Blockchain appliquée à un processus électoral — compte-rendu

Z. Aloui, Z. Hua — lu2in006

1 Introduction

L'objectif de ce projet est de mettre au point un système de vote basé sur une blockchain.

Le code comporte des fichiers de code décrits plus précisément dans le reste du rapport. À ces fichiers s'ajoutent un fichier de test (`tests.c`), comportant plusieurs fonctions de test, et un main réalisant une simulation de vote.

Pour exécuter les tests, compiler avec `make` et exécuter `./tests`.

Nous avons créé un constructeur par copie pour la plupart des structures pour minimiser les partages de pointeurs et simplifier la gestion de la mémoire.

2 Exercice 1

2.1 Question 2

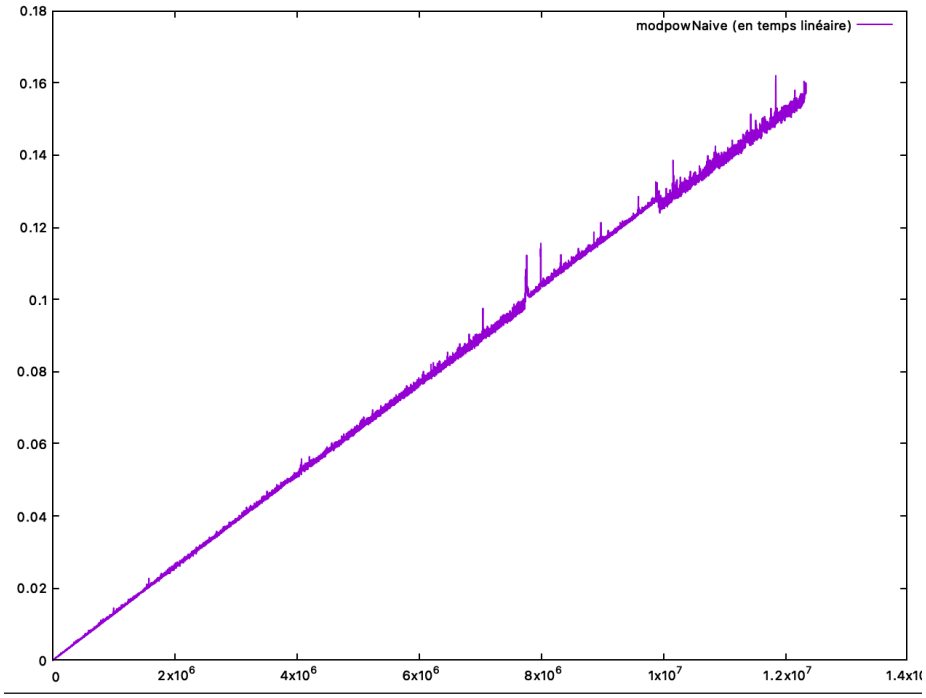
Le plus grand nombre premier que nous avons pu calculer en moins de 2 secondes est 239.317.339, calculé en 1.964 secondes.

