UNIVERSIDADE DE SÃO PAULO
FACULDADE DE ECONOMIA, ADMINISTRAÇÃO E CONTABILIDADE
DE RIBEIRÃO PRETO
DEPARTAMENTO DE ECONOMIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ECONOMIA APLICADA

PEDRO BIANCHI FRANCESCHIN

Uma introdução à teoria do Desenho de Plataformas Descentralizadas

Ribeirão Preto–SP

2022

Prof. Dr. Carlos Gilberto Carlotti Junior
Reitor da Universidade de São Paulo

Prof. Dr. André Lucirton Costa
Diretor da Faculdade de Economia, Administração e Contabilidade de Ribeirão Preto

Profa. Dra. Roseli da Silva
Chefe do Departamento de Economia

PEDRO BIANCHI FRANCESCHIN

Uma introdução à teoria do Desenho de Plataformas Descentralizadas

Versão corrigida

A versão original encontra-se na FEA-RP/USP.

Dissertação apresentada ao Programa de Pós-Graduação em Economia Aplicada da Faculdade de Economia, Administração e Contabilidade de Ribeirão Preto (FEARP) da Universidade de São Paulo (USP), como parte das exigências para a obtenção do título de Mestre em Ciências.

Orientador: Prof. Dr. Jefferson D. P. Bertolai

Ribeirão Preto–SP

2022

PEDRO BIANCHI FRANCESCHIN

An introdutcion to the theory of Descentralized Platforms Design

Corrected Version

The original version can be found at FEA-RP/USP.

Dissertation presented to Programa de Pós-Graduação em Economia Aplicada of Faculdade de Economia, Administração e Contabilidade de Ribeirão Preto (FEARP) from the Universidade de São Paulo (USP), as part of the requirements to hold the Master of Science degree.

Field of Study: Applied Economics.

Supervisor: Prof. Dr. Jefferson D. P. Bertolai

Ribeirão Preto–SP

2022

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

Franceschin, Pedro Bianchi
Uma introdução à teoria do Desenho de Plataformas Descentralizadas. Ribeirão Preto–SP, 2022.
148p. : il.; 30 cm.

Dissertação apresentada à Faculdade de Economia, Administração e Contabilidade de Ribeirão Preto da USP, como parte das exigências para a obtenção do título de Mestre em Ciências,
Área: Economia Aplicada.
Orientador: Prof. Dr. Jefferson D. P. Bertolai

1. Redes descentralizadas. 2. Contratos inteligentes. 3.Desintermediação. 4.DLT. 5.Blockchain. 6.Plataformas descentralizadas.

Pedro Bianchi Franceschin

Uma introdução à teoria do Desenho de Plataformas Descentralizadas

Dissertação de conclusão de programa de mestrado em conformidade com as normas ABNT.

Trabalho aprovado. Ribeirão Preto–SP, 25 de abril de 2022:

Orientador:
Orientador

Professor
Convidado 1

Professor
Convidado 2

Ribeirão Preto–SP
2022

Este trabalho é dedicado à minha família e a Satoshi Nakamoto,
sem os quais este projeto não seria possível.

Agradecimentos

O presente trabalho foi realizado com apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES).

Agradeço a todos que me apoiaram, de qualquer forma, antes e durante a realização deste projeto. Em especial, Bernardo Bianchi Franceschin pela contribuição direta na forma de sugestões e feedbacks.

Uma introdução à teoria do Desenho de Plataformas Descentralizadas

Resumo

A cerca de uma década, um ecossistema de plataformas descentralizadas vem se desenvolvendo globalmente através da internet. Essas plataformas representam novas aplicações da Teoria Econômica que devem ter impactos diretos sobre as formas como ela deve ser abordada. Atualmente, casos de uso incluem, principalmente, instrumentos financeiros, exchanges, certificados e sistemas de compartilhamento de arquivos, entre outros. Entretanto, o potencial deste tipo de tecnologia deve exceder o conjunto relativamente limitado de casos de uso que vemos hoje, sendo difícil prever o real impacto que ela terá na sociedade ao longo dos próximos anos. Essas aplicações devem representar um novo paradigma, não só para relacionamentos e processos conduzidos através de redes de comunicação, mas ao próprio arcabouço institucional das sociedades modernas. Este trabalho visa prover uma introdução ao que deve potencialmente se tornar uma nova disciplina da Teoria Econômica. Pretende-se não só prover uma base conceitual para um entendimento bem fundamentado do assunto, mas ilustrar essas aplicações através de uma linguagem familiar para economistas, em geral. Através desse conteúdo, espera-se que profissionais e pesquisadores da área possam ter uma introdução aos conceitos e tecnologias envolvidos nessas aplicações e contribuir para o desenvolvimento da teoria relacionada ao que pode ser considerada uma das disrupções mais significativas do novo milênio.

Palavras-chave: Redes descentralizadas. Contratos inteligentes. Desintermediação. DLT. Blockchain. Plataformas descentralizadas.

An introduction to the theory of Decentralized Platforms Design

Abstract

For about a decade, an ecosystem of decentralized platforms has been developing globally through the internet. These platforms represent novel applications of Economic Theory that can directly impact the way it should be viewed. Currently, use cases include mainly financial instruments, exchanges, certificates and file-sharing systems, among others. However, the potential of this type of technology should exceed the relatively limited set of use cases we see today, making it difficult to predict the real impact it is going to have in societies throughout the years to come. These applications should represent a new paradigm, not only to the relationships and processes conducted through communication networks, but also to modern societies' very institutional framework. This work intends to provide an introduction to what should become a new discipline in Economic Theory. The intention is not only to provide a conceptual basis for a fundamentally sound understanding of the subject but to expose these applications using a language familiar to economists in general. It is expected that, through this content, professionals and researchers of the area can be introduced to the concepts and technologies involved in these applications and contribute to the development of the theory related to what can be considered one of the most significant disruptions of the new millennium.

Keywords: Descentralized networks. Smart contracts. Disintermediation. DLT. Blockchain. Descentralized platforms.

Lista de Figuras

Figura 1 – Os componentes de uma Plataforma Descentralizada 35
Figura 2 – Níveis do ambiente institucional 38
Figura 3 – Inputs de uma transferência. 62
Figura 4 – Transição de estados na Ethereum 64
Figura 5 – blockchain após bifurcações. 70
Figura 6 – Encadeamento de blocos 77
Figura 7 – Manipulação oportunista do histórico de blocos 77
Figura 8 – Manipulação oportunista do histórico de blocos (II) 78
Figura 9 – Probabilidade de sucesso de um gasto duplo 79
Figura 10 – Nothing at Stake 86
Figura 11 – Penalização em PoS 87
Figura 12 – Bifurcação em PoS 88
Figura 13 – Nothing at Stake com previsibilidade n. 88
Figura 14 – Camada 2 97
Figura 15 – Camada de aplicações 99
Figura 16 – Transição de estados de um contrato de tokens 104
Figura 17 – Permissão e transferência bilateral 106
Figura 18 – Caso de uso de contrato de tokens: fundo de investimento 107
Figura 19 – Caso de uso de contrato de tokens: liquidação de investimento 109
Figura 20 – Transferência de uma NFT 111
Figura 21 – Criação de uma CDP 116
Figura 22 – Fechamento de uma CDP 117
Figura 23 – Gestão da oferta dos tokens S e G 126
Figura 24 – O modelo de Compensação 132

Lista de Tabelas

Tabela 1 – Um exemplo de *estado* do livro-razão. 63
Tabela 2 – Custos em gas 66
Tabela 3 – Função hash 73

Lista de abreviaturas e siglas

BTC	Denominação em bitcoin (criptomoeda).
ETH	Denominação em ether (criptomoeda).
USD	Denominação em dólar americano (moeda soberana).
BRL	Denominação em real brasileiro (moeda soberana).

Sumário

Introdução **27**

0.1 Literatura relacionada **31**

1 DESCENTRALIZAÇÃO E A SUA RELEVÂNCIA SOCIAL . **33**

1.1 Plataformas Descentralizadas **33**

1.2 O impacto sobre as Instituições **36**

2 SOBRE CONTRATOS E CONTRATOS *INTELIGENTES* **41**

2.1 Contratos **41**

2.1.1 Desenho de contratos 42

2.1.2 Certificados ao portador 43

2.1.3 Controles 44

2.2 Contratos inteligentes **45**

2.2.1 O ancestral 46

2.2.2 Custos de transação e Desintermediação 47

2.2.3 O tradeoff entre não-ambigüidade e flexibilidade 47

2.2.4 Observabilidade e ações ocultas 48

2.2.5 Medidas reativas, anonimato e seleção adversa. 49

2.2.6 Execução, medidas proativas e privatividade 50

2.2.7 Descentralização e Verificabilidade 52

2.2.8 Negociação e comprometimento 52

2.2.9 Eventos externos e o problema do oráculo 53

3 PRIMITIVOS DE DLT **55**

3.1 Redes descentralizadas **55**

3.2 Protocolos **55**

3.2.1 Protocolos vs. controles 56

3.2.2 Máquinas virtuais 56

3.2.2.1 Computação multipartes 57

3.2.2.2 Mecanismos 57

3.3 Chaves criptográficas e assinaturas digitais **58**

3.3.1 chaves 58

3.3.2 assinaturas digitais 59

4 BLOCKCHAIN E O COMPUTADOR GLOBAL 61
4.1 Sistemas de registro descentralizados 61
4.2 Um sistema de transição de estados 63
4.3 O computador global 64
4.4 Consenso: *proof-of-work* e *proof-of-stake* 69
4.4.1 *proof-of-work* 73
4.4.2 *proof-of-stake* 79
4.4.2.1 Nada em jogo 83
4.4.2.2 Simulação gratuita 89
4.4.3 Considerações 90
4.5 *blockchain* e a camada 1 91
4.5.1 O trilema 92
4.5.2 Anonimato 96
4.5.3 Camada 2 96

5 CAMADA DE APLICAÇÕES 99
5.1 Tokens 100
5.1.1 Tokens fungíveis 102
5.1.1.1 Exemplo: fundo de investimento 107
5.1.2 Tokens não fungíveis 109
5.1.2.1 Exemplo: segurança de automóveis 112
5.2 Ativos sintéticos 113
5.2.1 O modelo de Posição Colaterizada 115
5.2.1.1 Criação/Destruição de sintéticos. 115
5.2.1.2 Preço de S(A). 116
5.2.1.3 Estratégia de posição vendida. 120
5.2.1.4 Processo de liquidação 121
5.2.1.5 O Token de Governança. 122
5.2.1.6 Reserva do protocolo. 123
5.2.1.7 Compradores de última instância. 125
5.2.1.8 Posição vendida/comprada 126
5.2.1.9 Posição n-alavancada 127
5.2.1.10 MakerDAO. 128
5.2.1.11 Mirror. 129
5.2.2 O modelo de Compensação 129
5.2.2.1 Implementação Terra 132
5.2.3 Modelo Poço de Dívida 134
5.2.3.1 Criação/destruição de sintéticos 134
5.2.3.2 A dinâmica contábil 135
5.2.3.3 Posição vendida 137

5.2.3.4 Preço de S_i 138
5.2.3.5 Risco de sub-colateralização. 138
5.2.3.6 Synthetix 140
5.2.3.6.1 Processo de liquidação. 141

6 CONSIDERAÇÕES FINAIS 143

Referências Bibliográficas 145

Introdução

Historicamente, processos de relacionamento entre uma *rede*[1]de indivíduos ou entidades, dependeram de sistemas de intermediação para ganhar escala ou atingir determinados níveis de complexidade. Na maioria dos casos, esses intermediários representam focos de centralização aos processos conduzidos através da rede. Particularmente, se a quantidade de intermediários é relativamente pequena em relação à quantidade de pontas de comunicação da rede, então necessariamente essa intermediação representa um foco de centralização.

Um exemplo atual dessa dinâmica pode ser visto no sistema bancário moderno. Diferentes contas bancárias se comunicam entre si através do servidor central dos bancos comerciais, controlados pelas diretorias do respectivos bancos. As reservas dos diferentes bancos comerciais se comunicam entre si através do servidor dos bancos centrais, controlado pelo órgão de autoridade monetária de cada país. Existe ainda mais uma camada de intermédio para comunicar os sistemas bancários de diferentes países (e.g. o sistema SWIFT).

Focos de centralização podem representar uma série de vulnerabilidades a alguns processos conduzidos através de redes. Talvez a principal delas é a concentração de influência nesses intermediários sobre os processos gerados na rede. No exemplo do sistema bancário isso pode ser manifestado na forma de altos spreads e políticas monetárias desastrosas. De maneira geral, na forma de controle discricionário sobre as atividades conduzidas pelos membros da rede ou de fricção sobre transações em geral.

Por outro lado, em alguns tipos de rede, concentração de influência pode ser algo positivo. Pode gerar, por exemplo, agilidade no processo de tomada de decisão. Redes que operam em ambientes que demandam essa característica, se beneficiam de algum nível de centralização. Por esse motivo, a maioria das empresas se organizam através de hierarquias verticais em relação à concentração de influência sobre os processos internos à ela.[2] Quanto mais competitivo o ambiente onde a organização opera, mais ágil ela precisa ser para se adaptar a mudanças no ambiente, logo mais rápido precisa ser o processo

[1] Nesse contexto, podemos definir uma *rede* como sendo um conjunto de entidades, representativas de *pontas de comunicação*, que se comunicam entre si.

[2] Nesse caso podemos interpretar a própria empresa como sendo uma rede de funcionários e acionistas.

decisório. Portanto, internamente à empresa, focos de centralização podem melhorar sua performance em geral.

No entanto, se olharmos sobre a perspectiva de mercados, focos de centralização raramente produzem efeitos positivos em termos de performance. Se manifestam principalmente através de monopólios e oligopólios, que atuam como gargalos, tanto à oferta, quanto ao progresso tecnológico das atividades, limitando também a utilidade gerada para os diferentes membros da rede (do mercado neste caso).

Diversos outros tipos de rede podem ser limitadas por focos de centralização. Um exemplo interessante é provido pela rede da BitTorrent. Previamente ao desenvolvimento dessa tecnologia, o compartilhamento de arquivos online era feito através de conexões, diretas ou indiretas, com uma única fonte provedora do conteúdo.[3] Esse arranjo torna velocidades de download suscetíveis a quedas quando existe alta demanda pelo mesmo arquivo. A tecnologia implementada pela BitTorrent permite que usuários compartilhem *pedaços* do arquivo entre si, efetivamente descentralizando a fonte de oferta da informação de interesse.

Até então, mesmo em redes peer-to-peer (onde as pontas da rede compartilham conteúdos entre si), cada usuário baixava um arquivo de um único colega. A oferta de arquivos específicos era centralizada. Alta demanda por um arquivo do mesmo usuário iria gerar congestionamento e a velocidade do download iria despencar para todos. No modelo da BitTorrent, para que alguém se torne um ofertante, basta esse alguém ter baixado determinado pedaço de um arquivo. Dessa forma, o sistema torna consumidores em ofertantes. Como resultado, a velocidade de download fica muito menos vulnerável a congestionamentos gerados por alta demanda. Equanto redes peer-to-peer possibilitaram a descentralização da provisão de conteúdo online, a rede BitTorrent possibilitou mais uma camada de descentralização: a de instâncias específicas de provisão de conteúdo. Isso permitiu a rede a atingir um aumento de performance considerável nessa atividade.

Outro efeito negativo de focos de centralização é a introdução de *pontos centrais de falha*. Redes que funcionam através de servidores centrais dependem integralmente da operabilidade desses servidores, sendo completamente vulneráveis a potenciais panes localizadas neles. Mesmo com a introdução de backups, essas redes continuam vulneráveis a ataques coordenados que visam impossibilitar os backups, assim como o servidor principal. Sem contar com a probabilidade de pane em todos servidores, que deve ser positiva, por menor que seja. Ademais, grandes redes com alto tráfego demandam investimentos

[3] Esse formato é proeminente até hoje. As maiores plataformas de streaming funcionam de forma completamente centralizada, i.e., os arquivos de mídia são todos armazenados em servidores centrais controlados pelas organizações representantes da plataforma. Apesar dos arquivos não serem copiados para a máquina dos usuários, a analogia aqui pode ser feita pois existe uma única fonte provedora do conteúdo.

massivos na infraestrutura de servidores centrais, o que contribui para aumentar ainda mais a centralização, sob a perspectiva de mercados.

