Conflito de Transações em Redes Bitcoin

Hellan Dellamycow Gomes Viana
Departamento de Ciência da Computação
Universidade Federal da Bahia
Salvador, Brasil
dellamycow@gmail.com

Patrick Silva Ferraz
Departamento de Ciência da Computação
Universidade Federal da Bahia
Salvador, Brasil
patrick.ferraz@outlook.com

***Resumo*—Este relatório é parte do trabalho prático da disciplina MATD74 - Algoritmos e Grafos, semestre 2019.1. Nele aborda-se uma proposta de solução para o problema de transações conflitantes em uma rede Bitcoin, utilizando-se do método de Conjuntos Independentes Máximo, que pertence a classe NP-Completo.**

***Index Terms*—transações, Bitcoin, Conjuntos Independentes, duplo gasto**

I. Introdução

Bitcoin é uma nova moeda criptografada disruptiva baseada em um protocolo descentralizado de código aberto que vem gradualmente ganhando força. Apesar de diversos mecanismos associados a segurança e confiabilidade do funcionamento da cadeia de blocos gerada nesse ecossistema, vários tipos de ataques provenientes do problema de gasto duplo surgiram, e por sua vez uma das principais razões para tal fato é a existência de conflitos entre transações na rede Bitcoin. E é com o sentido de tratar as transações conflitantes que este relatório foi idealizado e está sendo apresentado. Nas seções seguintes iremos discutir sobre o Referencial Teórico, Descrição Formal do Problema, Justificativa de o problema ser NP-Completo, Revisão da Literatura e Metodologia.

II. Referencial Teórico

A. Transação de Bitcoin

Transação, no contexto da criptomoeda Bitcoin (BTC), é uma transferência do valor do BTC que é transmitido para a rede, coletada em blocos, e normalmente fazendo referência a outpus de transações anteriores como novas inputs de transação e dedica todos os valores de BTC de input a novos outputs. As transações tem por característica não serem criptografadas, permitindo a visualização de todas as transações já coletadas em um bloco. Estando as transações já consolidadas sob confirmações suficientes, elas podem ser consideradas irreversíveis.

Os outpus de transação padrão nomeiam endereços e o resgate de quaisquer inputs futuras requer uma assinatura relevante.

Todas as transações são visíveis na blockchain (ou *cadeia de blocos* em português) e podem ser visualizadas com um editor hexadecimal. Um navegador de blockchain é um site em que todas as transações incluídas na cadeia de blocos podem ser visualizadas em termos legíveis por qualquer pessoa, e possibilitam a leitura dos detalhes técnicos das transações e a verificação dos pagamentos.

A figura 1 apresenta o formato generalizado de uma transação Bitcoin dentro de um bloco, trazendo a descrição e tamanhos dos campos.

CAMPO	DESCRIÇÃO	TAMANHO
Versão	atualmente 1	4 bytes
Flag	Se presente, sempre 00001, e indica a presença de dados de Testemunhas	Array de 2 bytes (opcional)
Contador de Inputs	Inteiro Positivo VI = VarInt	1 - 9 bytes
Lista de Inputs	O primeiro input da primeira transação é também chamado de "coinbase"	<in-counter> vários inputs
Contador de Outputs	Inteiro positivo VI = VarInt	1 - 9 bytes
Lista de Outputs	Os outputs da primeira transação que gastam os bitcoins mineirados para o bloco	<ou-counter> vários outpus
Testemunha	Lista de testemunhas, 1 para cada input, omitido se a flag não está presente	varia de acordo com Segregated_Witness
lock_time	se não é zero e a sequência de números é < 0xFFFFFFFF: tamanho do bloco ou timestamp quando é a transação final	4 bytes

Figura 1. Formato generalizado de uma transação Bitcoin (dentro do bloco).

B. Como a mineração funciona e as transações são processadas

O foco desse trabalho não é a mineração de Bitcoin em si, mas as transações (mais especificamente os conflitos entre elas). Contudo, elas estão inseridas no contexto de mineração, e para que se tenha uma melhor observação e conhecimento do processo como todo, sequencialmente é apresentado de forma resumida um passo-a-passo do mecanismo de mineração e como as transações estão intrisecamente inseridas e são processadas.

