Tezos: 自己修正暗号台帳
ポジションペーパー
日本語訳 *

L.M Goodman

2014 年 8 月 3 日

「資本家に任せよ。」
— Pierre-Joseph Proudhon

概要

分散暗号通貨 Bitcoin の普及は代替暗号通貨、いわゆる「アルトコイン」の誕生を引き起こした。Ethereum、CryptoNote、Zerocash は暗号通貨技術の発展にそれぞれ独自の貢献を行なった。ほとんどの代替通貨が独自のイノベーションのアイデアを持っている一方、それらは他通貨のアイデアを利用して自通貨を成功に導くすべを持っていない。暗号通貨技術発展が萎縮してしまうこの可能性を打破するため、我々は汎用的で自己修正可能な暗号台帳である Tezos を提案する。

Tezos はブロックチェーンに基づいたプロトコルならば何でも選択し採用することができる。その開始時プロトコル[*1]には、**修正手順自体への修正も含む**、ステークホルダー (通貨保有者) によるプロトコル修正承認の手順が定められている。Tezos のアップグレードは段階的に行われる。修正はまずテスト環境に適用される。もしそこで問題が発見された場合は、ステークホルダーはその修正を廃案とすることができる。

Tezos の思想は Peter Suber の Nomic[1] というルールを自己修正できるゲームから発想を得ている。

この論文では、Tezos の利点、proof-of-stake システムの採用、実装プログラミング言語としての OCaml の採用、について説明したい。

---

* この文書の最新版は `https://gitlab.com/dailambda/tezos-papers/blob/ja/pdfs/position_paper_ja.pdf` にある。翻訳内容に関する問い合わせについては訳者の古瀬 淳 (`jun.furuse@tezos.or.jp`) まで。

*1 訳註: 原文は”seed protocol”。一般的ではないので、開始時、とした。

目次

1 動機

Tezos の開発において、我々は次のあげる Bitcoin[2] の 4 つの問題を解決することを目標とした:

- 「ハードフォーク」問題、つまり、Bitcoin が調整問題[*2]により動的に技術革新、更新ができない点。
- Bitcoin の proof-of-work システムによって引き起こされるコストと中央集権化の問題。
- スマートコントラクトを他のブロックチェーンに押しやることとなった Bitcoin トランザクション言語の表現力の限界。
- 暗号通貨を実装するにあたってのセキュリティ問題。

1.1 プロトコルフォークの問題

1.1.1 技術革新への追随

Bitcoin の成功をきっかけに、多くの開発者や起業家たちが代替暗号通貨「アルトコイン」を発表した。アルトコインのうちいくつかは Bitcoin のオリジナルのソースコードから劇的に分化したものではなかった[*3]が、

[*2] 訳註: ゲーム理論の用語。Coordination problem。
[*3] はぁ、なんと非創造的な...

その他いくつかは興味深い改良を提案した。たとえば、Litecoin は memory-hard proof of work 関数[*4]とより短いブロック確認時間を導入した。Ethereum は状態を持つコントラクトとチューリング完全なトランザクション言語を開発した [3]。より重要な貢献としてプライバシー保護機能のあるリング署名 (CryptoNote[4]) や SNARK を使った追跡不可能なトランザクション (Zerocash[5]) などがあげられる。

アルトコインの登場はソフトウェアイノベーションの巨大な競争を産み出した。しかしながら、このハイエク型成長[*5]のチアリーダーたちは基本的な点を見逃していた: 暗号通貨が貨幣としては機能するためには、それが富の安定した保有所有手段となる必要がある。台帳内の技術革新は、通貨に価値を与えるネットワーク効果を持続させることで富を保護する。

乱立したアルトコインの問題点を説明するために、暗号通貨とスマートフォンを対比してみよう。スマートフォンを買うとき、消費者は、たとえば、音楽プレーヤー、Email 機能、友達にメッセージを送れるとか、電話をかける、などの機能に対価を支払う。

数週間ごとにスマートフォンの新モデルが発表され、それらはしばしば機能が強化されている。古いモデルを持っている消費者は最新モデルの機能をうらやましく思うかもしれないが、新しいスマートフォンの登場によって古いスマートフォンが使えなくなるようなことはない。

しかし、もし最新のスマートフォンが古いモデルと通信できなかったとすれば、この力学は変化するかもしれない。もし多くのスマートフォンのモデルが隔たりなく相互使用できなかったとすれば、それぞれのスマートフォンの価値は同じモデルを持っている人々の数にまで減少してしまうことだろう。

互いが非互換なスマートフォンと同じ辛い運命を暗号通貨はたどっている。つまり、暗号通貨の価値はネットワーク効果、もしくはその暗号通貨の価値を見出しているユーザーの数に依存している。つきつめると、ある暗号通貨の外で技術革新が発生した場合、その革新は十分なネットワーク効果を産み出すことなく気づかれず捨て去られるか、成功したとしても、その今や陳腐化した暗号通貨が保持する価値を毀損してしまうことになる。

サイドチェーンは新しい通貨の価値と Bitcoin を連動させることで双方向交換を可能にして Bitcoin との互換性を保ちながら技術革新を進める試みだ。しかし、残念ながら、Bitcoin とは本質的に異なるプロトコルについて適用できるほどの柔軟性があるかは不明だ。今のところ、現実的な代替手段はプロトコルをフォーク (分岐) させるくらいしかない。

1.1.2 フォークの経済学

フォークの経済学について理解するには、まず、通貨の価値は主として社会合意であるということを理解する必要がある。暗号通貨をそのルールと台帳とに同一視してしまいがちだが、通貨とは実のところフォーカルポイント[*6]である: その価値はそれが貨幣として受け入れられているという共通認識から生じるのだ。一見これは循環論法のように見えるが、矛盾してはいない。ゲーム理論の観点からすると、価値の貯蔵手段としてのトークンの認知は、トークンが広く流通している限り、安定する。台帳としては Bitcoin は単なる 1 と 0 のビット列にすぎないことに注意しよう。未使用出力[*7]内にエンコードされた数値を残高として扱う選択は完全に**社会的**な合意なのであって、プロトコル自身の性質ではない。

