بسم الله الرحمن الرحیم

گزارش پروژه بلاکچین

دکتر محمد علی مداح علی

پوریا دادخواه

۹۶۱۰۶۴۸۵

SpaceMint : A Cryptocurrency Based on Proofs of Space

❋مقدمه

Space Mint یک پلت فرم بلاکچین Cryptocurrency است که برای اجماع نود ها به جای Pow از PoSpace (Proof of Space) استفاده می کند.

همان طور که در کلاس بحث شد proof of work اصلا روش بهینه و خوبی برای رقابت بین ماینرها برای ساخت یک بلوک نیست (توان مصرفی بسیار زیاد به اندازه برق مصرفی دانمارم برای تولید ۱۰ بلوک!! , TX/sec بسیار پایین و ...) و به همین دلیل پروتکل های زیادی ارائه میشوند که از جمله معروف ترین ها می توان به Proof of Stake و Proof of Space اشاره کرد.

ایده اصلی Proof of Space استفاده از فضای خالی به جای توان محاسباتی است.بسیاری از ما فضاهای خالی بدون استفاده ای در لپ تاپ ، هارد اکسترنال ، گوشی و .. داریم که میتوان با استفاده از این فضا ها کاندید برای تولید یک بلوک و دریافت جایزه آن بشویم.

بنابر این ایده اصلی ، پروتکل ها و روش های مختلفی برای پیاده سازی آن به وجود آمده اند و به تبع آن رمزارز های مختلفی نیز ساخته شده اند که Space Mint یکی از این رمزارز هاست.

در این مقاله به توضیح راه حل های اصلی این پروتکل برای حل مشکلات بنیادی بلاکچین ها ، چالش ها ومشکلات روبه روی آن و توضیح دقیق تر روش های اجماع و انتخاب ماینر ها می پردازیم و در آخرکاستی های مقاله این رمزارز و پیشنهادهایی برای بهبود آن می پردازیم.

❋مزایا و چالش های PoSpace

§ در ابتدا به اختصار به مهم ترین مزایای PoS نسبت به مشکلات PoW میپردازیم:

• **زیست محیطی** :همان طور که گفته شد برای تولید یک بلوک در بیت کوین انرژی مصرفی کل شبکه بسیار زیادی مصرف میشود که عملا به جز فرد برنده تمام انرژی فوق به هدر می رود.ولی در PoS در ابتدا با مقدار دهی اولیه مقداری از فضا به شبکه توسط ماینردرادامه برای تولید هربلوک تنها به میزان بسیار ناچیزی محاسبه نیاز هست و دیگر هیچ.

• **اقتصادی**: در بیت کوین هر ماینر برای تولید یک بلوک نیاز به مصرف زیادی برق دارد که هزینه آن اصلا کم نیست و به همین دلیل حتی باید دید که آیا اصلا نسبت به جایزه آن به صرف است یا خیر! هم چنین با توجه به سخت تر شدن تدریجی difficulty به مرور زمان نیاز به پروسسور های پیشرفته تر ، تعمیر و نگه داری آن ها هست !

اما در این روش پس از اختصاص دادن اولیه حافظه که همان یک بار انجام میشود در ادامه تنها هزینه های حاشیه داریم که بسیار ناچیز اند.

• **غیرمتمرکز بودن!** : همان طور که در شبکه بیت کوین بررسی شد با توجه به difficulty در حال حاضر احتمال موفق شدن یک ماینر عادی با یک توان محاسباتی معقول بسیار ناچیز(در حد ۱۰ سال یک بار !!) است. این امر باعث تشویق ماینر ها به تشکیل گروه های چند نفره برای افزایش توان محاسباتی خود میکند چرا که نوع اثبات چالش آن قابل موازی سازی است و میتوان یک کار دشوار را بین چند نفر تقسیم کرد و همزمان آن را جلو برد .بدین ترتیب در حال حاضر حدود ۱۶ mining farm بزرگ اصلی فعال هستند که توان بسیار زیادی از شبکه را دراختیار دارند. این دقیقا خلاف هدف اصلی یک شبکه بلاکچینی است و در حال متمرکز شدن است!

