Vol.2019-IOT-44 No.43
2019/3/8

Tangle の分裂攻撃に対する安全性検証

板倉悠馬[†1]　猪俣敦夫[†2]

概要：Tangle とは仮想通貨 IOTA で採用されている有向非巡回グラフ構造を応用したブロックチェーンに代わる独自技術である．Tangle はブロックチェーンが抱えるスケーラビリティ問題を解決するが，IOTA の公開日から 2 年半程であり，攻撃リスクの検証はブロックチェーン技術と比較すると不十分である．本研究では IOTA 財団が発行した White Paper で攻撃の可能性として述べられている分裂攻撃に着目し，分裂攻撃のリスクを検証する．検証では実際の Tangle に近づけたシミュレータに対して分裂攻撃を行い，ネットワーク全体に対する攻撃者の計算能力ごとの成功確率をまとめた．また，Tangle の構造に大きく影響を与えるパラメータを変更し，成功確率の変化を比較した．検証の結果，Tangle の構造と攻撃者の計算能力による攻撃成功率の変化傾向が明らかになった．

キーワード：Tangle，仮想通貨，分裂攻撃

1. はじめに

Bitcoin の基幹技術として登場したブロックチェーンは P2P ネットワークにおける合意形成を可能にし，銀行や IoT を初めとした様々な分野への応用が期待されている．しかしながら，ブロックチェーンにはトランザクション数の増加の際に，ブロックサイズがボトルネックとなり処理の遅延やマイナーに支払う取引手数料が高騰するスケーラビリティ問題が存在する．このスケーラビリティ問題を解決する手法の 1 つが DAG（Directed Acyclic Graph）技術を使用した仮想通貨 IOTA の Tangle[1]である．IOTA 財団が発行した White Paper[2]では，Tangle に対する数種類の攻撃の可能性が検討されている．本論文ではその中から Aviv Zohar が提唱した分裂攻撃[2]に着目する．分裂攻撃は正確な累積荷重の認識が困難であることから実現可能性が低いと考えられ，検証が十分に行われていない．そこで，本研究では分裂攻撃の脅威を認識するために，用意したモデルに様々な条件下で分裂攻撃を行う．攻撃の成功率を比較し，Tangle の分裂攻撃に対するリスクが変動する条件を明らかにすることを目標とする．

2. Tangle

本稿では Tangle における重要な概念と分裂攻撃について説明を行う．

2.1 Tangle

Tangle は大量のトランザクションを想定した有向非巡回グラフであり，図 1 に示す構造をとる．どのトランザクションからも承認を受けていないトランザクションはチップと呼ばれる．ノードが発行したトランザクションを承認してもらうためにはまず，先に発行されている 2 つのチップを承認する必要がある．承認作業では選択した 2 つのチップが矛盾する取引でないかを確認し，ブロックチェーンと同様に暗号パズルを解く．このように Tangle では承認作業を発行者自身が行い，トランザクション単位で記録していくことで，スケーラビリティ問題を解決する．

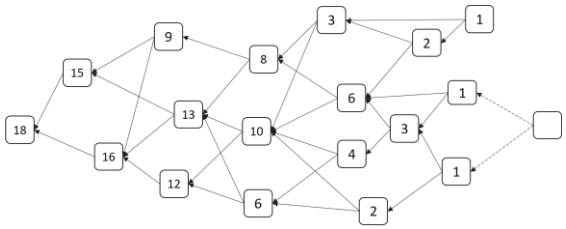

図 1　Tangle の構造と累積荷重

2.2 累積荷重

トランザクションにはそれぞれ荷重が存在し，IOTA では取引の金額の大きさといった指標から荷重を決定する．トランザクション自体が持つ荷重と間接的に承認するトランザクションの荷重の合計は累積荷重と呼ばれ，図 1 のそれぞれのトランザクションに書かれた数字である．累積荷重はチップ選択アルゴリズムの重要な指標となる．

2.3 信頼度

Tangle における取引の信頼の指標であり，累積荷重と並ぶ重要な概念である．自身より後に発行された重み付けされたチップに直接・間接的に承認されることで数値が増加し，0 から 1 の区間で増減する．