[*4] 今や scrypt マイニング用 ASIC(特定用途向け集積回路) を現在は入手できてしまうが

[*5] 訳註: ハイエク「貨幣発行自由化論」(1976) における、通貨の脱国営化論のことと思われる。

[*6] 訳註: focal point。ゲーム理論用語。期待値の焦点とも。

[*7] 訳註: Bitcoin の UTXO のこと。

プロトコルの変更は「フォーク」と言われる[*8]。これは、主に各ユーザーが古いプロトコルを使い続けるかどうか選択することができることからそう呼ばれる。そのため、フォークすると通貨は古いバージョンのものと新しいものの二つに分裂する。

フォークの成功はソフトウェアエンジニアリングを必要とするだけでなく、大多数のユーザーとの意見の調整を必要とする。現実には、この調整は難しい。実際、フォークの後には二つの台帳が存在するようになり、ユーザーはジレンマに陥る。フォークしたどちらの枝に価値があるのだろう。

これは、他のユーザーたちが最も評価するだろう枝を自分も最も評価するのが正解となる調整ゲームだ。もちろん、他のユーザーたちもおそらく同じ戦略をとって、同じ理由で枝を評価するだろう。この形態のゲームは経済学者 Thomas Schelling によって研究されたため、このフォーカルポイント (期待値の焦点) は「Schelling ポイント」[6] とも呼ばれる。

残念ながら、Schelling ポイントがステークホルダーにとって最も望ましい選択だという保証はない。それはただ単に「デフォルト」の選択であるというだけだ。「デフォルト」は、その価値と関係なく、中心開発チームの誘導や政府の命令に沿ったものとなる可能性がある。

社会合意を変更することができる攻撃者は思い通りに通貨をコントロールできる。元のトークンの価値が合意のシフトによって大きく毀損されるのならば、元のプロトコルに固執する選択は、とても非合理になる[*9]。

中心開発チームは中央集権化の危険な原因となりうる。ユーザーはどんなオープンソースプロジェクトでもフォークすることができるが、社会合意を変えてしまうことができる十分に強い力を持つ攻撃者の前には無力である。中心開発チームが善意にあふれていると仮定しても、彼らは攻撃者が付け入る隙となるだろう。

Tezos はプロトコルのフォークを思い切って非中央集権化することで、中央集権の源から作られる脆弱性を解決する。Tezos は自分自身の暗号台帳を使ってステークホルダーにフォークを調整させる。これにより調整問題は解決され、ステークホルダー間の調整の外から出てきたフォークは正当ではないという原則を確立でき、合意を動かすことでプロトコルを攻撃することを非常に難しくする。

たとえば、著名な開発者がプロトコル内で決められた手続きに従わずに Tezos をフォークしようと発表したとしよう。ステークホルダーは「なぜ彼は手続きを無視しようとするのか?」と疑問に思うだろう。おそらく確実に、この開発者は Tezos 内で自分のフォークの合意を取りつけることが無理であることを知っているのだ。

このことはステークホルダーたちに、彼らにとって最も好ましい選択はこのフォークを拒否することだと気づかせてくれる。それゆえ、その開発者にどれだけ影響力があろうとも、Schelling ポイントはこのフォークの拒否、となる。

1.2 Proof-of-Work の欠点

Bitcoin で使われている proof-of-work メカニズムは、二重支払い問題を避けるように注意深くバランスを保たれたインセンティブ群からなる。これは、マイナーの共謀の可能性がなければ良い理論的性質を持ってはいるが、実際にはひどい欠点がある。

[*8] プロトコル内部で発生するブロックチェーンの分岐 (フォーク) と混同しないように

[*9] 「フォークにより流通量上限を変更した Bitcoin は、もはや Bitcoin ではない」から Bitcoin は 2100 万 bitcoin 以上は流通し得ない、という議論はあまり実質的ではない。なぜなら、Bitcoin とは社会合意が定義するものだから。

1.2.1 マイニングパワーの集積化

暗号通貨の基盤としての proof-of-work にはいくつかの問題がある。もっとも顕著な問題は、2014 年の時点で完全に顕在化しているのだが、集積化されたマイニングプールが存在することで、少数の個人に権力が集中してしまうことである。

Proof-of-work メカニズムは非中央集権化されている、それは通貨を安全にするために、ユーザーが明示的に誰かを信用する必要がないことを意味する。しかしながら、非明示的に、Bitcoin はその通貨の安全性のために、全てのユーザーが 1 つか 2 つのマイニングプール運営者の善意を信用しなければいけないシステムと化してしまった。

50% を超えるハッシュ計算力を持つマイナーたちによる共謀は 51% 攻撃 [7] として知られている。これにより攻撃者はトランザクションを妨害し、トランザクションを巻き戻し、新たに鋳造されたコインを盗み、二重支払いができるようになる [8]。

中央集権化されたブロックをサインする造幣局は、はるかに無駄が少ないとはいえ、51% のハッシュパワーを持つマイナーと同様に安全ではない。もし中央集権化された造幣局が Bitcoin ユーザーにとって受け入れ難いのであれば、マイニングパワーのなし崩しの集積化にも耐えられないはずだ。

マイニングパワーの集積は偶然の産物ではない。大きなマイニングプールを持っていれば、競争相手よりも安定したリターンを得られるから、設備拡大が容易となる。巡り巡って、設備拡大はマーケットシェアをさらに増加させ、収入をより安定させる。

さらに悪いことには、ghash.io という巨大マイニングプールは彼らに直接提出された「特別」トランザクションを優先して処理するというビジネスモデルを臭わせたことがある。これは巨大マイナーは小さいマイナーと比べるとその大きさに比例してより多く稼げることを意味する。残念ながら、ほとんどのマイナーは利己的に中央集権化されたマイニングプールの利便性を好むから、p2pool[*10]がハッシュパワーを惹きつけてしまうという問題があった。