اما در Pos این انگیزه و مشکل وجود ندارد چراکه باید مقداری دیتا در یک حافظه ذخیره شود و ازمیان آن ها بخشی به عنوان اثبات ارائه شود.درواقع این اثبات قابلیتcomputing parallel ندارد...

§ حال به چالش های اصلی این پروتکل و راه حل های مختصر هر کدام می پردازیم و برخی ازراه حل های ارائه شده را در طول مقاله بررسی میکنیم.

• **تعاملی بودن پروتکل(interactive proofs) :** همانند شبکه بیت کوین در این شبکه نیز برای اثبات کار انجام شده(حافظه اختصاص داده شده) باید prover وverifier با رد و بدل کردن چند پیام این کار را انجام دهند. حال آن که در PoS این شبکه به تعاملات بیشتری نیز نیاز است (چرا که اثبات شامل دو فاز است) در نتیجه چالش های بیشتری در این زمینه مثل هماهنگ بودن دو نود در یک شبکه غیرمتمرکز آسنکرون خواهیم داشت.

*__راه حل__؟ استفاده از fiat-shamir به جای public بودن پیام ها + ذخیره کردم پیام ها در شبکه و استفاده از نوع خاص تراکنش برای verify کردن اثبات ها

• **انتخاب فرد برنده! :** در شبکه بیت کوین برنده بلوک بعدی به صورت احتمالاتی متناسب با میزان توان محاسباتی دراختیار هر ماینر تعیین میشد ونهایتا فردی که توانسته بود به هدف چالش برسد و بلوک بعدی را تولید کند ، با انتشار بلوک خود و بدون نیاز به تعامل اضافی از برنده بودن خود مطلع می شد! (به جز موارد خاص مثل fork)

حال اینجا نیز برنده متناسب با میزان فضای ماینر مشخص می شود ولی باید سازوکاری طراحی شود که فرد برنده بداند که برنده شده است یا خیر!

*__راه حل__؟ بر اساس فاکتور های موثر(حافظه ماینر ، میزان مشارکت کردن در زنجیر و ..) برای هر بلوک یک تابع احتمالاتیv = Quality (B)تعریف میکنیم که طبیعتا در اصل متناسب با میزان حافظه اختصاص داده شده است. در هر استپ تولید بلوک ، بلوکی که بیشترین v (کیفیت) را اشته باشد برنده میشود.

• **Nothing-at-Stake:** با برداشته شدن پیچیدگی محاسباتی برای اثبات ادعای چالش انجام شده هر ماینر با مشکلاتی رو به رو می شویم که به آن ها Nothing-at-Stake problems میگوییم.دو نمونه از این ها داریم:

- به دلیل نداشتن هزینه برای ارائه دوباره اثبات یک ماینر میتواند یک اثبات را به جای یک شاخه(chain) در چند شاخه فعالیت میکند (تا هر کدام که در آینده به عنوان زنجیر اصلی انتخاب شد او ضرر نکند..)
- با آسان بودن اثبات یک فرد میتوان یک اثبات را با تغییر دادن برخی محتویات بلوک چند بار در یک زنجیره استفاده کند و چند بلوک را انتشار دهد.!(Block-Grinding)

از مهم ترین مشکلاتی که این نوع حمله ها دارند می توان اشاره کرد به:

- سرعت شبکه را به شدت می کاهند چرا که verify کردن آن ها و تشخیص متقلب بودنشان ، بخشی از توان شبکه را به خود اختصاص می دهد
- انگیزه بسیار قوی برای ماینرهاست که از پروتکل پیروی نکرده و آنچه را که برای خود سود بیشتری به همراه دارد را انجام دهند.selfish-mining
- احتمال double-spending را افزایش میدهد!. در block-grinding دیدم یک ماینر میتواند چند بلوک را ارائه دهد با یک اثبات(یعنی میتواند بخشی را مخفی کرده و بعد پشت هم انتشار دهد)

*__راه حل__؟ این اساسی تر از بقیه بوده و راه حل های پیچیده تری می طلبد.

