社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

ネットワークルーティング攻撃に対する ブロックチェーンシミュレーション

松岡　主馬†　　鈴木　常彦†

† 中京大学工学研究科　〒470–0393 愛知県豊田市貝津町床立 101
E-mail: †t31907m@m.chukyo-u.ac.jp, ††tss@suzuki.sist.chukyo-u.ac.jp

あらまし　仮想ネットワークをプログラミングできる Ruby ライブラリ VITOCHA を Python で再実装した．Fika と名付けたそのライブラリは従来の VITOCHA の機能に相当するライブラリと，ブロックチェーンを用いたアプリケーションを開発するためのライブラリを持っている．これを用いてネットワークルーティング攻撃によるブロックチェーンへの影響をシミュレーションできるアプリケーションを作成した．Fika はブロックチェーンアプリケーションの開発環境を提供する．

キーワード　ブロックチェーン，VITOCHA，Fika，ネットワーク

Blockchain simulation against network routing attacks

Kazuma MATSUOKA† and Tsunehiko SUZUKI†

† Master's program of Graduate School of Engineering, Chukyo University　101 Tokodachi, Kaizu-cho, Toyota-shi, Aichi, 470–0393 Japan
E-mail: †t31907m@m.chukyo-u.ac.jp, ††tss@suzuki.sist.chukyo-u.ac.jp

Abstract　We re-implemented a Ruby library "VITOCHA", which can be used for programming virtual networks, using Python. The library named "Fika" has a library corresponding to conventional functions of VITOCHA and a library for developing blockchain applications. Using this, we have created an application that can simulate the effects of network routing attacks against a blockchain. Fika provides an environment for developing blockchain applications.

Key words　blockchain, VITOCHA, Fika, network

1. ま え が き

2009 年，信頼できる第三者機関を必要とせず，取引を行う二者間で直接ネット上で支払いを可能にする仮想通貨 Bitcoin が誕生した [1]．Bitcoin は取引情報をネットワークの参加者と共有するために分散型台帳システムが使われ，それが後にブロックチェーンと呼ばれるようになった．その後，ブロックチェーンは金融以外の様々な分野でも応用，研究がされてきた．

本研究では，ブロックチェーンを持ったノード（コンピュータ）で構成された仮想ネットワークを 1 台のマシンの中で自在に構築できる Python ライブラリ Fika を開発した．Fika は 2 つのライブラリ群で構成されている．1 つは仮想ネットワークを構築するするためのネットワーク構成ライブラリ（4.1 節）である．2 つ目はブロックチェーンを使ったアプリケーションを開発するためのブロックチェーンライブラリ（4.2 節）である．ブロックチェーンライブラリは再利用可能であり，ユーザはこれを用いて独自のブロックチェーンアプリケーションを書くことができる．本論文では Fika のライブラリを利用して，ネットワークルーティング攻撃 [2] によるブロックチェーンへの影響をシミュレートするアプリケーションを作成した．

本論文の構成は以下のようになっている．第 2 章ではブロックチェーンの概要について述べる．第 3 章では先行研究について述べる．第 4 章では Fika についての解説を行う．第 5 章では Fika の使用例として，2017 年に Maria Apostolaki らが示した仮想通貨システムに対するネットワークルーティング攻撃のシミュレーションを行い，評価と今後の課題について述べる．第 6 章にまとめを述べる．

2. ブロックチェーンの概要

この章では，本論文を読む上で必要なブロックチェーンの概要について述べる．ブロックチェーンは取引情報をネットワークの参加者と共有するための分散型台帳システムである．ネッ

トワークの各ノード（コンピュータ）は P2P で構成されており，何らかのコンセンサスアルゴリズム（4.2.2 節）を用いて取引情報の整合性を確立している．

2.1 ブロックチェーンの形成

ブロックチェーンは取引情報が格納された「ブロック」という単位のデータを生成し，鎖のように連結していくことでデータを管理するデータベースである．ブロックはネットワーク内の全てのノードで共有し，時系列順に管理されている．新たなブロックを作成することをマイニングと言い，マイニングに成功すると報酬が得られる．ブロックチェーンを持ったノードで構成された P2P ネットワークは，マイニングの報酬をインセンティブの 1 つとして維持されている．