Movimentos para descentralizar diversos tipos de rede existem desde antes do início do novo milênio, quando a expansão da acessibilidade à internet começou a atingir um escopo global. Na academia, as maiores contribuições vêm dos campos de Ciência/Engenharia da Computação e Criptografia[4]. No âmbito de implementações, tecnologias baseadas no conceito de rede peer-to-peer representaram uma grande quebra de paradigma nessa área.

A Napster, por exemplo, fundada em 1999 nos EUA, era um serviço que permitia usuários compartilharem entre si arquivos de áudio que acabou sendo um catalisador para a mudança do mercado de músicas para o formato que existe hoje em dia.[5] O principal foco de centralização da plataforma foi o que possibilitou o seu insucesso. Ela dependia de um servidor central para conectar usuários entre si, assim como fazer buscas de conteúdos entre eles. Como o principal caso de uso da plataforma era o compartilhamento de músicas, com distribuição restrita pelas legislações de direitos autorais, parte da indústria fonográfica, assim como autoridades americanas, encararam a prática como pirataria. A organização que controlava o servidor central representava um alvo à sanções burocráticas, além do próprio servidor ser um alvo de medidas coercivas de desligamento que iriam interromper o funcionamento da rede. A plataforma foi forçada a cessar operações em 2001.

A Gnutella e BitTorrent, que introduziram novos níveis de descentralização a esse tipo de rede, continuam operantes até hoje, apesar de apresentarem casos de uso similares aos da Napster, incluindo compartilhamento de filmes, séries e outros tipos de propriedades intelectuais. Pode-se dizer, então, que um efeito colateral desses movimentos de descentralização é tornar as redes mais evasivas em relação a medidas regulatórias de autoridades burocráticas. Esse efeito, apesar de possibilitar práticas formalmente ilícitas na maioria dos países, é socialmente benéfico para casos onde existe opressão regulatória, como censuras e outras formas de sanções que violem direitos amplamente tidos como fundamentais para sociedades funcionais.

Mesmo com o relativo sucesso dessas redes,[6]elas possuem uma vulnerabilidade bem conhecida por economistas em geral: sendo um bem público, a provisão de conteúdo na

[4] Mais adiante serão abordados alguns dos motivos pelos quais a criptografia é tão importante para esses movimentos.

[5] Anteriormente, a distribuição era predominantemente feita usando mídias físicas como discos, CD's, MD's, etc. Atualmente a distribuição é dominada por plataformas digitais de streaming.

[6] A BitTorrent, por exemplo, até 2019 representava um dos principais meios de compartilhamento de arquivos em geral pela internet [BitTorrent em *Wikipédia*]. Além do mais, a rede foi de fato alvo de sanções, por parte das autoridades de diferentes países, contra a práticas consideradas como pirataria. Entretanto, a rede como um todo pode prosperar, apesar das sanções, devido à sua arquitetura, per-

rede é sujeita ao *problema do carona*. Mesmo no caso da BitTorrent, por mais que o sistema torne consumidores em ofertantes, isso ocorre apenas temporariamente. Uma vez que um agente conclua um download, não existem incentivos *diretos*[7] para ele continuar ofertando o arquivo para a rede. Da mesma forma, não existem incentivos diretos para um membro da rede ser a fonte original do arquivo. Apesar disso, redes como a BitTorrent de fato ganharam bastante adoção devido à utilidade que geram aos seus usuários. Entretanto, em teoria, essa característica pode ser fonte de ineficiência ao processo endógeno à rede. Vários tipos de conteúdos provavelmente deixarão de ser providos.

A introdução de um sistema de pagamentos tradicional, por exemplo, representaria introdução de focos de centralização à rede. *Redes descentralizadas*[8] em geral, portanto, devem sofrer desse tipo de vulnerabilidade, certo?[9] Entra em cena a Bitcoin.

A reprodução de moedas ou sistemas de pagamentos descentralizados em redes digitais é extremamente difícil pelo fato que, nesse tipo de mídia, informações em geral são facilmente copiáveis e infinitamente reprodutíveis. Em meios digitais, simulação de *escassez*, uma das características mais importantes para uma moeda,[10] é um problema de difícil solução. Uma alternativa é recorrer a um sistema de crédito aonde um registro é mantido com todas as transações realizadas entre membros da rede. No entanto, é preciso alguma forma de impedir que agentes registrem transações fraudulentas. Isso é tipicamente feito se recorrendo a um intermediário *confiável* para manter o registro (e.g. bancos, fintechs, etc.), introduzindo focos de centralização à rede.

A Bitcoin, implementada em 2009, resolveu esse problema através de uma *rede descentralizada*. Pode-se dizer que a Bitcoin resolveu o problema da escassez e o problema da manutenção do registro. Na verdade, ambas as soluções estão atreladas. A tecnologia utiliza um registro descentralizado e impede a alimentação de entradas fraudulentas recorrendo à escassez de recursos materiais, além de um sistema de coordenação entre os membros da rede . Essa solução é ilustrada na seção 4.

Através dessa solução implementada pela Bitcoin, uma gama de aplicações começou a ser desenvolvidas para realizar diversos serviços online de forma descentralizada, com mecanismos de incentivo *direto* endógenos ao protocolo, para resolver também o problema do carona prevalecente em redes descentralizadas até então. Essas aplicações são o objeto foco da teoria que pretendemos abordar nesse trabalho. O termo proposto para

mitindo assim que outros casos de uso considerados legais pudessem ser conduzidos sem intervenções.

[7] Usa-se esse termo aqui para qualificar incentivos que são claramente observáveis (talvez até quantificáveis) por terceiros.

[8] Uma definição a este conceito é oferecida na seção 3.

[9] Um exemplo muito ilustrativo é provido pela Popcorn Time. Era uma plataforma que usava a rede da BitTorrent para prover streaming de filmes e séries de forma completamente gratuita para os usuários. Entretanto, ela cessou de operar esse ano, deixando a entender que um dos motivos foi baixo nível de utilização <https://www.bloomberg.com/news/articles/2022-01-04/popcorn-time-alternative-is-hard-to-find-as-app-shuts-down?sref=10lNAhZ9>.

[10] Ver, por exemplo, (KOCHERLAKOTA, 1998).

se referir a elas é *Plataformas Descentralizadas*. Mais adiante, na seção 1, uma definição mais detalhada ao termo é apresentada.

Essas plataformas representam, atualmente, uma importante aplicação da Teoria Econômica com enorme potencial disruptivo em vários setores da sociedade. Tipicamente, trabalhos que exploram essas aplicações demandam algum tipo de familiaridade com os conceitos e tecnologias subjacentes, que em vários casos não são novos, mas são ainda obscuros para a maioria das pessoas no exterior das áreas de estudo mais relacionadas, principalmente Ciência da Computação e Criptografia. Dada a importância desse assunto, assim como a demanda de amplo envolvimento de pesquisadores da área da Economia nele, este trabalho pretende prover uma introdução ao que provavelmente representará uma nova disciplina em economia, na opinião deste autor. Pretende-se abordar alguns conceitos fundamentais necessários para entender o funcionamento básico da tecnologia, assim como explorar a lógica por trás de algumas aplicações, de forma gradual, didática e com uma comunicação acessível ao leitor que não é familiarizado com assunto. Um conhecimento básico da Teoria Econômica, mais particularmente Teoria dos Jogos é propício.

A área da teoria mais relacionada com essas aplicações é a de Desenho de Mecanismos. Pode-se dizer que essa nova disciplina, *Desenho de Plataformas Descentralizadas*, deveria ser tratada sob a rubrica dessa área. No entanto, outras áreas como a da Economia Institucional, mais particularmente Economia de Custos de Transação, são muito impactadas por essas tecnologias disruptivas.[11]

Na seção 2 será abordado um conceito importante para o surgimento da tecnologia utilizada por essas aplicações, assim como para o desenvolvimento da teoria relacionada: *contratos inteligentes*. Será ilustrado ainda, como esse conceito se relaciona com o conceito tradicional de contratos, além de algumas vantagens e desvantagens que implementações desse conceito podem apresentar em relação ao paradigma tradicional. A seção 3 apresenta alguns conceitos e técnicas relevantes para o entendimento das exposições das aplicações que serão apresentadas nas seções 4 e 5. A seção 6 conclui o trabalho com algumas considerações finais.

0.1 Literatura relacionada

A principal literatura relacionada ao objeto principal de estudo deste trabalho, são as propostas e exposições das diferentes aplicações do conceito de plataformas descentralizadas estudadas no trabalho. Estes incluem (NAKAMOTO, 2008; BUTERIN, 2014; BENTOV; PASS; SHI, 2016; BUTERIN; GRIFFITH, 2017; CHEN; MICALI, 2019; DAMERON, 2018; GILAD et al., 2017; KIAYIAS et al., 2017; KING; NADAL, 2012; VASIN, 2014) e

[11] Ver, por exemplo, (HALABURDA; LEVINA; SEMI, 2019).

(ZAMFIR, 2017). Alguns conceitos e modelos abordados nesses trabalhos podem ser vistos na seção 4. No entanto, apesar de explicar a lógica por trás das respectivas aplicações, eles dão pouca ênfase à parte teórica que este trabalho visa focar, além de demandarem um entendimento prévio da tecnologia utilizada, em alguns casos.

(ANTONOPOULOS, 2017) provê um material educativo sobre como usar a plataforma *Ethereum*, que é o modelo referência para as seções 4.2 e 4.3. O trabalho explica ainda, de forma acessível, alguns conceitos e ferramentas específicas a essa aplicação. O autor, contudo, é focado na educação de usuários e desenvolvedores de aplicações.

(OPEN ZEPPELIN, a; OPEN ZEPPELIN, b; KEREIAKES; KWON; PLATIAS, 2019; MAKERDAO, b; MAKERDAO, a; MIRROR PROTOCOL, ; TERRA, ; SYNTHETIX, b; SYNTHETIX, a) apresentam especificações de implementações ligadas aos modelos abordados na seção 5. No entanto, no que diz respeito às informações relacionadas às exposições feitas abaixo, essas fontes são imprecisas e até imcompletas em determinados casos, fazendo necessário recurso de fontes alternativas.

(BABAIOFF et al., 2012; BAGARIA et al., 2019; BROWN-COHEN et al., 2019; CARLSTEN et al., 2016; DAIAN et al., 2020; EYAL; SIRER, 2014; EYAL, 2015; NAYAK et al., 2016; POELSTRA, 2015) e (SAPIRSHTEIN; SOMPOLINSKY; ZOHAR, 2016) são exemplos de trabalhos que analisam algumas aplicações sob uma perspectiva de incentivos dos agentes envolvidos. Pode-se dizer que estes trabalhos estão diretamente ligados com a teoria que pretendemos abordar. Esses trabalhos, focam particularmente em *mecanismos de consenso* das plataformas de *camada 1*, conceitos abordados nas seções 4.4 e 4.5, respectivamente. (KROLL; DAVEY; FELTEN, 2013; MA; GANS; TOURKY, 2018) e (SALEH, 2021) apresentam estudos desses mecanismos de consenso com linguagens e conceitos mais acessíveis a economistas pouco familiarizados com as tecnologias usadas, se alinhado um pouco mais com o intuito do presente trabalho. As conclusões obtidas nesses trabalhos, entretanto, não necessariamente são corroboradas pelo conteúdo que será exposto abaixo.

1

Descentralização e a sua relevância social

Nessa seção será apresentada uma breve descrição das aplicações que estão sendo referidas quando usado o termo Plataforma Descentralizada, pelo menos dentro deste trabalho. Em seguida será oferecido um argumento em relação a impactos que essas aplicações podem ter na sociedade.

1.1 Plataformas Descentralizadas

Conforme introduzido acima, essas plataformas são aplicações que usam redes de arquiteturas descentralizadas para possibilitar relações mutuamente benéficas entre os membros da rede. Essas redes são caracterizadas por um baixo grau de intermediação por parte de indivíduos e organizações. É importante ressaltar que, descentralização não será uma característica binária e pode ser um tema controverso em determinados casos, com uns considerando determinadas aplicações descentralizadas e outros não.

O importante a se atentar para os méritos deste trabalho é que, introdução de focos de centralização à rede, conforme ilustramos acima, descaracterizarão a descentralização da rede. Focos de centralização nesse contexto, podem ser definidos como a introdução de um processo de intermédio realizado por uma *entidade centralizada*, i.e., uma entidade no qual processos internos de tomada de decisão são realizados através de concentração de influência sobre ele. É difícil de fugir da relatividade dos conceitos *centralização* e *concentração*, entretanto, segue uma tentativa:

Definição 1 *Processos* `centralizados` *visam possibilitar tomada de decisão através da introdução de* `concentração` *de influência sobre ele. Tipicamente, essa concentração é baseada em autoridade ou em altos custos atrelados à influência sobre o processo. Por outro lado, processos* `descentralizados` *visam alcançar esse objetivo através de mecanismos*

para `nivelar` *a influência sobre ele. Tipicamente, o custo de participação no processo deve ser relativamente baixo e acessível a todos.*

Obviamente, o exemplo extremo de entidade centralizada será o indivíduo. Pode-se dizer que é a unidade atômica desse conceito. Organizações com baixa densidade atômica nos processos decisórios podem ser consideradas centralizadas. Um exemplo que já foi usado anteriormente são empresas. Empresas tipicamente usam sistemas de hierarquia para sustentar a tomada de decisão em processos internos. É um exemplo de concentração de influência via autoridade. Mesmo em empresas de capital aberto, onde algum tipo de influência é distribuído entre acionistas, o custo atrelado à participação é relativamente alto. O conselho administrativo geralmente é formado por uma fração pequena do número de acionistas, representada por investidores com nível de capital alocado maior que a maioria das pessoas terá acesso durante todas suas vidas. Organizações burocráticas, por sua vez, têm a distribuição de influência inteiramente baseadas em autoridade. Continuamos, contudo, expostos a um certo grau de relatividade. No entanto, para a definição que segue, o espaço para controvérsia não deve ser tão alto.

Definição 2 *Um* `foco de centralização` *à rede é o uso de intermediação a partir de* `processos centralizados`*.*

Uma regra de bolso é: se a quantidade de agentes envolvidos no processo de intermédio é pequena em relação aos afetados por ele (i.e., os membros da rede), ele representa um foco de centralização. Entretanto, fundamentalmente, esse não deve ser o único critério. Mesmo que determinada entidade tenha um grande número de envolvidos no processo decisório, é desejável que exista independência entre eles, caso contrário *coordenações adversárias* à rede podem ocorrer. Coordenações adversárias nesse caso significa que os envolvidos no intermédio agem de forma a maximizar suas utilidades, quebrando o contrato firmado previamente. Em organizações como empresas, o grau de independência entre os membros dela é mínimo, já que todos têm um objetivo comum, i.e., de maximizar o retorno da organização. Uma vez que esse objetivo esteja desalinhado com os termos do acordo, deve existir expectativa de quebra de contrato.

Para manter processos descentralizados, portanto, necessidades de intermédio devem ser supridas por redes abertas com baixo custo de entrada, quando não puderem ser usados algoritmos autônomos que eliminam a necessidade de intervenção humana. Isso deve ajudar a induzir maior grau de independência entre os envolvidos. Redes globais formadas através da internet, são capazes de induzir alto nível de independência entre os membros.

Podemos usar, portanto, a seguinte definição:

Definição 3 `Plataformas Descentralizadas` *são aplicações de processos formados através de redes* `sem` *focos de centralização.*

Tipicamente, essas aplicações serão formadas por 3 componentes principais:

i. um propósito;

ii. um processo;

iii. uma comunidade.

(i) geralmente é uma solução ou serviço, do qual a sua provisão é objetivo central à criação da plataforma. (ii) geralmente será o algoritmo pelo qual o resultado previsto por (i) deve ser atingido. (iii) será a rede formada pelos usuários da plataforma, que serão responsáveis por prover os recursos necessários para que o processo funcione e, dessa forma, possa gerar utilidade a todos os envolvidos.