2.4 チップ選択アルゴリズム

IOTA では非中央集権の構想と相容れないという理由からチップ選択アルゴリズムの強制は行っていない[3]．そのためユーザは承認するチップを自由に選択できるが，Tangle の同期を怠り，古いチップばかり承認する怠惰なノードの出現を防ぐマルコフ連鎖モンテカルロアルゴリズム（以降 MCMC アルゴリズムと表記）をチップ選択アルゴリズムとして推奨している．MCMC アルゴリズムでは承認されてから時間が経っているトランザクションのいずれかを選択し，トランザクションの累積荷重に基づいた遷移確

†1 東京電機大学
†2 東京電機大学

率に従いながらチップに辿り着くまで遷移を行う．この遷移は重み付けランダムウォークと呼ばれ，辿り着いたチップを承認するチップとして選択する．もしチップ選択時に矛盾した 2 つのチップを選択した場合は MCMC アルゴリズムを複数回繰り返し，辿り着いた回数が多いチップを承認するチップとして選択する．

2.5 分裂攻撃

分裂攻撃の概要と考えられている対策について本稿で説明する．

2.5.1 攻撃の概要

一般ノードが矛盾するトランザクション同士の承認を避けることを利用して Tangle を分裂させる攻撃である．攻撃者は分裂の起点となる矛盾するトランザクション（図 2 の黒いトランザクション）を発行し，それぞれを自身で承認して信頼度を均等に近づけ続ける．承認には一般ノードも加担するので，50%未満の計算力で分裂状態の維持が可能である．分裂を維持している間，攻撃者はそれぞれの分岐で同じ資金を利用する二重支払いが可能になる．分岐が一方に集中して分裂状態から回復した場合も，捨てられた分岐に加担したトランザクションの信頼度は 0 に近づいていき，多くのトランザクションが取り残されてしまう．分裂攻撃には以上のリスクが存在するが，実際にこの攻撃を成立させる難易度は高い．Tangle 上のトランザクションがネットワーク全体に伝播されるまでにはタイムラグが存在し，攻撃者が正確な累積荷重を認識することは困難であるためである．また，Tangle の同期状況によってノード毎に見えている Tangle が異なるため，MCMC アルゴリズムを採用する一般ノードは攻撃者が維持する Tangle とは異なる遷移確率でチップを選択する可能性が高い．

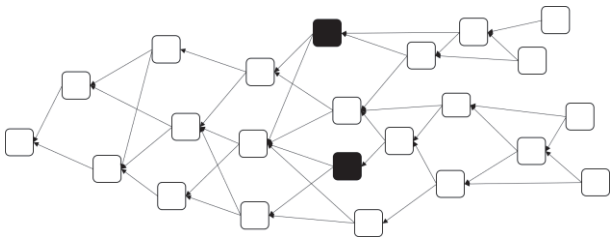

図 2　矛盾する取引を起点とした分裂攻撃

2.5.2 攻撃の対策

重み付けランダムウォークの乱雑さを指定する α[3]の値を高く設定し，閾値を厳密にすることで分裂を維持する難易度が上昇する．重み付けランダムウォークでほぼ同じ値の累積荷重を取る二つのトランザクションのどちらかに遷移を行う場合，累積荷重が大きいトランザクションに遷移する確率が高いほど，攻撃の難易度が上昇し効果的な対策となる．

3. 関連研究

Bartosz Kusmierz らは Tangle 上でトランザクションが取り残される可能性について数値的な推定を行った[4]．α=∞とした粒子は GHOST 粒子と呼ばれ，無限の制限時間内に GHOST 粒子の最終的な遷移先のチップから一度も承認を受けていないトランザクションを取り残されたトランザクションと定義している．実験では α と，単位時間あたりのトランザクションの期待発生回数である λ の値を変更してトランザクションが取り残される割合を求め，結果をグラフで可視化している[5]．λ と α の値が大きくなるほどトランザクションが取り残される確率が高くなることが明らかになり，結論ではトランザクションが取り残される確率を一定にするためには，λ が増加した際に α の値を下げる必要があると述べている．分裂攻撃の対策として有効な α であるが，λ の値によっては α の値を下げる可能性があり，分裂攻撃のリスクを高めてしまうことが判明した．

