CM-006

Ethereum ネットワークにおけるスマートコントラクトの経時変化に関する分析
An Analysis on Temporal Change in Progress of Ethereum Smart Contracts

安冨 勝貴 † 内田 真人 †
Katsuki Yasudomi Masato Uchida

1. はじめに

近年，ブロックチェーンは金融分野だけでなく，多分野にわたる適用が期待できる技術として注目を集め，さまざまな暗号通貨が開発され利用されている．ブロックチェーンネットワークのひとつに Ethereum がある．Ethereum の主な特徴として，スマートコントラクトを実行できることがあげられる．ブロックチェーン上でのプログラムであるスマートコントラクトを実行すると，取引履歴が改ざんされないことが保証される上に，仲介人なしで取引を行うことができる．スマートコントラクトの実現により，第三者を必要としない，新たな決済方法が誕生し，人と人が取引を行うことに対して大きな利便性をもたらした．

一方で，スマートコントラクトを悪用した犯罪やスマートコントラクトに対する攻撃が報告されている．2016 年に起きた DAO 事件では，3,600,000ETH (当時の日本円で約 52 億円相当の通貨) が悪意を持った攻撃によって盗み出された．このような攻撃や犯罪の対策も，スマートコントラクトに関する分野では研究の対象になっている．Chen ら [1] は，過去に横行したポンジスキームと呼ばれる金融詐欺を実装したスマートコントラクトが存在していることを示し，ポンジスキームを実装したスマートコントラクトの検知手法を提案した．Atzei ら [2] は，スマートコントラクトの脆弱性の分類を行い，堅牢なスマートコントラクトの設計に必要な情報を示した．また，スマートコントラクトの分析および検証技術の開発を促進する研究者向けの手引きになるように，多種にわたるスマートコントラクトを用いた攻撃の分析を行った．

さらに，スマートコントラクトの解析や分類に関する研究も行われている．Grishchenko ら [3] は，Ethereum ネットワークにおけるスマートコントラクトの静的解析を行う EtherTrust というツールに着目し，EtherTrust について詳細に分析した．Tikhomirov ら [4] は，SmartCheck というスマートコントラクトにおけるソースコードの静的解析ツールを提案した．SmartCheck を用いることで，スマートコントラクトを記述するための言語である Solidity におけるバグや問題を検知することができる．さらに，彼らは SmartCheck で検知可能な問題の概要を説明し，Solidity が抱える問題を分類した．また，Bartoletti ら [5] は，スマートコントラクトの分類を行っている．彼らは，スマートコントラクトが単なる契約という目的以外で作成されていることを示した．

†早稲田大学基幹理工学研究科情報理工・情報通信専攻
Department of Computer Science and Communication Engineering, Waseda University

このように，特定のスマートコントラクトに焦点を当てた研究や，安全なスマートコントラクトの作成方法に関する研究，およびスマートコントラクトを分類する研究は存在する一方で，その特徴の経時変化に関する分析は行われていない．

作成回数や複雑さに関する経時変化を分析することで，Ethereum におけるスマートコントラクトの変化の特徴を知ることができる．さらに，スマートコントラクトが将来どのように変化するのかを予想することができる．また，Chen ら [1] の研究では，同じ目的で作成された悪性スマートコントラクトのソースコードは，部分的に一致，類似していたことが示されている．このことから，ソースコードから得られるオペコードが同一であるスマートコントラクトの分析を行うことは，悪性スマートコントラクトの検知の足掛かりになると考えられる．さらに，取引回数の分析を行うことで，現存するスマートコントラクトが抱える問題が浮かび上がると考えられる．

本研究では，スマートコントラクトの作成数の経時変化に関する分析や，取引回数の分析を行う．また，スマートコントラクトの複雑さをオペコードの出現回数を用いて定義し，スマートコントラクトが時間と共に複雑化していることを示す．一方で，オペコードが完全に一致しているスマートコントラクトの件数と，完全に一致はしていないが各オペコードの出現回数が同一なスマートコントラクトの件数を合計すると，全体の半数近くを占めていることを示す．