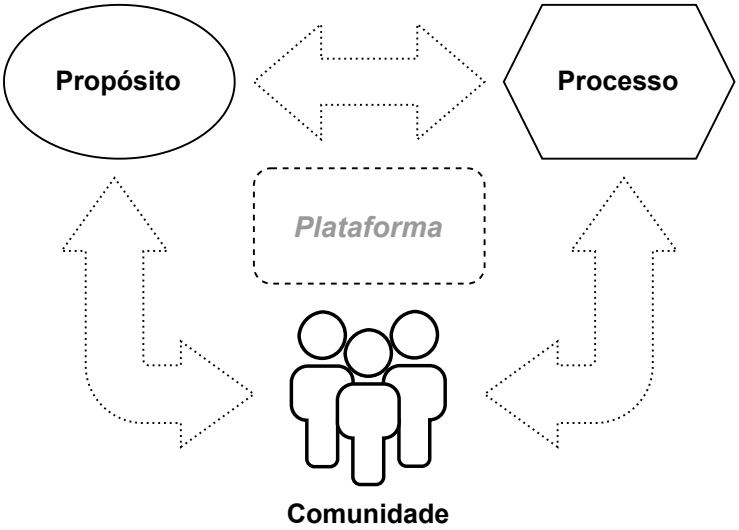


Figura 1 – Os componentes de uma Plataforma Descentralizada

Descrição: A relação simbiótica entre os componentes que formam a rede da plataforma: a *abstração*, a *implementação* e a *socialização*

A Bitcoin, por exemplo, tem como propósito resolver o problema da escassez no meio digital. Ela faz isso através de um processo descentralizado de manutenção de um registro de transações. Para possibilitar o funcionamento desse processo, uma rede de usuários provê a implementação de protocolos, recursos computacionais, assim como propostas de melhorias que devem ser decididas sem recurso à focos de centralização.

Nas seções 4 e 5 serão abordados alguns exemplos de plataformas que utilizam soluções ao problema da escassez para possibilitar que mecanismos de incentivo direto sejam implementados endogenamente. Na sua forma mais aberta, plataformas descentralizadas

são um *bem público*. A propriedade revolucionária dessas implementações, entretanto, é que existem formas de estabelecer incentivos diretos para a provisão desse bem público.

1.2 O impacto sobre as Instituições

Conforme já introduzido acima, aplicações desse conceito podem ser disruptivas para alguns tipos de instituições sociais atuais. No exemplo da BiTtorrent, órgãos regulatórios perderam certo grau de eficácia na aplicação de leis de proteção à distribuição de propriedades intelectuais. É possível imaginar que, dependendo do grau de descentralização de redes como essa, a própria instituição da Propriedade Intelectual pode ser ameaçada em relação a sua praticabilidade, possivelmente se tornando obsoleta.

Muito já foi dito em relação ao potencial diruptivo que aplicações como a Bitcoin podem ter sobre os sistemas monetários.[1] De fato, em alguns países com cenários de alta inflação, essas aplicações possibilitaram alguns agentes a protegerem seu patrimônio convertendo suas poupanças em criptomoedas. Devido a algumas características, que serão abordadas a seguir, essas plataformas tornam medidas como sanções ao seu uso de difícil implementação.[2] Historicamente, autoridades monetárias de alguns países recorreram a sanções e proibições à conversões cambiais para reter poder de controle sobre a economia, frente à quedas sobre a demanda da moeda nacional (e.g. *cepo cambial* na Argentina). A dificuldade apresentada à esse tipo medida, pela utilização de criptomoedas, representa uma potente disrupção ao Sistema Monetário como conhecemos hoje.

Mas o potencial disruptivo de aplicações de Plataformas Descentralizadas não se restringe a esses casos. Antes de prosseguir com o argumento, no entanto, é interessante oferecer definições a alguns conceitos. Se baseando nas definições oferecidas por (NORTH, 1990),

Definição 4 *`Instituições` são conceitos e/ou entidades que formam a estrutura de regras e restrições, formais ou informais, sob a qual interações de indivíduos (e organizações) são governadas.*

Exemplos bem claros incluem, constituições (formal) e normas sociais (informal).

Definição 5 *`Organizações` são grupos de indivíduos que operam dentro da estrutura institucional e implementam as regras e normas providas pelas instituições.*

Exemplo incluem legislaturas, partidos políticos, universidades, empresas, etc.

[1] referencias
[2] Nigéria

Dentre os papéis principais de instituições, estão a preservação de competitividade em mercados, proteção da propriedade privada e geração de um ambiente no qual contratos em geral são respeitados e aplicados propriamente. Organizações terão propósitos e objetivos diversos, e.g., implementação de normas burocráticas, maximização de lucro, filantropia, proteção do meio ambiente, etc. Segundo (NORTH, 1990), elas primariamente procurarão otimizar custos de transação relacionados às suas operações. Entretanto, como elas precisam operar dentro das regras estabelecidas pelas instituições (na maioria dos casos), elas normalmente têm essa otimização restrita pela estrutura institucional, principalmente devido à capacidade limitada das instituições em se adaptar às constantes mudanças tecnológicas e econômicas. Tipicamente, instituições são pouco ágeis nesse processo de adaptação, o que pode ser fonte de infeficiência econômica.

(WILLIAMSON, 2000) oferece uma contextualização bem propícia para ilustrar essas idéias. Ele separa o ambiente institucional em 4 níveis, que podem ser vistos na Figura 2. Os níveis mais relevantes para o presente contexto são os níveis 2 e 3, formados respectivamente por, instituições formais e sistemas de governança organizacional. O nível 2 abriga o processo de formação de leis e normas regulatórias, enquanto o nível 3 é aonde acontece a otimização de custos de transação mencionada anteriormente.

O nível 2 estabelece as restrições sob as quais os processos do nível 3 podem operar formalmente. Para induzir que essas restrições sejam seguidas, uma série de sanções são previstas para agentes que operarem fora delas. Em alguns casos, elas se manifestam informalmente na forma de perdas de reputação e repercussões sociais e econômicas associadas à ela. Em outros casos deve existir, em última instância, a ameaça de coerção física realizada pelas organizações com competências de aplicação da lei, i.e., o "braço forte do estado". O segundo tipo é mais diretamente associado às normas formadas no nível 2.

Finalmente, existem algumas carcterísticas de aplicações de Plataformas Descentralizadas que dificultam esse tipo de aplicação das normas representativas do nível 2. Primeiro, a óbvia impossibilidade de cometer atos de coerção física através de redes de comunicação virtual como a internet.

Segundo, devido à *indefinição jurisdicional* característica dessas redes globais (SZABO, 1997). As instituições formais que compõem o nível 2, assim como seus agentes de autoridade, tipicamente têm jurisdição restrita a determinados territórios.

Terceiro, devido à potencial ausência de pontos centrais de falha, *ergo*, de alvos à sanções regulatórias. Mesmo que autoridades de diferentes jurisdições se coordenassem para tentar aplicar sanções, não existiriam alvos claros para essa prática. Os próprios membros e usuários da rede, seriam a única alternativa.[3] Essa estratégia, contudo, teria eficácia limitada pela capacidade de monitorar as atividades de cada membro da rede.

[3] Isso aconteceu no caso da BitTorrent. Na falta de um ponto central de falha, as autoridades passaram a executar sanções aos usuários que utilizavam a plataforma.

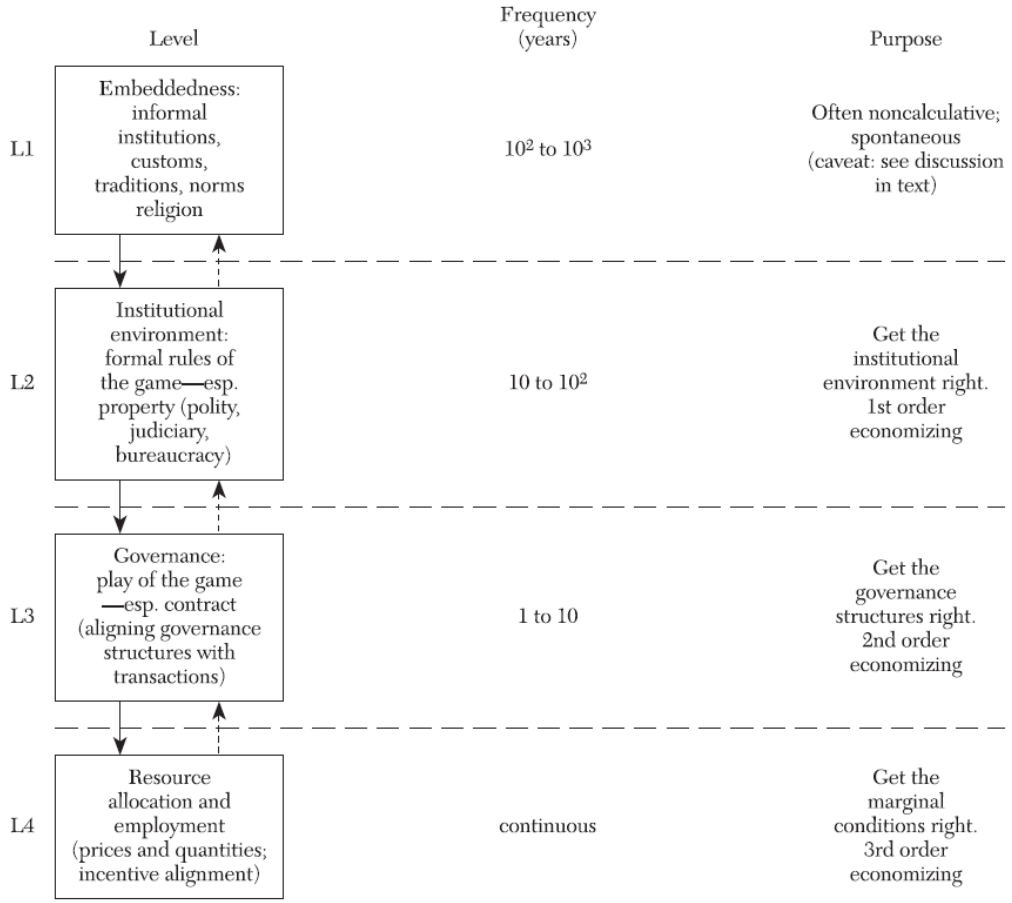


Figura 2 – Níveis do ambiente institucional

Descrição: Os níveis do arcabouço institucional. As setas apontadas para baixo representam direções e restições oferecidas pelas camadas superiores. As setas pontilhadas, apontadas para cima, representam feedback oferecido pleas camadas inferiores. Fonte:(WILLIAMSON, 2000)

O que finalmente, nos traz para a quarta propriedade. Devido a possibilidade da introdução de anonimato à esses processos, essa última medida também se tornaria inviável, caso isso fosse implementado.

A partir da introdução de cada uma das características acima, as medidas necessárias para induzir o respeito às restrições potencialmente ineficientes estabelecidas pelo nível 2, se tornam cada vez mais draconianas e ineficientes. Por exemplo, dada a ausência de um ponto central de falha e possibilidade comunicação direta entre usuários, seria necessário monitorar a atividade online de todos indivíduos da sociedade para identificar membros e usuários da rede.

É possível, portanto, que alguns tipos de Instituições formais, que compõem o nível 2 do esquema acima, se tornem ineficazes e obsoletas, caso aplicações de Plataformas Descentralizadas se tornem comuns e ganhem escala na sociedade. Dadas as claras vantagens que essas aplicações oferecem, apontadas acima e mais adiante neste trabalho, espera-se que elas deem surgimento à novos tipos de instituições, talvez mais focadas em

processos de governança e coordenação descentralizadas mais típicos do nível 3.[4] Ademais, como veremos mais adiante, a tecnologia utilizada por elas possibilita a realização de transações previamente inviáveis, devido a problemas de fricção, assimetria de informação e ausência de confiança.[5] É possível também, portanto, que novos mercados paralelos surjam, relacionados a atividades que se beneficiem desse tipo de tecnologia e do tipo de governança oferecido por essas novas instituições, estabelecendo um novo paradigma de relações econômicas e contratuais, em geral.

[4] Ver, por exemplo, (WRIGHT; FILIPPI, 2015).

[5] Ver (EENMAA-DIMITRIEVA; SCHMIDT-KESSEN, 2019) e (HALABURDA; LEVINA; SEMI, 2019).

2

Sobre contratos e contratos *inteligentes*

Antes de prosseguirmos com a exposição da teoria por trás de algumas implementações de plataformas descentralizadas, conforme as definimos anteriormente, é conveniente explorarmos alguns conceitos relevantes ao assunto. Ao longo das próximas duas seções serão abordados alguns conceitos fundamentais ao desenvolvimento da teoria objeto deste trabalho, assim como alguns conceitos específicos às implementações abordadas abaixo, dos quais uma definição prévia se faz oportuna. É claro que a intenção aqui não é apresentar um estudo extenso desses assuntos, mas sim fornecer uma base conceitual para as análises oferecidas nas seções subsequentes.

O leitor mais familiarizado com os fundamentos por trás do conceito de *smart contacts* pode talvez pular essa seção e consultá-la conforme seja necessário. No entanto, é recomendado que seja feita a leitura dessa seção pois ela apresenta informações importantes para uma compreensão robusta do assunto.

2.1 Contratos

O conceito *contrato* é antigo e tem uma literatura ampla. Na maioria das sociedades modernas, ele é formalizado e regulamentado por sistemas de leis baseados na teoria do Direito. Nesse trabalho, no entanto, utilizaremos o conceito bem amplo descrito abaixo, pois ele se provará útil para nossos propósitos.

Definição 6 *Para todos os efeitos, chamaremos de* ***contratos*** *um conjunto de promessas feitas por duas ou mais partes através de um acordo consensual. Essas promessas podem, ou não, ser condicionadas a eventos que podem ser* ***internos*** *ou* ***externos*** *ao processo gerado pela relação contratual.*

O conjunto de promessas formalizadas no acordo compõem os termos—ou cláusulas—do contrato.

Um exemplo simples e não muito intuitivo é um jogo de xadrez: nesse jogo, os participantes de uma partida concordam em seguir determinadas regras assim como aceitar o resultado final da partida. Essas são *promessas*. Cada jogada deve ser condicionada ao estado corrente do tabuleiro, que por sua vez, é definido pelas jogadas anteriores. As jogadas, portando, são eventos *internos* ao contrato, pois são determinadas pelo próprio contrato (i.e., pelas regras), e condicionam as as ações das partes envolvidas.

Um outro exemplo ainda mais simples é uma aposta em um jogo de futebol. A promessa é o pagamento que deve ser feito ao(s) ganhador(es) pelo(s) perdedor(es). O resultado da partida é o evento *externo* ao contrato que condicionará as ações que devem ser tomadas por cada parte, de forma a respeitar o acordo. É um evento externo pois não sofre qualquer tipo de influência exercida pelo contrato em si.

Em suas instâncias mais simples, contratos assumem a forma de singelas transações comerciais, i.e., compra e venda de bens de consumo.

2.1.1 Desenho de contratos

Dada a definição bem ampla apontada acima, deve existir uma variância enorme no nível de complexidade de contratos em geral. Tipicamente, contratos mais complexos vão demandar o envolvimento de terceiros para prover suporte à sua execução, de forma a respeitar o acordo formalizado.

Pode-se dizer, contudo, que o escopo da realização de contratos envolve atividades além da *execução* dos termos do contrato. Esse processo pode ser separado em duas fases: (i) *ex-ante*, formada por *procura*, *negociação* e *comprometimento*; (ii) *ex-post*, composta por *execução* e *adjudicação* (SZABO, 1997). Todas as atividades listadas podem ter influência direta sobre os resultados do processo.

Contratos bem desenhados devem levar em conta três importantes propriedades: *observabilidade*, *verificabilidade* e *privatividade*[1](Ibid.). *Observabilidade* indica o quão claro estão expressos os termos do contrato e o quanto ele possibilita que as partes envolvidas avaliem a performance de suas contrapartes, assim como provar a sua própria performance das ações previstas no acordo. A prática da contabilidade é um exemplo de atividade voltada para aumentar a observabilidade de contratos. Faltas nessa propriedade podem levar à problemas de *seleção adversa* (ex-ante) e *ações ocultas* (ex-post) (Ibid.).

Verificabilidade indica como o contrato possibilita que um processo de adjudicação descubra se o acordo foi ou não cumprido pelas partes envolvidas. Essa propriedade

[1] Tradução do termo inglês *privity*.

está diretamente ligada à implementações de *medidas reativas* de segurança de contratos (explicadas mais adiante).

