次世代暗号通貨プラットフォーム Ethereum の実験的評価

フォン　ヤオカイ†‡,　松本 晋一♣‡, 穴田 啓晃‡, 川本 純平†‡, 櫻井 幸一†‡

†九州大学大学院システム情報科学研究院

♣九州大学大学院システム情報科学府

819-0395 福岡市西区元岡 744 番地

‡九州先端科学技術研究所

814-0001 福岡市早良区百道浜 2-1-22

fengyk@ait.kyushu-u.ac.jp, smatsumoto@isit.or.jp, anada@isit.or.jp,
kawamoto@inf.kyushu-u.ac.jp, sakurai@inf.kyushu-u.ac.jp

あらまし 暗号通貨はインターネット上で流通しているデジタル通貨である．Bitcoinは暗号通貨の代表の１つであり，金融機関による管理を必要としない通貨の流通を可能とする概念として提唱され，近年は一般的人にも良く知られている存在となった．またBitcoinの別の側面である，集中管理者の存在なしに分散環境で価値を流通/交換し合意形成する方法に着目した方式もいくつか生まれており，Ethereumはその一つで，分散型アプリケーションを構築するためのプラットフォームである．本発表では，Ethereumとその周辺の暗号通貨に関する研究と実装状況について報告する．

A Next-Generation Cryptocurrency Platform Ethereum and Its Evaluation

Yaokai Feng†‡,　Shinichi Matsumoto♣‡, Hiroaki ANADA‡, Junpei Kawamoto†‡, Koichi Sakurai†‡

†Faculty of Information Science and Electrical Engineering, Kyushu University

♣Graduate School of Information Science and Electrical Engineering, Kyushu University

744 Motooka, Nishi-ku, Fukuoka City, 819-0395, Japan

‡Institute of Systems, Information Technologies and Nanotechnologies

2-1-22 Momochihama, Sawara-ku, Fukuoka City, 814-0001, Japan

fengyk@ait.kyushu-u.ac.jp,　smatsumoto@isit.or.jp,　anada@isit.or.jp,
kawamoto@inf.kyushu-u.ac.jp,　sakurai@inf.kyushu-u.ac.jp

Abstract A cryptocurrency is a medium of exchange on the Internet using cryptography to secure the transactions and to control the creation of new units. Bitcoin became the first decentralized cryptocurrency in 2009 and has become one of the representative cryptocurrencies. Since then, several projects aiming at distributing/exchanging on decentralized platforms based on global consensus have been proposed. Ethereum is one of them. In this paper, we report on an implementation of Ethereum after cryptocurrency is briefly introduced.

1　暗号通貨の歴史とBitcoin

暗号を利用するネットワーク型電子マネーは，1983年に匿名の電子マネーや電子現金システムとしてDavid Chaumによって考案された[1]．それは彼の会社Digicash（オランダ）により実現され，1995年からアメリカ銀行Mark TwainのMicropayment systemとして使用されていた．買い手と売り手の間は次のようにやりとりをする．

1）買い手は，銀行から暗号化されたノートを取得する．

2）買い手は売り手にそのノートを渡す．

3）売り手は貰った電子ノートを自分の銀行口座に預金する．

日本では1997年3月に初めてDigiCash社の「ecash」のライセンスを取得した．電子現金の実験を開始したのがその年の6月だった[2]．

DigiCash社は最終的にクレジットカードに負け，1998年に倒産した[3]．

1998年11月にWei Daiがcypherpunksメーリングリストで Hashcash の可能応用に関する討議の際にB-moneyを提案した[4]. B-money では初めて Hashcash proof-of- work を暗号通貨に導入した．これは後で誕生するBitcoinの基礎の1つである．

それまでの第3者に頼んで信用を担保するモデルは次の問題がある[5].