1) Um usuário tenta enviar uma certa criptografia ou token para outra pessoa, através de uma carteira;
2) A transação é transmitida pela carteira e agora está esperando em um ”conjunto, ou pool, de transações não confirmadas”para ser pego por um minerador no blockchain correspondente.
3) Mineradores na rede, ou nós, selecionam transações desses pools e os formam em um ”bloco”, que é uma coleção de transações (até então ainda não confirmadas), além de alguns metadados extras. Todo minerador constrói seu próprio bloco de transações, podendo vários mineradores selecionar a mesma transação a ser incluída em seu bloco;
4) Ao selecionar as transações, os mineradores criam um bloco de transações. Para adicionar este bloco de transações ao blockchain (para que todos os outros mineradores e nós registrem as transações), o bloco primeiro precisa de uma assinatura (também chamada de prova de

trabalho). Essa assinatura é criada pela solução de um problema matemático muito complexo exclusivo de cada bloco de transações e todos são igualmente difíceis de resolver, sendo necessário poder computacional elevado. Este é o processo conhecido como mineração;

5) O minerador que encontra uma assinatura elegível para o seu bloco primeiro transmite este bloco e a sua assinatura para todos os outros mineradores;
6) Outros mineradores verificam a legitimidade da assinatura pegando a sequência de dados do bloco transmitido e fazendo um hashing para ver se o hash de saída realmente corresponde à assinatura incluída. Sendo válido, os outros mineiros confirmarão sua validade e concordarão que o bloco pode ser adicionado ao blockchain. A assinatura é a "prova"do trabalho realizado (o poder computacional gasto). O bloco agora pode ser adicionado ao blockchain e é distribuído por todos os outros nós da rede. Os outros nós aceitarão o bloco e o salvará em seus dados de transação, desde que as transações dentro do bloco correspondam corretamente aos saldos atuais do histórico de transações naquele momento;
7) Após um bloco ser adicionado à cadeia, todos os outros blocos adicionados em cima agem como uma "confirmação"para esse bloco. Toda vez que outro bloco é adicionado sobre ele, o blockchain alcança o consenso novamente no histórico de transações completo, incluindo sua transação e seu bloco. Quanto mais confirmações a transação tiver, mais difícil será para os invasores alterá-lo. Depois que um novo bloco é adicionado ao blockchain, todos os mineradores precisam selecionar outras transações, formando novos blocos de transações.

III. Descrição Formal do Problema

As transações, no contexto de Bitcoin, podem eventualmente possuir dependências e conflitos, mas como o processo de solução de conflitos afeta diretamente as dependências (que por si só já poderiam ser tema para um outro trabalho dessa natureza), então o problema a ser atacado nesse trabalho tem por temática os conflitos que ocorrem entre transações, visando propor uma alternativa para que elas sejam eliminadas, ou atenuadas, no processo de criação de novos blocos.

Transações conflitantes nada mais são do que duas ou mais transações que gastaram o mesmo UTXO (do inglês, Unspent Transaction Output, ou seja, Uma Saída de Transação Não Utilizada) que pode ser gasto como uma entrada em uma nova transação, ou seja, em dobro, sendo esse um problema conhecido como gasto duplo.

Visando remediar essa situação, o método aplicado é não tratar uma transação como bem-sucedida até que ela tenha um número de confirmação que seja satisfatória a quem esteja transacionando, pois é considerado como impossível a recuperação de um ataque de gasto duplo bem sucedido se a transação conflitante tiver sido incluída na blockchain.

Na rede Bitcoin, um bloco possui o limite superior de 1 MB e as transações possuem taxas, que são as mais diversas possíveis a depender da transação em si. Contudo, por questão teórica vamos ignorar o limite de tamanho de bloco e assumir uma taxa constante para todas as transações.

Para abstrair a visualização de uma mensagem, iremos considerar uma transação como um nó, ou um vértice, da forma que é apresentada na figura 2

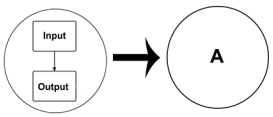

Figura 2. Abstração de uma transação como um vértice A.

Considere então, o seguinte cenário, demostrado na figura : Se as transações Y e Y' reivindicarem fundos da transação X, somente uma das transações poderá ser publicada, Y ou Y", e assumimos então que elas estão em conflito. Supondo esta como a única restrição, é possível representar em forma de grafo todas as transações, tendo as transações como vértices e atribuindo arestas entre transações que estejam em conflito e dessa forma o minerador precisará encontrar o maior conjunto de vértices (transações) sem arestas entre eles (sem conflito).

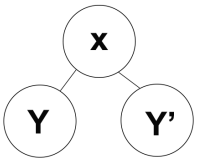

Figura 3. Cenário com transações conflitantes.

O problema do tipo gasto duplo acaba sendo um dos motivos de vulnerabilidades da rede de blockchain, gerando a possibilidade de vários ataques a rede. Vários ataques são conhecidos, dos quais podemos mencionar: Ataque de Corrida, Ataque Finney, Ataque Vector76, Ataque de histórico alternativo e Ataque da majoritariedade.