2.2 フォーク

各ブロックのヘッダーは 1 つ前のブロックのヘッダーをハッシュ化したものを持っている．各ブロックの 1 つ前のブロックのことを親ブロックという．各ブロックの親ブロックは 1 つしか存在しないが，一時的に子ブロックが複数存在する場合がある．このようにブロックチェーンが分岐した状態をフォークと呼ぶ．ノードは，自分の管理するブロックチェーンとは異なるブロックチェーンを持っているかどうかを隣接ノードに定期的に確認する．隣接ノードが自分とは異なるブロックチェーンを持っていた場合，フォークが起きていると認識する．

フォークが起きる条件は主に 2 つある．1 つは複数のノードがブロックのマイニングにほぼ同時に成功した場合に起こる．マイニングされたブロックは直ちにネットワーク内の全ノードと共有する必要があるため，各ノードがリレー形式でブロックを全ノードに伝搬させる．ブロックチェーンに含まれるブロックの数（以下，これを長さと呼ぶ）がフォークした各ブロックチェーンで等しい場合はどちらも正当なブロックチェーンとみなされる（図 1）．フォークしたどちらかが他方より長い場合，そのブロックチェーンが正当なものと判断され，他方のブロックチェーンは正当性を失う（図 2）．2 つ目はプロダクトの仕様変更により発生する．前者の場合は，いずれ正当なブロックチェーンは 1 つに決定される．本論文では，これをブロックチェーンが収束すると表現する．本論文で特に指定がなくフォークと言う場合は，前者を指す．

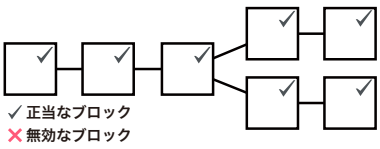

図 1　ブロックチェーンのフォーク
Fig. 1　Blockchain forks.

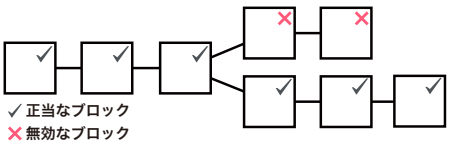

図 2　ブロックチェーンの収束
Fig. 2　Blockchain convergences.

3. 先行研究

この章では，Fika の一機能となっているライブラリと既存のブロックチェーンシュミレータに関する先行研究について述べる．

3.1 VITOCHA

VITOCHA [5] は FreeBSD の OS パーティショニング機能である jail とネットワーク仮想化機能 VIMAGE を操作し，自由に仮想ネットワークを 1 台の仮想マシンの中で構築できる Ruby ライブラリである．jail は 1 つの FreeBSD を仮想的に複数の FreeBSD にパーティショニング（隔離）する技術である．VIMAGE は jail が提供する隔離機構に加えて，ネットワークスタックやルーティングテーブルなどの隔離も行ってくれる機構である．Fika はこのライブラリを Python で再実装しており，ネットワーク構築機能を継承している．

3.2 SimBlock

SimBlock [4] はブロックチェーンネットワーク(注1)を模擬するソフトウェアである．SimBlock ではノードやネットワーク，ブロックチェーンの性質をパラメタで簡単に調整することが可能である．また，SimBlock は可視化機能も備えておりシミュレーション結果を Web ブラウザ上で確認することができる．これらによって，Bitcoin などの既存のシステムや独自に考案したブロックチェーンを PC 上でシミュレートすることで，その性能や安全性を検証できる．

SimBlock は地理的なノード分布（例：ヨーロッパ x%，アメリカ y% など）を設定することができるが，ネットワークの詳細は設計することができない．対して，Fika では物理的な機器のネットワークやブロックチェーンネットワークにおけるノード間の接続関係の設計が可能である．また，実際にパケットを構築した仮想ネットワークに流しているためブロックチェーンが形成される様子をリアルタイムに観察することができる．

4. Fika の構成

この章では，Fika の構成について解説する．Fika は役割によって分けられた 2 つのライブラリ群で構成されている．

4.1 ネットワーク構成ライブラリ

このライブラリは VITOCHA（3.1 節）の機能を継承しており，FreeBSD の jail を用いて仮想ネットワークを 1 台のマシンの中で構築することができる．各 jail は仮想ネットワークにおけるルータ，ブリッジ，そしてブロックチェーンネットワークに参加するノードのいずれかとして機能する．

このライブラリを実行し仮想ネットワークを構築すると，接続されている機器同士は epair と呼ばれる仮想 LAN ケーブルで結ばれ両端にそれぞれ IP アドレスが割り振られる．ブロックやブロックチェーンの情報は，各機器のインターフェースに設けられた IP アドレスとポート番号を用いてソケット通信を行うことで送受信されている．