به عنوان مثال برای مشکلblock-grinding مطمئن می شویم که اثبات ارائه شده در یک استپ unique است. در این جهت برای اثباتchallenge ای را ارائه میدهیم که وابسته به سابقه زنجیره است . همچنین از دو زنجیرهproof chain (اصلی ؛ برای چک کردن صحت) و signature chain(برای verify کردن و اتصال تراکنش ها به زنجیر اصلی) استفاده میکنیم.

- **challenge Grinding** : مشکل دیگر همانند PoStake این است که انتخاب ماینر های بلوک های بعدی وابسته به بلوک های قبل می باشد و ماینر فعلی میتواند الگوریتم برای تولید بلوک داشته باشد که درآینده نیز به نفعش باشد و در واقع زنجیره را بایاس کند.

*__راه حل__؟ برای این مشکل هم یکی از پارامتر های تابع کیفیت بلوک را تعداد بلوک های دنباله بلوکی تولید شده توسط آن ماینر قرار می دهیم به طوری که احتمال افزایش تعداد آن به صورت نمایی کاهش می یابد.

1*پروتکل های مشابه

اکنون برخی از شبکه هایی که ازPos و موارد مشابه استفاده میکنند را معرفی میکنیم و امکان استفاده از ایده های آن هارا بررسی می کنیم

- Proof of storage/retrievability : verifier یک فایل به prover میفرستد و سپس یک اثبات از آن فایل میخواهد که نشان میدهد فضای لازم را داشته یا نه. در این الگوریتم فضا و محاسبات **V** در اردر polylogarithmic **P** است.
- Proof of secure erasure (PoSE) : در این روش از تابع های یکبار مصرف(one time comutable functions) استفاده میشود . مثلا فایل مورد نظر کاملا پاک شود! ولی از آن جایی که اثبات در PoS شبکه ما ۲ فاز را شامل می شود نمیتوان از این روش استفاده کرد.

- Permacoin : این الگوریتم بیس و پایه ی Powدارد و ایده اصلی آن انجام یک کار مفید به جای اتلاف فضا و پردازش برای اثبات است. مثلا یک شبکه بلاکچین میتواند به عنوان یک archive یک دیتا بزرگ (مثل اطلاعات تاکسی های اینترنتی ... , uber) مصرف شود. ولی در PoSشبکه ما این اتفاق نمیافتد و همان روش سنتی اجرا میشود چرا که به عمد قصد استفاده از هیچPow نداریم.

- Burstcoin : ایده اصلی این شبکه PoSاست ولی در آن میتوان trade-off بین فضا و محاسبات را انجام داد و به جای ارائه فضای زیاد مقداری محاسبه ارائه کرد.

- Chia network : این پروپوزال یکی از جدیدترین و محبوب ترین هاست. در این روش هر چه "کیفیت " یک بلوک بیشتر باشد ، نهایی شدن آن سریع تر میشود. به طور کلی الگوریتم های استفاده شده کاملا متفاوت از ماست. از دلایل محبوبیت آن می توان به عدم نیاز به سنکرون بودن با حفظ بازده شبکه اشاره کرد.

2*Proof of Space in SpaceMint

شمای PoS در شبکه :

در این شبکه در ابتدا هر ماینر ابتدا موقع عضو شدن به مقدار فضای موردنظر برای اختصاص دادن به شبکه را دیتا Sγ به اندازه N بیت ذخیره میکند و verifier (V) یک کسری از آن(γ)را دریافت و ذخیره میکند(فاز اولیه)

سپس در فاز اجرایی execution phase ,V یک چالش یه prover (P) میفرستد و او جواب a را میدهد.

حال به تبیین این پارامتر ها میپردازیم.