Sendo assim, esse trabalho visa produzir uma alternativa de seleção de transações menos propensas a sofrer do problema do gasto duplo, pois não são provenientes de transações conflitantes para serem inseridas em um novo bloco minerado. Para isso, trata-se transações generalizadas contidas em forma de um grafo, onde vértices são as transações e as arestas são os conflitos existentes entre alguns vértices do grafo em questão.

Nós modelamos a rede Bitcoin como um grafo G = (V, E) e cada nó v corresponde a uma transação e assumimos que cada aresta $e \in E$ tem uma reivindicação de fundos associado a ela.

IV. Justificativa (ou demonstração) de o problema ser NP-difícil (ou NP-completo)

Resolver isso é exatamente o problema do conjunto máximo independente. Mais uma vez, isso significa encontrar a solução ideal NP-hard e decidir se uma solução existe acima de um

determinado tamanho (o que indicaria a taxa de transação total desde que assumimos taxas fixas de transação) é NP-completo.

Para se provar que um problema é NP-Completo são necessários dois passos:

1) Mostrar que o problema pertence aos problemas NP.
2) Mostrar que um problema NP-Completo conhecido pode ser polinomialmente redutível para ele.

Levando em conta as considerações anteriores, será provado que criar um conjunto máximo de transações não conflitantes se trata de um problema NP-Completo, para um tamanho fixo k de taxa de transação total.

1) Prova de que pertence aos problemas NP: A prova pode ser feita de duas formas:

- Através de um algoritmo não determinístico polinomial para o problema (Algoritmo 1).
- Demonstrando que, a partir de uma solução para o problema, esta pode ser verificada em tempo polinomial.

Algoritmo 1: CMTNC NÃO DETERMINÍSTICO

Entrada: V
Saída: Conjunto máximo de transações não conflitantes ($CMTNC$)

início
  $i = 1$
  $CMTNC = inicializa\_cmtnc()$
  para cada $t \in V$ **faça**
    $j = escolhe(lista\_naoadj(i))$
    $i = j$
    $add\_cmtnc(CMTNC, j)$
  fim
fim
retorna $CMTNC$

Observando o Algoritmo 1, é definido uma transação inicial i e para cada iteração t menor ou igual ao conjunto de transações V, é escolhido uma transação j, não adjacente à i, e adicionado ao conjunto independente. Sendo assim, temos a iteração $t = 1...V$ onde V é o tamanho da entrada, problema já conhecido para o cálculo de complexidade de algoritmos, logo é possível determinar o algoritmo assintoticamente como $O(n)$.

2) Prova de que o Maximum Independent Set é redutível para o Conjunto de Transações não Conflitantes: Considerando o grafo da Figura 4 sendo uma instância para identificar conjuntos independentes, é possível determinar diferentes tipos:

- Conjuntos independentes (CI): Todos conjuntos que consistem de vértices não adjacentes. p.ex (A, E), (E, D), (C, D), (B, C), (B, E), (A, B, E)
- Conjuntos independentes máximos (CIM): (E, D), (C, D), (B, C), (A, B, E). Apesar do conjunto (A, E) ser um conjunto independente, o mesmo não é um conjunto independente máximo devido ser subconjunto de (A, B, E).

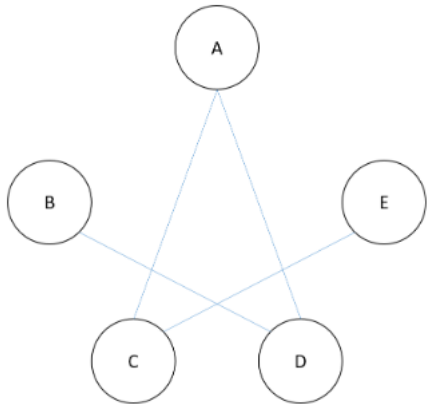

Figura 4. Grafo com 5 vértices e 4 arestas.

- Conjunto máximo independente (CMI): Entre todos os possíveis conjuntos independentes máximos, (A, B, E) é o maior, portante é o conjunto máximo independente para o grafo.

Uma redução polinomial do CMI para o Conjunto Máximo de Transações não Conflitantes (CMTNC) pode ser feito da seguinte forma:

- Para transações usa-se os vértices
- Para os conflitos entre transações usa-se as arestas (arestas adjacentes em nós representam tais conflitos)

Conforme citado na Seção III, uma maneira de contornar transações conflitantes é encontrar o maior número de transações (vértices) que não se conflitam (não adjacentes).