本論文の構成は以下の通りである．第 2 節では本論文に関連する 2 つの研究について紹介する．第 3 節では本研究で行った分析結果を述べる．最後に，第 4 節で本研究のまとめと今後の課題について述べる．

2. 関連研究

2.1. 悪意を持ったスマートコントラクトの検知

Chen ら [1] は，ポンジスキームと呼ばれる，投資詐欺を実装したスマートコントラクトの検知手法を提案した．この投資詐欺では，「あなたがお金を投資してくれれば，それを運用して増やし，増えた分支払う」などと謳い，投資資金を集める．しかしながらその資金は（全くあるいはほとんど）運用されず，以前の出資者に「配当」として渡すことで，まともな資金運用をしているかのように装う詐欺である．彼らは，スマートコントラクトのオペコードの出現回数や，スマートコントラクトのアカウントの特徴に着目したポンジスキームの検知手法を提案している．彼らの提案手法では，最も頻繁に出現する，PUSH，DUP および SWAP の 3 つを除いたオペコードが特徴として用いられている．また，アカウントの特徴よりもオペコードの情報のほう

表 1: スマートコントラクトのカテゴリとその特徴

カテゴリ	特徴
Financial	このカテゴリに含まれるスマートコントラクトは，通貨を管理，収集，または分配する．また，プロジェクトの資金調達のために投資家から資金を集めてクラウドファンディングサービスを実施するスマートコントラクトも含まれる．
Notary	このカテゴリに含まれるスマートコントラクトは，ブロックチェーンの不変性を利用して，一部のデータを永続的に保存し，場合によってはその所有権を証明する． 一部のスマートコントラクトは，ユーザーがブロックチェーンにドキュメントのハッシュを書き込むことを許可しているため，ユーザーはドキュメントの存在と整合性を証明することができる．
Game	このカテゴリには，運により勝ち負けが決まるゲームと技術や腕が勝敗を左右するゲームを実装するスマートコントラクト，およびこれらを組み合わせたゲームを実装するスマートコントラクトが含まれる．
Wallet	このカテゴリのスマートコントラクトは，ブロックチェーンとのやり取りを簡単にするために，キーの処理，トランザクションの送信，通貨の管理，契約の展開および監視を行う．Wallet は，一人または多数の所有者によって管理されることができ，後者の場合，複数の承認を必要とする．
Library	このカテゴリに含まれるスマートコントラクトは，他のスマートコントラクトによって使用されるための汎用演算（例えば，数式処理および文字列変換など）を実装する．

がポンジスキームの検知に効果的であったことを示している．彼らは，Ethereum 上のスマートコントラクトの内 0.15%がポンジスキームを実装したスマートコントラクトであると予測した．

2.2. スマートコントラクトの分類

Bartoletti ら [5] は，ブロックチェーンネットワークが存在する様々なプラットホームの特徴をまとめた．さらに，作成された目的によるスマートコントラクトの分類や，デザインパターンの分類を行った．彼らの研究では，スマートコントラクトを 5 つに分類して，それらの特徴を示している．5 つのカテゴリと特徴は表 1 の通りである．また，彼らの研究によると，対象とした，834 件の Ethereum におけるスマートコントラクトのうち，373 件のスマートコントラクトが Financial に分類され，各カテゴリのスマートコントラクトとの取引回数では，80 %以上の取引が Financial に分類されたスマートコントラクトとの取引であった．