多くの人が市場の集中は大したことではないと主張してきたが、彼らの議論は実世界経済の事例から拙速な一般化を行なっている。実世界のビジネスは、そのプレーヤーがシュンペーターの言う創造的破壊による不断の進化圧に晒され続ける、急速に変化する環境のおける競争である。実世界ビジネスは地域の知識を必要するし、組織の問題や、プリンシパル＝エージェント問題[*11]に対処しなければいけない。Bitcoin のマイニングはハッシュパワーを中心に形作られた純粋に人工的な経済部門であり、純粋に代替可能なコモディティ[*12]である。拙速に一般化して、この不毛な環境が複雑で豊かな実世界経済[*13]と同じ有機的堅牢性を持っていると考えるのは間違いである可能性がある。

さらに、経済学的には、自然な独占はその地位をわざわざ脅かすようなインセンティブを持たないことが一般的に明らかである。Bitcoin マイナーについても同じことが言えるだろう、つまり、一体、支配的なマイナーが通貨を危険にさらすことで彼らの投資を破壊したがるだろうか?(いやないだろう。) それでも残念ながら、巨大マイナーが不正直な攻撃者に操作されうるという大きなシステムリスクがある。ネットワークに対し

[*10] 訳註: P2Pool. 非中央集権化 Bitcoin マイニングプール

[*11] 訳註: 経済学、ゲーム理論用語。

[*12] 訳註: 実世界経済には純粋なコモディティは存在しないことが念頭にある。

[*13] FPGA ボードを追いやった ASIC をさらに駆逐してしまうような新しい技術が生まれる可能性はある。しかし、そのような技術革新は特定のマイナーが長期間にわたって支配的な地位を築くのを防止できるほど頻繁には起こらない。また、このような技術革新は、新しい技術を一番に入手したか、さもなければ同じパターンを繰り返すだけ十分に資本のある、ごく少数の新しい (もしくは前と同じ) 派閥を利するだけだろう。

二重支払い攻撃を実行するコストは、大きなマイニングプールのいくつかを転覆させるコストと**変わらない**からだ。

(訳註: この段落は訳者のマイニングプールに関する知識不足からうまく翻訳できていません。) この問題を解決するため、プール運営者にその参加者達が不正をしないと信用できなくなるようなプロトコル変更がいくつか提案されてきた。しかし、これらの提案は、不正をしてもプールが報復することのできない匿名参加者からの採掘力集約を防ぐだけである。プーリングは非匿名の人々の間ではまだ可能なままだ。プール運営者は参加者がシェアを持っている間、全てのマイニングハードウェアを使うかもしれない、さもなければ、運営者はハッシュしようとするブロックに認識番号 (nonce) を含ませることを強制して不正を行うものを追跡するかもしれない。これらの提案はこのように匿名マイニングの変化を増大させることになり、マイニングカルテルの手にさらなる集中を推し進めることになるだろう。

Proof-of-stake は、Tezos で使われているように、この問題には無縁である。51% のマイニングパワーを所持することは不可能ではないが、それには全流通量の 51% の通貨を保持する必要がある。それは 51% のハッシュパワーを操作するよりもかなり面倒なだけでなく、基本的により良い**インセンティブ**の一致をもたらす。

1.2.2 インセンティブの不一致

Proof-of-work には、マイニングパワーの集中よりももっと解決することが難しいさらに深い問題がある。マイナーとステークホルダー間のインセンティブの方向が一致しないのだ。

実際、長期的に見ると、マイニングによる総売上はマイナーに支払われたトランザクション手数料の合計になる。マイナーはハッシュ値を生成するのを競うから、マイニングに使われる経費は売上よりほんの少し小さくなるだろう。トランザクションに使われる量はトランザクションの需給に依存するが、ブロックチェーン上のトランザクションの供給はブロックサイズという固定された値で決まっている。

残念なことに、トランザクションの需要は非常に低いレベルにまで落ちていくだろうと思われる理由がある。人々は、特に少額取引の場合、取引の完了確認を待つのを嫌って、信頼できる第三者機関によるオフチェーントランザクションを利用するだろう。そして、支払い処理業者たちはお互いの貸し借り解消をそう頻繁には行わないだろう。

このシナリオは経済的にありうるだけでなく、Bitcoin がサポートする比較的低いトランザクション処理レートからしても必要だと思われる。ブロックチェーントランザクションはオフチェーントランザクションと競争しなければならないから、トランザクションに使われる量はそのコストに近づいていく。それは、近代のインフラにおいてはゼロに近いはずだ。

トランザクション手数料に最低額を課すのは問題をさらに悪化させ、ユーザーはオフチェーントランザクションにより頼ることになる。トランザクション手数料に支払われる額が暴落すると、マイナーの収入も暴落し、それは 51% 攻撃にかかるコストも下がることになる。一言で言うと、proof-of-work ブロクチェーンの安全性にはコモンズ問題 [9] がある。中心開発者のマイク・ハーンは資金調達約束型のマイニング助成特殊トランザクションを使ってはどうかと提案した [10]。しかし、堅牢な通貨は、その安全な運営のために慈善に依存する必要があってはいけない。

Proof-of-stake はマイナーとステークホルダーのインセンティブの方向を揃えることで、これらのインセンティブ問題を修正する。まずその定義そのものから明らかなように、マイナーはステークホルダー**自身**である。だから彼らはトランザクションのコストを低く保ちたい。同時に、proof-of-stake マイニングは資源の無駄遣いに基づかないから、トランザクションにかかる費用 (直接の手数料と間接的なインフレーション費用) の全てはマイナーが得ることができる。彼らは富の破壊を通した競争をすることなく運用コストをカバーする

ことができる。

1.2.3 コスト

他の解決策として、Dogecoin[11] が考えたように、マイニングに永久に報酬を与え続けるという方法がある。残念ながら、proof-of-work はマイナーの利益を上昇させることなくどんどんユーザーのコストを上昇させ、価値の喪失を引き起こす。確かに、マイナーはハッシュ値を生成する競争を行うから、マイニング費用は売上よりほんの少し小さくなる。長い目で見ると、マイナーはトランザクションサービスの価値として相応しい利益を上げることができるが、マイニングコストは誰の手にも渡らず無駄になってしまう。

この影響は決して僅かなものではない。実在する経済的な財 (工場での時間、電気使用、エンジニアリング努力) は proof-of-work マイニングによって経済から取り払われてしまう。2014 年 6 月の時点で、Bitcoin の年間インフレ率は 10% を少し超えるくらいで、毎日約 216 万ドルが、ghash.io に中央集権化されているあまり安全性を提供しないシステム維持のために浪費されている。

