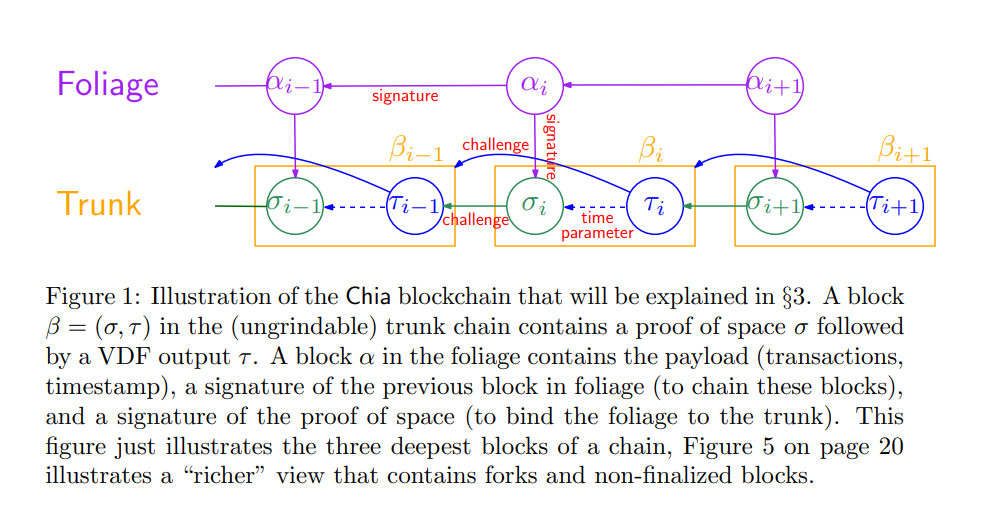
Chia and VDF (verifiable delay function)

Introduction

VDF是一個在區塊鏈的世界中加入延遲的重要工具，或是說一段真實世界時間經過的證明。VDF的特色是就算使用平行運算加速，VDF依然能夠保證一定時間的經過。VDF的output需要經過一連串的迭代計算完成，無數次的迭代使得電腦需要耗費一定時間計算，並且每次迭代的數值是由上一次迭代的輸出決定，因此很難被平行運算加速。

在Dan Boneh 提出了VDF概念以及形式化之後，Wesolowski (2018)和Pietrzak(2018)分別提出的兩種構造VDF的方法，兩個方法都是對一個未知階的群上做連續平方運算。Wesolowski的證明比較短、驗證更快，但是Pietrzak的構造中，生成證明的速度比較快。產生群的方式一般是使用RSA群，即生成兩個非常大且不相近的質數p, q，使得pq = N。在無法對N做因式分解的情況下，prover只能乖乖地對VDF進行連續平方的運算。使用RSA群的問題是需要一個trusted setup生成兩個p, q，而且不能被洩漏出去，否則安全性將大打折扣。在Dan Boneh對兩篇論文的調查中提到，使用一個虛二次數域的類群會是一個更好的方式。

Chia Network在Proof of Space 的基礎上加入 VDF，延遲農夫出快的時間，產生新的 Proof of Sapce and Time。Chia Network最後選擇使用Pietrzak構造的VDF。



Chia Network共有兩條鏈，一條是Trunk (樹幹)，一條是Foilage (樹葉)。Trunk中包含有proof of space以及VDF的輸出，Foilage中則有交易資訊。所有proof of space和VDF的挑戰都來自於前一個Trunk之中。

Group and Order

Group(群)是一個由集合以及二元運算組成的代數結構，舉例來說 (N, +)是最常見的群，叫做整數加法群。一個群必須滿足群公理 : 封閉性、結合律、單位元素、反元素。

封閉性:兩個群內的元素計算結果也會在群之中。

結合律: 對群內的元素a, b, c, (a\*b)\*c = a\*(b\*c)

單位元素:存在e使得 a\*e = e\*a = a

反元素: 對每個a 皆存在一個b使得b\*a = a\*b = e

Order(階): 一個群的元素個數。

Wesolowski 和 Pietrzak 使用的都是整數模n乘法群，常被記做 。這個群內的元素就是所有小於等於n並且與n互質的整數。群的階可以用歐拉函數，因為1, 3, 5, 7與 8 互質。

對於任一個質數p來說，。

若， 除了p的倍數以外，其餘都跟n互質。

若m, n 互質，。

從以上兩個條件，我們可以對N做因式分解並計算他的order

RSW Time Lock Puzzle

VDF的設計都是基於Rivest, Shamir and Wagner(1996)的time-lock puzzle問題構造而成。Time-Lock Puzzle的目的是加密一段文字使得這段文字在一定時間內都無法被解密完成。

Time Lock Puzzle 由 (N, x, T)構成，其中N = p\*q，T是時間參數，且。問題是去計算。 如果知道N的因式分解，亦即知道p, q，就可以把上述問題分解成兩個小步驟。

雖然步驟比較多，但是在T非常大的時候( )，可以節省很多時間，因為e會被限縮在0到N的範圍之間。如果不知道N的因式分解，那麼就只能乖乖的進行連續平方的運算。

以下舉個例子:

假設 x = 11, T = 8, N = 161=7\*23:

依照下面表格的操作知道最後答案等於95。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| T | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| y | 121 | 151 | 100 | 18 | 2 | 4 | 16 | 95 |

然而利用上面提到的兩次指數運算:

What is Verifiable Delay Function?