دیتایی که V در اختیار P قرار میدهد یک "hard-to-pebble Graph"* است که هر نود آن به این صورت تعریف می شود:

$$l_i := hash(u, i, l_{p1}, \dots, l_{pt})$$

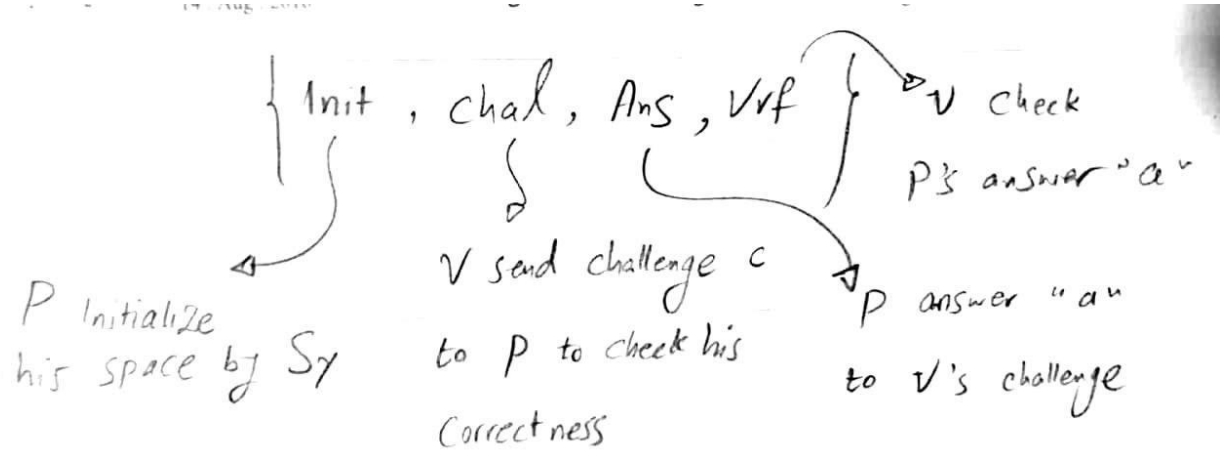
که در آن u ثابتی است که آن را به طور خاص در شبکه خودمان V تعیین میکند و P باید براساس آن دیتا را ذخیره و اثبات کند. و $l_{p1}, \dots, l_{pt}$

مقادیر متناظر parent node l هستند. بدین ترتیب برای محاسبه li میبایست ابتدا مقادیر parent آن را حساب کرده باشیم.

*به طور خلاصه گرافی است که تعذاذ پرنت های هر نود زیاد باشد و وابستگی هر نود به سابقه اش ساده نباشد.

برای این هدف دو نوع گراف hard to pebble می توان استفاده کرد.
در نوع اول P حداقل باید $\Omega(|V|/\log(|V|))$ از حافظه مورد نیاز را بین دو فاز را داشته باشد(که طبیعتا $|V|$ اندازه گراف مورد نظر است و Ω حد بالای کران است) و یا همین مقدار حافظه را در حین فاز اجرایی.
درنوع دوم P باید $\Theta(|V|)$ حافظه بین دو فاز و یا $\Theta(|V|)$ زمان در فاز اجرایی مصرف شود.(که Θ دقیقا همان اردر $|V|$ با اختلاف یک ضریب است.)

روند اثبات این طور است که الگوریتم های {Init, Chal, Ans,Vrfy } را با استفاده از الگوریتم های پایه ای al-1 , al-2 , al-3 که صفحه بعد تعریف کردیم در دو فاز بین P, V اجرا می شوند.



initialization phase: al-1 (P commit the space storing Sγ) + al-2 (P prove the commitment by γ)

Execution phase : V checks the proof by challenge c;

به جز ارسال u توسط verifier الگوریتم معرفی شده غیر تعاملی است و از چالش های مربوط به آن دور است.

Algorithm 1 Space commit

Common input: A hard-to-pebble graph G with n nodes and a function hash: $\{0,1\}^* \to \{0,1\}^L$.