É possível utilizar CMTNC para encontrar o conjunto máximo de transações não conflitantes que seja menor ou igual a V (vértices). Sendo assim, o conjunto máximo é o CMI do grafo, (A, B, E).

V. CONJUNTO INDEPENDENTE

Levando em consideração a teoria dos grafos, um Conjunto Independente (CI), também conhecido como conjunto estável, coclique ou anticlique, é um conjunto de vértices em um gráfico, dos quais dois não são adjacentes. Ou seja, é um conjunto *S* de vértices tal que para cada dois vértices em *S*, não existe arestas os conectando.

Um CI de um grafo *G* é um conjunto *S* de vértices de *G*, tal que não existem dois vértices adjacentes contidos em S. Em outras palavras, se *a* e *b* são vértices quaisquer de um conjunto independente, não há aresta entre *a* e *b*.

Por definição temos que todo grafo tem no mpinimo um CI: o conjunto vazio e um único grafo pode ter vários CIs distintos e o tamanho de um CI equivale ao número de vértices nele contido

Se *S* é um CI de *G* e não existe um CI de *G* maior que *S*, diz-se que *S* é um CI máximo de *G*. O problema de, dado um grafo *G*, determinar se há um CI de tamanho k é um problema NP-completo.

Um CI máximo (*ou em inglês, Maximal Independent Set* é um CI, de forma que adicionar qualquer outro vértice ao

conjunto força o conjunto a conter uma aresta ou o conjunto de todos os vértices do gráfico vazio.

Já o máximo de um CI (*ou em inglês, Maximum Independent Set* é um CI de maior tamanho possível para um dado gráfico *G*. Esse tamanho é chamado de número de independência de *G* e denotado α(G). O problema de encontrar tal conjunto é chamado de problema do conjunto máximo independente e é um problema de otimização NP-difícil. Como tal, é improvável que exista um algoritmo eficiente para encontrar um conjunto independente máximo de um gráfico.

Todo conjunto máximo independente também é maximal, mas a implicação inversa não necessariamente se mantém.

VI. Revisão da Literatura

Uma breve revisão da literatura foi realizada, conforme descrição da atividade, e assim vamos citar três trabalhos relacionados encontrados que tratam direta ou indiretamente da temática aqui proposta.

A. Trabalhos Relacionados

1) Secure High-Rate Transaction Processing in Bitcoin: Este artigo, dos autores Yonatan Sompolinsky e Aviv Zohar, trata de uma investigação das implicações de segurança em se ter uma alta taxa de processamento de transações em Bitcoin contra ataques do tipo gasto duplo. É mostrado que em uma alta taxa de transferência, os invasores substancialmente mais fracos são capazes de reverter os pagamentos que fizeram, mesmo depois de serem considerados aceitos pelos destinatários.

2) Handling Bitcoin Conflicts Through a Glimpse of Structure: Dois tópicos considerados principais pelos autores (Thibaut Lajoie-Mazenc, Romaric Ludinard e Emmanuelle Anceaume) são abordados, são eles Gastos duplos e forks de blockchain, confrotando o sistema de criptografia Bitcoin. O primeiro se refere à habilidade do adversário de usar a mesma moeda mais de uma vez, enquanto a segunda reflete a ocorrência de inconsistências transitórias no histórico da estrutura de dados distribuída do blockchain.

3) Towards Risk Scoring of Bitcoin Transactions: Aqui, os autores (Malte Moser, Rainer Bohme, and Dominic Breuker) demostram a preocupação em o Bitcoin se tornar o sistema de pagamento predominante na Internet e acreditam que supostos combatentes de crimes se unirão para regular e reforçar a lista negra de prefixos de transação nas partes que oferecem produtos reais e serviços em troca de bitcoin.

Eles argumentam que Bitcoins na lista negra serão difíceis de serem gastos e, portanto, são menos valiosos. Assim sendo, acreditam que isso requer que todos os recebedores de pagamentos do Bitcoin não só verifiquem todas as listas negras de transações, mas também para avaliem o risco de uma transação na lista negra no futuro.