1）仲介紛争が発生する可能性がある；

2）仲介コストが必要である．特に小さいトランザクションは大きな影響を受ける；

3）トランザクションの不可逆性を保証できない；

4）ある程度の詐欺を容認させられる．

2009年5月Satoshi Nakamotoを名乗る人物によってBitcoinを提案した論文がCyperpuncksメーリングリストに投稿された[5]．Bitcoinは初めてディセントラルなpeer-to-peerネットワークの上に構築したオンライン暗号通貨で，proof-of-workをベースとしたブロックチェインによって，トランザクションの順序を公的に合意するというコンセプトを初めて暗号通貨に導入した．従って，第3者（金融機関など）による管理・信用担保を必要としない．Bitcoinは提案された後，世界からの注目を浴びた．

Bitcoinは，商品の買い手が売り手に対して直接Bitcoinを送ることによって決済を行う．商品の買い手は，売り手のウォレットのBitcoinアドレスを教えてもらい，自分のウォレットから相手のウォレットに対してBitcoinを送金するようにウォレットに対して指示をする．Bitcoinがちゃんと相手に届いて，それが公的に承認される．

Bitcoinの二重使用問題は次のような仕組みで解決する[6].

１）コインが所有者の手元から離れた（送金された）際に，所有者がそのコインに自分のサインで電子署名をする．即ち，それぞれのコインに，そのコインの所有者の履歴を全て残す．

２）送金側（支払う側）が自分の所有しているコインを送金先に送金する際，「xxxx さんに yyyy 識別のコインを送る」という取引の情報をBitcoinネットワーク全体に通知し，この取引の承認を申請する． Bitcoin ネットワーク上の各PCが，そのコインに付いている取引

履歴（所有者の電子署名の一連）を確認し，取引内容と所有者の情報が合っていれば，「この取引を認める」アクションを行う．問題がある（例えば，送金側が自分の所有していないコインを送ろうとする）ようだったら，「認めない」アクションを行う．最後に，取引が成立できるかどうかは，「投票」によって決められる．

3）ユーザが自分の所有しているあるコインを，2 人以上に同時に送るという不正行為を防ぐこともできる．その際，ネットワーク上の各 PC に，[A-->B]の取引情報と[A-->C]の取引情報が届いている（届く順序が PC により異なる可能性がある）．各 PC はどちらか１つ（例えば，先着順で）しか認めないので，「投票」の結果としても１つしか認められない．

2　次世代暗号通貨プラットフォーム Ethereum

Ethereum は，Bitcoin で使用されている技術等を応用・発展して，通貨以外の役割・機能も持たすことを主な目的とした技術・プロジェクトの１つである．このような目的をもって Bitcoin を拡張するプロジェクトはいくつかあり，まとめて Bitcoin2.0 とも呼ばれている[7]. Ethereum の上で通貨以外の役割・機能として，分散型アプリケーション（DApps: Decentralized Applications）を開発することである[6]. Ethereum の重要で根幹的な機能・技術として，

1) スマートコントラクト
2) 分散型自動化組織
3) ユーザー独自通貨（個人/団体で自由に通貨発行）
4)所有権のデジタル化

などは上げられる．

これらの機能(サービス)を利用するには基本的に暗号通貨が必要となる．Ethereum は Bitcoin からの発展と考えられるが，実はブロックチェーンも通貨も独自のもので，Bitcoin を一切利用していない．一からプログラムされたものである．

スマートコントラクトとは，契約行動をプログラミング化し，自動的に実行することにより契約の相手方を信用する必要がなくなり(trust-free)，コストが大きく低下するものである．また，ブロックチェーン技術により過去の契約の実行履歴が全て記録および公開されるので透明性もある．従って，紙の契約文書を作成する必要もなくなり，契約に関する訴訟も大幅に減る．簡単な例としては，こどもが電子遊具を遊ぶ場合で，300 円を入れて 10 分間遊べるとしたら，10 分間が経つと，電子遊具は自動的に止まる．この例でのコントラクトは子供の信用保証が要らない．

分散型自動化組織は，組織（会社など）の分散化，自動化および自律化を意味する．このような組織はより長期・永続的に行うスマートコントラクトの集合になる．このような会社は企業の製品，サービスは暗号通貨（例えば，Bitcoin）で決済し，株式は暗号通貨で，給料も暗号通貨である．