1. $\mathcal{P}$ generates a unique nonce μ and then computes and stores $(\gamma, S_\gamma) := \mathsf{Init}(\mu, n)$, and sends the nonce[5] μ and the commitment γ to $\mathcal{V}$. S_γ contains the labels of all the nodes of G computed using Eq. (1) and γ is a Merkle-tree commitment to these n labels. The total size of S_γ is $N = 2 \cdot n \cdot L$ (graph + Merkle tree).

Algorithm 2 Prove commit

Initial state: $\mathcal{V}$ holds commitment γ and nonce μ; $\mathcal{P}$ stores S_γ and μ. Both are given the challenges $c = (c_1, \ldots, c_{k_v})$ to be used.

1. $\mathcal{P}$ computes openings $b := (b_1, b_2, \ldots)$ of all the labels of the nodes $\{c_i\}_{i\in[k_v]}$ *and of all their parents* and sends them to $\mathcal{V}$. This is done using Ans where $\mathsf{Ans}(\mu, S_\gamma, c)$ returns the Merkle inclusion proof of label l_c w.r.t. γ.
2. $\mathcal{V}$ verifies these openings using Vrfy, where $\mathsf{Vrfy}(\mu, \gamma, c, a) = 1$ iff a is a correct opening for c. It then checks for all $i = 1, \ldots, k_v$ if the label l_{c_i} is correctly computed as in Eq. (1).

Algorithm 3 Prove space

Initial state: $\mathcal{V}$ holds commitment γ and nonce μ; $\mathcal{P}$ stores S_γ and μ. Both are given the challenges $c = (c_1, \ldots, c_{k_p})$ to be used.

1. $\mathcal{P}$ computes openings $\{a_i := \mathsf{Ans}(\mu, S_\gamma, c_i)\}_{i\in[k_p]}$ and sends them to $\mathcal{V}$.
2. $\mathcal{V}$ verifies these openings by executing $\mathsf{Vrfy}(\mu, \gamma, c_i, a_i)$.

Mining in SpaceMint*3

در ابتدا ماینر که وارد شبکه میشود pub key , secret key خود را ساخته و با آن و تابعInit مقادیرموردنظر حافظه و فرکشن آن را تولید می کند

$$(\gamma, S_\gamma) := \mathsf{Init}(pk, N) \ .$$

سپس با (Pk, γ) منتشر می کند تا هم V بتواند آن را چک کند و این که از grinding attack جلوگیری شود.

حالا بعد از مقدار هایی اولیه با پیروی از الگوریتم های تعریف شده در ۲ به ترتیب زیر تلاش برای تولید بلوک جدید میکند:

۱. هش بلاک قبلی + حساب کردن challenge . این چالش برخلاف بیت کوین که صرفا بر اساس بلوک قبل بود براساس بلوک Δ - I تعریف می شود تا از حمله grinding که یک نوع nothing-at-stake است در امان باشیم! خروجی چالش دو عدد رندوم بزرگ Sv,Sp است

۲. سپس براساس پارامتر های n,Kp,Sp ، Chalدر al-3 را جواب میدهد و a را به عنوان جواب حساب میکند.
۳. نهایتا Quality block را نیز محاسسه میکند(جلوتر پرداخته شده)
۴. اگر Q قابل قبول داشت بلاک را ساخته و با اثبات al-2 منتشر میکند. حال سایر ماینر ها کار V را انجام میدهند.

*برخلاف Pos اصلی ما صرفا نیاز بهal-2 داریم نه 1,2 که این باعث بازدهی بهتراست.(چرا که درستی پاسخ b بخش ۱ الگوریتم را نیز به صورت احتمالاتی بالا تضمین میکند.