B. Técnicas Utilizadas

No primeiro artigo, foi abordado a preocupação com a segurança através de métricas estabelecidas por meio da regra GHOST (The Greedy Heaviest-Observed Sub-Tree), uma modificação na forma como os nós Bitcoin constroem e reorganizam o cadeia de blocos, a estrutura de dados distribuída central do Bitcoin. O método GHOST foi adotado e uma variante dele foi implementada como parte de projeto em Ethereum, uma criptomoeda alternativa ao Bitcoin, uma plataforma de aplicações distribuídas de segunda geração. O modelo da rede Bitcoin é trazido na forma de um grafo orientado

Já no segundo artigo, os autores apresentam uma nova abordagem para enfrentar estas questões: adicionar algumas restrições à sincronização local de validação de transações do Bitcoin, e tornar essas restrições independentes do protocolo nativo da blockchain. Restrições de sincronização são tratadas por nós da rede quee são escolhidos aleatoria e dinamicamente no sistema Bitcoin. Assim sendo, mostram que com tal abordagem, o conteúdo da blockchain é consistente com todas as transações validadas e blocos que garantem a ausência de ambos os ataques de gasto duplo e garfos blockchain. São feitas também análise de safety, liveness e not triviality do ecossistema Bitcoin. Assim como é apresentado um modelo de Orquestração de serviços de detecção de conflitos considerando os nós da rede Bitcoin e Tabelas de Hash Distribuidas.

Por fim, o terceiro artigo traz a elaboração de um cenário, com um modelo de risco especificado, e a concepção de uma abordagem de previsão usando o conhecimento público apresentando resultados preliminares usando dados de roubos conhecidos selecionados. Os autores ainda discutem as implicações em mercados onde os bitcoins são negociados e trazem uma crítica sobre a capacidade do Bitcoin de servir como uma unidade de conta. É fornecido um Modelo de temporização da lista negra de transações com políticas diferentes (Poison, Haircut), assim como predição e análise de riscos e impacto.

VII. Metodologia

Algumas abordagens podem ser utilizadas para o cálculo do CMTNC utilizando programação dinâmica, backtracking e algoritmos gulosos. O Algoritmo 2 descreve uma metodologia utilizando conceitos de algoritmos gulosos na tentativa de encontrar um ótimo global.

O algoritmo recebe como entrada um grafo $G(T, C)$ onde T é o conjunto de transações (vértices) e C o conjunto de conflitos (arestas). Cada transação t_i possui um grau $d(t_i) \mid t_i \in T,\ i \in \mathbb{N}$, que é o número de conflitos existentes em uma transação, ou seja, a quantidade de vértices adjacentes a um vértice.

O algoritmo se inicia escolhendo uma transação t_i de menor conflito $d(t_i) \forall t_i \in T$ adicionando-o ao conjunto máximo de transações não conflitantes $CMTNC$. Durante cada iteração é escolhido a próxima transação de menor conflito, podendo acontecer duas situações:

1) A transação escolhida é realmente a menor
2) $\exists$ duas menores transações com a mesma quantidade de conflitos

Para a segunda opção é escolhida a transação de forma aleatória.

Em ambos casos é verificado se a nova transação $t_i \in CMTNC$, caso afirmativo, outra transação é escolhida, caso contrário, a transação é adicionada ao conjunto $CMTNC$ e removida do conjunto de transações T (Algoritmo 2).

Algoritmo 2: CMTNC

Entrada: $G(T, C)$
Saída: Conjunto máximo de transações não conflitantes ($CMTNC$)

```
1  início
2      se T = ∅ então
3          retorna ∅
4      fim
5      CMTNC = inicializa_cmtnc()
6      repita
7          t = escolhe_menor(T)
8          se t ∉ CMTNC então
9              add_cmtnc(CMTNC, t)
10             remove(T, t)
11         fim
12     até T = ∅;
13 fim
14 retorna CMTNC
```

Referências

[1] Nakamoto, Satoshi. "Bitcoin: a peer-to-peer electronic cash system (2008).", 2008.

[2] Sompolinsky, Yonatan, and Aviv Zohar. "Secure high-rate transaction processing in bitcoin."International Conference on Financial Cryptography and Data Security. Springer, Berlin, Heidelberg, 2015.

[3] Lajoie-Mazenc, Thibaut, Romaric Ludinard, and Emmanuelle Anceaume. "Handling bitcoin conflicts through a glimpse of structure."Proceedings of the Symposium on Applied Computing. ACM, 2017.

[4] Möser, Malte, Rainer Böhme, and Dominic Breuker. "Towards risk scoring of Bitcoin transactions."International Conference on Financial Cryptography and Data Security. Springer, Berlin, Heidelberg, 2014.

[5] "Bitcoin Wiki". En.Bitcoin.It, 2019, https://en.bitcoin.it/wiki/MainPage. Accessed 1 June 2019.

[6] Cachin, Christian, et al. "The Transaction Graph for Modeling Blockchain Semantics.", IACR Cryptology ePrint Archive 2017 (2017): 1070.