les faiblesses décelées chez Iota. Le système des grands électeurs est le fruit de nombreux idées qui nous sont venus tout au long du projet, j'ai trouvé cela très intéressant, néanmoins, l'implémentation l'était beaucoup moins.

6.1.3 COMBARET LEO

Tout d'abord j'ai choisi ce sujet car c'est un domaine qui me passionne beaucoup et dans lequel je porte mon intérêt depuis septembre dernier. Avec ce projet j'ai pu découvrir le fonctionnement de l'intérieur des blockchain avec les différents concepts que nous avons pu évoquer. Cependant j'aurais bien voulu avoir plus de temps pour pouvoir aller plus en détail dans l'implémentation car il y a encore bcp de chose à découvrir.

6.1.4 KALLEL ABDELAZIZ

Ce projet m'a permis d'approfondir mes connaissances dans le domaine de la blockchain, un domaine qui me passionne depuis quelques mois, surtout avec les différents aspects théoriques de la blockhain IOTA qui se caractérise par une architecture différente des autres et qui cherche à faciliter l'adoption massive de cette technologie. Ce travail que j'ai eu la chance de faire avec mes collègues m'a permis de comprendre les défaillances des blockchains traditionnelles tels que bitcoin et ethereum, j'aurais bien voulu avoir plus de temps pour pouvoir implémenter.

Références

[1] Coordinator. part 1 : The path to coordicide. https://blog.iota.org/coordinator-part-1-the-path-to-coordicide-ee4148a8db08/, 2018.

[2] Nkn : a scalable self-evolving and self-incentivized decentralized network. https://nkn.org/wp-content/uploads/2020/10/NKN_Whitepaper.pdf, 2018.

[3] Andrewch. Public watts-srogatz github repository. https://github.com/andrewrch/strogatz, 2012.

[4] S. H. Strogatz D. J. Watts. Collective dynamics of 'small-world'networks. nature, 1998.

[5] Maria Potop-Butucaru Gewu Bu, Wassim Hana. E-iota : an efficient and fast metamorphism for iota. https://ieeexplore.ieee.org/document/9223294, 2020.

[6] Osama Rashid-Gewu Bu Maria Potop-Butucaru Hamed Mamache, Gabin Mazué. Resilience of iota consensus. https://hal.archives-ouvertes.fr/hal-03427543, 2021.

[7] Sabri Ameziane-Abdelaziz Kallel Leo Combaret, Amaury Rodriguez. Implementation du cellular automata consensus algorithm. https://github.com/Shimaadakunn/Cellular_consensus, 2022.

[8] Hans Moog. A new "consensus" : The tangle multiverse [part 1]. https://husqy.medium.com/a-new-consensus-the-tangle-multiverse-part-1-da4cb2a69772.

[9] C. Darcy-C. Angelo D. Vassil-G. Alon G. Andrew K. Bartosz M. Sebastian P. Andreas S. Olivia S. William V. Luigi W. Wolfgang P. Serguei, M. Hans and A. Vidal. The coordicide. https://files.iota.org/papers/20200120_Coordicide_WP.pdf, 2020.

[10] S. Popov. The tangle. http://www.descryptions.com/Iota.pdf, 2019.

[11] Emile Pichard-Ghassen Hachani Tahar Amairi, Léo Lassalle. Implementation of different tsa for iota using omnet++. https://github.com/T-amairi/Implementation-of-different-TSA-for-IOTA, 2021.

7 Annexe

— **Attaque Sybil** : Lorsqu'un système est mis en danger par un utilisateur contrôlant plusieurs identités censées appartenir à des peronnes différentes

— **Consensus** : Acoord entre les noeuds (acteurs du réseau) sur la validité d'une transaction.

— **Genesis** : L'adresse fondatrice du Tangle et la première transaction a être émise dans le Tangle (elle est donc approuvée par toutes les autres transactions).

— **Double spending** : Utilisation d'un même token pour effectuer deux (ou plusieurs) transactions .

— **Ledger** : Une structure qui sert à répertorier toutes les transactions (ce rôle est joué par le tangle dans le cas de IOTA).

— **Mana** : Score de confiance évoluant selon plusieurs critères

— **Miner** : Une opération consistant à valider une transaction sur un réseau blockchain par le biais d'un calcul mathématique.

— **Rate Control** : Controle du flux des nouvelles transactions

— **Tangle** : Le graph acyclique créé par l'ensemble des transactions (Ledger).

— **Tip** : Une transaction non encore approuvée émise par un noeud dans le Tangle.

— **Token** : Une entité de valeur créée et protégée par la technologie blockchain.

— **Topologie** : La manière dont on choisit de relier les noeuds entre eux.