4. 提案手法

分裂攻撃は実現難易度が高いことに加え，重み付けランダムウォークの閾値に関わる指標 α を大きくすることで効果的な対策が行える．しかし，関連研究でも述べられているように α を大きくすると，取り残されるトランザクションの割合が増加する．また，λ の値が大きくなると取り残されるトランザクションがさらに増加し，一定の承認率を維持するためには α の値を下げざるを得ない．このことから分裂攻撃への対策は常に万全を期せるわけではなく，攻撃のリスクを知る必要がある．そこで本研究では分裂攻撃のリスクを認識するために，シミュレータ上で Tangle に対して分裂攻撃を行う．攻撃者の計算能力や Tangle の形状に関わるパラメータ α，λ の値を変化させ，分裂攻撃の成功率との関係を分析することで，分裂攻撃のリスクが変動する条件とその対策について明らかにすることができると考えた．

5. 実装と実証実験

本稿では分析結果の出力に必要な実装部分の説明と，実験の分析結果の報告を行う．

5.1 実装

分析結果の出力は，シミュレータによるデータの出力と出力結果の分析の 2 段階の作業に分かれる．本稿ではそれぞれの概要を説明する．

5.1.1 シミュレータ

実験で使用するシミュレータは，IOTA 財団が作成した Tangle の承認を可視化するシミュレータ[6]を実験に適した形に変更を加えたものである．図 3 に示すように単一モ

デルの攻撃者と一般ノードがそれぞれ承認を行い，分裂状態を維持できるか観測可能なシミュレータを作成した．

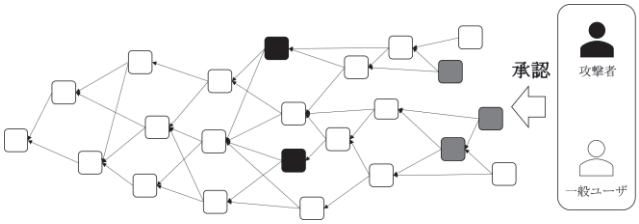

図 3　シミュレータ上で行う承認のイメージ

シミュレータは Javascript で記述されたプログラムであり，ブラウザ上で起動させる．出力結果は分析が行いやすい JSON 形式のファイルをダウンロードして取得する．出力結果に記録する内容を表 1 に示す．

表 1　出力結果に記録する情報

No.	記録内容
1	シミュレート開始時のトランザクション数
2	Lambda
3	Alpha
4	シミュレート終了時のトランザクション数
5	攻撃の失敗位置
6	分裂元トランザクション
7	分裂元トランザクションの信頼度の推移
8	攻撃者の発行したトランザクションの割合
9	攻撃者の発行したトランザクション数
10	一般ノードが発行したトランザクション数

シミュレータ上で各トランザクションが持つ情報はトランザクションが承認したチップや累積荷重，信頼度，発行時間といった最低限のものであり，本来含まれる送信先アドレス，送金額は含まず，トランザクションが矛盾しているか確認する作業や暗号パズルを解く作業も行わない．そのため実際のモデルとは異なるが，パラメータの変化による傾向を知ることは十分可能であるため，情報が省略されたことによる実験への影響は小さい．

5.1.2 分析プログラム

シミュレータによって出力された JSON 形式のデータは，Python で記述したプログラムで分析を行う．各条件において行った分裂攻撃の成功率や，成功・失敗位置の分布が出力される．

5.2 実証実験

本稿では実験の前提条件や実験で変更するパラメータの詳細について説明を行い，実験の結果を報告する．

5.2.1 条件設定

本実験では一般ノードが全て MCMC アルゴリズムに従い，承認するチップを選定するモデルを考える．トランザクション数の初期値は実験に使用するマシンが比較的短時間で生成できる限界である 1000 とし，トランザクションの荷重は全て 1 にする．生成された Tangle に対して一般ノードと攻撃者が同時にリアルタイムでの承認を開始し，条件ごとに 50 個のログを取得する．このモデルでは，トランザクションの発行間隔に偏りがなく一定であるため，実際の環境よりも分裂攻撃の成功率は高くなる．