A terceira propriedade, *privatividade*, indica que as informações relevantes ao processo descrito no contrato, assim como influência sobre ele, devem ser distribuídas entre as partes envolvidas somente dentro do necessário para executar as atividades previstas e alcançar bons resultados. Essa propriedade diz tanto que algumas informações relevantes ao contrato devem ser protegidas da ciência de terceiros, quanto que direitos e responsabilidades previstos nos termos do contrato devem ser restritos às partes envolvidas.[2] [3] É fácil notar como conceitos como privacidade e propriedade privada estão relacionados à esse objetivo.

Privatividade requer que se minimize exposição do contrato à terceiros. No entanto, observabilidade e verificabilidade requerem algum tipo de exposição à agentes externos, como intermediários e arbitradores (Ibid.). Desenho de contratos devem trabalhar em cima do tradeoff que existe entre esses objetivos para alcançar bons resultados.

Medidas para proteger a integridade do contrato podem ser *proativas* ou *reativas*. Medidas proativas visam principalmente tornar quebras de contrato ou interferências externas impossíveis ou inviáveis. Um exemplo relevante ao contexto deste trabalho é o uso de criptografia para proteger informações sensíveis ao contrato. Medidas reativas visam remediar essas situações, assim como prover incentivos contra a prática, através de penalizações. Exemplos mais comuns são sistemas de reputação e recursos à justiça comum e o "braço forte do estado" (aplicação da lei).

2.1.2 Certificados ao portador

Certificados ao portador são contratos que estabelecem certos direitos e/ou qualificações específicos ao indivíduo em posse do certificado. Os termos do contrato são os direitos que o indivíduo que estiver em posse do certificado pode gozar através da apresentação do mesmo.

Dinheiro é uma instância bem primitiva desse tipo de contrato. Ele representa o direito de adquirir determinada quantidade de bens de consumo no mercado, para aquele que esta em posse dele. É, contudo, um exemplo especial pois a contraparte ao portador do contrato (i.e., da moeda) não é um indivíduo específico, mas sim toda a comunidade que o utiliza como dinheiro. Não existe, portanto, muito espaço para aplicação do conceito

[2] Interpretações modernas do conceito focam mais na restrição de direitos e responsabilidades, no entanto, (SZABO, 1997) argumenta que o significado mais tradicional do conceito engloba também a proteção da informação sensível ao contrato contra agentes externos.

[3] Em algumas jurisdições existem regulamentos que preveem exceções à premissa de privatividade de contratos, criando o conceito de *direitos de terceiros*. Ver *Privity of contract*, em *Wikipedia* <https://en.wikipedia.org/wiki/Privity_of_contract>.

de privatividade.

Um exemplo mais simples é o de cheques ao portador. Um cheque representa o direito de reivindicação de uma quantia da conta bancária do emissor, por parte do indivíduo em posse do cheque. É um contrato com baixa observabilidade, dada a possibilidade de passar cheques sem fundos. Da mesma forma, notas de banco (*bank notes*) na era do *Free Banking* representavam reivindicações sobre a reserva dos bancos emissores.

Outros exemplos comuns incluem ingressos, passagens, senhas de acesso, etc. Nesses casos, os contratos devem estabelecer que indivíduos na posse desses itens, podem ter acesso a determinados serviços providos pelo emissor. Diplomas e certificados de graduação são exemplos de certificados que estabelecem qualificação, no entanto geralmente são registrados a um identificador (nome, rg, etc.) ao invés do portador.

2.1.3 Controles

Contratos dão surgimento a processos, relacionados às atividades referentes ao primeiro, apontadas acima. Á medida que contratos escalam em complexidade, assim como na quantidade de partes envolvidas, os processos relacionados devem ser robustos o suficiente para entregar resultados satisfatórios. Existe uma antiga tradição de processos usados no curso de execução de contratos que lidam com fluxo de dados e procedimentos de verificação para alcançar esse objetivo. (SZABO, 1997) usa o nome "controles" para se referir a esses processos. Operações de alta complexidade com múltiplas atividades interdependentes se beneficiam do uso deste tipo de processo. Eles preenchem os espaços onde as capacidades humanas de gerência e gestão se mostram insuficientes. Eles contribuem ainda para viabilizar auditorias de performance de contratos, atividades tipicamente custosas e vulneráveis à assimetria de informação.

Um exemplo clássico é o método contábil de partilhas dobradas: são mantidos dois registros separados e o *controle* requer que haja reconciliação aritmética entre ambos. A identificação de irregularidades se torna mais fácil e a ocultação delas mais difícil, sendo necessária a manipulação de ambos os registros ou feitas entradas que compensem a irregularidade (Ibid.).

Controles geralmente envolvem tanto a *geração de documentação*, quanto a *descentralização de responsabilidades* e *verificação de resultados* (Ibid.). O primeiro deve garantir que existirá o fluxo de dados necessário para o funcionamento do processo. O último oferece um método para avaliar o funcionamento dele e geralmente envolve a reconciliação de quantidades de bens (ou serviços) e dinheiro.

O segundo tem uma importância especial para o contexto desse trabalho. Ele além conferir mais transparência ao processo, permitindo que auditorias façam inferências mais

precisas sobre o comportamento de agentes específicos, ele cria *necessidade de conspiração* para fraudar o processo.

Um exemplo básico é a operação na bilheteria de um cinema.[4] A responsabilidade de receber o pagamento e gerar a documentação inicial (o ingresso) é separada da responsabilidade de operar a cancela. Cada função é realizada por partes independentes. Periodicamente, é feita uma verificação através da contagem de ingressos e dinheiro. Se a quantia em dinheiro é inferior ao que a quantidade de ingressos sugere, então provavelmente a irregularidade partiu do primeiro agente. Para poder lucrar com essa irregularidade sem ser detectado, é necessário que o operador dos bilhetes coluda com o operador da cancela. Quanto mais agentes envolvidos, mais custosa fica essa colusão e menos lucrativa a fraude. Pode existir, todavia, dificuldade de diferenciar erros de fraudes em determinados casos, o que seria representativo de carência de verificabilidade do processo.

Quando usados para gerir relação entre empregado e empregador, controles são muito mais baseados em autorizações fundamentadas em hierarquia do que em termos contratuais (Ibid.). Nesse casos algum nível de autoridade é creditado a cargos de gerência para que o processo possa ser adaptado sem a necessidade formalizar todos os detalhes na forma de contratos. Quanto menos detalhes são formalizados em contrato, mais autoridade e discricionariedade sobre o processo deve ser conferida à esses cargos de gerência podendo deixar o processo mais vulnerável à problemas de assimetria de informação (problema do principal/agente). Essa dinâmica ilustra como, em muitos casos, contratos são conduzidos de forma muito implícita com baixa observabilidade.

2.2 Contratos inteligentes

Os desenvolvimentos da tecnologia e ciência da computação possibilitaram a implementação de algoritmos previamente inviáveis. A ideia fundamental por trás do conceito de contratos inteligentes (*smart contracts*) é aplicar o poder desse campo da ciência e tecnologia para otimizar os processos e propriedades contratuais apontadas acima (entre outros). Aliados à tecnologia de transmissão de informação, em rápido crescimento a décadas, assim como tecnologias de segurança por criptografia, é possível que implementações desse conceito fomentem a total globalização de relações contratuais, derrubando barreiras geográficas e acabando com necessidades de alguns tipos de intermediários.

Um dos primeiros estudos sobre a possibilidade de codificar termos contratuais em programas de computador pode ser visto em (ALLEN; WIDDISON, 1996). Nesse trabalho os autores focam mais na discussão da filosofia legal em torno do assunto. O trabalho mais comumente citado como pioneiro no desenvolvimento do conceito de smart

[4] Exemplo usado por (SZABO, 1997).

contracts como conhecemos hoje é (SZABO, 1997). Nesse trabalho o autor não só aponta alguns fundamentos e propriedades de relações contratuais que podem se beneficiar da prática, mas também explora algumas ferramentas propícias para tais implementações.

A princípio, smart contracts podem ser usados em todas as atividades do escopo de contratos listadas acima, da procura à adjudicação. Contudo, esse trabalho cobrirá exemplos de *negociação*, *compromisso* e *execução*, com ênfase ao último (seções 4 e 5).

2.2.1 O ancestral

O exemplo primitivo de smart contract citado por Szabo como sendo a implementação ancestral de smart contracts são as máquinas de venda. De fato, essa tecnologia, já conhecida por todos, talvez não desperte mais o fascínio das pessoas em geral, mas representa um importante passo no desenvolvimento conceitual dessas tecnologias tão disruptivas que podemos ver hoje. As máquinas funcionam através de um algoritmo computacional e mecânico executado de forma completamente autônoma. Esse algoritmo pode ser descrito como segue: o gatilho para rodar o algoritmo é a inserção de moedas por parte do cliente; após esse passo, a máquina espera o cliente selecionar o produto digitando o código referente, que deve ser disponibilizado visualmente; após esse passos executados pelo cliente, a máquina executa os seguintes passos,

(1) Conta o valor inserido, x;

(2) verifica se o preço do produto selecionado, y, é menor ou igual ao valor inserido ($y \leq x$?);

(3a) se sim, libera o produto e o troco, z, sendo $z = x - y$;

(3b) se não, libera o dinheiro inserido, x.

O algoritmo é simples, porém eficaz para cumprir a função prevista. A estrutura da máquina deve ser forte o suficiente apenas para tornar o seu corrompimento, com fins de subtração do conteúdo, um mau negócio. Ou seja, o custo da violação deve extrapolar o seu retorno. Uma máquina de venda de diamantes demandaria um estrutura com custo de violação alto demais, muito provavelmente inviabilizando a implementação, enquanto uma máquina de refrigerantes demanda uma estrutura com custo de violação muito mais baixo, tornando sua implementação um negócio lucrativo. O uso de um container de difícil corrompimento físico é um exemplo de *medida proativa de segurança*.

2.2.2 Custos de transação e Desintermediação

A vantagem clara da implementação acima é acabar com a necessidade de ter uma pessoa executando essas atividades que são tão facilmente programáveis, assim como reprodutíveis mecanicamente. Como resultado, o processo não só ficou mais barato, mas também sem as restrições de horário que podem surgir quando utilizada mão de obra humana. Pode-se dizer que o custo de transação da atividade foi reduzido. Caixas eletrônicos são outro exemplo que ilustra esse passo conceitual nessas tecnologias de desintermediação.

Os custos relacionados a praticamente todas as atividades referentes ao escopo de uma contratação são comumente agregados sob a rubrica *custos de transação* (SZABO, 1997). Pode-se dizer, portanto, que o propósito principal de implementar smart contracts é a redução de custos, ou fricções, de transações. Em várias ocasiões essas implementações viabilizarão transações previamente inviáveis, podendo dar origens a novos tipos de mercados.

No entanto, sob a perspectiva do desenho desses contratos, é importante haver uma separação dos diferentes tipos de custos que podem afetar transações e contratações. Dessa forma é possível analisar diferentes abordagens, assim como implementações de diferentes ferramentas, para alcançar o objetivo final que é reduzir fricções em transações diversas, não só otimizando custos, mas viabilizando novos tipos de transações.

No exemplo da máquina de vendas, o custo da mão de obra foi substituído pelo custo menor de manutenção do aparelho. A mão de obra pode ser interpretada como uma espécie de intermediário no processo. O uso deste tipo de intermediário incorre em custos além do salário pago ao empregado. O uso de *controles*, por exemplo, para garantir que o processo corra conforme previsto, tem custos atrelados à ele. Sob a forma de descentralização de responsabilidades, por exemplo, será introduzida a necessidade de pagar ainda mais salários e outros custos associados à gerência de pessoal. O conceito de *desintermediação* é também um tema prevalente quando o assunto é a implementação de smart contracts, especialmente em implementações *descentralizadas*. Em implementações descentralizadas, várias funções de intermédio são substituídas por *protocolos* aliados a *redes descentralizadas*.[5] Veremos mais adiante (seção 4) exemplos de como esses protocolos podem funcionar.

2.2.3 O tradeoff entre não-ambigüidade e flexibilidade

Programas computacionais são, por natureza, *completos*, i.e., eles funcionam através de uma estrutura lógica exaustiva em relação aos possíveis cenários que podem ocorrer no processo endógeno ao programa. As instruções que o programa segue são necessariamente

[5] Esses dois conceitos são definidos mais adiante.

condicionadas por critérios objetivos, não existindo espaço para a subjetividade prevalente na linguagem usada em contratos tradicionais.

Smart contracts devem, portanto, herdar essas características. Como resultado, esses contratos serão predominantemente *não-ambíguos*. Não-ambiguidade do contrato confere alta previsibilidade ao processo, evitando, em vários casos, recursos e processos de arbitração/adjudicação que geram alto custo à sociedade. Ademais, incerteza em relações contratuais pode ser uma fonte perniciosa de ineficiência econômica e instabilidade jurídica. Smart contracts são extremamente eficazes em mitigar incerteza, devido a característica acima.

Não-ambiguidade implica que o contrato será rígido à interpretações subjetivas do seu conteúdo. Se isso é positivo sob o ponto de vista da incerteza ex-ante, é negativo sob a perspectiva da flexibilidade ex-post. Rigidez do contrato implica que adaptações à cenários não-previstos no seu estabelecimento serão, na maioria dos casos, impossibilitadas. Implica ainda a impossibilidade de quebras pareto eficientes, podendo levar a execução excessiva do contrato (EENMAA-DIMITRIEVA; SCHMIDT-KESSEN, 2019).

2.2.4 Observabilidade e ações ocultas

Implementações de smart contracts podem ter efeitos adversos. Problemas de *ações ocultas* são um exemplo. Dada a natureza "invisível" de processos executados por computadores, é difícil agentes saberem exatamente o que um smart contract está fazendo. Por um lado isso pode ser positivo, pois pode poupar usuários e clientes do esforço de se familiarizar com o processo e tornar sua experiência mais conveniente em alguns tipos de transações. No entanto, essa falta de familiaridade com o processo pode dar margem a atividades não consentidas que colocam uma das partes do contrato em vantagem em relação à outra.

Um exemplo muito comum é o armazenamento de informações de clientes, por parte de algum prestador de serviço, para serem compartilhadas posteriormente com terceiros. Esse problema é especialmente comum em plataformas online, como mídias sociais, plataformas de email, etc. As políticas de privacidade são providas, porém escondidas no meio de um texto interminável de termos e políticas de uso, de forma que praticamente nenhum usuário considera toda a leitura um esforço justificável. É muito comum inclusive os termos e condições estarem em um página diferente da que contém o comando de aceitá-las.

Esse tipo de prática é antiga e prévia a smart contracts e plataformas digitais, sendo comumente referida como as "letrinhas pequenas" (*fine print*) em contratos tradicionais. No contexto de smart contracts, especialmente nas implementações em plataformas

descentralizadas, esse problema pode ser mitigado tornando o código do contrato aberto para consulta do público, prática conhecida como código *open source*. No entanto, a complexidade semântica das linguagens computacionais assim como a complexidade estrutural dos prgramas, tornam a identificação dessas ações ocultas inconcebível para a grande maioria das pessoas, problema batizado por Szabo de "smart fine print".[6] Esse problema, no entanto, não é particular a linguagens computacionais, sendo bastante prevalente em linguagem jurídica.

2.2.5 Medidas reativas, anonimato e seleção adversa.

Para assegurar a integridade de um contrato, dois tipos de proteção precisam ser providos: (i) proteção contra quebras de contrato; e (ii) proteção contra interferências externas. Smart contracts apresentam algumas características que dificultam a aplicação de leis através do recurso ao braço forte do estado, que é a principal solução utilizada sob o paradigma tradicional. Primieiramente, devido a óbvia impossibilidade de cometer atos de coerção física através das redes de comunicação sobre a qual esses ferramentas operam, como a internet. Plataformas digitais em geral continuam, no entanto, sujeitas a respeitarem determinadas leis e normas devido à possibilidade de sancionar as organizações e coagir os indivíduos ligados à elas.

Esse fato nos traz à uma segunda característica dessas instâncias de contratos que dificultam essas soluções, que é o caráter quase-anônimo de processos online. Dada a dificuldade de identificar agentes na internet, torna-se muito difícil puní-los através da aplicação da lei em alguns casos, se tonando necessário realizar investigações e rastreios que, além de ter um custo atrelado a eles, possuem uma possibilidade de sucesso limitada. Uma resposta é simplesmente desligar o processo gerado pelo contrato (e.g. a plataforma digital), por exemplo caçando o site sobre o qual o processo é gerado.