ユーザー独自通貨とは，個人や団体でだれでも通貨・コインを発行できることを意味する．実は，航空会社のマイルは独自通貨で，スターバックスの商品券も独自通貨である．

所有権のデジタル化とは，所有権を暗号通貨のコインで表すことで，しかも取引所で交換可能である．このように，誰にでも迅速に低コストで所有権を売買することができる．

また, Ethereum では暗号通貨の分散型取引所を実現することも可能である[7][8]．暗号通貨 Bitcoin の取引所はこれまで，ユーザーから Bitcoin を直接預かり，サーバー上に Bitcoin を保管する方法を取り入れてきた[8]．このような取引所は「集中型取引所」と呼ばれている．しかし，サーバーは常にサイバー攻撃の脅威に晒されているのみでなく，管理者に継続的

にユーザーの Bitcoin が不正に引き出されたことで取り戻せなくなったこともあった.

P2P 方式で取引を行うプラットフォームで実現した分散型取引所は次の長所がある[7].

1) 安全性が高い. 取引所を P2P 化することにより暗号通貨を盗難するのは非常に困難になる.

2) コストが低い. 理論的には取引所を P2P 化することで, 取引所へ手数料を払わなくて済む分だけより低コストになる.

3) 取引量が増える. 実質的に世界中の取引が一つのプラットフォームで行われることになるため, ユーザー独自通貨の実現と組み合わせて, 世界のあらゆる価値をすぐに交換できるようになる.

3 Ethereum の設計

3.1 アーキテクチャ

図 2 に, Ethereum の動作モデルの概略を示す. Ethereum は Bitcoin 同様に P2P モデルで動作するが, このための P2P プロトコルを DEVp2p wire protocol と呼んでいる. 当該プロトコルは, 予め定められた bootstrap ノードに接続しピア情報を得る.

Ethereum におけるアプリケーションは, 各

Python-based App. (e.g.pyeth)	Lisp-based app.	JavaScript-based app.	Other apps.
Serpent	LLL	Solidity	
EVM(Ethereum VM)			
Ethereum Blockchain Ethash, Merkle-Patricia Tree, RLP			
DEVp2P Wire Protocol			

図 1 Ethereum アーキテクチャ

ピア上の EVM (Ethereum Virtual Machine) と呼ばれる仮想機械による. EVM は図 1 に示すプログラミング言語とのバインディングが用意されている.

ツリー構造としては, Bitcoin の採用する Markle Tree に対し, Ethereum では Markle Particia Tree (詳細は 3.3 節を参照) を採用している.

3.2 Ethash

3.2.1 要件

暗号通貨のプロトコルにおいては, ハッシュ値の計算が重要な役割を担っている. ハッシュ計算として, Bitcoin では SHA-256 アルゴリズムを用いているが, Ethereum では複数のアルゴリズムを組み合わせたものを用いて

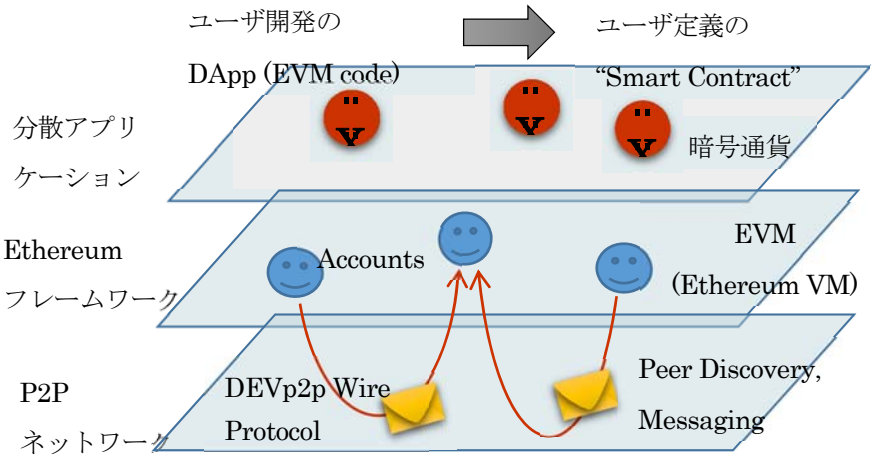

図 2 Ethereum の動作モデル概略

おり，これを Ethash と呼んでいる．Bitcoin とは異なるハッシュアルゴリズムを採用するに至った経緯として，Ethereum ではハッシュアルゴリズムに求められる要件として以下を設定している[9].