Proof-of-work 方式の安全性そのものは、攻撃者が賄えるコストよりも高いこの実際の運用コストにあって、それは通貨の成功に伴って増える運命にある。

Proof-of-stake は、攻撃に必要なコストを下げることなく、この無駄の原因を取り除く — 実際、通貨価値の上昇にともなって自動的に攻撃のコストは増えていく。マイニングのために証明しなければいけないものは既存資源の破壊ではなく、既存資源の調達だから、proof-of-stake 通貨は広い流通を得るために膨大な資源浪費に頼る必要はない。

1.2.4 コントロール

最後に、重要なことだが、proof-of-work システムはステークホルダーではなく、マイナーに権力を持たせる。例えば、フォークにはマイナーの多数の合意を必要とする。これは潜在的な利益相反をもたらす。たとえば、マイニング報酬の増加をもたらすようなプロトコルフォークをステークホルダーたちに認めさせるまで、マイナーの多数は結託してブロックチェーンを人質にすることが可能だ。より一般的には、マイナー達は仮想通貨を、そのユーザーにとって経済的利益がある期間よりも長い期間、マイナー自身に権力を与えるとてつもなく無駄なシステムとして維持し続けるだろう。

1.3 スマートコントラクト

Bitcoin ではスマートコントラクトを書けるにもかかわらず、そのオプコードの多くは使用禁止となってしまった経緯があり、その可能性は限られている。イーサリアムはいくつかの重要な違いのあるスマートコントラクトシステムを導入した。そのスクリプト言語はチューリング完全であり、Bitcoin の未使用出力の代わりに状態を持つ口座を使用する。

言語がチューリング完全であることに重きが置かれがちだが、第二の性質のほうがはるかに重要である。Bitcoin ではアウトプットは使用額と未使用額という二つの状態しか記憶できないとみなせるが、Ethereum では (秘密鍵で守られた) アカウントは残高、コントラクトのコード、データ領域を保持できる。アカウントの記憶領域はアカウントに対するトランザクションを行うことで変更することができる。トランザクションはその金額とコントラクトコードに渡すパラメタによって指定される。

チューリング完全であるコントラクトスクリプト言語の欠点はスクリプトの実行に必要なステップ数に上限がない可能性があることで、この上限は一般には計算不可能である。

この問題に対処するため、Ethereum ではトランザクションを認証するマイナーが、コントラクト実行に必要な計算複雑性とステップ数に比例する手数料を要求するシステムを作り出した。

しかし、ブロックチェーンが安全であるためには全てのアクティブノードがトランザクションを検証する必要がある。悪意のあるマイナーが自分の生成するブロックに無限ループに陥るようなトランザクションをわざと紛れ込ませ莫大な手数料を自分自身に支払うことができてしまう。他のマイナー達はそのトランザクションを認証するのに非常に長い時間を浪費するか、最悪、検証に失敗するかもしれない。しかし実際には、実用性のあるスマートコントラクトのほとんどは非常に簡単なビジネスロジックを使って実装され、複雑な計算を必要としない。

我々の解決法は各トランザクションでプログラムを走らせることのできるステップ数に上限を置くことである。ブロックにはサイズ上限があってブロックごとのトランザクション数も制限されるから、ブロックごとの計算ステップ数にも上限があることになる。このレート制限は CPU 濫用を狙う DoS 攻撃を防ぐことができる。一方、悪意のないユーザーは、このレート制限の下でも複数のトランザクションを発行することで、単一のトランザクションで許されるステップ数よりも多くの計算を行うことが可能だ。

マイナーは、トランザクションに設定された手数料が、その長い実行時間に対してあまりに小さすぎると感じた場合、そのトランザクションを除外することができる。Tezos プロトコルは修正可能なので、必要が生じればこの実行ステップ上限を将来見直したり、スクリプト言語に新しい暗号プリミティグを追加することが可能だ。

1.4 正しさ

Bitcoin は比較的小規模なコードベースで 80 億ドルの価値を支えている。セキュリティ研究者 Dan Kaminsky によれば、理論上は Bitcoin はコンピュータセキュリティの悪夢に見える。カスタムバイナリプロトコルを伴った `C++` コードベースがインターネットに接続された電子マネーを管理するノードを動かす、というのは大災厄を約束しているようなものだ。`C++` プログラムはしばしばメモリ崩壊バグによって穴だらけになっている。これらのプログラムがインターネットに接続されれば、それはリモート攻撃者にとって格好の脆弱性を作り出す。そこに蓄えられた電子マネーは、脆弱性を見つけ出し利用できるほど賢い攻撃者にとっては手っ取り早い収入となる。

幸運なことに、Bitcoin の実装は今までのところ攻撃に対して非常に堅牢であるようだが、いくつかの例外があった。2010 年 8 月、二つの出力の和がオーバーフローして負値になるバグにより攻撃者は 0.50 コインの入力から 92233720368.54 コインの出力二つを作り出した。最近では、OpenSSL ライブラリに Heartbleed バグのような深刻な脆弱性が複数見つかっている。これらの脆弱性には共通点が一つある。これらは `C` や `C++` のようなプログラミング言語が実行される命令の正しさを全くチェックしないから発生したのだ。計算効率のため、これらのプログラムはメモリのランダムな場所をアクセスできたり、結果が計算機がサポートできないほど大きい整数の加算などを行ってしまえる。これらの脆弱性は今の所は Bitcoin をめちゃくちゃにしてはいないが、そのシステムの安全性のためには良いものではない。

このような問題のないプログラミング言語もある。OCaml は INRIA で 1996 年から (それ以前の成果をふまえて) 開発されている関数型言語である。その実行速度は `C++` と同程度で、ベンチマークではだいたい最速のプログラミング言語の部類に入る [12]。より重要なのは、OCaml は強く型付けされており強力な型推論システムが備わっていることだ。その強力なパターンマッチや高階モジュールを含む表現力のある文法や意味論によって、ブロックチェーンのプロトコルを支えるロジックを簡潔に正しく、それも容易に記述することがで