3. 分析結果

3.1. データセット

本研究の分析で使用したデータセットは，Etherscan[1]から収集したデータを使用して作成したものであり，スマートコントラクトのアドレス，名前，取引回数 (データを収集した 2018 年 11 月 22 日時点) および作成日の情報は Etherscan 内の，作成されたスマートコントラクトが記載されているページ[2]から取得した．また，各スマートコントラクトのソースコードから得られるオペコードの情報は Etherscan の API を用いて取得した．オペコードとは，機械語におけるひとつの命令で，実行する操作を指定するコードのことである．スマートコントラクトを作成するために記述されたソースコードをコンパイルすることで，バイトコードが得られる．バイトコードは，オペコードが示す操作とその操作の対象 (オペランド) に対応した数字の羅列として構成されている．実際に存在する ETH_GAME[3]という名前のスマートコントラクトのオペコードの断片を図 1 に示す．図 1 における，PUSH1，MSTORE といったものがオペコードであり，本研究では，このオペコードの出現回数に着目した．取得したスマートコントラクトの総数は 50,998 件であり，最も古いものは 2016 年 3 月 24 日に作成されたスマートコントラクトであった．本

```
PUSH1 0x60
PUSH1 0x40
MSTORE
PUSH1 0x04
CALLDATASIZE
LT
PUSH2 0x0077
JUMPI
PUSH4 0xffffffff
PUSH29 0x0100000000000000000000000000000000000000000000000000000000
PUSH1 0x00
CALLDATALOAD
DIV
AND
PUSH4 0x3853682c
DUP2
EQ
PUSH2 0x0079
JUMPI
DUP1
PUSH4 0x3e3ee859
EQ
PUSH2 0x008c
JUMPI
DUP1
PUSH4 0x3fad9ae0
```

図 1: ETH_GAME のオペコードの断片

[1]https://etherscan.io/
[2]https://etherscan.io/contractsVerified
[3]0x9dE841bAEa33789B88403913f8c4fb43213aB3Bd

表 2: Etherscan から収集した情報

情報	説明
Address	スマートコントラクトのアドレス
ContractName	スマートコントラクトの名前
TxCount	スマートコントラクトが過去に取引した回数
DateVerified	スマートコントラクトの作成日

表 3: データセットに追加した情報

情報	説明
SourceID	オペコード全体を入力とした SHA256 によるハッシュ出力値
オペコード各種	各オペコードの出現回数
Count	各オペコードの出現回数の和

研究で作成したデータセットには，Etherscan から収集した表 2 に示す情報 が含まれている．また，オペコードが完全に同一であるかどうか判断するため，API を使用して得られたオペコード全体を入力としたハッシュの出力値をデータセットに加えた．オペコード全体とは，図 1 における，PUSH1，MSTORE のようなオペコード単体に分解する前のテキスト全体を指す．その後，API を用いて取得したオペコードの情報から各オペコードを分解し，各オペコードの出現回数，各オペコードの出現回数の和をデータセットに加えた．これらの追加したデータを表 3 に示す．

3.2. 分析方法の概要

Etherscan から入手した実データの中に，スマートコントラクトが作成された日付の情報がある．これを用いて，スマートコントラクトの作成数の変化を分析する．作成数の推移を把握することで，Ethereum におけるスマートコントラクトの規模の変化がわかる．同様に，実データに含まれるスマートコントラクトの過去の取引回数の情報を用いて，スマートコントラクトの取引回数の分布を分析する．スマートコントラクトの取引回数を分析することにより，作成されたスマートコントラクトの使用頻度を把握することができる．

次に，スマートコントラクトの複雑さの分析を行う．一般に，ソースコードからそのプログラムがどの程度複雑であるかを計測する指標として用いられるものには，変数の数，関数の数，オブジェクト間の結合度や継承の深さなどを使用することがある．一方，本研究では，各オペコードの出現回数の和を利用し，よりシンプルにスマートコントラクトの複雑さを評価する．スマートコントラクトの複雑さを分析することで，スマートコントラクトのソースコードや機能の複雑さがどのように変化したのかを大まかに把握することができる．

さらに，API を使用し得られたオペコード全体のハッシュ値を比較することにより，同一なオペコードを持つスマートコントラクトの分析を行う．同一なオペコードを持つスマートコントラクトを分析することで，関連研究 [1] で示されていたような，悪意を持ったスマートコントラクトの存在を示すことができる．また，同一な機能を持ったスマートコントラクトが多く存在していることは，利用者にとっての煩わしさに繋がると考えられる．同一なオペコードを持つスマートコントラクトの分析を行うことでこのような煩わしさを示すことができると期待される．