Essa possibilidade nos traz a uma terceira característica, particular de implementações descentralizadas. Nessas instâncias não é possível desligar o processo devido ao caráter descentralizado da rede que o opera. Caso as organizações e indivíduos ligados a plataforma não possam ser identificados e punidos, esse tipo de solução se torna ineficaz.

Plataformas descentralizadas aliam essas três características. Em várias instâncias elas funcionam também como organizações descentralizadas, sem coordenação central, i.e.,

[6] Uma forma de contornar esse problema é através de serviços especializados de auditoria/consultoria, que devem tentar identificar essas práticas, analogamente ao serviço prestado por advogados com contratos tradicionais. Na falta de oferta desse tipo de serviço, muitos acabam dependendo que agentes provenham esse bem público por boa fé. No contexto de plataformas descentralizadas, essa prática é relativamente comum e esses agentes são comumente chamados de "white hat hackers".

seguem um processo de governança totalmente descentralizado, provido pelos próprios usuários membros da rede descentralizada sobre a qual ela opera. Nesses casos não existe um alvo de sanções a serem impostas pelo estado. A organização ligada à plataforma é impessoal de certa forma, não possui representantes individuais, ela é representada por uma comunidade na maioria das vezes com escopo global.

Essa característica de descentralização dessas plataformas introduz ainda a dificuldade de aplicação da lei sob uma perspectiva jurisdicional. Leis e regulamentos, assim como as organizações que as impõem, são instituições com alcance restrito a territórios geográficos. Além de redes descentralizadas não apresentarem um alvo central sobre o qual essas instituições podem impor sanções, elas extrapolam fronteiras nacionais e, consequentemente, as jurisdições sobre as quais essas sanções podem operar.

Sistemas de reputação continuam sendo uma possibilidade em redes descentralizadas. No entanto o caráter quase-anônimo dos processos aliado à possibilidade de um indivíduo criar múltiplos identificadores, prejudica a eficácia desses sistemas, dando uma enorme margem a um problema de *seleção adversa*. Agentes que acumulem reputação negativa no sistema podem simplesmente criar um novo identificador e começar do zero. Como é difícil diferenciar novos entrantes de *maus* agentes, existirá um alto risco atrelado a algumas transações e esse risco deverá ser absorvido pelos termos dos contratos, prejudicando a eficiência do mercado gerado por esses processos. Em efeito, novos entrantes pagarão a conta dos maus agentes.

A quebra da anonimidade é uma potencial solução à esse problema. No entanto, existe o clara desvantagem em relação à privacidade dos agentes, especialmente em redes descentralizadas onde os registros são acessíveis por qualquer um (redes de alta *verificabilidade*). Esse tipo de solução pode ser uma ameaça ao tipo de proteção (ii) dos contratos, i.e. de interferência externa. Existe aqui um claro tradeoff entre *observabilidade* e *privatividade*.[7]

2.2.6 Execução, medidas proativas e privatividade

Dadas as características apontadas acima, implementações de smart contracts em redes descentralizadas, na sua melhor forma, devem priorizar medidas próativas de segurança. A vantagem desse tipo de medida é evitar recursos custosos à justiça comum e à aplicação da lei, assim como os potenciais problemas de seleção adversa e privacidade atrelados à dependência de identificar os históricos de transação dos agentes. Em plataformas descentralizadas, essas medidas são tipicamente providas através do desenho dos protocolos, que representam os termos dos contratos, aliados ao uso da criptografia.

[7] Ver (SZABO, 1997), Reputation Systems.

Os desenhos dos contratos devem ser cruciais para evitar potenciais quebras por alguma das partes. O caráter autônomo da execução de smart contracts, que são realizadas por programas computacionais, confere um alto nível de segurança contra essas quebras. Dado que a execução do programa é condicionada às ações das partes e tanto a verificação dessas ações, quanto a de potenciais eventos condicionais, deve ser estritamente objetiva (pelo menos sob a perspectiva do programa de computador), quebras se tornam difíceis, no melhor dos cenários sendo inviáveis ou impossíveis. Essa característica é capaz de eliminar a necessidade de confiança entre as partes para viabilizar acordos.

Essa vantagem se torna especialmente saliente na execução de contratos que devem seguir algum tipo de ordem sequencial entre as ações das partes envolvidas (SZABO, 1997). Alice não tem incentivos de cumprir sua parte do contrato no primeiro momento se não acredita que Bob terá incentivos de cumprir sua parte no segundo momento. Um exemplo comum desse tipo de dinâmica são contratos de empréstimo. Formas tradicionais de lidar com essa vulnerabilidade desses tipos de contrato são os sistemas que mencionamos acima, reputação e aplicação da lei. No entanto, como foi apontado, em instâncias decentralizadas desses contratos, esses sistemas não são eficazes ou apresentam tradeoffs que podem não ser muito vantajosos. Outra forma é recorrer à intermediários imparciais, como advogados, arbitradores, etc. Essa solução, no entanto, geralmente apresenta um custo alto, além de ser corrompível e prejudicar a privatividade do contrato. Smart contracts podem cumprir o mesmo papel, porém de forma potencialmente incorruptível e com um custo muito mais baixo, ao mesmo tempo sem abrir mão da privatividade dos contratos.

O uso de *redes distribuídas* como a internet para implementar smart contracts pode representar um alto risco à interferências externas se não tratado com cautela. Nessas implementações, interferências podem ser realizadas de forma remota e anônima, ou seja, com um risco muito mais baixo ao interferente do que em transações tradicionais. Os termos *vandalismo* e *ataques racionais* podem ser usados para identificar essas interferências (Ibib.). Vandalismos podem ser definidos como estratégias de payoff relativamente baixo, por exemplo onde o agente precisa gastar mais recursos do que ele recebe em retorno. Ataques racionais, por outro lado, são estratégias claramente compatíveis com incentivos, onde o agente pode lucrar com a prática sob um critério puramente objetivo. Prudentemente, smart contracts devem ser planejados para serem robustos contra ataques racionais *sofisticados*, i.e., que dispendem grande quantidade de recursos no seu planejamento/execução, em contraste com ataques *ingênuos*. Uma estratégia comum usada no desenho desses protocolos, é tornar o custo da interferência mais alto que o seu potencial retorno. Isso pode ser alcançado aliando algoritmos a tecnologias de criptografia.

2.2.7 Descentralização e Verificabilidade

A partir desse ponto iremos focar exclusivamente em instâncias descentralizadas de smart contracts. Como veremos mais adiante essas aplicações são construídas usando redes descentralizadas, que introduzem a grande vantagem de eliminar o ponto central de falha do sistema. O sistema fica não só mais resiliente à falhas internas como à ataques externos conforme apontados acima. O uso desse tipo de rede para criação de smart contracts e plataformas de serviço pode ser tratado sobre a rubrica de *Descentralized Ledger Technology*, ou DLT (Tecnologia de Registros Descentralizada).[8]

Outra grande vantagem introduzida por aplicações da DLT, é a alta verificabilidade do processo. Dado que os registros são compartilhados por uma rede de computadores, e não um por servidor central, as informações registradas estão muito mais disponíveis a serem auditadas a qualquer momento por agentes independentes. Para que esse tipo de sistema funcione, contudo, é necessário que os registros sejam protegidos contra manipulações maliciosas. A seção 4 mostra como esses sistemas podem funcionar.

2.2.8 Negociação e comprometimento

As seções 4 e 5 apresentam exemplos de sistemas de smart contracts usados tanto para negociação quanto execução de contratos, com ênfase no segundo. A *execução* dos contratos está ligada a instâncias de instruções computacionais que representam termos e cláusulas previamente fixadas. Já negociação pode ser vista de diferentes formas. Um exemplo bem claro é visto na compra e venda de cripto ativos nas diferentes plataformas usadas com esses propósito. O casamento entre compradores e vendedores pode ser realizado por um algoritmo de forma a reduzir fricção e aumentar a eficiência do mercado. Exemplos incluem *livros de ofertas*, *poços de liquidez*, *leilões*, etc.[9]

Outra forma menos intuitiva que processos de negociação podem tomar pode ser vista na *governança* das plataformas. Nesses processos agentes tentam entrar em acordo em relação às regras que devem ser seguidas pelos usuários e provedores da plataforma, i.e., os termos do contrato. Como será apontado mais adiante, esse processo pode (ou não) ser fomentado pelo próprio protocolo, de forma que o contrato estabelece previamente como alterações nele devem ser feitas.

Endogenamente aos sistemas formados por essas aplicações, *comprometimento* é representado pelo uso de *assinaturas digitais*. Essas são ferramentas criptográficas que permitem que a autoria de mensagens na rede seja verificada. São extremamente importantes, tanto para a observabilidade, quanto para a verificabilidade do processo. Ao

[8] Detalhes são providos mais adiante na seção 4.
[9] Ver, por exemplo, (SCHÄR, 2021).

assinar uma mensagem de transação contratual, o agente se comprometeu a aceitar os resultados da execução daquele contrato, de forma que não existem meios de reverter esse processo.

Sob uma perspectiva externa, o próprio desenho dos protocolos deve representar a principal tecnologia de comprometimento. Protocolos bem desenhados devem prevenir ações que burlem as regras estabelecidas pelos termos do contrato através de medidas proativas. Nesses casos, a própria adesão ao protocolo representa comprometimento dos agentes a seguirem as regras estabelecidas (BERTOLAI; OLIVEIRA, 2020).

2.2.9 Eventos externos e o problema do oráculo

Quando eventos externos a um contrato são usados para condicionar os termos estabelecidos, existe a necessidade de um processo de verificação desses eventos que seja capaz de fornecer as informações necessárias com certo grau de precisão. Isso gera um problema nada trivial a ser resolvido e uma potencial vulnerabilidade ao contrato. Para se comprometer com os termos do contrato, agentes precisam estar confiantes que a informação provida é precisa o suficiente. Quando essas informações estão contidas nos registros da DLT utilizada, a solução pode ser provida através de algoritmos.

Quando esse não é o caso, ou seja, a informação precisa ser alimentada por uma fonte externa à DLT, existe a necessidade de reintroduzir um intermediário no processo. Essa introdução de intermédio pode representar uma ameaça à integridade do contrato através da possibilidade de corrompimento da informação provida pelo intermediário, ou simplesmente pela incompetência dele em produzir informações suficientemente precisas. O intermediário introduz uma brecha sobre a qual o contrato pode novamente ser corrompido, de forma deliberada ou não.

Fontes centrais imparciais com credibilidade são uma possível solução, no entanto essa solução reintroduz necessidade de confiança na transação e/ou um processo de monitoramento e auditoria da fonte. Sob o contexto de processos de desintermediação via redes descentralizadas e contratos algorítmicos/autônomos, essa solução é indesejável. Ela derrota o propósito da tecnologia. Se o processo tiver que depender de intermediários auditáveis e monitoráveis, talvez seja melhor utilizar métodos tradicionais de relações contratuais e intermédios.

No contexto de plataformas descentralizadas, esse problema é conhecido como o *problema do oráculo.* O termo "oráculo" é utilizado para se referir às fontes provedoras de informações externas aos registros mantidos pela rede. Oráculos centralizados são indesejáveis pelos motivos apontados acima. Oráculos descentralizados devem utilizar ferramentas similares às utilizadas pelas demais plataformas, para criar robustez ao pro-

cesso de provisão de informação. A base para essa solução será aliar a descentralização da responsabilidade, estabelecendo uma rede de fontes independentes, a um mecanismo de incentivos para a provisão de boas informações. Estudos aprofundados sobre implementações de oráculos podem ser vistos em (AL-BREIKI et al., 2020) e (BENIICHE, 2020).

3

Primitivos de DLT

Nessa seção são oferecidas breves descrições e definições[1] a alguns conceitos e ferramentas importantes para o entendimento da teoria abordada no restante do trabalho.

3.1 Redes descentralizadas

Redes distribuídas, são redes compostas por múltiplas unidades de processamento que se comunicam entre si para produzir determinados resultados. *Redes descentralizadas* são instâncias de redes distribuídas, nas quais cada unidade tem autonomia para produzir o resultado pretendido. Cada unidade (computador) que forma a rede é referida como um *nó*. A rede não depende do funcionamento de nenhum nó específico para operar, o que implica a inexistência de pontos centrais de falha.

Uma característica importante dessas redes é que, ao contrário de redes baseadas em um servidor central, a comunicação entre os nós não pode ocorrer de forma simultânea. Mensagens viajam pela rede sendo transmitidas de nó a nó, de forma que a comunicação entre os nós é realizada de forma assíncrona, i.e., alguns nós recebem uma dada mensagens antes que outros.

3.2 Protocolos

No contexto deste material, *protocolos* podem ser definidos como um conjunto de algoritmos que comunicam, por meio de mensagens, dois ou mais computadores (SZABO, 1997). Pontas de comunicação, ou nós, são comumente referidos através de nomes como "Alice" e "Bob", mas na realidade são representativos de computadores que podem ou não estar

[1] Definições de alguns termos podem variar dependendo da fonte. Serão usadas definições propícias ao contexto do trabalho.

sendo controlados por humanos [Ibid.]. Tipicamente, usuários humanos não conhecem plenamente o funcionamento do protocolo usado. Eles dependem, por sua vez, de um conhecimento metafórico desse funcionamento que pode ser obtido através de interfaces de usuário [Ibid.]

3.2.1 Protocolos vs. controles

Protocolos são a base sobre a qual plataformas descentralizadas (e smart contracts em geral) são construídos. Eles podem ser vistos como uma evolução tecnológica de processos de *controles* no contexto de relações contratuais. Controles, tipicamente. dependem da geração de documentação e da verificação de resultados através de contagens e manipulações aritméticas. Resultados são normalmente registrados através números, que são informações facilmente forjáveis. Além do mais, pouca ênfase é dada à confidencialidade das transações.(Ibid.) Um auditor pode ter acesso à informações que podem ser usadas contra o interesse do contratante.

Em contraste, protocolos tipicamente fazem uso de tecnologias de criptografia para proteger certas informações. Resultados, por exemplo, podem ser registrados de forma criptografada, impedindo que sejam forjados. A agilidade de comunicação e processamento em relação a métodos tradicionais usados em controles, permite que os processos gerados por protocolos alcancem alto nível de eficiência relativa, incluindo em processos de auditoria. Melhorias em verificabilidade e privatividade são, portanto, duas das principais vantagens alcançadas pelo uso dessa ferramenta.

Controles são técnicas antigas e testadas com eficácia comprovada, porém possuem vulnerabilidades e ineficiências que podem ser mitigadas com uso de tecnologias computacionais e criptográficas para desenvolver algoritmos que tornem relações contratuais mais explícitas e seguras. Protocolos representam, para a era digital, o que controles representaram para a "era do papel" (Ibid.).

3.2.2 Máquinas virtuais

Máquinas virtuais são programas de computadores que visam emular um sistema computacional independente ao sistema sob o qual operam. Representam a *virtualização* de uma máquina física.[2] Tipicamente, funcionam de forma isolada ao sistema operacional paralelo, i.e., não trocam informação com o meio externo. Podem ser: (i)*orientadas a sistemas* (ou de *virtualização completa*), onde oferecem funcionalidade necessária para executar um

[2] *Virtual machine* em *Wikipedia* <https://en.wikipedia.org/wiki/Virtual_machine>.

sistema operacional completo; ou (ii)*orientadas a processos*, onde são desenvolvidas para executar programas específicos em um ambiente independente.[3]

3.2.2.1 Computação multipartes

Máquinas virtuais orientadas a processos podem ser usadas para compor protocolos implementados em uma rede descentralizada, onde cada nó compartilha o mesmo estado da máquina, formando um sistema de computação coletivo em cima da rede. Nesses arranjos, como os nós possuem máquinas virtuais com a mesma especificação, é garantido que se alimentados com o mesmo input eles produzirão o mesmo estado. Sistemas como esse podem ser usados para implementar qualquer tipo de algoritmo para formar processos de intermediação entre os diferentes nós da rede. Esses tipos de sistemas foram desenvolvidos no campo da Criptografia sob a rubrica *Multiparty Computation*. Nesse área de estudo, propõe-se geralmente o uso de criptografia para proteger o input de um determinado nó da rede, da ciência dos demais nós.