きる。

OCaml の意味論はとても厳密で、そのかなりの部分はすでに形式化されており [13]、プロトコル修正の意図がなんなのか、曖昧さなくコードから理解することができる。

さらに、Coq という、もっとも進んだ形式証明ソフトウェアの一つはその証明コードから OCaml プログラムを抽出することができる。Tezos が成熟した暁には、プロトコルの正しさの数学的証明からプロトコルの鍵となる部分のソースコードを抽出することができるようになるだろう。

恐ろしいソフトウェア失敗事例は枚挙にいとまがない。Heartbleed バグは数百万ドルレベルの損害を引き起こした。2013 年、高頻度取引会社 Knight capital のちょっとしたバグは 5 億ドルに相当する損害を引き起こした。1996 年には、70 億ドルの開発コストをかけた Ariane 5 ロケットが算術演算のオーバーフローバグによって破壊された。そのロケットと積荷の価値はは 5 億ドルほどと見積もられている。

これらすべてのバグは形式検証を使っていれば防ぐことができた。形式検証技術は近年飛躍的に進歩しており、今や実際のシステムに適用する時だ。

2　抽象ブロックチェーン

Tezos はブロックチェーンプロトコルをできるだけ一般的な形であらわし、同時にネイティブプロトコルとしても効率的であるようデザインされている。ブロックチェーンの目的は、複数から並行に編集されうる一つの状態を記述することである。並行編集間の衝突を避けるため、状態を台帳、つまり初期状態からの変更列として表す。これらの変更はブロックチェーンの「ブロック」と呼ばれる。そして、Bitcoin の場合は、状態とはほぼ未使用出力の集合となる。ブロックは多数の並行に走るノードにより非同期に作成されるから、ブロックのツリーが形成される。このツリーのそれぞれの末端 (葉) は可能な状態の一つを表すチェーンの末端となる。Bitcoin では一つの枝だけ、もっとも難易度の合計が高いものだけが、正当な枝だと見なされる。ブロックは、名前が指し示す通り、複数の編集操作 (Bitcoin の場合、トランザクションと呼ばれる) をまとめたものである。これらの編集操作は状態に対して順番に適用される。

2.1　三つのプロトコル

ネットワーク・プロトコル、トランザクション・プロトコル、そしてコンセンサス・プロトコル、暗号台帳におけるこの三つのプロトコルの区別は重要である。

メタシェルは、トランザクションとコンセンサス・プロトコルを抽象実装として分離する一方、できるだけ透過なな方法でネットワーク・プロトコルを扱う。

2.1.1　ネットワークプロトコル

Bitcoin のネットワーク・プロトコルは基本的にはトランザクションのブロードキャスト、公開されるブロックのダウンロード、ピアノードの発見などを行うゴシップネットワークである。ここはもっとも盛んに開発が行われる場所だ。例えば、2012 年に BIP0037 によって導入されたブルームフィルタは、クライアントがブロックチェーン全体をダウンロードすることなく単純な支払いの検証をスムーズに行えるようにするものだ。

ネットワーク・プロトコルの変更は比較的論議を呼ぶことはない。これらの変更要求に対する初期の反論はあるかもしれないが、最終的にはすべてのステークホルダーが基本的な合意に到達する。

これらの変更は公に議論される必要さえない。たとえば、誰かが Bitcoin のトランザクションを猫ちゃん画像にステガノグラフィとして埋め込んでネットに公開する方法を開発したとする。もし十分多数の人がトランザクションを同じ方法で公開し始めたとすれば、マイナーはトランザクションを猫ちゃん画像から探し出しそれをブロックチェーンに組み込もうとしはじめるだろう。

健全なネットワークには互換性が必要だが、ネットワーク・プロトコルにおける技術開発競争は一般的に暗号通貨をより強力なものとする。

2.1.2 トランザクション・プロトコル

トランザクション・プロトコルはトランザクション検証を行う。例えば Bitcoin ではそれはスクリプティング言語を通して定義される。まず、コインは、マイナーがブロックを発見すると生成される。それから、マイナーは自分が採掘したコインにスクリプトを取り付ける。

このようなスクリプトは「未使用出力」と呼ばれる。トランザクションは複数の未使用出力のスクリプトが真に評価されるような引数を与えることで出力を組み合わせる。スクリプトは南京錠で、引数はその鍵だと考えることができる。

単純なトランザクションではこれらのスクリプトは単なる署名検証を行うが、より複雑なスクリプトも作ることができる。これらの出力は積み上げられ新しい出力の集合を形成する。もし消費出力の合計が支払い総額量より多ければ、マイナーがその差額を手数料として得ることができる。

トランザクション・プロトコルの変更はネットワーク・プロトコルの変更と比べてより多くの議論を引き起こす。少数の人々が猫ちゃん画像ブロードキャストアルゴリズムを勝手に使い始めることはできるだろうが、トランザクション・プロトコルを変えるにはより工夫がいる。ただ、そのような変更は大抵ブロックの正当性を変更しないので、マイナーの大多数の協力を得ることさえできれば可能だ。これらの変更は普通「ソフトフォーク」と呼ばれる。

比較的議論の余地がないのであれば、変更のいくつかはここで実装される可能性がある。例えばトランザクション属性問題の修正はトランザクション・プロトコルレベルの変更であろう。Zerocash の導入もトランザクション・プロトコルレベルの変更ではあるが、あまりに議論を引き起こすので受け入れられないリスクを犯している。

2.1.3 コンセンサス・プロトコル

Bitcoin のコンセンサス・プロトコルは、もっとも難易度の高いチェーンを中心にコンセンサスが確立されることと、マイナーの報酬スケジュールを規定している。これによってマイナーは採掘報酬からトランザクションを作り出すことが可能となり、どのように難易度が時間にともなって変化するかが指定され、どのブロックが正当であって、「公式」なチェーンの一部であるのかが決定される。

このレベルのプロトコル変更は最も核心的で難しい。それゆえしばしば「ハードフォーク」、過去のブロックを非正当としてしまうようなフォーク、を必要とする。たとえば、proof-of-work システムの SHA256 への依存を変更するなどである[*14]。

[*14] 訳註: ここの原文はうまく訳せていません: For instance, the proof of work system, as is the reliance on SHA256 as a proof-of-work system, etc.