3.3. スマートコントラクトの作成数推移

はじめに，Ethereum ネットワークにおけるスマートコントラクトの作成数の推移を分析した．スマートコントラクトの日ごとの作成数の推移を図 2，月ごとの作成数の推移を図 3 に示す．スマートコントラクトの作成数の最大値は，2018 年 7 月 16 日と同年 7 月 19 日の 293 件であった．また，月ごとの集計では，2017 年 6 月中に作成された 1 日あたりのスマートコントラクト数の平均が 166.7 件で最大であった．2016 年 3 月から 2018 年 8 月頃までは 1 日あたりの作成数は増加傾向にあると言える．一方でそれ以降は減少傾向にあると言える．Ethereum ネットワークにおけるスマートコントラクトが徐々に多くの人に作成されるようになり，普及していった一方で，近頃は急激な作成数の増加はなくなり，月ごとの作成数は安定してきたことがわかる．

3.4. スマートコントラクトの取引回数の分析

次に，スマートコントラクトの取引回数の分布を図 4，図 5 に示す．全スマートコントラクトの平均取引回数は 2,196 回であった．1 度のみ取引を行ったスマートコントラクトの数が最も多く，11,380 件で全体の 22.3%を占めている．また，取引回数が 10 回以下のスマートコントラクトは 33,583 件 (65.8%) であった．このことから，作成されたスマートコントラクトの半数以上が，ほとんど取引を行っていないことがわかる．また，取引回数が少ないスマートコントラクトを分析した結果，名前が “My～ ” であるスマートコントラクトが多数存在していることがわかった．このような名前のスマートコントラクトは 575 件存在し，平均取引回数は