3.2.2.2 Mecanismos

Na teoria econômica um *mecanismo* pode ser definido como um modelo abstrato que, através de um sistema de comunicação com agentes participantes e regras estabelecidas na forma de um algoritmo, produz determinados resultados, tipicamente na forma de alocação de recursos. Geralmente é usada a suposição que intermediários *confiáveis* operam esses mecanismos. Esses sistemas compartilhados mencionados acima podem ser eficazes em cumprir essa função de intermédio, em alguns casos de forma incorruptível. Dada a flexibilidade computacional de máquinas virtuais, a princípio qualquer mecanismo de complexidade arbitrária pode ser implementado, desde que a não-simultaneidade da comunicação na rede seja levada em conta e não prejudique a eficácia do mecanismo (SZABO, 1997).

Um exemplo simples de implementação é a seguinte: Alice é a potencial compradora e Bob o ofertante. Alice envia uma oferta x e Bob envia o preço inicial y á máquina virtual que ambos compartilham. A máquina tem a seguinte instrução:

$$X = \begin{cases} 1 & \text{, se } x \geq y \\ 0 & \text{, cc.} \end{cases}$$

Caso o resultado seja positivo, i.e. $X = 1$, a transação pode ser efetuada ao preço de Alice, x, sendo que ela não saberia o preço inicial y, pois a informação é protegida pelo protocolo. Este mecanismo cobre apenas a fase de *negociação* do processo, deixando as demais fases por conta de outros mecanismos externos à ele.

[3] Ibid.

Dada a capacidade de executar mecanismos de complexidade arbitrária e a possibilidade de proteger informação dos demais participantes, sistemas como esses devem possibilitar a implementação de mecanismos tradicionais à teoria econômica onde se alcança resultados propícios como *revelação direta* e *compatibilidade de incentivos*.

3.3 Chaves criptográficas e assinaturas digitais

3.3.1 chaves

Conforme mencionado acima, protocolos tipicamente fazem uso da criptografia para prover segurança aos processos internos a ele. As tecnologias criptográficas comumente utilizadas nesses processos são baseadas em cima de um conceito chamado *chave criptográfica*. Uma *chave* é um pedaço de informação, e.g. um número, com um alto grau de aleatoriedade atrelado a ela, que pode ser usada para criptografar informações em códigos assim como decifrar os códigos criptografados nas informações originais. Funcionam como focos de *obscuridade* (SZABO, 1997), que possibilitam a proteção de informações confidenciais.

O alto grau de aleatoriedade torna tentativas de corrompimento da criptografia através da força bruta, uma atividade proibitivamente custosa. De fato, existe uma enorme assimetria de custo entre aumentar o tamanho da chave e tentar adivinhá-la por tentativa e erro. Por exemplo, suponha que a chave é composta por um número de n algarismos. A quantidade de combinações possíveis será dada por 10^n. A probabilidade de se adivinhar a chave em x tentativas pode ser calculada por,

$$p_n(x) = \frac{\binom{10^n-1}{x-1}}{\binom{10^n}{x}} = \frac{x}{10^n}$$

ou seja, se $n = 10$, a probabilidade de adivinhar a chave em mil tentativas seria $p_{10}(10^3) = 10^{-7}$. Se aumentarmos o tamanho da chave para onze algarismos, i.e., $n = 11$, a probabilidade de sucesso para o mesmo x seria dividida por 10. Isso significa que para manter a mesma probabilidade de acerto é preciso fazer 10 mil tentativas. Enquanto o custo do corrompimento aumentou em uma ordem de grandeza, o aumento no custo computacional para o algoritmo foi insignificante. Quando se considera que chaves tipicamente utilizadas atualmente são compostas por códigos alfa-numéricos de 32 bytes ou mais (ou seja, uma quantidade muito maior de combinações do que os 10 algarismos usados acima), essa assimetria acontece numa escala de grandeza muito mais alta.

3.3.2 assinaturas digitais

Em protocolos criptográficos assinaturas digitais são baseadas em cima da tecnologia de criptografia de *chave-pública*. Uma assinatura é fundamentalmente um meio de assegurar a ciência e/ou consentimento de um agente a uma informação de interesse. Em protocolos elas representam uma tecnologia de comprometimento. Para serem eficientes nesse papel é necessário que exista alta observabilidade, assim como verificabilidade, nesse processo. Ou seja, deve ser fácil se certificar de que as assinaturas são autênticas, i.e., foram de fato criadas pelo agente de interesse e não forjadas por um terceiro.

Para isso é possível usar chaves para criptografar e decifrar códigos contidos nas mensagens trocadas em cima do protocolo. Se Alice deseja verificar se determinada mensagem foi assinada por Bob, ela pode usar uma chave criptográfica que Bob usou para criptografar um pedaço de informação. No entanto, em uma rede descentralizada essa solução é ineficaz, pois demandaria que cada nó da rede soubesse as chaves todos os demais nós e inviabilizaria a implementação da assinatura pois ela seria facilmente forjável (qualquer nó poderia assinar em nome de qualquer outro nó).

Para resolver esse problema cada nó deve criar um par de chaves: uma *pública* e uma *privada*. A chave privada cumpre o propósito de realizar a assinatura, enquanto a chave pública cumpre o propósito de possibilitar a verificação da autenticidade sem a necessidade de usar a chave privada. Para isso, não deve ser possível descobrir a chave privada através da chave pública, embora elas devam ser matematicamente relacionadas.

Assim, Bob pode publicar sua chave pública para rede e, se Alice deseja verificar a assinatura de Bob, ela pode usar a chave pública de Bob. Se Alice usar a chave pública de Bob para assinar uma mensagem, quando o restante da rede usar a mesma chave para realizar a verificação o processo não produzirá o resultado previsto, indicando que a mensagem não foi assinada por Bob.

4

Blockchain e o computador global

Conforme já foi antecipado, redes decentralizadas podem ser usadas para manter registros compartilhados por todos os nós e esses registros, por sua vez, podem ser usados para implementar contratos e plataformas de serviços que aliam os benefícios de smart contracts com a resiliência de redes descentralizadas.

No entanto, dada a natureza de redes descentralizadas, algum sistema é necessário para garantir que os nós sigam determinadas regras ao realizarem atualizações do registro, de forma que fraudes sejam dificultadas e informações conflitantes sejam impossibilitadas. Esses sistemas são desenvolvidos através de protocolos baseados, principalmente, nos conceitos e técnicas apontadas nas seções 3 e 2.

Nesta seção será abordada a lógica por trás de alguns modelos de protocolos baseados no conceito *blockchain*. Será dado foco a modelos que podem ser descritos como *plataformas de propósito geral*, pois acomodam implementações de smart contracts de complexidade arbitrária, ao contrário de plataformas de pagamentos que propositalmente bloqueiam essa possibilidade.[1]

A primeira proposta de plataforma de propósito geral foi feita em (BUTERIN et al., 2014), que acabou servindo de base para a implementação original da Ethereum. As seções 4.2 e 4.3 são primariamente baseadas nessa proposta. O ponto de partida para ela foi, contudo, a implementação da Bitcoin[2], a desbravadora das blockhains, proposta por (NAKAMOTO, 2008).

4.1 Sistemas de registro descentralizados

O propósito central da Bitcoin na sua idealização foi ser um sistema de pagamentos descentralizado, colocado de outra forma, uma plataforma de moeda digital (NAKAMOTO,

[1] O motivo para isso é abordado mais adiante na seção 4.3, problema 1.
[2] (RUSSO, 2020).

2008). O seu funcionamento é análogo ao de um livro-razão (*ledger*). Esse livro-razão, contudo, é compartilhado por uma `rede descentralizada`, o que implica que cada nó deve ser capaz de manter o livro-razão assim como atualizá-lo.

Cada entrada no livro-razão representa um transferência entre dois agentes, um remetente e um recipiente. Essas entradas no livro-razão são feitas através da troca de mensagens entre nós, que verificam as autorias das mensagens através de `assinaturas digitais`. A validação das transferências é feita através da verificação de seus *inputs*, i.e. as "moedas" usadas para alimentar a transferência. Após uma mensagem de transferência ser enviada à rede, o protocolo verifica se aqueles inputs usados foram de fato recebidos pelo remetente em uma transação passada. Para possibilitar tal verificação, a mensagem carrega com si uma referência ás transferências anteriores nas quais as moedas foram recebidas (Figura 1). Em seguida é verificado se as moedas ainda não foram usadas em uma transação posterior. Caso ambas as condições sejam atendidas, a transação deve ser considerada válida e os *outputs* da transferência são os devidos *inputs* registrados como transferidos ao recipiente. Dessa forma, o sistema de validação de transferências do protocolo é feito através da verificação de *outputs* recebidos que não viraram *inputs* em transações posteriores, ou *unspent transaction outputs* (*UTXO*).

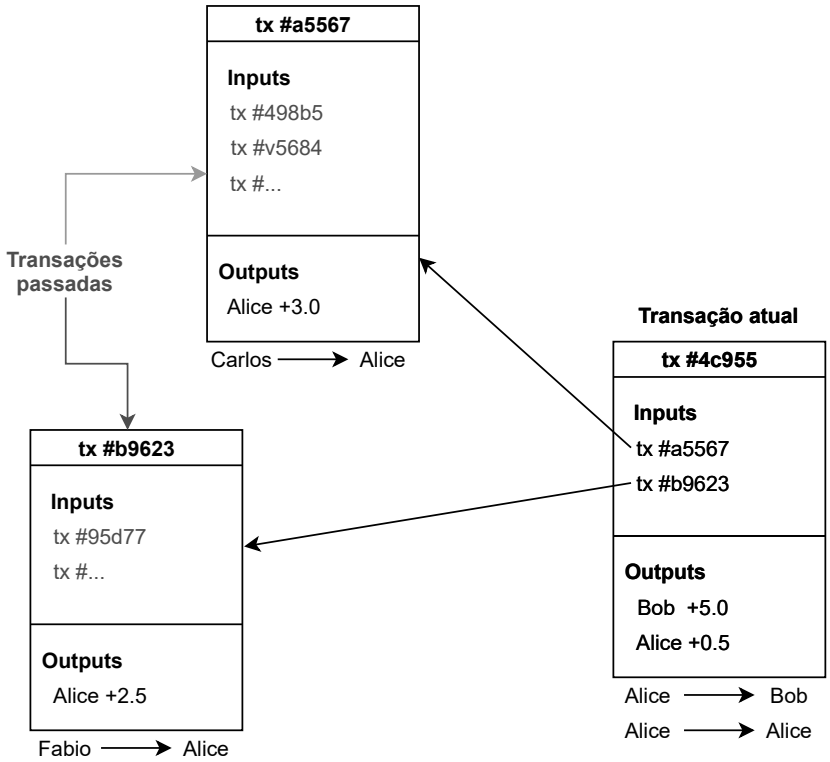


Figura 3 – Inputs de uma transferência.

4.2 Um sistema de transição de estados

Apesar de eficaz para o controle de simples transferências, o sistema de validação acima limita a implementação de transações mais complexas, como contratos com múltiplos estágios e determinados tipos de transferências condicionais a eventos paralelos (BUTERIN et al., 2014).

Uma abordagem alternativa é olhar o funcionamento do protocolo como um *sistema de transição de estados* do livro-razão (Ibid.). Em um livro-razão tradicional, seu estado seria composto pelos saldos correntes de todos os usuários, como mostrado na tabela 1, enquanto no caso do Bitcoin o estado seria o conjunto de todas UTXO—registradas aos seus respectivos donos.

Alice	2.5
Bob	120
Carlos	500
Jesus	3
Henrique	1.2
Fernanda	.000002
...	...

Tabela 1 – Um exemplo de *estado* do livro-razão.

A validação de transações pode ser vista como uma função que recebe como argumento um estado, S, e uma transação, t, e retorna um novo estado, S' (figura 4).

Supondo que S registra os saldos de cada usuário, ao invés da posse de cada UTXO, considere o exemplo de uma rede com dois usuários, Alice e Bob, onde $S = (s_A, s_B)$ é o estado corrente; s_A e s_B representam os saldos de ambos, respectivamente. Suponha que $s_A = 20$ e $s_B = 5$. Seja $f : \mathbb{R}^2_+ \times \mathcal{T} \mapsto \mathbb{R}^2_+$, a função de transição onde $\mathcal{T}$ é o conjunto universo de transferências. Se $t = (5, A, B)$, i.e., transferência de 5 u.m. de Alice para Bob, segue que,

$$f(S,t) = (15,10) \quad , \quad \text{onde} \begin{cases} S &= (20,5) \\ t &= (5,A,B) \end{cases}$$

sendo $S' = (15, 10)$ o novo estado. Se, a partir do estado original S, tivéssemos $t = (10, B, A)$, a transação não seria executada e seria retornado o estado original inalterado. Formalizando o conceito para uma rede de n usuários, teríamos $f : \mathbb{R}^n_+ \times \mathcal{T} \mapsto \mathbb{R}^n_+$, tal que,

$$f(S,t) = \begin{cases} S & \text{, se } s_i < x \\ S' & \text{, cc.} \end{cases} \tag{4.1}$$

$$\text{onde,} \begin{cases} S & = (s_1, s_2, ..., s_n) \in \mathbb{R}^n_+, \\ t & = (x, i, j) \in \mathcal{T}, \\ i, j & \in (1, ..., n), \text{ tal que } i \neq j, \\ S' & = S - S^x_i + S^x_j, \end{cases}$$

e S^x_k é uma tupla de n elementos todos iguais a zero exceto a k-ésima entrada, igual a x.

O protocolo da rede Ethereum usa um método similar para fazer a transição entre estados da rede. Exceto que nela pode ser registrado qualquer tipo de dado que possa ser expressado como uma tupla *(chave,valor)* (ANTONOPOULOS; WOOD, 2018). No exemplo anterior a *chave* seria o nome do usuário, enquanto o *valor* seria o saldo correspondente. Dessa forma, qualquer tipo de informação arbitrária pode ser registrada

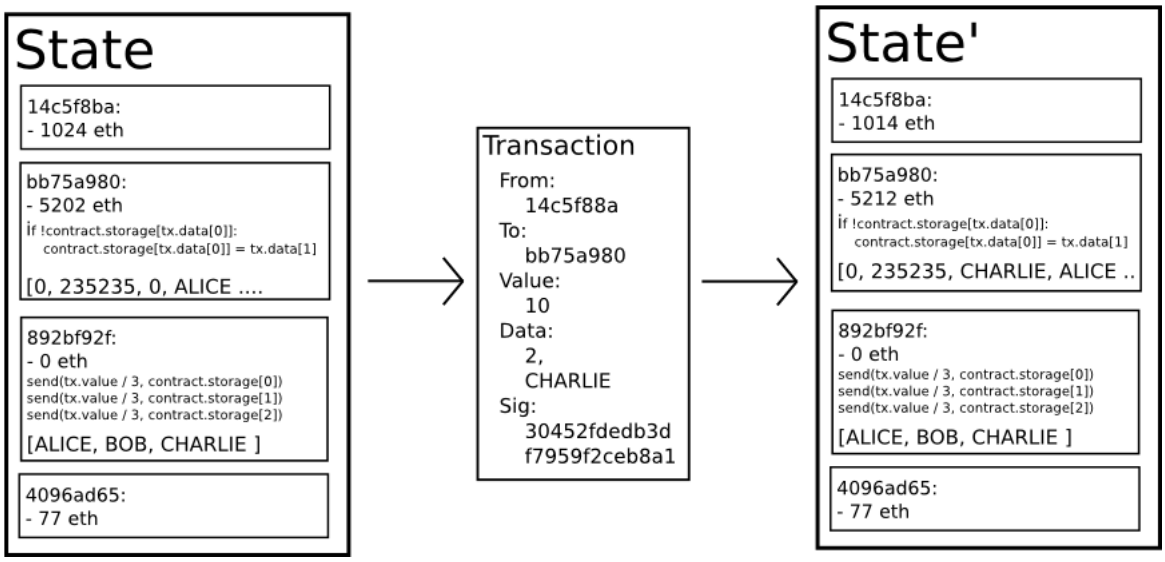


Figura 4 – Transição de estados na Ethereum

Fonte: (BUTERIN et al., 2014).

na rede, vinculadas á *chaves* que funcionam como endereços, exemplificados na figura 4 pelo valor contido no topo de cada retângulo interno (e.g. 14c5f8ba). Tipicamente, essas informações são *scripts* de programação que, através de programas computacionais chamados de *máquinas virtuais* aliados á uma unidade de processamento, podem ser usados para executar uma série de instruções complexas para alimentar a transição de estados da rede.