2.2 ネットワークシェル

Tezos は以上の三つのプロトコルを分離する。トランザクション・プロトコルとコンセンサス・プロトコルは独立したモジュールの中に実装され、ブロックチェーン維持の役割を持つ汎用的ネットワークシェルにプラグインとして挿入される。

プロトコルを汎用的なままにするために、以下のインターフェースを定義する。ブロックチェーンを使って表したい経済の現在の「状態」を Tezos では**コンテクスト**と呼ぶ。コンテクストには、口座残高や、現在のブロック番号などの情報が含まれる。ブロックは、古い状態を新しい状態に変換する演算子とみなされる。

この点で、プロトコルは次の 2 つの関数を与えるだけで記述することができる:

- **apply** 関数はコンテクストとブロックを引数に取り、正当なコンテクスト[*15]かエラー (ブロックが不正である場合) を返す。
- **score** 関数はコンテクストを受け取り、評価スコアを返す。これはブロックチェーンの葉を比較し最も正準なものを選択するために使われる。Bitcoin では、この関数は単純にコンテクストのチェーンの総難易度を計算し、その値を返す。

驚くべきことに、これらの 2 つの関数を与えるだけで**任意の**ブロックチェーンに基づく暗号台帳を実装することができる。さらに、これらの関数をコンテクスト自体に付け加え、さらに次の 2 つの関数をプロトコルに公開する:

- **set_test_protocol** 関数はテストネット環境で使われているプロトコルを新しいもの (典型的にはステークホルダーの投票によって選ばれたプロトコル) で置き換える。
- **promote_test_protocol** 関数は現在のプロトコルを現在テスト中のプロトコルで置き換える。

これらの 2 つの手順は、プロトコルがそれ自身の置換を検証することを可能にする。開始時プロトコルは、定足数のある単純な多数決を採用しているが、将来、より複雑なルールを採用することもできる。例えば、ステークホルダーは、将来のプロトコルが一定の性質を満たすべきだと規定する投票を行うこともできる。これは、プロトコル内に形式証明チェッカーを搭載し、すべてのプロトコル修正に合憲性証明の添付を要求することで実現できる。

3 Proof-of-Stake

Tezos はいかなるブロックチェーンのアルゴリズム、proof-of-work、proof-of-stake、中央集権的なものでさえ、実装することができる。Proof-of-work メカニズムの欠点のため、Tezos の開始時プロトコルは proof-of-stake を採用している。きちんと機能する proof-of-stake システムをデザインするには相当な理論上のハードルがあるが、それらを扱う方法について説明しよう[*16]。

[*15] 訳註: 元のコンテクストにブロックを適用したもの

[*16] 我々の proof-of-stake システムの完全な技術的説明は Tezos white paper を参照してほしい

3.1 Proof-of-Stake は不可能なのか

どんな proof-of-stake システムにも、非常に深刻な理論上のハードルがある。Proof-of-stake システムの可能性そのものに対する一番の疑問は次のとおりだ: 新しいユーザーがクライアントをダウンロードし、ネットワークに初めて接続する。ユーザーはジェネシスハッシュから始まる 2 つの大きな枝を持つブロックツリーを受け取ったとしよう。どちらの枝も非常に活発な経済活動を示しているが、それらは根本的に異なる 2 つの歴史を表している。このうち 1 つは明らかに攻撃者によって偽造されたものだが、どちらが本当のチェーンだろうか?

Bitcoin の場合、正規のブロックチェーンは最大計算量を持つものとなっている。これは、履歴の書き換えが不可能であるということを意味するものではないが、書き換えに使うハッシング・パワーを真正のブロックチェーンのブロック採掘に使える[*17]ことを考えると、非常にコストがかかる。ブロックがステークホルダーによって署名される proof-of-stake システムでは、(すでに所持コインを全て売り払った) 以前のステークホルダーがその古い署名を用いてブロックチェーンのフォークをコスト無しに行うことができてしまう— これは nothing-at-stake 問題として知られている。

3.2 緩和策

この理論上の反論は厳しそうに見えるが、効果的な緩和策がある。フォーク[*18]には大きく分けて 2 種類あることに注目することが重要だ: まず歴史のかなりの部分を書き換えようとする非常に深いもの、そして二重支払いを試みる短いもの、がある。表面的にはこれら両者には量的な違いしかないが、実際にはそれぞれのインセンティブ、動機づけ、対する緩和戦略はそれぞれ異なる。

無条件に安全なシステムというものはない。それは Bitcoin も、そして公開鍵暗号システムでさえ例外ではない。システムは、設定された**脅威モデル**に対して安全であるように設計するものだ。そのモデルが現実をどれだけとらえているかどうか、それは**一言で言うと**実証的な質問である。

3.2.1 チェックポイント

時々のチェックポイント生成は非常に長いブロックチェーンの再編を防止する効果的な方法だ。しかし、チェックポイントはハックに過ぎない。Ben Laurie が指摘しているように、Bitcoin のチェックポイントの使用は、完全に分散した通貨としての地位を貶めている [14]。

しかし、実際には一年ごとまたは半年ごとのチェックポイントに何か問題があるとは考えられない。人類は、何ヶ月にも渡って一つのハッシュ値に対するコンセンサスを形成することくらい、全く安全に達成できる。ハッシュ値は世界中の主要新聞に載せることができるし、新入生の机の彫り込んだり、橋脚にグラフィティとして描きこみ、それで歌を作り、まだ柔らかいコンクリートに刻印したり、ペットのフェレットに刺青することだって[*19]... できる。時々のチェックポイントを記録する方法は数え切れないほどあり、偽造することは不可能だ。対照的に、分単位のコンセンサス合意の問題は、非中央集権プロトコルによってより安全に解決できる。

[*17] 訳註: 当然、そこから正当な報酬を受け取ることができる
[*18] 訳註: ブロックチェーンのフォーク。プロトコルのフォークではない。
[*19] 訳註: 原著者はペットへの刺青を読者に推奨しているわけではない。