VDF是一個需要消耗一定時間運算的函數，但是一旦被計算完成之後，利用其output以及附帶的證明可以很快的進行驗算。VDF含有三個步驟的演算法:

其中

，G是一個未知階的有限群，H是一個雜湊函數。

的過程如下:

計算

計算證明

輸出

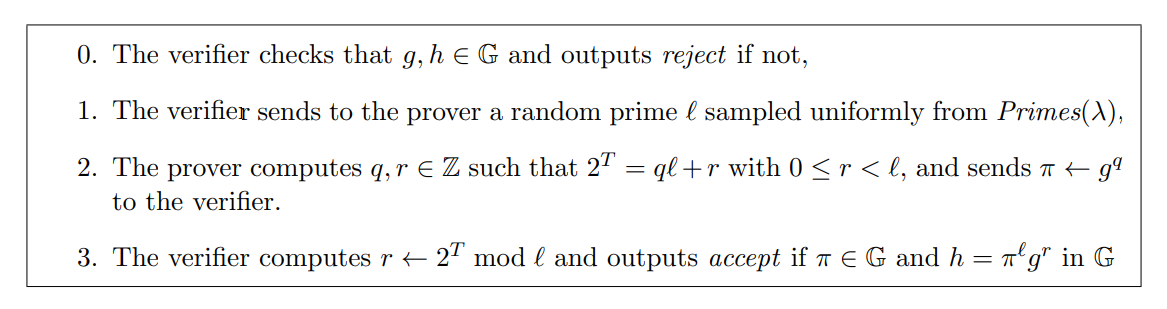
計算證明也需要耗費時間導致實際計算時間增加到，實作上。

注意到這邊Verifier是不知道G的生成方式，因此也無法利用因式分解進行速算法，只能透過證明去驗證Y是否正確。

輸出y 的正確性

為了讓這個問題更抽象化，先定義一些符號:

Wesolowski 論證



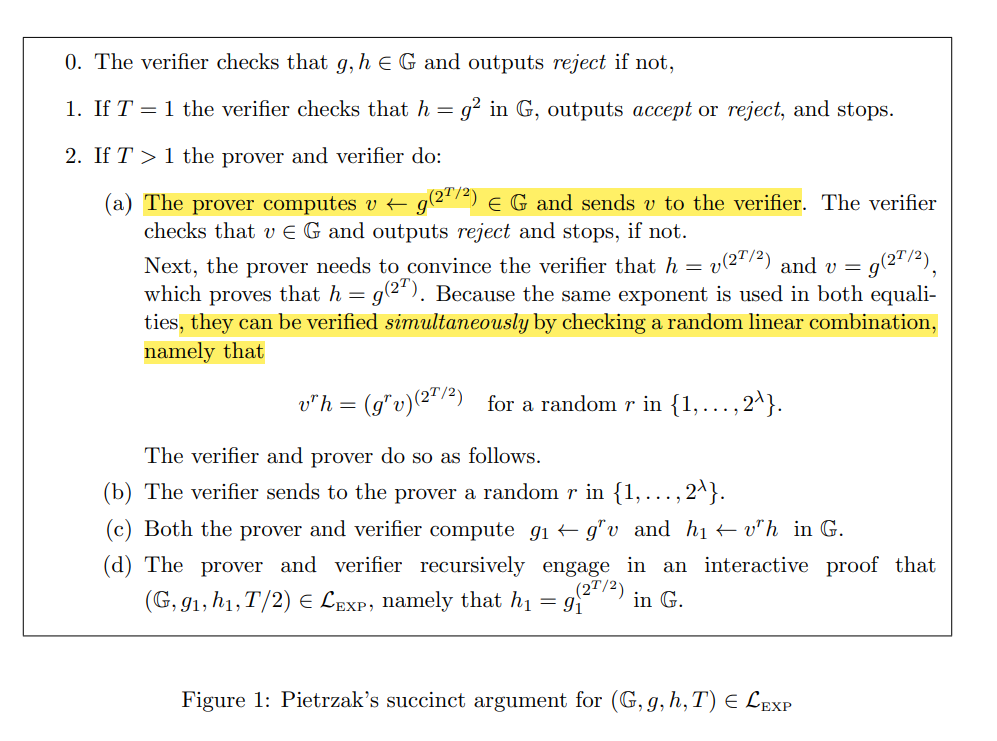
在證明過程中，prover需要計算的部分是

Verifier需要計算的部分則是

注意到prover需要計算q, r比起verifier只需要考慮同餘項需要更多運算，加上q涉及到2的T次方，是一個非常大的數字。

Pietrzak的論證

Pietrzak提出的構造方式相對複雜很多，是一個遞迴的證明過程。



這裡第一個重點是prover必須告訴verifier:

假設T = 4 :

Verifier需要計算:

Prover需要計算:

完成一次證明之後會產生新的一組tuple

這組tuple會在一次通過證明產生，一直持續到T = 1 為止。

我們也可以發現在每一次遞迴prover都需要計算v值:

在計算VDF得到的時候，VDF會儲存個元素 : 。

兩個VDF構造的差異

Wesolowski提供的proof更短，並且驗證這個proof也更快速。

Pietrzak的構造有兩個好處:

Pietrzak的優勢是生成proof的速度更快。

Pietrzak的VDF使用 adaptive root assumption來證明健全性，Wesolowski使用low order assumption。adaptive root assumption 比low order assumption 更強，也就是說如果adaptive root assumption成立那 low order assumption必定成立。

Reference

<https://blog.priewienv.me/post/verifiable-delay-function-1/>

<https://www.chia.net/assets/ChiaGreenPaper.pdf>

<https://eprint.iacr.org/2018/712.pdf>

<https://eprint.iacr.org/2018/627.pdf>