(注1)：ノード間の繋がりのみを示したネットワーク

4.2 ブロックチェーンライブラリ

このライブラリではブロックチェーンアプリケーションを開発するために必要な要素を定義している．ライブラリの構成を簡潔に示すと図 3 のようになっており，ライブラリ内の各プログラムが図の楕円部分を定義している．

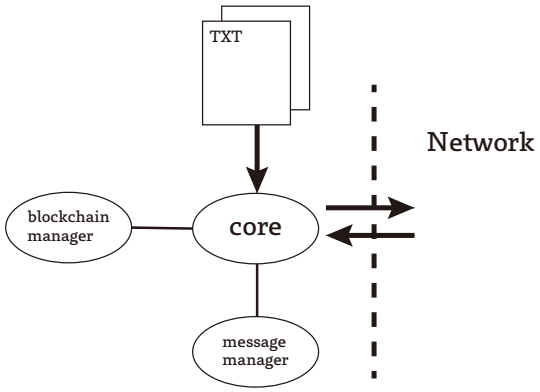

図 3　ブロックチェーンライブラリの構成図
Fig. 3　Blockchain library configuration diagram.

core はノードの窓口となっており，core を通じてノード間で通信を行っている．core は起動時に，同じディレクトリに置かれたテキストファイルから自分が接続する他ノードの IP アドレスを得ている．message manager はメッセージ（4.2.1 節）管理を行っており，受け取ったパケットの解析と他ノードに送るメッセージを作成する機能を備えている．blockchain manager はノードの持つブロックチェーンを管理することに加えて，ブロックのマイニングを行ったり，他ノードから受け取ったブロックやブロックチェーンの検証をする機能（4.2.2 節）を備えている．

パケットがノードの core に届いたとする．core は内容を解析するためにそのパケットを message manager に渡す．パケット解析の結果を再び core に返し，core は解析結果に応じて処理を行う．例えば，他ノードから新しいブロックが届いた場合，blockchain manager にそのブロックに関するデータを渡してブロックの正当性を検証をする．blockchain manager はそのブロックを正当と判断した場合，自分の管理するブロックチェーンに追加して処理が正常終了したことを core に伝える．正常終了したことを確認した core は，そのブロックを自分と接続している他ノードに伝えるためにメッセージの作成を message manager に依頼する．message manager は作成したメッセージを core に返す．最後に，core はそのメッセージを自分と接続している他ノードたちに送る．

4.2.1 ノード間メッセージ

ノードは他ノードとのやり取りのためのメッセージタイプとして MSG_BLOCK，MSG_CHAIN，MSG_RSP_CHAIN(注2) の 3 つを備えている．

ノード A がブロックのマイニングに成功したことを想定する（図 4）．ノード A はブロックチェーンネットワーク上で接続しているノード B にマイニングしたブロックを送信する．MSG_BLOCK はブロックを他のノードに伝達するときに用いるメッセージタイプである（図 4 : ①）．ノード B は受け取ったブロックの正当性を検証（4.2.2 節）し，正当と判断した場合次の処理を実行する．自分の管理するブロックチェーンに追加可能な場合はそのまま追加し，処理を終了する．そうでない場合，自分の管理するブロックチェーンとは異なるブロックチェーンを送り主であるノード A が持っている可能性があると判断する．ノード B はノード A に対して管理しているブロックチェーンを送るように要求する（図 4 : ②）．MSG_CHAIN は MSG_BLOCK を送ってきた相手に，管理しているブロックチェーンを送信するように要求するためのメッセージタイプである．MSG_CHAIN を受け取ったノード A は自分の管理するブロックチェーンを返信する（図 4 : ③）．MSG_RSP_CHAIN は MSG_CHAIN メッセージの返信用のメッセージタイプである．ノード B は受け取ったブロックチェーンと自分の管理しているものとの長さを比較する．もしノード A から受け取ったブロックチェーンが自分の管理するものよりも長い場合，ノード B は自分が管理しているブロックチェーンをそれに更新する必要がある．

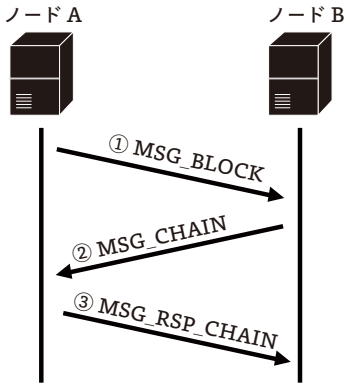

図 4　ノード間メッセージの例
Fig. 4　An example of messages between nodes.