4.3 O computador global

Como uma rede que armazena dados genéricos vinculados a endereços de armazenamento, a Ethereum pode ser comparada à memória RAM de um computador convencional. Ela

possui uma memória capaz de armazenar valores arbitrários e usa a *blockchain* para rastrear como esta memória muda ao longo do tempo (ANTONOPOULOS; WOOD, 2018). A rede ainda é capaz de carregar e rodar códigos de programação que executam instruções computacionais complexas que serão usadas para atualizar sua memória. Efetivamente, seu funcionamento é análogo ao de um computador genérico, exceto que o estado da máquina é compartilhado por uma rede global, e as atualizações são providas por cada nó e regidas por um conjunto de regras de consenso seguidas por todos os membros da rede.[3]

O estado da rede Ethereum pode ser descrito como um mapeamento entre `endereços` e `contas` (WOOD et al., 2014). O primeiro cumpre o papel de um identificador/chave pública, enquanto a segunda carrega informações de estado correspondentes a um `endereço` particular e pode ser de dois tipos:

- **externa**: controlada por uma `chave privada`;
- **contrato**: controlada por um código com instruções computacionais.

`contas` também armazenam o balanço correspondente ao `endereço` denominado na moeda nativa da rede (`ether` no caso da Ethereum), que cumpre um papel fundamental no seu funcionamento, como veremos mais adiante.

Quando os `endereços` de contratos recebem `mensagens` as instruções descritas nos códigos são executadas. Para executar as instruções contidas nos contratos, cada `nó` da rede usa uma `máquina virtual` (MV), que deve ser igual entre todos os nós. A MV deve ser um sistema *Turing-completo*, o que implica que ele é capaz de executar qualquer algoritmo computacional de complexidade arbitrária. Essa propriedade, apesar de muito vantajosa, não é dificil de ser atingida. De fato, existem exemplos de sistemas "acidentalmente" *Turing-completos*.[4] No entanto, essa liberdade computacional expõe o sistema á potenciais brechas de segurança, especialmente aquelas relacionadas ao *problema da parada*[5]:

Problema 1 *(problema da parada) Dada a liberdade de construir programas recursivos, é possível induzir a rede a rodar um programa indefinidamente, i.e., um programa que roda em um* ***loop*** *infinito.*

Um programa como o descrito no Problema 1 pode esgotar os recursos computacionais dos `nós` que o rodarem, efetivamente incapacitando a rede. Isto pode ser feito de

[3] Ver a seção 4.4.

[4] Andreas Zwinkau, *Accidentally turing-complete* <http://beza1e1.tuxen.de/articles/accidentally_turing_complete.html>.

[5] *Halting problem*: conforme demonstrado por Alan Turing na década de 1930, não existe solução para o seguinte problema: 'dado o par arbitrário (*algoritmo*, *input*), decida se o algoritmo encerrará ou será executado infinitamente.'

forma deliberada por um agente malicioso—um ataque *DoS*[6]—ou surgir espontaneamente pela complexidade de interação entre algoritmos e condições iniciais. Em um computador global, um programa que abusa dos seus recursos efetivamente abusa de recursos globais (ANTONOPOULOS; WOOD, 2018).

A solução ao Problema 1 é implementada através de um mecanismo de contabilização computacional. A medida que a MV executa um programa, ela faz contabilidade de cada instrução computacional usando uma unidade denominada *gas*. Cada tipo de instrução (e.g uma transferência simples, registro de dados, operações matemáticas, etc.) tem um custo correspondente pré-determinado denominado em `gas`. A tabela 2 mostra um exemplo de custo de algumas operações básicas.

Quando uma transação aciona a execução de um programa, ela deve carregar em si especificados:

A) *preço do* `gas`:

- o valor a ser pago por unidade de `gas`, denominado na moeda nativa da plataforma;

B) `gas` *máximo*:

- a quantidade máxima de `gas` a ser consumida pela transação.

instrução	*custo* (`gas`)
operação $+$	3
operação $-$	3
operação $\star$	5
operação $\div$	5
relação $<$	3
relação $>$	3
relação $=$	3
$\cdots$	$\cdots$

Tabela 2 – Custos em gas

Descrição: Exemplos de custos em `gas` por operação/relação matemática usada.

O *gas máximo* funciona como uma cota superior ao número de passos computacionais que um programa pode executar. Cada unidade de `gas` consumida pela transação deve ser paga com o valor especificado pelo *preço do gas*. Dessa forma a taxa T paga pela transação pode ser calculada simplesmente por $T = pg$, sendo p o preço especificado e g quantidade de `gas` efetivamente consumida pela transação. Como o usuário paga pelo recurso computacional consumido pela transação, ele deve priorizar utilizar programas

[6] *Denial of Service.*

eficientes e, consequentemente, desenvolvedores serão incentivados a escrever programas que minimizam o consumo de `gas`.

Solução 1 *O sistema de contabilidade e cobrança de* ***`gas`*** *incentiva os agentes a tratarem recursos computacionais como um* ***bem escasso****. Ainda, aliado à exigência de estabelecer o* ***gas máximo*** *ele resolve o problema Problema 1.*

Quando a quantidade de `gas` a ser consumida por uma `transação` deve exceder a quantidade prevista pelo *gas máximo*, a transação é revertida—i.e. o estado original é restaurado—mas a taxa de transação T ainda deve ser paga para ressarcir o esforço computacional aplicado. Assim ataques que ativam programas computacionalmente custosos com intuito de incapacitar a rede são desincentivados. *Loops infinitos* são impossibilitados pela cota superior estabelecida pelo *gas máximo*.

Assim, a função de transição (4.1) deve acomodar mais um critério de validação: checar se os parâmetros A e B são acomodáveis pela transação. Um estado S não é mais composto apenas por saldos, mas por contas, c, que além do saldo podem armazenar informações de estado arbitrárias, potencialmente instruções transacionais. Temos agora: $S = (c_1, c_2, ..., c_n)$; $t = (\xi, x, i, j, \bar{g}, p)$, onde $\bar{g}$ é a cota máxima de `gas` e p o seu preço; ξ captura instruções arbitrárias contidas na mensagem de transação. $\phi(\xi, c_j, \bar{g}) = g$ representa o processo de contabilização do `gas` e retorna um valor positivo menor ou igual a cota máxima $\bar{g}$, correspondente à quantidade de recursos computacionais consumidos por t, ou seja $g \in [0, \bar{g}]$. Temos então,

$$f(S,t) = \begin{cases} 0 & \text{, se } x + \bar{g} > s_i \\ \bar{S} & \text{, se } g = \bar{g} \text{ [7]} \\ S' & \text{, cc.} \end{cases} \tag{4.2}$$

onde $\bar{S}$ é o estado original exceto que o valor $\tau = pg$ é subtraído do saldo de c_i (s_i) para ser posteriormente adicionado ao saldo da conta responsável pelo registro da transação. Em S' a mesma subtração é feita junto com todas as outras operações acionadas por t. Se $f(S,t) = 0$, o processamento da transição não é executado.[8]

Mas afinal, onde são feitas as computações e validações das transações? Todos os `nós` que rodam a MV fazem a computação de todas as transições passadas e futuras,

[7] Essa restrição não permite que o `gas` consumido coincidida com a cota máxima, de forma que o primeiro deve ser estritamente menor que o segundo para que a transação seja validada. A desigualdade estrita foi usada para facilitar a notação e não é crucial para o propósito dessa ilustração. Outra forma de fazer a representação seria usar a restrição $g > \bar{g}$ em (4.2) sendo que $g \in [0, \bar{g} + 1]$.

[8] Existem outros parâmetros técnicos que precisam ser verificados para que a transação seja processada, como, por exemplo, a assinatura digital do remetente para verificação de autoria da transação/mensagem. Ver (BUTERIN et al., 2014) para uma explicação simplificada, (DAMERON, 2018) e (WOOD et al., 2014) para a especificação técnica.

de forma que o registro é auditado milhares de vezes por dia por agentes independentes. Qualquer pessoa, com posse do equipamento necessário, pode rodar um `nó` e fazer a auditoria dos registros. As computações são feitas, portanto, nas unidades de processamento de cada `nó`. Não é necessário, no entanto, rodar um `nó` para enviar uma `transação` à rede, são necessários apenas um `endereço` com sua respectiva *chave privada* e acesso à internet. Quem usa a rede sem rodar um `nó` confia, em última instância, que outros estão o fazendo e verificando se o registro está sendo atualizado de maneira correta.

Todos os `nós` auditam o registro, porém nem todos os `nós` o atualizam. Á medida que transações são enviadas à rede, elas são recebidas por determinados `nós` que estão interessados em receber uma recompensa pelo serviço de atualizar o registro. Esses nós reúnem um *bloco* de transações, computam e validam as transições e, em seguida, adicionam o bloco ao registro. Cada bloco é registrado fazendo uma referência ao bloco antecessor, formando uma cadeia de blocos ordenada, ou *blockchain*. O último bloco a ser adicionado à cadeia contém o estado atual da rede, enquanto cada bloco antecessor contém o estado correspondente àquele ponto no *tempo*.[9]

O algoritmo de validação de um bloco pode ser descrito como segue,[10]

i. Defina o estado atual da cadeia como S_0;

ii. Defina $G = 0$ e seja $\bar{G}$ a cota superior de `gas` do bloco;

iii. Seja $t = \{t_0, t_1, ..., t_n\}$ o conjunto de transações. Faça, sequencialmente, para todo $i \in \mathbb{N}$ tal que $i < n$,[11]

$$S_{i+1} = f(S_i, t_i)$$

$$G = \sum_{j=0}^{i} g_i$$

se $S_{i+1} = 0$ ou $G > \bar{G}$ encerre e retorne erro;

iv. Seja r a recopensa do bloco. Defina $\tau = r + \sum_{j=0}^{n} p_j g_j$. A partir de S_n, adicione τ ao saldo de c_m.

A variável c_m representa a conta responsável pela criação e registro do bloco. A cada bloco adicionado, um montante r é emitido e creditado à conta responsável pelo registro do bloco. Esse montante r é fixado e é efetivamente a política monetária do sistema.

[9] No contexto de *blockchains*, tempo é medido em quantidade de blocos. É muito comum se usar como referência cronológica para eventos relevantes à essas plataformas, o número ordinal do bloco correspondente, ao invés da data do calendário.

[10] Exitem passos com verificações de especificações técnicas que estão sendo omitidos.

[11] Considerando $0 \in \mathbb{N}$.

Como $\tau = r + \sum p_j g_j$ é coletado pelo nó m e dado que m só pode adicionar um número finito de transações por bloco, ele priorizará transações j com valores $T_j = p_j g_j$ maiores. Usuários, por sua vez, com intuito de ver suas transações processadas rapidamente, são incentivados a oferecerem preços competitivos. Dessa forma um mercado de transações emerge com usuários interessados em executar transações a um baixo custo de um lado (demanda) e `nós` interessados em maximizarem seu lucro através do registro de blocos do outro (oferta).

Em *blockchains sem permissão* qualquer `nó` pode executar o algoritmo acima. Em *blockchains privadas* ou *permissionadas*, participação no processo de atualização—ou até envio de transações, verificação e consulta do registro—pode ser restrita.[12] Nesse trabalho focaremos em blockchains sem permissão, i.e., a única barreira de entrada é o acesso ao equipamento necessário para realizar cada atividade—tanto a simples usuários quanto a `nós` que atualizam os registros.[13] Apesar de ser uma característica crucial para se alcançar altos graus de descentralização da plataforma, essa abertura também aumenta a probabilidade de conflito entre blocos, já que haverá um maior número de `nós` independentes tentando adicionar blocos na mesma posição. Para resolver este problema um *mecanismo de consenso* se faz necessário.

4.4 Consenso: *proof-of-work* e *proof-of-stake*

A falta de permissão para atualizar o estado da *blockchain* é uma característica crucial para manter este processo descentralizado. A possibilidade de qualquer pessoa rodar um `nó` e/ou alimentar o registro faz pressão para aumentar a quantidade de agentes independentes executando estas atividades. Quão maior o número de `nós` independentes alimentando e auditando a blockchain, mais descentralizada será a plataforma desenhada em cima dela e mais robusta será a economia emergente à atividades indesejadas como censuras, golpes e capturas da rede para servir o interesse de cartéis e outros tipos de instituições centralizadas.[14]

No entanto, essa falta de barreiras ao processo de alimentação da blockchain vem com seu preço. Primeiro, assumindo que todos agentes concordam em relação às regras apontadas até aqui, existe uma enorme margem para blocos conflitantes. Como a todo momento existem múltiplos `nós` tentando atualizar a blockchain, diferentes `nós` podem

[12] Exemplos mais conhecidos de implementação de blockchain privada incluem a Hyperledger Fabric e Corda.

[13] Em algumas blockchains consideradas sem permissão, existem outras barreiras de entrada para nós que atualizam o registro, e.g., quantia mínima alocada na plataforma em mecanismos PoS (apresentado a seguir).

[14] Conforme apontado na seção 2.1.3, um processo mais descentralizado, torna colusões mais custosas e improváveis. Os sistema de consenso de blockchains, como veremos mais adiante, são construídos em cima dessa hipótese.

adicionar diferentes blocos ao mesmo bloco, que contém o estado corrente da rede. Quando isso acontece, existem efetivamente múltiplas versões conflitantes da blockchain.

Problema 2 *(bifurcação) Se qualquer* `nó` *pode adicionar um novo bloco, a cada atualização existe uma probabilidade positiva da blockchain se dividir em múltiplas versões conflitantes.*

O Problema 2 toca no conceito de `bifurcação` da *blockchain*. Todos os blocos são criados "apontando" para o seu bloco antecessor, mas não para o sucessor visto que até então ele ainda não existe. Dessa forma, a partir de qualquer bloco existe um único caminho "para trás" em direção ao primeiro bloco (*genesis block*). No entanto, dada a possibilidade de bifurcação da blockchain, é possível que não exista, a partir de um dado bloco, um único caminho "para frente" em direção ao final da cadeia. Como resultado, a blockchain deve assumir um formato semelhante ao de uma árvore com múltiplos *ramos* surgindo do tronco (figura 5).[15]

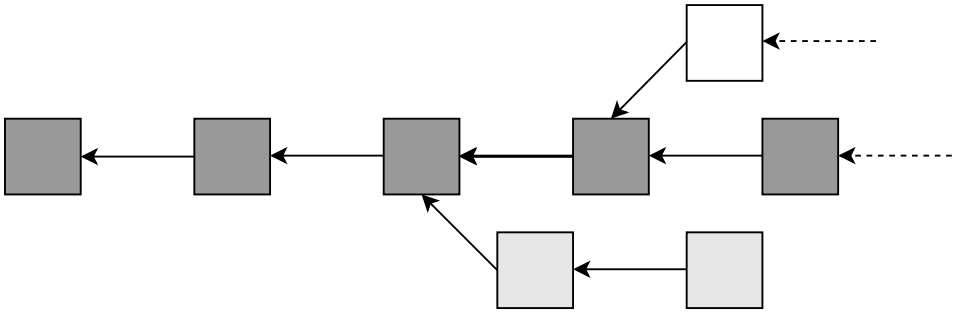


Figura 5 – blockchain após bifurcações.

Segundo, considerando que agentes são maximizadores de utilidade, é plausível que um agente *oportunista*[16]desvie das regras aceitas pela maioria, agindo em benefício próprio e prejudicando o funcionamento pleno da plataforma. O principal exemplo seria o caso em que um agente se aproveita da posição de atualizador do registro e tenta reverter uma transação (ou fazer uma transação conflitante) após a transação ter sido adicionada à blockchain. Basta o agente bifurcar a blockchain a partir do bloco antecessor ao bloco que contem a transação, criando uma nova versão da blockchain que não a contém. Usando essa técnica seria possível, por exemplo, que um agente gastasse múltiplas vezes o mesmo saldo, a fraude conhecida como *double spending*.

Problema 3 *(gasto duplo) Dada a possibilidade de* `bifurcação` *da blockchain, agentes oportunistas podem agregar blocos conflitantes, efetivamente fraudando o processo de validação.*

[15] Usaremos aqui os termos *bifurcação* e *ramificação* como sinônimos, ambos para se referir ao conceito de *forking*, e *ramos* (da blockchain) para se referir a *branches*.

[16] Podemos usar aqui a definição de "oportunismo" oferecida por (WILLIAMSON, 1975): "Opportunism is the pursuit of self-interest through taking advantage of incomplete information possessed by other parties."