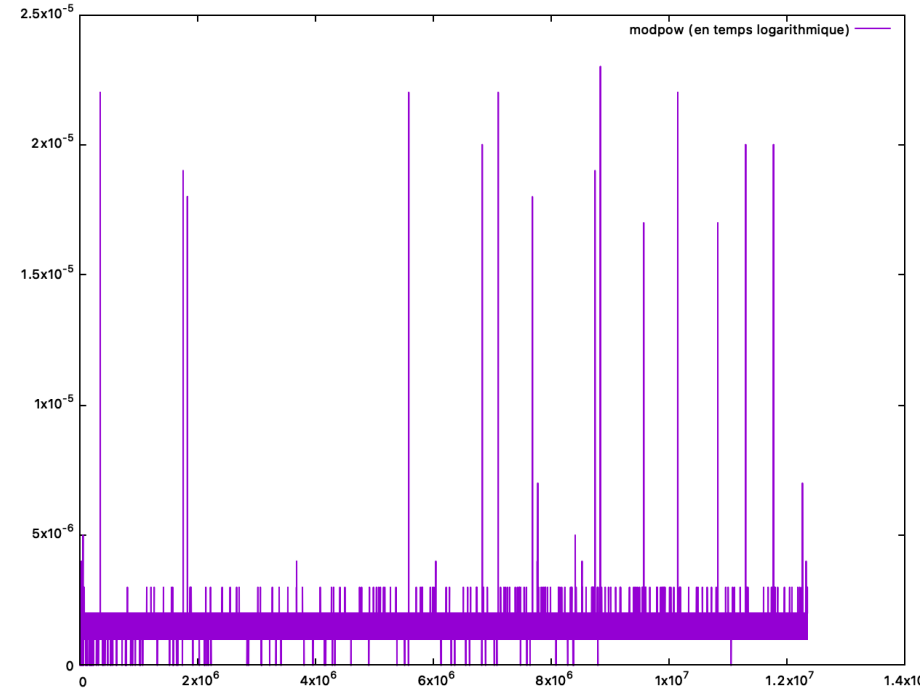
2.2 Question 3

L'algorithme naïf tourne en temps linéaire par rapport à l'exposant.

2.3 Question 5

Nous avons effectué 10.000 tests avec comme base 12, comme résidu 51 et un exposant de $1234i$, pour i variant de 1 à 10.000. Les graphes ci-dessous montrent l'esposant en abscisse et le temps d'exécution en ordonnée. La différence était trop grande pour pouvoir afficher les deux courbes sur le même graphe.





2.4 Question 7

On modélise les k tirages faits par un appel de l'algorithme par une suite de k éléments à valeurs dans $\{T, T^c\}$, où T indique que le nombre tiré est un témoin et T^c indique qu'il ne l'est pas. Si p n'est pas premier, la probabilité des témoins est de $\frac{3}{4}$. On cherche à calculer la probabilité de ne trouver aucun témoin après k expériences.

Les évènements "le $i-$ème tirage donne T^c" sont indépendants, donc $\mathbb{P}\left(\bigcap_{i=1}^{k} \textit{ le i-ème tirage donne } T^c\right) = \frac{1}{4^k}$.

La probabilité d'avoir un faux positif est donc inférieure $\left(\frac{1}{4}\right)^k$.

3 Exercice 2

Les fonctions de l'exercice 2 sont dans le fichier `rsa`. La fonction de test fournie est implémentée dans `test_rsa` dans le fichier `tests.c`.

4 Exercice 3

La structure `Key` est définie dans le fichier `rsa.h`. `protected` et `signature` sont définies dans `voting.h`.

Les fonctions de gestion de chaque structure sont définie dans le fichier associé.

La fonction de test fournie est implémentée dans la fonction `testEX3` du fichier de tests.

5 Exercice 4

Pour le tirage aléatoire des nc candidats, on commence par tirer nc entiers distincts entre 1 et nv. On génère alors les clés des candidats, que l'on stocke dans une table de hachage. Cela permet égalemement de se souvenir des indices qui ont déjà été tirés et donc de garantir qu'il n'y a pas d'électeur tiré deux fois pour être candidat.

Une fois les nc candidats choisis et leurs clés générées, on itère sur $[\![0, nv-1]\!]$, on crée les clés de chaque électeur non-candidat, on choisit la personne pour qui il va voter, on génère la déclaration de vote et on remplit les fichiers qui sont à remplir.

On utilise une table de hachage clé vers entier pour s'assurer qu'aucune clé n'est tirée plusieurs fois.

6 Exercice 5

Les fonctions de cet exercice sont dans le fichier `voting.h`.

7 Exercice 6

La table de hachage est implémentée dans `HashTable`. Les fonctions permettant de manipuler une table entier vers clé commencent par `htIk`, et `htKi` pour les tables clé vers entier.