4.2.2 コンセンサスアルゴリズム

このライブラリではブロックの正当性を判断するコンセンサスアルゴリズムとして PoW（Proof-of-Work）[3] を実装している．PoW ではマイニングの対象となるブロックのヘッダーとナンス（nonce）と呼ばれるものを結合してハッシュ値を計算し，そのハッシュ値があらかじめ決められた値よりも小さくなればそのブロックは正当なものと判断される．各ノードはハッシュ値を計算した際に基準値を下回るようなナンスを見つける必要があるため，ナンスの値を 1 から順にインクリメントしながらハッシュ計算を反復実行している．条件に合うナンスを見つけた場合，そのナンスをブロックのヘッダーの要素に追加し，ブロックを他ノードに伝搬する．他ノードから受け取ったブロックの正当性を検証する場合は，ブロックのヘッダーからナンスを取り出し，ハッシュ計算を行いその値が基準値よりも小さいかどうか確認すれば良い．PoW の特徴はナンスを見つけるには大量のハッシュ計算を試みる必要があるが，検証は 1

(注2) : RSP = Respose

回のハッシュ計算で可能ということである．

5. シミュレーション

この章では，Fika の使用例を示す．2017 年に Maria Apostolaki らが示した仮想通貨システムに対するネットワークルーティング攻撃手法の 1 つ Partition 攻撃（5.2 節）をシミュレーションするアプリケーションを作成した．

5.1 BGP 経路ハイジャック

インターネットでは，自分の管理するネットワーク情報を広告することで，他のネットワークにそのネットワークまでの経路を知らせている．BGP（Border Gateway Protocol）は AS（Autonomous System）間で経路情報を交換するルーティングプロトコルである．悪意のある攻撃者が BGP の設定によって自分の管理下にない他のネットワーク情報を広告すると，そのネットワークへのパケットが攻撃者に届くようになる．これを本論文では，経路を乗っ取ると表現する．このように BGP を用いてあるネットワークの経路を乗っ取ることを BGP 経路ハイジャックと言う．

5.2 Partition 攻撃

Partition 攻撃は BGP 経路ハイジャックによってブロックチェーンネットワークを相互に到達できない 2 つ以上のネットワークに分離させる攻撃である．分離された各ネットワーク内のノードがその外部のノードと通信できないようにすることで，分離されたネットワーク内のみでブロックチェーンを作成していくことになる．その結果，攻撃者は意図的にブロックチェーンのフォークを引き起こすことができる．攻撃者が攻撃をやめると分離していたネットワークが再び 1 つになるため，ノードたちはブロックチェーンがフォークしていたことに気づく．その後，互いのネットワークで作成されたブロックチェーンの長さを比べることでフォークしたブロックチェーンが 1 つに収束する．その結果，短いブロックチェーンは正当性のないものと見なされる．攻撃中にその短いブロックチェーンを作っていたノードたちの取引やマイニングによる報酬は無効となる．もし攻撃者が短いブロックチェーンを作成したネットワークに属していた場合，自分の取引を無効にすることが可能となる．

5.3 攻撃手順

今回のシミュレーションで用いる仮想ネットワークは図 5 のようになっている．ブロックチェーンネットワークにおける各ノードの接続関係を図 5 の①で示している．図の中央より右側のネットワーク（ISP6，ISP7，ISP8 に属するノード群）と左側のネットワーク（ISP1，ISP2 に属するノード群）を分離するように攻撃を行う．Partition 攻撃の具体的な攻撃手順は以下の通りである．

（1） 攻撃者（ISP4）は BGP 経路ハイジャックを行うことで，右側のネットワークから左側のネットワークへ向かうパケットが自分を通過するように引き寄せる（図 6）．ネットワーク構築時（図 6）は，左右のネットワークを行き来するパケットがどの ISP を通過するかはわからないが，攻撃を行うことで確実に ISP4 を通過するようにできる．

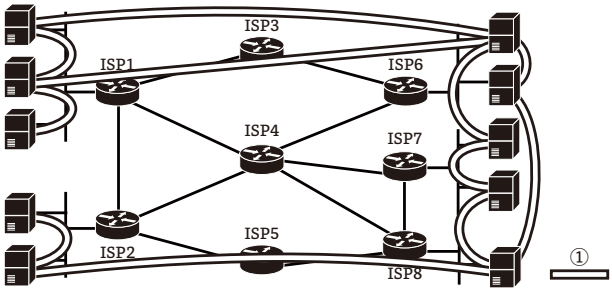

図 5 初期ネットワーク構造
Fig. 5 Initial network structure.