3.2.2 統計的検出

トランザクションは正規ブロックチェーンにあるブロックを参照することができるから、暗黙的にチェーンに署名していることになる。チェーンの長い再編成を試みる攻撃者は、彼が制御下にあるコインに関するトランザクションについてしか最終チェックポイントに付け加えることができない。長い正規チェーンでは、通常は流通通貨のより大きな部分で活発な取引が行われているはずだから、統計的に偽造されたチェーンと区別することができる。

この種のテクニック (しばしば TAPOS、「proof-of-stake としてのトランザクション」と呼ばれる) はサンプルが小さ過ぎて信頼的な統計的検査を行うことができない短いフォークに対してはうまく働かない。しかしながら、短期フォークに対処する技術と組み合わせて、短期と長期両方のフォークに対する強固な複合選択アルゴリズムを形成することができる。

3.3 Nothing-At-Stake 問題

Nothing-at-stake 問題を解決する興味深いアプローチが Vitalik Buterin によって Slasher アルゴリズム [15] において概説されている。しかし、Slasher はまだブロック採掘に proof-of-work メカニズムを採用しており、可能なフォークの深さに限界を仮定している。

我々の主アイデアは二重署名に対する罰則からなっている。署名に対する報酬支払いを遅らせることで、二重支払いの試みが検出された場合、その報酬支払いを止めることができる。これは、フォークに楽観的に署名することで、そのフォークがたまたま成功した場合の報酬を得ようとする利己的なステークホルダーを防ぐには十分だ。一旦報酬が支払われてしまうと、正直に行動しようとするこのインセンティブはなくなってしまうから、TAPOS が統計的に意義を持つか、またはチェックポイントが発生するまでの十分な時間、報酬支払いを遅らせることになる。

また、ステークホルダーに正直に行動させるために、チケットシステム[*20]を導入している。マイニングしたい者は、採掘権を行使するために一定のコインを担保とする必要がある。もしマイニングに失敗するか、(訳註: マイニングに成功した場合は) 長い期間をおいて、この金額は自動的に返還される[*21]。

ステークホルダーがインターネットに永続的に接続する必要をなくし、かつその秘密鍵を公開しないですむよう、署名には異なる鍵が使用される。

3.4 脅威モデル

無条件に安全なシステムというものはない。それは Bitcoin も、そして公開鍵暗号システムでさえ例外ではない。システムは、設定された**脅威モデル**に対して安全であるように設計するものだ。そのモデルが現実をどれだけとらえているかどうか、それは**一言で言うと**実証的な質問である。

Bitcoin は興味深い保証を提供している：それは非道徳的で利己的な参加者を容認している。マイナー達が共謀しない限り、参加者がネットワークを破壊するよりもお金を稼ぐことを好んでいさえすれば、どの参加者も正直であると仮定する必要はない。しかし、この共謀不在という鍵となる条件はしばしば忘れられ、Bitcoin の「トラストレス性」の主張は熟考されることなく熱狂的に繰り返されている。

[*20] 訳註: 保証金を取ることを指している。
[*21] 訳註: 不正が見つかった場合はこの保証金は没収される。

チェックポイント (年に一度) を使用することで、proof-of-stake システムは同じ性質を達成できる。

チェックポイントの無い proof-of-stake システムではこの主張は成り立たない。事実、攻撃者が多数の元ステークホルダーから古い鍵を購入することは理論上可能だし、元ステークホルダーにとって古い鍵を売ることは大したことでは無い。この場合、参加者についてより強い前提が必要となる。つまり、現在または以前のステークホルダーの大多数が安価に買収されネットワークへの攻撃に参加しないことが必要だ。この場合、proof-of-stake における「ステーク」の役割は、コンセンサスグループにおける悪意あるプレーヤーによる逆選抜 (訳注 adverse selection) を避けることに過ぎない。

4　将来の開発の方向性

このセクションでは、Tezos プロトコルに組み込もうと考えているアイディアをいくつか紹介する。

4.1　プライバシー保護トランザクション

最も重要なプロトコル更新の一つは、プライバシー保護トランザクションの導入だ。これを達成するために我々は 2 つの方法を考えている:リング署名と非インタラクティブゼロ知識証明 (NIZKPK) だ。

4.1.1　リング署名

CryptoNote はプライバシー保護のためのリング署名を使ったプロトコルを構築した。ユーザーは、N 個あるアドレスのどの一つがコインを使ったかを明らかにすることなくコインを使用することができる。それでも、二重支払いは発見することができ、その場合、取引は無効とできる。これは、トランザクション難読化するための他アドレスの協力を必要とし**ない** CoinJoin プロトコルと同等に機能する。

リング署名の主な利点の一つは、NIZKPK より実装が比較的簡単で、長い歴史のテストに耐えてきたより成熟した暗号プリミティブにしか依存していないことだ。

4.1.2　非インタラクティブゼロ知識証明

Matthew Green らはブロックチェーンベースの暗号通貨におけるトランザクション非追跡性を実現する NIZKPK の使用を提案した。最新の提案である Zerocash では、コインセットを Merkle ツリー内に秘密情報を付加して維持している。コミットされたコインは、ツリーのコインに付加された秘密情報の NIZKPK を提供することによって償還される。これは、比較的新しい暗号プリミティブ、SNARKs を使用して、効率的に検証可能な非常に小さな証明を構築する。

この手法は魅力的だが、欠点がある。使われている暗号プリミティブはかなり新しく、Bitcoin で使われている比較的単純な楕円曲線暗号ほどには精査されていない。

第二に、これらの証明の構築は CRS モデルに依存している。これは事実上、信頼されるセットアップが必要であることを意味するが、安全なマルチパーティ計算を使用することで、そのようなセットアップが危険にさらされるリスクを低減できる。

4.2 修正ルール

4.2.1 立憲主義

さらに進んだアイデアだが、特定の性質を持つという形式証明の備わったプロトコル修正のみを受け入れるよう、証明検査器をプロトコルに内蔵させることができる。これによって実質的に一種の合憲性を強制できる。

4.2.2 フュターキー

Robin Hanson は、価値観に投票し、信念に賭けることを提案した[*22]。彼はそのようなシステムを「フュターキー」[16] と呼んでいる。このフュターキーの中心的なアイデアは、価値観は大多数の合意によって最もよく補足することができ、その価値感を実現するための政策の選択は予測市場に任せるのが最良だというものだ。