5.2.2 使用するパラメータ

(1) 計算能力

ネットワーク上の計算能力は実際の環境ではトランザクションの発行頻度，暗号パズルの計算時間など常に変動する要素の影響を受けるため，一定の値を取ることはなく正確に測れるものではない．実験では Tangle の初期状態以降に発行された攻撃者のトランザクションの割合を計算能力とみなし，ほぼ一定の値を取るものとして扱うことで結果の比較を行いやすくする．攻撃者が 50%以上の計算能力を保有する場合，今回実験に用いるモデルはトランザクションの発行回数が一定であるため 100%攻撃が成功する．そのため，検証は 50%未満で行い，良い比較結果が得られると考えられる 45%，40%，35%の 3 段階の計算能力での検証を行う．実験の出力結果は，指定した計算能力と実際に出力された攻撃者のトランザクションの割合との誤差が ±0.5%以内に収まるものを使用する．

(2) Lambda

単位時間当たりのトランザクションの期待発生回数を表すパラメータである．λ の値が大きいほど単位時間あたりに到着するトランザクション数が多くなっていく．λ の値を小さく設定すると図 4 に示すような一本のチェーンに近い低負荷状態の Tangle を形成し，値を大きく設定すると図 5 に示すようなチップが乱立する高負荷状態の Tangle を形成する．λ の初期値は比較的低負荷状態である 1.5 に設定し，λ の値が高くなるにつれて分裂攻撃の成功率がどのように変化するかを比較する．

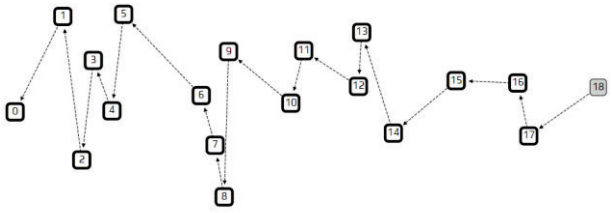

図 4　λ = 0.1，α=0.5 の Tangle

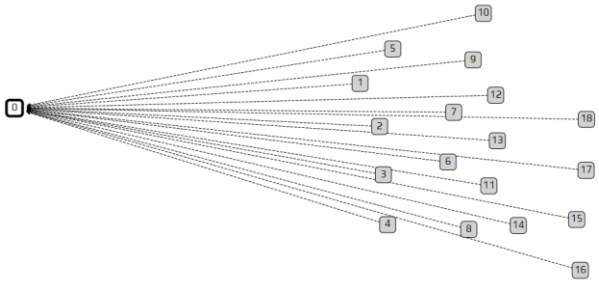

図 5　λ = 20，α=0.5 の Tangle

(3) **Alpha**

MCMC アルゴリズムによるチップ選択時に，累積荷重を重視する度合いを設定するパラメータである．α の値を小さく設定すると累積荷重を無視したランダムな遷移を行い，図 6 のような多くのチップが承認される Tangle を形成する．α の値を大きく設定すると累積荷重の大きいトランザクションを優先した遷移を行うようになり，図 7 のような取り残されるチップの数が多い Tangle を形成する．関連研究で α の値は 0 から 1 の範囲で十分に変化することが明らかになっているため，初期値は中間値である 0.5 を指定する．