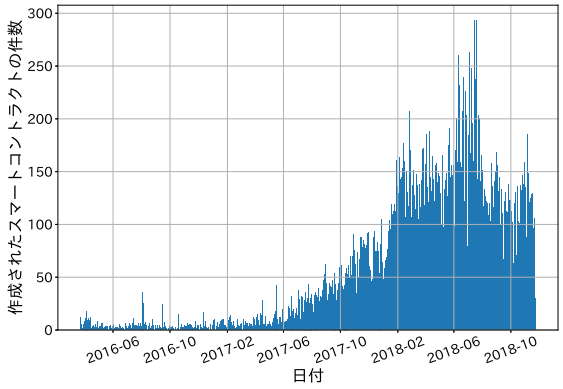

図 2: 1 日あたりのスマートコントラクトの作成数推移

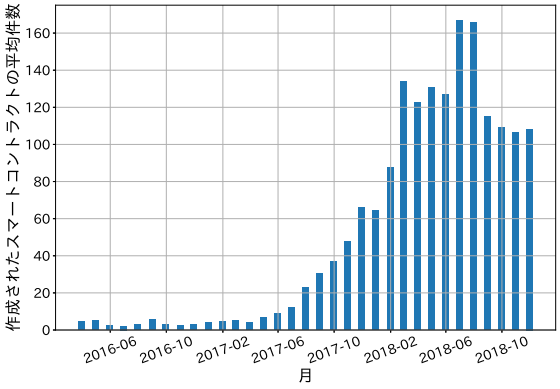

図 3: 1 月あたりのスマートコントラクトの作成数推移

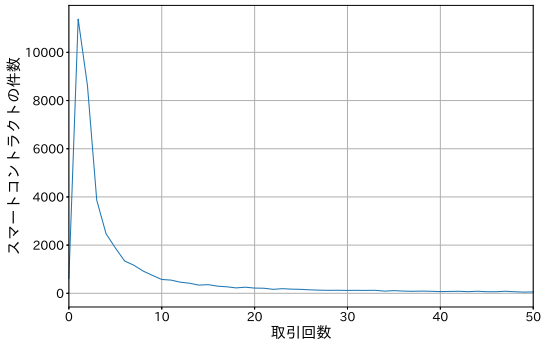

図 4: スマートコントラクトの取引回数の分布 1

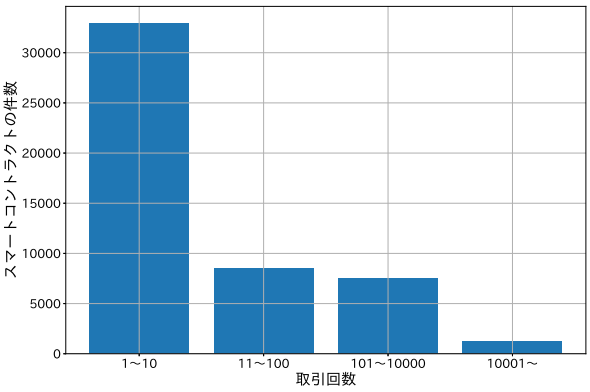

図 5: スマートコントラクトの取引回数の分布 2

808 回であった．これらのスマートコントラクトは名前が示唆しているように，個人的な目的で作成されたと考えられる．これらのことから，個人的な目的で作成されてたスマートコントラクト，あるいは複数の人が使用することを目的に作成されたが実際はほとんど使用されていないスマートコントラクトが多く存在していることがわかる．また，作成された月ごとの取引回数が 10 回以下のスマートコントラクトの割合を分析した．2016 年 3 月と 2017 年 9 月を除いた全ての月において，作成されたスマートコントラクトの半数以上のスマートコントラクトの取引回数が 10 回以下であった．作成月が 2016 年 3 月では 31.6%，2017 年 9 月では 46.8%のスマートコントラクトの取引回数が 10 回以下であった．このことから，作成されてから時間が経っていても，取引がほとんど行われていないスマートコントラクトが多く存在していることがわかる．

3.5. スマートコントラクトの複雑さの分析

オペコードの情報を用いて，スマートコントラクトの複雑さの分析を行った．データセットに含まれる，各オペコードの出現回数の和 (Count) の月ごとの平均の推移と最小二乗法を用いた近似直線を図 6 に示す．Count の情報は，スマートコントラクトが実行された時に行う操作の回数を表しているため，スマートコントラクトの潜在的な計算量を表しており，複雑さの指標になる．図 6 より，2016 年から 2017 年上旬までは，作成されたスマートコントラクトの数が少ないため，ばらつきはあるが増加傾向であったことが読み取れる．それ以降に関しても増加しているのがわかる．つまり，時間が経つにつれて複雑なスマートコントラクトが作成されてきたということが示せる．さらに，最小二乗法を用いて描いた近似直線からも，日が経つにつれて，複雑化していることがわかり，将来作成されるスマートコントラクトも複雑化していくと予測できる．近似直線から導出した，1 月あたりの複雑さの増加量は 28 であり，1 年あたりの増加量は 332 であった．複雑さの増加率は，1 月あたり 1.4%であり，1 年あたりの増加率は 16.2%であった．複雑さの 1 月あたりの増加量と増加率，1 年あたりの増加量と増加率を表 4 に示す．既存のプログラムを部分的に変更し組み合わせることが容易になった現在，ソースコードが複雑化しているということは，作成者が以前よりも高度で，多機能なスマートコントラクトを作成していることを示している．