このシステムはまさに文字通り Tezos で実装することができる。ステークホルダーはまず価値の満足度を表す信頼できるデータフィードに投票する。これは、例えば国際通貨バスケットに対するコインの為替レートなどになるかもしれない。採用されようとしているさまざまなコード修正によるこの指標の変化予想を行う内部の予測市場が形成されるだろう。価格発見と流動性を改善するためにマーケット・メーカーにコインを発行することで、これらの契約のマーケット・メーキングを助成することができる。最終的に、指標を改善する可能性が最も高いと思われる修正案が自動的に採用されこととなる。

4.3 集合行為問題の解決

集合行為問題[*23]は、複数の当事者が行動を取ることで恩恵を受けることができるが、個々がその行動を独立に取っても益を受けない場合に生じる。これはタダ乗り問題としても知られている。暗号通貨の所有者がその法的な困難に対して立場を改善したり、防衛したりするため、集団で取れるいくつかの行動がある。

4.3.1 意識向上

2014 年 7 月現在、Bitcoin の時価総額は約 8 億ドルである。毎月 Bitcoin 総額の約 0.05% を費やすことで、Bitcoin は**毎週** 100 万ドルの目に見える寄付を行うことができる。2014 年の時点で、この毎週の寄付金によって Bitcoin の価値を 0.6% 以上、上げることができるだろうか? 我々の答は明らかに「Yes」だ。Bitcoin のステークホルダーは、善行を積みながら上手くやっていくことができるだろう。

しかしながら、Bitcoin のステークホルダーは、大規模な拘束力のある約束を形成することが困難なため、このような事業を行うことができない。この種の集合行為問題は、Tezos で解決できる。プロトコル修正は、ステークホルダーが毎月いくつかのコントラクトアドレスに投票することで全流通量の 0.05% をどこに寄付するか決定するという手続きを作り上げることができる。ステークホルダーのコンセンサスは、無効なコントラクトアドレスに投票することによって (訳註: 寄付を回避して) 自分達の持つ価値の希釈を避ける、ということになるかもしれないが、そうならずに、そのお金が慈善的寄付金に使われるということもまたあり得るだろう。

[*22] 訳註: `http://mason.gmu.edu/~rhanson/futarchy.html`

[*23] 訳註: Collective action problem: 集合行為問題。経済学者、社会科学者 Mancur Olson により議論された。

4.3.2 資金調達の革新

技術革新のための資金調達は、プロトコルの中に直接奨励金を埋め込むことで促進できるだろう。プロトコルは単体テストを定義して、テストに合格するコード提案に対して自動的に報奨金を支払うことができる。

逆に、新しいプロトコルを設計する発明者は、そのプロトコル内に自分自身への報酬の支払いを組み込むことができる。彼のプロトコル案は報酬部分を取り除いてコピーされることもありうるが、ステークホルダーのコンセンサスはおそらく原作者に報酬を与えることになるだろう。ステークホルダーはこのゲームを繰り返し行うのだから、正当な報酬を出し渋ってゲームから降りるのは愚かな選択だからだ[*24]。

結論

我々は既存の暗号通貨の問題点を提示し、解決策として Tezos を提案した。新しい暗号通貨を提案することで暗号通貨界の断片化を防ぐという皮肉から我々は逃れることができないが、Tezos は本当に**最後の**暗号通貨たらんとしている。

他のプロトコルでどんな技術革新が生み出されたとしても、Tezos のステークホルダーはそれを Tezos に採用することができる。さらに、集合行為問題を解決する能力と OCaml によって簡単に実装できるプロトコルのおかげで Tezos はもっとも即応的な暗号通貨となるだろう。

参考文献

[1] Peter Suber. Nomic: A game of self-amendment. `http://legacy.earlham.edu/~peters/writing/nomic.htm`, 1982.

[2] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. `https://bitcoin.org/bitcoin.pdf`, 2008.

[3] Vitalik Buterin et al. A next-generation smart contract and decentralized application platform. `https://github.com/ethereum/wiki/wiki/%5BEnglish%5D-White-Paper`, 2014.

[4] Nicolas van Saberhagen. Cryptonote v 2.0. `https://cryptonote.org/whitepaper.pdf`, 2013.

[5] Matthew Green et al. Zerocash: Decentralized anonymous payments from bitcoin. `http://zerocash-project.org/media/pdf/zerocash-extended-20140518.pdf`, 2014.

[6] Thomas Schelling. *The Strategy of conflict*. Cambridge: Harvard University Press, 1960.

[7] Bitcoin Wiki. Weaknesses. `https://en.bitcoin.it/wiki/Attacks#Attacker_has_a_lot_of_computing_power`, 2014.

[8] Gaving Andresen. Centralized mining. `http://bitcoinfoundation.org/centralized-mining/`, 2014.

[9] Bitcoin Wiki. Tragedy of the commons. `https://en.bitcoin.it/wiki/Tragedy_of_the_Commons`, 2014.

[10] Bitcoin Wiki. Dominant assurance contracts. `https://en.bitcoin.it/wiki/Dominant_Assurance_Contracts`, 2014.

[*24] 訳註: 報酬を渋れば次からは誰も改善を提案してくれなくなるので。

[11] Simon de la Rouviere. Not actually capped at 100 billion? https://github.com/dogecoin/dogecoin/issues/23, 2013.

[12] Debian project. Computer language benchmarks game. http://benchmarksgame.alioth.debian.org/u32/index.html, 2014.

[13] Scott Owens. A sound semantics for ocaml light. http://www.cl.cam.ac.uk/~so294/ocaml/paper.pdf, 2008.

[14] Ben Laurie. Decentralised currencies are probably impossible, but let's at least make them efficient. http://www.links.org/files/decentralised-currencies.pdf, 2011.

[15] Vitalik Buterin. Slasher: A punitive proof-of-stake algorithm. https://blog.ethereum.org/2014/01/15/slasher-a-punitive-proof-of-stake-algorithm/, 2014.

[16] Robin Hanson. Shall we vote on values, but bet on beliefs? http://mason.gmu.edu/~rhanson/futarchy2013.pdf, 2013.