Blockchain Format*

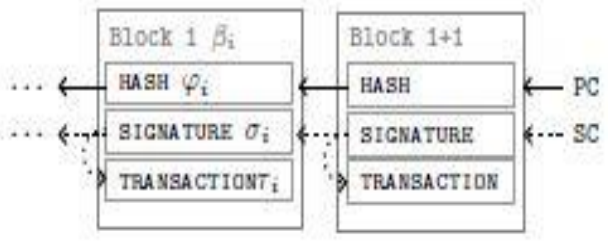
هر بلاک از سه قسمت تشکیل شده

ς Hash sub-block که شامل امضای هش بلاک های قبلی می باشد و proof chain که زنجیر اصلی ارتباط بلوک هاست را تشکیل میدهد. از این زنجیره برای اثبات صحت ماینر ها استفاده می شود

ϯ Tx sub-block شامل تراکنش های بلاک است(۳ نوع تراکنش داریم. عادی برای پرداخت – تراکنش های اثبات – تراکنش های جریمه)

$$ctx = (\text{commit}, txId, (pk, \gamma)) \qquad ctx = (\text{penalty}, txId, pk, prf) \qquad ctx = (\text{payment}, txId, \vec{in}, \vec{out}) \ .$$

σ signature sub-block ک شامل امضای تراکنش هاست و sig sub های قبلی را نیز امضا میکند. این زنجیره signature-chain است که باعث میشود از double-spend و حمله های نظیر آن جلوگیری شود.



❊Quality of Proofs and Chains

Quality of a proof.1.

یکی از تعاریف خلاقانه و جالب این مقاله میتوان به نحوه تعریف کیفییت یک proof اشاره کرد.

از میان proof های π1 و π2 و .. و πm قرار است آنی به عنوان ماینر بلوک بعد انتخاب شود که احتمالاتی شانس آن متناسب با فضای اختصاص داده شده اش بیشتر باشد .
بنابراین باید از یک تابع رندوم هش استفاده کنیم که برای هر i احتمال اینکه(H(Q i از سایر(H(Qj ها بزرگ تر باشد با خودQ ها باشد و از دید احتمالی برابر میزان بیت های i

$$\Pr_{\text{hash}}\left[\forall j \neq i \,:\, \text{Quality}(\pi_i) > \text{Quality}(\pi_j)\right] = \frac{N_{\gamma_i}}{\sum_{j=1}^{m} N_{\gamma_j}},$$

نسبت به کل بیت ها باشد.

در ادامه این هش ها را در [0,1] مپ میکنیم توسط Dn که یک توزیع احتمالاتی است و سپس مکس آن ها را به عنوان ماینر انتخاب می کنیم.

$$\text{Quality}(pk, \gamma, c, a) := D_{N_\gamma}(\text{hash}(a))$$

با قوانین احتمال و وارون CDF درمیابیم که خروجی را میتوان از رابطه زیر به دست آورد

$$D_{N_{\gamma_i}}(\text{hash}(a_i)) := \left(\text{hash}(a_i)/2^L\right)^{1/N}.$$

که L تعداد بیت های خروجی تابع هش فوق می باشد.

Quality of a chain.2

از بین شاخه های به وجود آمده باید سازوکاری مشابه longest chain در بیت کوین تعبیه کنیم که بهترین شاخه را انتخاب کنیم. مشابه قبل طبیعتا شاخه ای که فضای بیشتری در شبکه قرار میدهد برای ما مفید تر است. بنابراین برای هر شاخه $(\varphi_0, \varphi_1, \varphi_3, \ldots, \varphi_i)$ برای هر بلوک آن Q آن را حساب کرده و تابع زیر این گونه تعریف می کنیم که میزان فضای لازم برای این که با احتمال بیشتر از ½، Q بالاتری نسبت به Q بلوک فوق داشته باشد .

$$\mathcal{N}(v) = \min\left\{N \in \mathbb{N} \,:\, \Pr_{w \leftarrow D_N}[v < w] \geq 1/2\right\},$$

از طرف دیگر میخواهیم تاثیر هر بلاک در کیفیتQPc کل آن شاخه به میزام موثر بودن آن نیز ربط داشته باشد. به طوری که بلاک هایی با محتوایی بهتری هستند وزن بیشتری داته باشند. این کار باعث جلوگیری از Grinding attack می شود(که یک ماینر در خفا تعداد زیادی بلوک با Q بالا ولی بی محتوا تولید کند و آن ها را در زمان مناسب انتشار دهد. به همین جهت یک فاکتور دیگر Λ^{i-j} در QPc موثر است. نهایتا آن را ازین رابطه به دست می آوریم:

$$\mathsf{QualityPC}(\varphi_0, \ldots, \varphi_i) = \sum_{j=1}^{i} \log(\mathcal{N}(v_j)) \cdot \Lambda^{i-j}$$