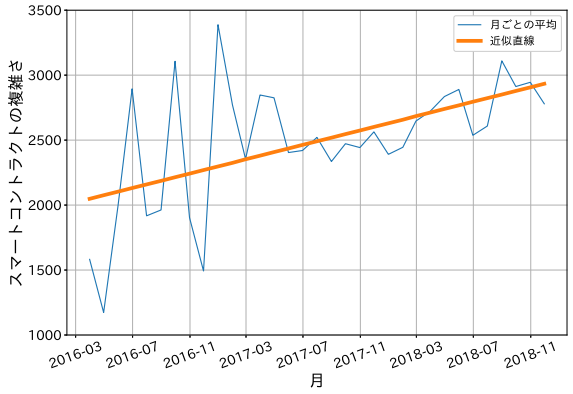

図 6: スマートコントラクトの複雑さの推移

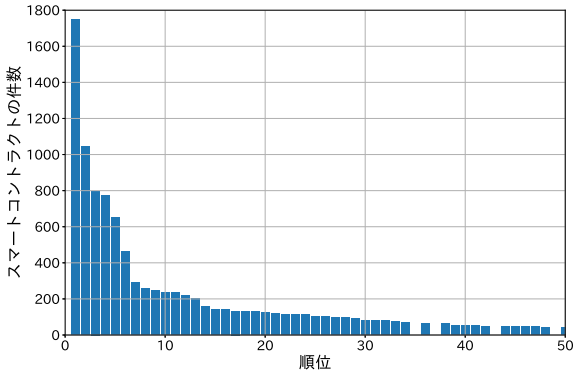

図 7: 同一オペコードの件数と順位の関係

表 4: 複雑さの増加量 (率)

	1 月あたり	1 年あたり
増加量	28	332
増加率	0.014	0.162

3.6. オペコードが同一なスマートコントラクトの分析

オペコードが同一なスマートコントラクトの分析を行った．オペコードが同一なスマートコントラクトの件数と順位の関係を図 7 に示す．また，オペコード全体が完全に一致していないにも関わらず，各オペコードの出現回数が同一であるスマートコントラクト （以降「酷似スマートコントラクト」と呼ぶ.） が存在した．本研究で調査した 50,998 件のスマートコントラクトのうち，32,318 件のスマートコントラクトはオペコードが固有（ユニーク）であった．また，26,934 件のスマートコントラクトは各オペコードの出現回数が固有であった．オペコードが同一であれば各オペコードの出現回数も同じになる．一方で，オペコードが完全に一致していなくても，そのスマートコントラクトが機能面で酷似しているときに各オペコードの出現回数が一致することがある．酷似スマートコントラクトの件数は，オペコードが固有なスマートコントラクトと各オペコードの出現回数が固有なスマートコントラクトの差である，5,384 件であった．オペコード全体で比較したスマートコントラクトの件数を表 5，各オペコードの出現回数で比較したスマートコントラクトの件数を表 6 に示す．同一なオペコードを持つスマートコントラクトには，名前も同一で，どのようなスマートコントラクトなのかが明解なものがある．一方で，名前が異なっているものも存在した．このようなスマートコントラクトの機能を理解するには，ソースコードを読み取るための専門的な知識を要するため，利用しにくいものになってしまっている．

4. まとめ

本研究では，Ethereum ネットワークにおけるスマートコントラクトの経時変化に関する分析を行った。作成数の分析から，スマートコントラクトの作成数は最近までは増加傾向にあったが，現在は減少傾向にあることを示した．また，オペコードの出現回数を用いてスマートコントラクトの複雑さを評価することで，スマートコントラクトが複雑化していることも示した．作成数が減少傾向にある一方で，スマートコントラクトは作成数に関係なく複雑化している．このことから，さまざまなスマートコントラクトを大量に作成する潮流から，ひとつひとつのスマートコントラクトの機能性に重点をおいて作成されるような潮流に変化してきていると考えられる．

また，取引回数の分析から，ほとんど取引を行っていないスマートコントラクトが多く存在していることを示した．ほとんど取引を行っていないスマートコントラクトには，もともと個人的な目的で作成され，他人に使用されることを想定していないと思われるものも存在していると考えられるが，便利でなく今後もほとんど使用されないと予測できるスマートコントラクト，つまり，使用しにくいスマートコントラクトが多く存在していると考えられる．また，取引回数が少ないスマートコントラクトの割合と作成月の関係から，作成されてから時間が十分に経っているが取引回数が少ないスマートコントラクトが多く存在していることを示した．このような，他人に使用されることを想定されていないスマートコントラクトや，使用しにくいスマートコントラクトの存在は利用者がスマートコントラクトを使用する上での煩わしさに繋がっていると考えられる．