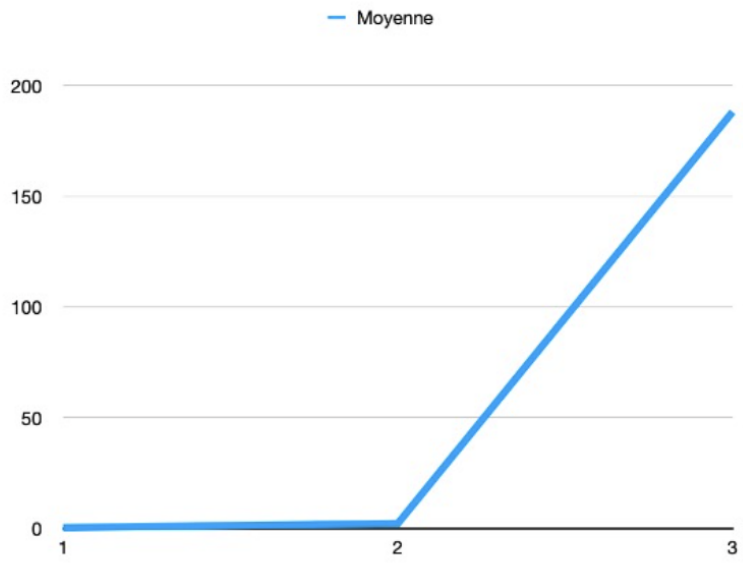
La fonction `computeWinner` est implémentée dans `voting`.

8 Exercice 7

Les fonctions de cet exercice sont implémentées dans `blockchain`, à l'exception des fonctions de traitement des hash, qui sont définies dans `Hash`.

8.1 Question 8

Le temps de calcul dépasse une seconde dès que $d = 2$, et le calcul devient irréalisable pour $d \geq 4$. Nous avons effectué des tests sur dix configurations différentes des champs d'un bloc affectant son hash. Le graphe suivant montre la moyenne des temps d'exécution, en seconde, pour $d indiqué en abscisse.



Le temps moyen est de 0.3 seconde pour $d = 1$, 2.2 secondes pour $d = 2$ et 188 secones pour $d = 3$.

9 Exercice 8

Les fonctions de cet exercice sont toutes dans le fichier `Arborescence`, à l'exception de `merge` (question 8) qui est dans `voting`, avec les autres fonctions de traitement général des listes de clés et de déclarations.

9.1 Question 8

`merge` s'exécute en temps linéaire par rapport à la taille du premier paramètre. Elle aurait pu s'exécuter en temps constant si l'on avait accès en temps constant à un pointeur vers le dernier élément du premier paramètre, ce qui aurait pu s'implémenter à l'aide d'une structure définie comme suit :

```
struct Cellule {
    Type contenu;
    Cellule suivant;
}

struct ListeChaînée {
    Cellule tête;
    Cellule dernierÉlément;
}
```

10 Exercice 9

Les fonctions de cet exercice sont dans le fichier `Simulation` et sont testées dans la fonction `test_voteSimulation` du fichier de test.

10.1 Conclusion

Nous listerons quelques avantages et inconvénients auxquels nous avons pensé :

- En supposant que la preuve de travail soit suffisamment longue à calculer, il est très difficile de frauder en injectant des faux blocs.
- Le comptage est plus rigoureux, et s'effectue instantanément.
- Les électeurs ne seraient plus obligés de se déplacer pour voter, ce qui pourrait augmenter le taux de participation (à condition d'assister les personnes n'ayant pas d'accès à internet ou de compétences suffisantes).

–

- Retour sur le premier point : on peut cependant imaginer une attaque de type 51%, ou des attaques par force brute pour usurper la clé d'un autre électeur.

- Il est difficile d'imaginer qu'aucune autorité régulatrice n'ait accès aux clés publiques et privées des électeurs. Cela implique donc un risque pour l'anonymat.
- Le processus est technique et difficilement compréhensible par la plupart des électeurs, ce qui peut diminuer la confiance.
- Cela pose un problème écologique substantiel du fait de la quantité d'énergie requise.

Il est donc difficilement envisageable d'utiliser ce système pour des élections ayant de grands enjeux : malgré les défauts de sécurité du système de vote par papier, son fonctionnement est clair et ses limites sont bien connues, ce qui en fait sûrement un meilleur choix.