❋Nothing-at-stake problems in details

همان طور که در ابتدا بحث شد ، برداشتن پیچیدگی محاسباتی مشکلات زیادی به وجود می اورند که منجر به کاهش سرعت شبکه ، افزایش توان adversary نسبت نود های صادق و از همه مهم تر double-spending می شود.

اکنون چند مورد از این مشکلات را همراه با راه پیشنهادی ارائه میدهیم؛

۱.Grinding-block : صورت مسئله در ابتدا بحث شد؛ هر بلاک به قبلی وابسته است و با آسان بودن اثبات ؛ تولید چندین بلاک با یک اثبات وانتخاب آن که در آینده سود بیشتری دارد.

؟راه حل : در SpaceMint به جای استفاده از یک chain دو چین داریم. ابتدا در چین اصلی (φ proof chain) صحت اثبات بررسی می شود و با رجوع بع بلاک موردنظر یکسان بودن تراکنش ها و درواقع unique بودن بلاک چک می شود.

۲.Mining on multiple chains.

باز بحث شد با آسان بود پیچیدگی ماین کردن در چند شاخه به صرفه و مفید است !باید جلوی این کار گرفته شود.

؟راه حل:اگر در چالش ماینر ها به جای هش بلاک قبل ، از آن ها هش Δ بلاک قبل را بخواهیم که در آن Δ نسبتا بزرگ است باعث میشود احتمال اینکه او بتواند Δ بلاک تولید کند و به جای زنجیره اصلی قالب کند کم میشود.به طور دقیق تر این مسئله دو حالت دارد . fork تولید شده بیشتر/کمتر از Δ داشته باشد. اگر کمتر باشد که هش فوق با هش زنجیر اصلی یکسان است و تقلبی بودن بلوک فعلی او آشکار می شود و با تراکنش جریمه! نصف جایزه به ماینر تشخیص دهنده و نصف دیگر از بین می رود.

اما اگر بیشتر باشد توانایی تشخیص نداریم و او میتواند به هدفش برسد. اما احتمال این کار بسیار کم است.

۳..Grinding challenges :

جمع کردن یک سری بلاک درخفا و آزاد کردن آن ها به عنوان زنجیره اصلی . نیمه اول زنجیر فیک Qبالا داشته ولی بی محتوا ولی نیمه دوم نیاز به Q بالا نیست ولی در عوض تراکنش های مطلوب (مثل... ,double-spend)

؟ راه حل: ابتدا مشابه بالا Δ را استفاده میکنیم . همچنین ۱) در تعریف Q به جای جمع تک تک مولفه های یک زنجیره ضرب آ نها رامیگیریم که باعث می شود کار ماینر خراب کار برای ساخت یک دنباله فیک با طول Δ 2 سخت تر شود. هم چنین۲) همین چالش c که اکنون برای بلاک فعلی استفاده میکنیم را برای δ اینده هم مطرح می کنیم تا خراب کار هرچه قصد طمع بیشتری داشته باشد ، نمایی کار سخت تری برای تولید بلاک high-Quality داشته باشد.

❋تعیین پارامتر های شبکه

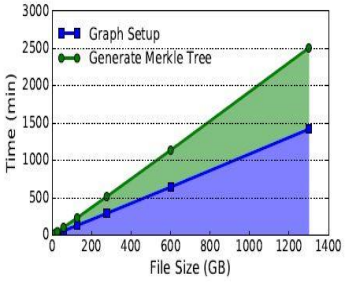
Δ :طبق استدلال های بالا هرچه عدد بزرگ تری باشد برای جلوگیری از **Grinding challenges** بهتر است.