さらに，同一オペコードを持つスマートコントラクトの分析から，同一なオペコードを持つスマートコントラクト，各オペコードの出現回数が同一であるスマートコントラクトが多く存在していることを示した．今回の研究で使用したスマートコントラクトの中には，名前が同一である一方でオペコードが異なるスマートコ

表 5: オペコードで比較したスマートコントラクトの件数

	オペコードが固有でない	オペコードが固有
件数	18680	32318
割合	0.366	0.634

表 6: 各オペコードの出現回数で比較したスマートコントラクトの件数

	各オペコードの出現回数が固有でない	各オペコードの出現回数が固有
件数	24064	26934
割合	0.472	0.528

ントラクト，名前は異なっているが互いにオペコードが同一であるスマートコントラクトが存在していた．これらのスマートコントラクトの存在は，スマートコントラクトを使用する上で，Ethereum や，スマートコントラクトに関する専門的な知識を必要としていることを示しており，知識がない人には，自分の目的と一致するスマートコントラクトを使用する妨げになっていることを示している．

4.1. 今後の課題

本研究では，50,998 件のスマートコントラクトに関して分析を行った．図 3，図 6 からわかるように，1 ヶ月分のスマートコントラクトが少数であると，グラフが不安定になり，どのような傾向があるか捉えることができない．Soska ら [6] は，多数のエンドポイントから収集したデータを用いて，日々作成され続ける数億ファイルの分析を行った．ブロックチェーンの性質上，一度承認されたスマートコントラクトは削除することができない．そのため，スマートコントラクトの件数は今後も増え続けると考えられる．したがって，将来スマートコントラクトが膨大な数になることが予想される．このような問題を解決するためには，本研究に用いたスマートコントラクトの件数より多くの件数を対象とした分析が課題としてあげられる．

また，本研究で行った経時変化に関する分析と，関連研究 [5] のようなスマートコントラクトの分類を組み合わせることにより，スマートコントラクトの潮流を分析することができると考えられる．さらに，悪性か否かの分類が可能となれば，スマートコントラクトによる犯罪を検知することができ，より安全にスマートコントラクトを利用できる．一方，本研究ではスマートコントラクトをその役割や作成された目的について分類せず包括的な分析を行ったが，悪性のものではなくても，必然的にソースコードが同一なものや酷似しているもの，取引をほとんど行わないスマートコントラクトも存在している．役割や目的に関する分類を行えば，作成者の意図したこととは違った振る舞いをしているスマートコントラクトを見つけ出すことができると考えられる．このようなことを可能にするために，スマートコントラクトを分類する技術が必要になる．

謝辞

本研究の一部は、日本学術振興会における科学研究費補助金基盤研究 (B)(課題番号 17H01742) による支援を受けている。ここに記し謝意を表す。

参考文献

[1] Weili Chen et al. “Detecting Ponzi Schemes on Ethereum: Towards Healthier Blockchain Technology”. In: *Proceedings of the 2018 World Wide Web Conference*. WWW ’18. Lyon, France: International World Wide Web Conferences Steering Committee, 2018.

[2] Nicola Atzei, Massimo Bartoletti, and Tiziana Cimoli. “A Survey of Attacks on Ethereum Smart Contracts SoK”. In: *Proceedings of the 6th International Conference on Principles of Security and Trust - Volume 10204*. 2017.

[3] Ilya Grishchenko, Matteo Maffei, and Clara Schneidewind. “Foundations and Tools for the Static Analysis of Ethereum Smart Contracts”. In: *Computer Aided Verification - 30th International Conference, CAV 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings, Part I*.

[4] Sergei Tikhomirov et al. “SmartCheck: Static Analysis of Ethereum Smart Contracts”. In: *1st IEEE/ACM International Workshop on Emerging Trends in Software Engineering for Blockchain, WETSEB@ICSE 2018, Gothenburg, Sweden, May 27 - June 3, 2018*.

[5] Massimo Bartoletti and Livio Pompianu. “An empirical analysis of smart contracts: platforms, applications, and design patterns”. In: *CoRR* ().

[6] Kyle Soska et al. “Automatic Application Identification from Billions of Files”. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’17. 2017.