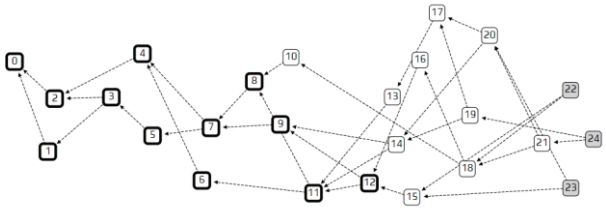

図 6　α =0，λ= 1.5 の Tangle

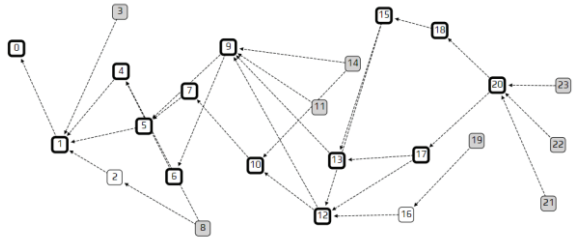

図 7　α = 1，λ= 1.5 の Tangle

5.3 分裂攻撃の手順

(1) **分裂元トランザクションの選定**

実際のモデルでは攻撃者は分裂の起点となる 2 つの矛盾するチップを発行するが，今回使用するシミュレータではトランザクションが送信先や送金額といった情報を持たないので，他の方法をとる必要がある．本実験では発行された最新 15 個のトランザクションから 2 つのチップを選択し，分裂元トランザクションとして扱う．一般ノードは分裂元トランザクションの両方を承認することを避けるように設定することで分裂状態を生み出す．

(2) **分裂の維持**

攻撃者は分裂元トランザクションの信頼度を参照し，それぞれが 0.5 に近づくように調整を行う．調整は信頼度が低いほうのトランザクションを攻撃者が直接・間接的に承認し，信頼度を上げることで実現する．

5.3.1 分裂攻撃の成否判断基準

攻撃の成否の判断は分裂状態を一定回数維持できたかによって行う. トランザクション数が 2000 前後になると計算時間が 1000 の時と比較して何倍にも跳ね上がるため, トランザクション数が 2000 に到達した時点で検証を切り上げる．

(1) **成功条件**

- 500 回連続で分裂元トランザクションの両方の信頼度が 0.1 以上 0.9 以下の値を取り続ける．

(2) **失敗条件**

- 500 回連続で分裂元トランザクションのどちらか一方の信頼度が 0.1 未満もしくは 0.9 より上の値を取り続ける．
- 検証の途中でトランザクション数が 2000 を超える．

5.4 実験結果

実験の結果は表 2 に示す通りとなった．

表 2　分裂攻撃の成功率

	攻撃者の計算能力		
パラメータ	35%	40%	45%
Default	0%	0%	48%
λ=10	2%	22%	50%
λ=20	0%	20%	34%
α=0.1	84%	98%	100%
α=0.9	0%	0%	0%

5.4.1 失敗位置の分布

攻撃者の計算能力ごとの分裂攻撃の失敗位置を図 8, 図 9, 図 10 にまとめた. 本実験は 500 回連続で分裂を維持できなかった場合に失敗と判断するため，失敗と判断される大元となるのは失敗した時点のトランザクション数から 500 を引いた数値である．この数値を失敗位置として定義する．失敗位置はすべて 1000 から 1500 の範囲に分布しており，100 ごとに区切ったものを積み上げ縦棒グラフで表す．