- I/O 飽和: ASIC などのハードウェアアクセラレーションによる占有的なマイニングを防ぐため，メモリアクセス帯域により処理が律速されるよう設計している．
- GPU 処理への適性: CPU 処理によるマイニングが低速となる一方で GPU での処理時に特に高速なマイニングが可能となるよう設計している．
- 軽量クライアントによる検証: 軽量クライアントによる高速な検証を可能とするよう設計している．
- 軽量クライアントのスローダウン: 軽量クライアントの処理がフルクライアントによる処理よりも遅くなるように設計することで，軽量クライアントによるマイニング実装(専用ハードウェアによるものも含む)が経済的に割に合わないよう設定する．
- 軽量クライアントの高速な起動: 軽量クライアントは起動から動作可能になるまでの時間を短縮するよう(JavaScript 実装の場合 40 秒以内).

3.2.2. 設計

前節で述べた要件に基づき，Ethereum では以下の手順でハッシュ計算を行う（図 3 を参照）．

- ブロックチェーン内のブロックヘッダからSHA-3演算によりシードを求める．
- シード値より疑似乱数を用いてキャッシュを生成する(16MB)．軽量クライアントは当該キャッシュのみを保持する．
- キャッシュを元に DAG(Directed Acyclic Graph)データセットを生成する．当該データセットの容量は1GBであり，生成にはメモリ帯域を必要とするが，検証には帯域を必要としない計算を実現

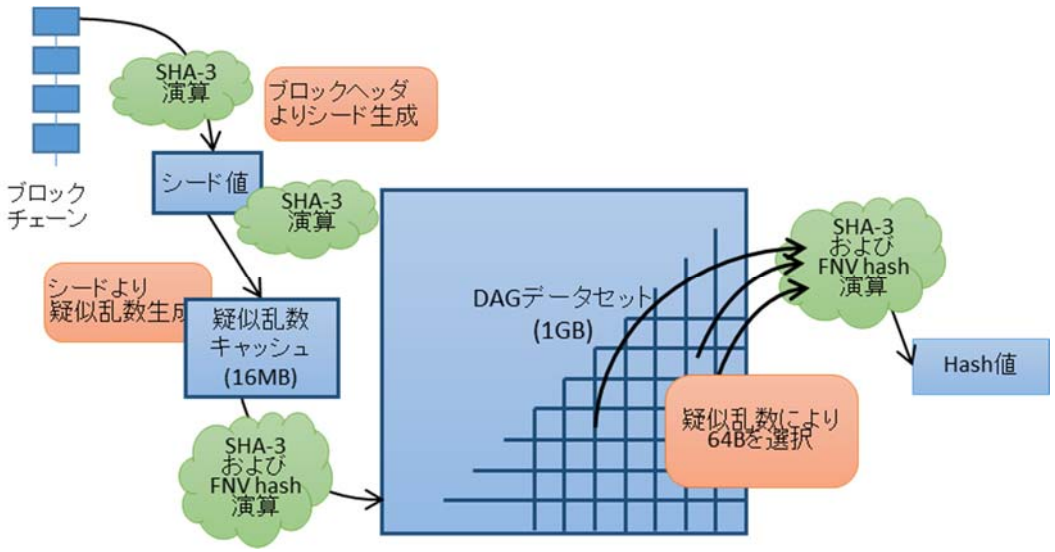

図 2 Ethash の概要

している．(軽量クライアントでない)フルセットのクライアント，またマイニングを行うクライアントは当該データセットを保持して処理を行う．

- マイニング時にはDAGデータセットから疑似乱数で選択したデータチャンクを基にハッシュを計算する．

上記の手順では，マイニング時には巨大なサイズ(携帯機器などにとっては)である 1GB の DAG データセットに対しランダムアクセスが必要であるのに対し，検証は小容量(16MB)のキャッシュを元に DAG データセットの必要なチャンクのみを計算すればよいことから，携帯機器などの一次記憶の容量に制限のあるデバイスでも実行可能となる．