（2） 攻撃者は分離させたいネットワーク間のパケットを受け取るようになる．その後，攻撃者は中継したパケットを全て破棄することでネットワーク間の通信の到達性がなくなる（図 7）．攻撃中，各ノードは分離した同じネットワーク内の他ノードとのみ通信を行うことができるが，ネットワークが分離したことになる．

（3） 攻撃中はブロックを作成しても分離したネットワーク間で伝搬することができないため，そのブロックの存在を認識することができない．分離した各ネットワーク内のみでブロックチェーンを作成するためフォークが発生する．攻撃を止めると，分離したネットワーク間で再び通信を開始する．その時にノードたちはブロックチェーンのフォークを認識し，やがて 1 つのブロックチェーンに収束する．

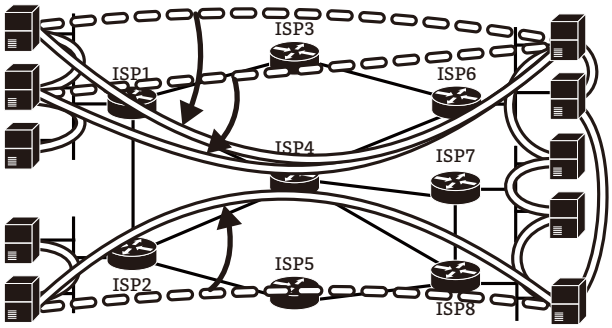

図 6 攻撃ステップ 1
Fig. 6 The 1st step of the attack.

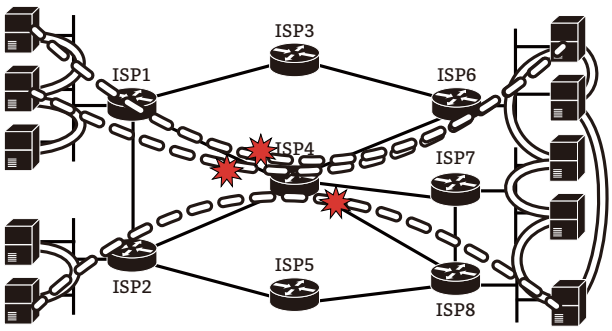

図 7 攻撃ステップ 2
Fig. 7 The 2nd step of the attack.

ノードを停止させると，その時点で管理していたブロック

チェーンをテキストファイルに書き出しを行う．分離したネットワークのブロックチェーンを見比べると，ある時点を境に内容が異なる（フォークしている）ことが確認できる．nonce と previous_hash の値を見ると height が 10 のブロックからフォークしていることがわかる（図 8，図 9）．これによって，Partition 攻撃によって意図的にブロックチェーンのフォークを引き起こすことが可能だと確認できた．


```
[
    {
        "height": 1,
        "nonce": 0,
        "previous_hash": "This is genesis block!",
        "timestamp": 0
    },
    {
        "height": 2,
        "nonce": "4422",
        "previous_hash":"e4a9e6d1bee0344151641e86058c33841978a7286f55fbd688c1c3f195088d19",
        "timestamp": 1570302775.3496032
    },
    ...
    {
        "height": 9,
        "nonce": "121736",
        "previous_hash":"2bb0f53cafdfc405a42ea214a3fd43efcc9f1660c1a386a0650804fe72ff1420",
        "timestamp": 1570302822.3661394
    },
    {
        "height": 10,
        "nonce": "18165",
        "previous_hash":"2405a2f73b2c2be2fc789466010ea4f92b03f90177700b32df6925f7a1787173",
        "timestamp": 1570302826.729632
    },
    {
        "height": 11,
        "nonce": "173814",
        "previous_hash":"5fe33d17d1bb7b6d48ee74477c15d4ac1052e5236d6fe46d2cabadab20c12621",
        "timestamp": 1570302830.773725
    }
]
```