اما ازطرفی بنا بر سایر ویژگی های شبکه بهتر است کوچک باشد. با trade-off بین این دو نتیجه گرفته شده $\Delta=50$ بهتر است.

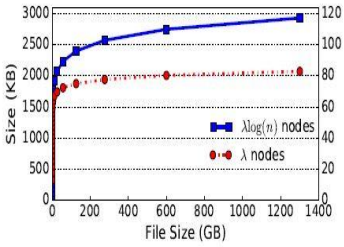
Frequency: تولید یک بلاک زمان زیادی نمیگیرد(30 s) و برای تثبیت شدن یک بلاک لازم بود که Δ بلاک بگذرد پس 50*30 sec اما انتظار داریم Tx/sec بالایی داشته باشیم و تنها محدود به ویژگی های فیزیکی بلاکچین باشیم! در کل f = 1 min بهترین نرخ را از لحاظ امنیتی و ریت تراکنش می دهد.

Λ: با در نظر گرفتن rate افزایش حافظه میزان تاثیر گذاری را تعیین میکنیم.در حالت معمولی ضریب ۰,۹۹۹ خواهد داشت.

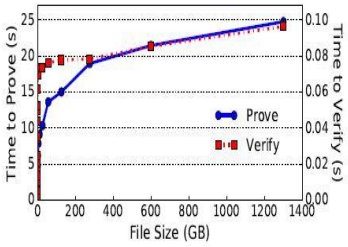
Evalution*



(a) Time to initialize space.



(b) Proof size for varying space sizes with $\lambda = 30$. The left and right vertical axes represent proof size when opening $\lambda \log(n)$ and λ nodes (respectively).



(c) Time for a miner to prove and verify when $\lambda \log(n)$ nodes are opened for $\lambda = 30$.

Game Theory of SpaceMint

یکی دیگر از کار های جالب و خلاقانه این مقاله پیاده سازی و شبیه سازی شبکه با n نود است. در این بازی n ماینر بازی می کنند که هر یک در هر استپ(ماین شدن هر بلاک) هر نود بهترین استراتژی خود را نسبت به سابقه کار های خود و انچه در شبکه

به ثبت رسیده ، را انجام میدهد تا utiliy خود که همان reward حاصل از ماین شدن هاست را ماکس کند. واضح است که لزومی ندارد هر نود از اقدامات سایر نود ها اطلاعی داشته باشد.

همان طور که از تعریف برمی آید این بازی extensive است و نه one shot

در گام اول ثابت میکنیم این بازی Nash equibliurm است(با فرض One shot بودن اضافه) و بهترین استراتژی تولید و انتشار بلاک است و نه selfish mining که در خفا بلاک تولید کنیم و زنجیر فیک تولید کنیم.

نهایتا در گام دوم این اثبات را برای sequential rational تعمیم می دهیم که نیاز به فرض one shot ندارد

معدلات و اثبات هل و تعاریف در مقاله آمده که در صورت نیاز حضوری شرح داده خواهد شد.

** مزایا و معایب کلی مقاله))

به نظر من از مزایای اصلی این پروتکل نسبت به مشابهانش میتوان گفت:

- تعریف تابع کیفیت رندوم هش در هین حال متناسب با حافظه
- شبیه سازی ریاضی شبکه در فضای انتزاعی با استفاده از game theory

معایب نیز می توان موارد زیر را نام برد:

- میزان Tx/sec همانند بیت کوین میزان قابل توجهی برای ما نیست!

- حمله های کمی برای شبکه بررسی شدند که حتی برای برخی پاسخ همه جانبه داده نشده است
- میتوان پروتکل را طوری تغییر داد که از شبکه بتوان به عنوان data archieve استفاده کرد.

نهایتا برای بهبود پروتکل می توان موارد گفته شده در معایب بالا را همانند Postake تغییر داد.

با استفاده از سیستم دانایی صفر سایر حمله های ممکن را هم پوشش داد با هزینه کمتر.

استفاده از trade-off بین فضا و محاسبه میتواند آزادی عمل بیشتری میدهد.

---------- پایان