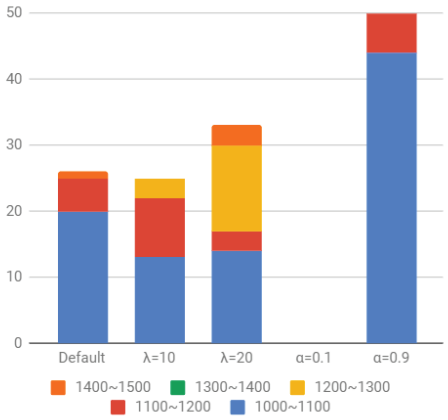

図 8　計算能力 45%の失敗位置

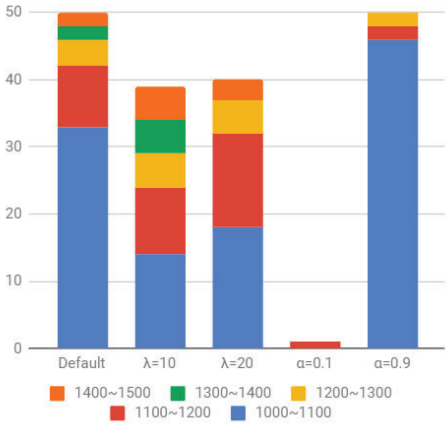

図 9　計算能力 40%の失敗位置

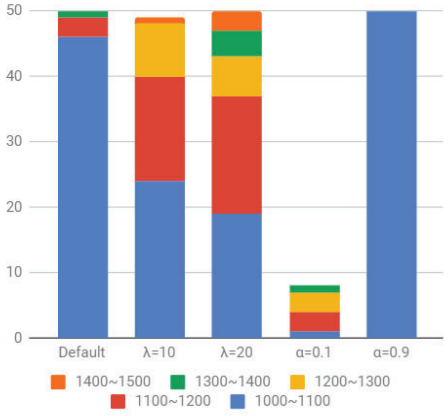

図 10　計算能力 35%の失敗位置

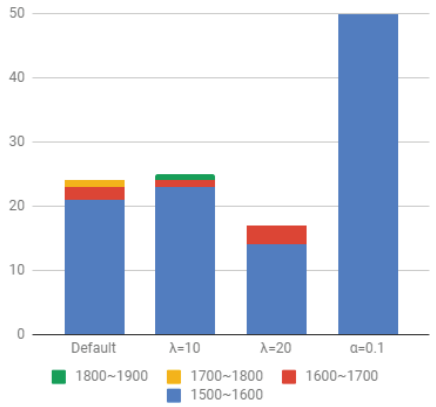

図 11　計算能力 45%の成功位置

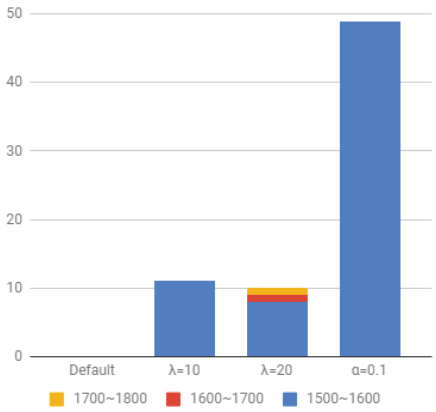

図 12　計算能力 40%の成功位置

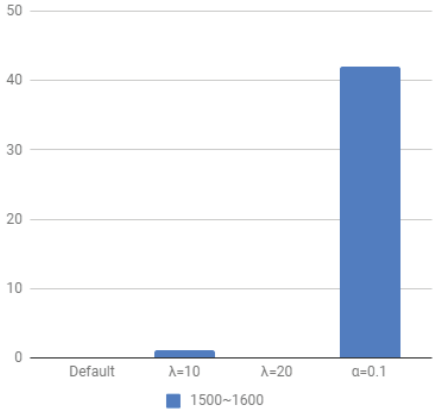

図 13　計算能力 35%の成功位置

5.4.2 成功位置の分布

分裂攻撃が成功した時点のトランザクション数を成功位置と定義し，その位置を図 11，図 12，図 13 に積み上げ縦棒グラフで表した．

6. 考察

6.1 分裂攻撃の成功確率について

Default のパラメータでは攻撃者の計算能力が 45%の状況で 5 割に近い成功率を示した．40%，35%においても λ や α の値の変動で成功例が見られた．特に α の値が増減す

ると 0%や 100%といった極端な数値を出した．このことから，α は分裂攻撃の対策において非常に重要なパラメータであることが分かる．一方 λ では，λ=20 よりも λ=10 の攻撃の成功率が高いという結果が出た．攻撃者の計算能力 45%，λ=20 の条件下で攻撃の成功率は Default よりも低くなっており，λ の増加と攻撃の成功率の関係は単純な比例ではないと考えられる．このような結果になった理由は λ の増加には攻撃の成功率を上げる要因と下げる要因が両方存在するためであると予想する．λ の値が高くなるとチップも乱立するので，攻撃者が分裂元を承認しても信頼度の増加は微々たるものになる．その分，一般ノードが分裂元トランザクションを直接・間接的に承認する確率も低くなることが攻撃の成功率を下げる要因であると思われる．しかし，チップの乱立数が攻撃者の承認によって一般ノードを十分に巻き込む信頼度の上昇が見込める数であれば，攻撃の成功率は上昇すると考えられる．実験の検証回数が十分でなく，値が入れ替わる可能性が高いため，λ について正確な結果を出すためには更なる検証が必要である．