図 8　ノード A のブロックチェーン

Fig. 8　A blockchain which node-A has.


```
[
    {
        "height": 1,
        "nonce": 0,
        "previous_hash": "This is genesis block!",
        "timestamp": 0
    },
    {
        "height": 2,
        "nonce": "4422",
        "previous_hash":"e4a9e6d1bee0344151641e86058c33841978a7286f55fbd688c1c3f195088d19",
        "timestamp": 1570302775.3496032
    },
    ...
    {
        "height": 9,
        "nonce": "121736",
        "previous_hash":"2bb0f53cafdfc405a42ea214a3fd43efcc9f1660c1a386a0650804fe72ff1420",
        "timestamp": 1570302822.3661394
    },
    {
        "height": 10,
        "nonce": "65708",
        "previous_hash":"2405a2f73b2c2be2fc789466010ea4f92b03f90177700b32df6925f7a1787173",
        "timestamp": 1570302831.4974113
    },
    {
        "height": 11,
        "nonce": "121189",
        "previous_hash":"25de6d98f446d3c1030147230c0ac26d72674e486198fccec50a26bc4e853950",
        "timestamp": 1570302836.7626328
    }
]
```


図 9　ノード B のブロックチェーン

Fig. 9　A blockchain which node-B has.

5.4　評価と課題

Fika は仮想ネットワークにおけるルータやノードなどの機器のネットワーク設計に加えて，ブロックチェーンネットワークの設計，ルータのルーティング設定も必要とするため SimBlock に比べてシミュレーションに多くの手間がかかる．しかし，ネットワークを設計し実際にパケットを流すことができる点が Fika のメリットであり，それによって従来のライブラリでは再現されていない，攻撃による影響のシミュレーションを可能にしている．今回のシミュレーションではメモリを約 120MB 消費していた．また，ブロック 1 つを生成するにあたり，増加するブロックチェーンのデータと出力するログ等で約 24KB のメモリ消費をしていた．SimBlock は数万ノードが動いているネットワークを想定したブロックチェーンのシミュレーションが可能であるが，Fika ではメモリの消費効率が悪いため大量のノードを動かすことができないことが課題の 1 つである．

現段階の Fika は，ブロックチェーンネットワークにおけるノード間の接続関係（図 5：①）を静的に設定している．改善点として各ノードが新たに他ノードと接続したり，既存の接続が切断されたときは代わりの接続先を探すなどの動的な接続管理を行う機能が必要だと考えている．動的な接続管理ができると，Partition 攻撃だけでなく Eclipse 攻撃 [6] などのブロックチェーンネットワークにおけるノード間の接続関係を利用した攻撃手法についてのシミュレーションも可能となる．

ユーザはブロックチェーンライブラリ（4.2 節）の各プログラムで定義されているクラスを継承して再定義することで，ブロックの要素やメッセージタイプを変更，または追加定義することができる．これによって仮想ネットワーク内にユーザ独自のブロックチェーンを実装することが可能である．しかし現段階のブロックチェーンライブラリは，各プログラムが互いに強く依存するような設計になっているため，アプリケーション開発をするためのフレームワークとして活用することが難しい．そこで，各プログラム間の依存度が少ない設計に修正し，ブロックチェーンを用いたアプリケーション開発を行うフレームワークとして十分に活用できるライブラリに改善していきたい．

6.　ま　と　め

ブロックチェーンという技術が登場して約 10 年が経ち，Web や書籍等で様々な情報を得ることが可能となった．しかし，主な仕組みを大まかに理解できても実践的に学習する環境が存在しなかった．Fika はブロックチェーンを持ったノードで構成されたネットワークを 1 台のマシンの中で自在に構築することができる．また，ブロックチェーンライブラリを再利用することでブロックチェーンアプリケーションを開発する環境を提供する．今後の課題を解決することにより，Fika をより良い学習ライブラリとしていきたい．

文　　献

[1] Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", 2008.

[2] Maria Apostolaki, Aviv Zohar, Laurent Vanbever, "Hijacking Bitcoin: Routing Attacks on Cryptocurrencies", 2017.

[3] "Proof of Work", https://en.bitcoin.it/wiki/Proof_of_work

[4] Yusuke Aoki, Kai Otsuki, Takeshi Kaneko, Ryohei Banno-yand, Kazuyuki Shudo, "SimBlock: A Blockchain Network Simulator", 2019.

[5] 鈴木常彦, "仮想ネットワーク構築ライブラリ VITOCHA とネットワーク技術者教育", 2018.

[6] Ethan Heilman, Alison Kendler, Aviv Zohar, Sharon Goldberg, "Eclipse Attacks on Bitcoin’s Peer-to-Peer Network", 2015.