3.2.3. 実装

Frontier リリースでは[10]，Ethash アルゴリズムは主として libethash ディレクトリ内で実装されている(C++実装の場合)．当該ディレクトリ内のコード量は 3k 行弱である．マイニング処理は主として libethcore ディレクトリ内の EthashCPUMiner.cpp, EthashGPUMiner. cpp にて実装されている．GPU マイニングは OpenCL ライブラリを用いた実装である．

3.3. Merkle-Patricia Tree

3.3.1. 設計

Merkle-Patricia Tree の例を図 4 に示す．ブロックチェーンのデータ構造として，Bitcoin の採用する Merkle Tree に対し，Ethereum では Radix Tree(Patricia Tree)と Merkle Tree を組み合わせた，Merkle-Patricia Tree を採用している[11]．データはキー-バリューストアの形式で収容されるが，キーは 4bit(ニブル)値の配列としてあらわされ，枝に対応する．当該データ構造は，kv ノードと diverge ノードから構成される．kv ノードはキーと，それに対応するバリューもしくはノードからなる．diverge ノードは 16 のノードもしくは空白が収容される配列に加え，一つのバリューもしくは空白を収容する．

kv ノード中のキーのエンコードにおいて，先頭のニブルの最下位ビットはニブル値長が基数であることを示すフラグとなる．当該ビットが 0 の場合はニブル値長が偶数であることを示し，次のニブルは¥x0 が挿入される．ま

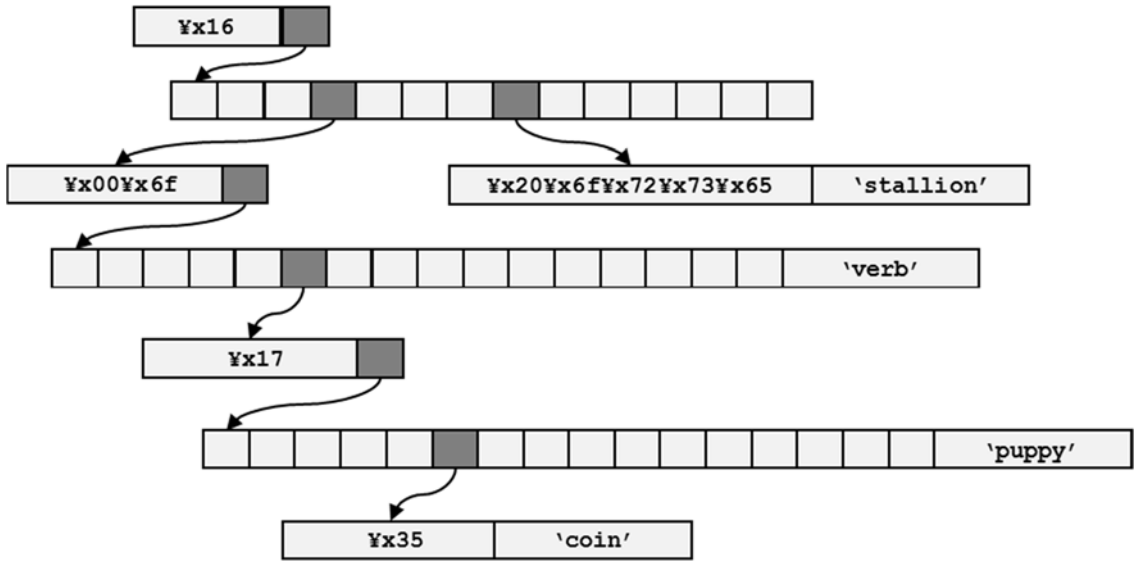

図 3 Merkle-Patricia Tree の例

表 1 Bitcoin と Ethereum の比較

	Bitcoin	Ethereum
ブロックチェーン	Proof-of-Work	Proof-of-Work
ハッシュDB	Merkle Tree	Merkle Patricia Tree
ハッシュ	SHA-256	Ethash
アプリケーション開発	JSON-RPC(限定的)	JSON-RPC, Serpent(Python) Solidity(JavaScript), LLL(Lisp) Mutan(C言語)