6.2 分裂攻撃の失敗位置について

攻撃の失敗位置は，後になるほど分裂状態を長く維持していることを表す．各計算能力のグラフを比較すると，計算能力の低下によって分裂の維持期間が短くなり，早い段階で攻撃が失敗する傾向にあることが確認できる．今回の実験で α=0.9 の成功確率は全ての計算能力で 0%という結果が出ているが，失敗位置が 1100，1200 を超えるものが 40%，45%のグラフに数例存在することから，計算能力による差が若干ながら存在することが分かる．

6.3 分裂攻撃の成功位置について

すべてのグラフにおいて成功位置の大半を 1500~1600 の範囲が占めており，この結果から分裂攻撃成功可否は最初の段階が非常に重要となることが分かる．1600 を超える成功位置の遷移は，途中で分裂の成功ラインである信頼度が 0.1 以上 0.9 以下の条件を僅かに超えてしまい，その後持ち直すケースがほとんどであった．

7. 課題

今回使用したシミュレータは，パラメータの変化による攻撃の成功率の変化傾向を知ることができるが，多くの部分を簡易化し，攻撃者側が有利な環境となっているため，実際の環境における成功率よりも高い結果が出ている．実際の環境に近づけるためにはトランザクションの発行時間にランダム性を持たせる，ノード毎に見えている Tangle の状態に差を設ける，偶に Tangle の同期によって多数のトランザクションが追加されるといった変更が考えられる．一方で攻撃者の分裂元トランザクション選定はランダムで行っていたため，攻撃の成功率に上昇の余地がある．Tangle が分裂攻撃の成功しやすい形状をとるときに攻撃を行うようなアルゴリズムへの改良が考えられる．検証回数とパラメータの種類が十分でなかったために，λ の変化による分裂攻撃の成功確率の変動について確実な結果を出すことができなかった点については，検証回数とパラメータの種類を増やし，検証の精度を上げていくことで正確な結果を求めていく必要がある．シミュレータを実行すると計算に多くの時間を要するため，分裂の成功条件を 500 回連続の維持としたが，分裂をどこまで維持できるかを長期的に確認するためには，保持するデータの削減をはじめとしたプログラムの改良を行い，計算時間を短縮する必要がある．

8. まとめ

実験によって，Tangle の形状や攻撃者の計算能力が分裂攻撃の成功率に影響を与えることが確認できた．今回の実験では λ の変化による攻撃の成功率の変化は単純な比例関係では表せず，攻撃のリスクを正確に認識することはできなかった．一方で α の変化は攻撃の成功率に顕著な影響を与え，分裂攻撃の対策において重要な要素であることが実験でも確認できた. 今後の改善点として検証回数の増加や，シミュレータをより実際の環境に近づけるといった変更が考えられる．

参考文献

[1] “What is IOTA?”. https://www.iota.org/get-started/what-is-iota, (参照 2019-01-30).
[2] 〝The Tangle”. https://assets.ctfassets.net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3.pdf, (参照 2019-01-30) .
[3] “The Tangle: an illustrated introduction Part 3: Cumulative weights and weighted random walks”. https://blog.iota.org/the-tangle-an-illustrated-introduction-f359b8b2ec80, (参照 2019-01-30).
[4] 〝Probability of being left behind and probability of becoming permanent tip in the Tangle v0.2”. https://assets.ctfassets.net/r1dr6vzfxhev/6FMwUH0b4WIyi6mm8oWWgY/8f1d7b30f7b652098a5e68b6634c63df/POLB-02.pdf, (参照 2019-01-30).
[5] 〝Alpha: playing with randomness”. https://blog.iota.org/alpha-d176d7601f1c, (参照 2019-01-30).
[6] 〝Tangle visualization”. https://github.com/iotaledger/iotavisualization, (参照 2019-01-30).