た先頭のニブルの下から 2 ビット目は葉ノードであることを示すフラグとなる．ニブル配列の最後はターミネータシンボルとして16が収容される．

3.3.2. 実装

Ethereum における Merkle-Patricia Tree 実装では，基数 16 の Patricia Tree を用いているが，木操作の高速化のために文献[11]の例に挙げられたデータ('dog', 'puppy'), ('horse', 'stallion'), ('do', 'verb'), ('doge', 'coin')を収容する場合，まずキーの文字コードをニブル値に変換し，

```
[6,4,6,15,16]:'verb'

[6,4,6,15,6,7,16]:'puppy'

[6,4,6,15,6,7,6,5,16]:'coin'

[6,8,6,15,7,2,7,3,6,5,16]:'stallion'
```

を元に作成したツリー構造は，前節のエンコード規則に従い，**図**3のようになる．

Merkle-Patricia Treeは，C++実装においては/libdebcoreの下のTrie*.cpp, Trie*.hにて実装されている．

4. インストール

Ethereumクライアントは，実装言語により複数の種類が存在する．Linux OS用のgo言語実装は，バイナリが提供されている．UbuntuのPPA(Personal Package Archive)でも提供されているため，Ubuntuのlaunchpad経由でインストール可能である．コマンドラインの場合は，

```
> apt-get install software-properties-common

> add-apt-repository –y ppa:ethereum/ethereum

> add-apt-repository –y ppa:ethereum/ethereum-dev

> apt-get update

> apt-get install ethereum
```

インストール後はEthereumのgo言語実装(geth)が利用可能になる．gethインストール後はアカウントを作成することで利用可能となる．“account new”オプションを付与して起動し，disclaimerに対するacknowledgeにyesと答えた後，パスフレーズを入力するとアドレスが表示される．

```
> geth account new
```

gethをCLIとして用いる場合は “console”オプションを付与する．マイニング開始には，gethコンソールにて “miner.start()”コマンドを発行する．

```
> geth consol

> miner.start()
```

5. まとめと今後の課題

本論文では，暗号通貨の歴史および第3者に頼んで信用を担当するモデルの短所を説明した上で，完全に分散型の暗号通貨Bitcoinを簡単に紹介した．そして，Bitcoinの発展とされる次世代暗号通貨プラットフォームEthereumを紹介して，Ethereumで何ができるのかについては説明した．Ethereumのアーキテクチャおよび Ethash，Merkle-Patricia Tree，EVM (Ethereum VM)などの核心技術の設計と実装も紹介した．

今後の課題として，Ethereumの上で具体的なアプリケーション(DApps)を開発することによりEthereumへの理解を一層深める．

謝辞

この研究の一部は，科学研究費（基盤研究(B) 研究課題番号:15H02711)および科学研究費（基盤研究(C)研究課題番号 : 26330169）の支援を受けている．ここに記して謝意を表す．

参考文献

1. Chaum David(1983). "Blind signatures for untraceable payments", Advances in Cryptology: Proceedings of Crypto 82 (3): 199–203, Springer.

2. ニュース/電子マネー, http://internet.watch.impress.co.jp/www/article/970331/ecash.htm

(accessed on 2015/8/15)

3. https://en.wikipedia.org/wiki/Ecash

(accessed on 2015/8/15)

4. Wei Dai, "b-money", cypherpunks mailing-list, 1998.11 http://www.weidai.com/bmoney.txt (accessed on 2015/8/15).

5. Satoshi Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System”, cypherpunks mailing-list, 2019.5 https://bitcoin.org/bitcoin.pdf (accessed on 2015/8/15)

6. A.N.Lab., http://anlab.jp/service/bitcoinmechanism/ (accessed on 2015/8/15)

7. Bitcoin日本語情報サイト, http://jpbitcoin.com/bitcoin2s (accessed on 2015/8/15)

8. Bitcoin News: http://btcnews.jp/decentralized-bitcoin-exchange-coinffeine-beta-released/ (accessed on 2015/8/21)

9.https://github.com/ethereum/wiki/wiki/Ethash-Design-Rationale (accessed on 2015/8/10)

10. https://github.com/ethereum/go - ethereum/wiki/Frontier (accessed on 2015/8/24)

11. https://github.com/ethereum/wiki/wiki/Patricia-Tree (accessed on 2015/8/24)