Para começar a endereçar essas questões, precisamos tocar no conceito de *consenso* da rede. Conforme colocado por (KROLL; DAVEY; FELTEN, 2013), usando como exemplo a Bitcoin, o sucesso da plataforma depende de três tipos de consenso:

1 **Consenso sobre as regras**: agentes precisam concordar em relação ao critério de validação das transações. Apenas transações consideradas válidas pelo critério devem ser aceitas.

2 **Consenso sobre o estado**: agentes precisam concordar em quais transações ocorreram de fato, e em que ordem (i.e., precisam concordar em relação ao histórico de transações da economia).

3 **Consenso sobre valor**: agentes precisam concordar que a moeda nativa do protocolo tem valor e que outros a aceitarão como pagamento no futuro.

Os três tipos de consenso acima são interdependentes (KROLL; DAVEY; FELTEN, 2013). Por exemplo, se agentes não concordam em relação às regras, eles não chegarão à mesma conclusão em relação ao estado da rede em dado momento. Se não existe concordância em relação ao estado da economia em nenhum momento, não é possível se ter uma crença defensável que a moeda será aceita como pagamento no futuro.

Consenso sobre as regras é em grande parte um processo social (Ibid.). É preciso haver concordância entre os participantes da rede em relação ao que deve ser permitido para que as regras sejam codificadas nos programas de cada usuário. A forma na qual esse consenso é atingido varia entre plataformas. Algumas plataformas usam sistemas de votação internos ao processo de validação da blockchain, às vezes aliados a mecanismos para alinhar incentivos e induzir participação. A Ethereum e Bitcoin não possuem sistemas internos ao processo de validação da blockchain para coordenação da comunidade em relação ao consenso tipo 1. Nessas plataformas, agentes usam canais externos à blockchain, como mídias sociais, fóruns e até eventos e conferências presenciais para discutir e decidir implementações referentes à este tipo de consenso. São exemplos onde este processo é amplamente maleável e indefinido, apesar de já possuir práticas bem comuns.[17] Em contraste, plataformas que usam o processo de validação da blockchain para atingir esse tipo de consenso, possuem sistemas mais rígidos porém mais formalizados e definidos.[18] No contexto de plataformas descentralizadas, esse processo é comumente referido como *processo de governança*.

[17] Uma ótima referência para se familiarizar sobre como este processo pode acontecer é (BIER, 2021). No trabalho o autor explica detalhadamente a saga sobre a definição de tamanho do bloco na plataforma Bitcoin, que acabou dividindo parte da comunidade de usuários e provedores em duas facções e culminou na criação de blockchains independentes.

[18] Um bom ensaio sobre tradeoffs entre esses tipos de processos pode ser visto em *Notes on Blockchain Governance*, V. Buterin <https://vitalik.ca/general/2017/12/17/voting.html>.

Consenso sobre o estado é um problema tecnológico em redes distribuídas (Ibid.). Agentes precisam cooperar em grandes números, sob uma rede potencialmente inconfiável, para chegar á um entendimento comum sobre o estado global do sistema. Esse consenso tecnológico precisa ser alcançado apesar da possibilidade de alguns agentes tentarem desviar das regras publicadas. Veremos a seguir exemplos de mecanismos usados para resolver esse problema de consenso.

Consenso sobre o valor é similar ao consenso inerente á sistemas de moeda fiduciária (Ibid.). Se o objeto tem como única funcionalidade o seu papel como moeda, um grande número de agentes precisa concordar que o objeto tem valor e pode ser trocado por bens de consumo. Quanto mais agentes aderirem a este consenso mais valor o objeto terá, pois mais bens poderão ser trocados por ele. Expectativas sobre esse consenso podem influenciar o valor real corrente do objeto.[19]

Esses mesmos conceitos podem ser extrapolados para plataformas de *smart contracts*. O consenso sobre o valor da moeda nativa—o `ether` no caso da Ethereum—tem suma importância pois ela é a unidade usada para pagar pelos recursos computacionais utilizados, a exemplo do `bitcoin` na Bitcoin. Quanto mais aplicações são construídas em cima dessas plataformas—veremos alguns exemplos nas seções subsequentes—mais valor é agregado à moeda nativa, dando assim mais suporte ao consenso sobre o valor. A seguir consideraremos os tipos 1 e 3 de consenso como dados e iremos focar nos mecanismos desenhados para induzir o consenso do tipo 2.

Atingir o consenso tipo 2 em redes distribuídas não é uma tarefa trivial. Alguém poderia imaginar que, em caso de transações conflitantes que passaram no crivo de validação representado pelas regras do consenso 1, bastaria considerar a transação que ocorreu primeiro como válida e descartar todas as outras conflitantes como inválidas. Feito isso, teria-se uma garantia de que um histórico coerente de transações seria formado, impossibilitando ataques como descrito no problema 3.

No entanto, não existe uma referência global de tempo—i.e. um relógio comum a todos os nós da rede—e, por conta disso, a *marca de tempo* (*timestamp*) de cada transação pode ser facilmente manipulada. Além disso, a informação na rede não chega instantaneamente a todos os nós. Diferentes nós recebem mensagens em diferentes ordens, impossibilitando a utilização do sistema acima. Essas são características de *redes assíncronas*.

[19] Variações nessas expectativas estão provavelmente entre as principais fontes da alta volatilidade histórica das principais criptomoedas (KROLL; DAVEY; FELTEN, 2013).

4.4.1 *proof-of-work*

O mecanismo para induzir consenso sobre o estado da rede utilizado pela Bitcoin—e pela Ethereum inicialmente—é baseado num conceito cunhado de *proof-of-work*. Essencialmente, proof-of-work (PoW) é um instrumento usado para se provar que determinada quantidade mínima de *trabalho*[20] foi aplicada em determinado procedimento.

Nesse sistema, para adicionar um bloco na *blockchain* é necessário resolver um problema matemático de custo computacional relativamente alto. Esse problema envolve uma *função randômica verificável*, que utiliza uma tecnologia criptográfica chamada `hash`. Uma função `hash` transforma um argumento de tamanho arbitrário em um valor (*hash*) de tamanho pré-fixado—e.g. 32 bytes. Ela é uma função determinística, porém com comportamento totalmente imprevisível, sendo extremamente difícil invertê-las na prática.[21] A tabela 3 exemplifica esse comportamento. Uma mínima mudança no valor alimentado à função (um acento) produz um resultado sem nenhuma relação aparente com o resultado anterior.

x	$h(x)$
'Alice e Bob'	6da227156d931a720136c5e3d51b0005
'Alice é Bob'	d2889e0322b9c198856867c29ea8167c
0	dbd315c8e6f342d62799fa6669249ead

Tabela 3 – Função hash

Descrição: Exemplo de *hash criptográfico*, a função MD2. Na coluna esquerda estão valores alimentados à função; na direita valores entregues pela função, i.e. os *hashes*, com tamanho padrão de 16 bytes.

Seja $h(\cdot)$ representativa da função hash, Θ a informação contida no bloco e δ uma cota pré-estabelecida. A solução ao problema matemático pode ser descrita como segue:[22]

Encontre $\mathbf{y}$ tal que,

$$h(\Theta, \mathbf{y}) < \delta \tag{4.3}$$

(proof-of-work)

[20] Uma boa definição para trabalho no contexto desse mecanismo é *energia* $\times$ *tempo*.

[21] Algumas funções hash foram invertidas no passado tornando-as inseguras para serem utilizadas em sistemas de proteção criptográficos (e.g. SHA-1, MD2, entre outras). A função SHA-256, da família de funções SHA-2, é amplamente usada em sistemas de segurança online a quase 20 anos e ainda não foi violada. Ela é usada no protocolo da Bitcoin, enquanto a Ethereum usa Keccak-256, publicada pela primeira vez em 2016.

Como não é possível determinar, através de um valor $x < \delta$, o argumento $\mathbf{y}$ tal que $h(\Theta, \mathbf{y}) = x$—e dada a imprevisibilidade da imagem da função—a melhor estratégia conhecida para resolver o problema acima é testar valores arbitrários para $\mathbf{y}$ até achar uma solução.[23] Como a função é determinística, ela sempre gera o mesmo resultado quando alimentada com o mesmo argumento, tornando a verificação de veracidade do *proof-of-work* uma operação trivial, bastando alimentar a solução do problema à função e verificar se o mesmo resultado (*hash*) é encontrado. Ou seja, se um bloco é registrado apontando $\mathbf{y}^*$ como solução à inequação 4.3 e $x < \delta$ como *hash* do bloco, para verificar validade do argumento basta verificar se $h(\Theta, \mathbf{y}^*) = x$. A cota δ é inversamente proporcional ao nível de dificuldade do problema e ela pode ser manipulada para aumentar ou diminuir a quantidade média de recursos necessária para resolver o problema. Consequentemente, o tempo médio de produção de blocos pode ser manipulado através do ajuste de δ, já que o consumo de recursos por unidade de tempo é finito e previsível.[24]

Suponha, então, que uma rede funciona sob uma hipótese de sincronicidade até um parâmetro ϵ—i.e., espera-se que qualquer informação seja disponibilizada para a grande maioria da rede dentro de um período ϵ. Suponha que υ_δ é o tempo médio de produção de novos blocos pela rede sob a dificuldade δ.[25]Para se garantir que a maioria da rede terá acesso ao último bloco produzido antes que outros blocos sejam produzidos, basta fazer $\upsilon_\delta \gg \epsilon$. Dessa forma, o criador do último bloco, ou nós que tenham acesso mais rápido à publicação dele, não possuem vantagem injusta sobre os demais.

A solução à inequação 4.3 funciona, então, como uma prova de que recursos suficientes foram gastos no processo de registro do bloco. Para incentivar agentes a proverem esse bem público, o nó responsável pelo registro do bloco na cadeia tem, como recompensa ao seu trabalho, direito de adicionar, no mesmo bloco, uma transação enviando moedas previamente inexistentes ao seu endereço.[26] Esse processo de produção de blocos foi cunhado de *mineração*, fazendo alusão à mineração de ouro.

Mas mesmo com o emprego do *proof-of-work*, bifurcações espontâneas e deliberadas (ataques) continuam viáveis. Para completar o mecanismo deve haver um critério para todos concordarem qual versão da *blockchain* deve ser seguida no eventual caso de bifurcações. No trabalho que introduziu o protocolo original implementado na plataforma Bitcoin, (NAKAMOTO, 2008) sugere que seja utilizado o critério da cadeia mais longa: seguindo esse critério, quando se depararem com múltiplas versões da cadeia (ramificações),

[22] Para ser mais preciso, a condição diz que o código binário representativo do resultado de $h(\Theta, \mathbf{y})$ deve conter pelo menos η_p zeros como prefixo, sendo este último calibrado de forma que a probabilidade de que um argumento satisfaça a condição não exceda um valor p. E.g. se $\eta_p = 5$ e $h(\Theta, \mathbf{y}) =$ 0000001011..., então a condição é atendida.

[23] Nos casos da SHA-256 e Keccak-256, existem 2^{256} *hashes* possíveis e são necessárias, em média, 2^η tentativas, sendo η a quantidade mínima de zeros no prefixo do resultado.

[24] Essa última quantidade é definida pela tecnologia de chips de processamento disponíveis.

[25] E sob um dado poder computacional da rede (somando todos os nós criadores de blocos).

[26] Essa recompensa é representada por r no algoritmo de validação apresentado na seção anterior.

agentes deveriam favorecer a versão da blockchain (ramo) que tem maior quantidade de blocos registrados. Se todos os nós seguirem este critério sempre haverá consenso em relação ao estado da blockchain—desde que os demais consenso (1 e 3) também sejam mantidos.

No entanto, ao contrário do algoritmo *proof-of-work*, esse critério não é codificado nas regras do protocolo. Ele é meramente uma estratégia que, quando seguida pela maioria, estabelece um equilíbrio vantajoso para o funcionamento da plataforma, dada sua proposta inicial—i.e. o consenso sobre as regras. Chamemos essa estratégia de $\mathcal{S}_L$.[27] Por que seria racional, então, esperar que os demais nós sigam a estratégia $\mathcal{S}_L$?

Definição 7 *Seja $\mathcal{S}_L$ a estratégia na qual, quando deparado com uma ramificação, um nó segue o ramo com maior quantidade de blocos produzidos.*

O que está realmente sendo feito ao seguir essa estratégia, é favorecer a cadeia na qual mais trabalho computacional foi aplicado, ou seja, a cadeia com a maior quantidade de soluções ao problema 4.3, ou mais *proof-of-work*. Se a maioria do nós seguirem esta estratégia,[28]eles se asseguram de que a probabilidade de que um nó, ou cartel, decida o estado da *blockchain*, burlando algum tipo de regra, é relativamente baixa. Se um cartel bifurcar a cadeia a partir de certo bloco para tentar induzir uma cadeia de consenso diferente, ele precisa manter um ritmo de produção de blocos superior ao restante da rede, caso contrário sua versão da cadeia nunca será a mais longa. Para isso ele precisa competir em poder computacional com todos os demais nós que não estão em colusão com ele.

A cadeia mais longa, portanto, e por consequência a estratégia $\mathcal{S}_L$, é em efeito a alternativa mais segura para os nós que desejam seguir as regras aceitas pela maioria. Através do *proof-of-work* o processo de mineração faz nós competirem entre si para agregar novos blocos à cadeia e induz a rede a concordar em qual versão da cadeia é a mais segura e deve ser seguida. Esse processo torna-se uma "corrida computacional", tanto entre nós (para agregar blocos à cadeia), quanto entre facções da rede, ou versões da blockchain (para definir o estado). A versão que atrai a maior alocação de recursos será a vencedora. Para considerar a plataforma segura contra ataques como o do problema 3, basta assumir que a maior parte do poder computacional total da rede ($> 50\%$) está no controle de nós *honestos*—i.e. que seguem as regras de consenso.

[27] (KROLL; DAVEY; FELTEN, 2013) modela o jogo gerado pelo processo de agregação de blocos (`mineração`) e encontra que a adoção da estratégia $\mathcal{S}_L$ por todos é um entre vários equilíbrios de Nash possíveis.

[28] Na verdade o parâmetro que deveria ser usado aqui é quantidade de poder computacional, não quantidade de nós. Foi usado este último para simplificar a explanação.

Hipótese 1 *Seja ρ a fração do poder computacional total da rede em controle de nós honestos. Para a plataforma ser segura contra ataques como o do problema 3 basta que, $\rho > 1/2$.*

A hipótese acima, largamente usada nas implementações deste tipo de mecanismo, ilustra por que o grau de descentralização de uma rede é um aspecto tão importante para sua segurança. Caso um cartel de nós em conluio controle mais do que 50% do poder computacional de toda a rede, suas chances de definir o estado da blockchain se aproximam de 1 conforme sua versão da cadeia cresce. Uma rede com poder computacional bem distribuído entre partes independentes, torna um potencial conluio como esse mais custoso e improvável, conforme apontado na seção 2.1.3 com o conceito de *distribuição de responsabilidades*.

Por enquanto o mecanismo acima parece ser seguro para induzir consenso sobre o estado da rede, desde que consideremos que a fração maior de todo o poder computacional da rede é controlada por nós *honestos*. Para sustentar esse consenso, no entanto, é preciso haver concordância em relação a todo o histórico de transações, desde a origem do registro. Mas o que impede um agente oportunista de manipular blocos previamente agregados à *blockchain*? Se isso fosse possível, transações poderiam ser revertidas e o registro histórico da rede como um todo, que define o seu estado em dado momento, poderia ser arbitrariamente manipulado quebrando o consenso do tipo 2.

Suponha que a cadeia de consenso em dado momento é representada por, $B_0, B_1, ...,$ $B_i, B_{i+1}, ..., B_{i+n}$, onde B_j representa o bloco j. Á medida que blocos são agregados à blockchain eles carregam em si o *hash* do bloco anterior.[29] Ou seja, se Θ_i é a informação do bloco i e $h(\Theta_i, \mathbf{y}_i) = \theta_i$, temos que $\theta_i \in \Theta_{i+1}$ (Figura 6). Dessa forma, se alterarmos a informação de Θ_i produzindo Θ_i^*—e achando o respectivo $\mathbf{y_i^*}$ que resolva a inequação 4.3—teremos $h(\Theta_i^*, \mathbf{y}_i^*) = \theta_i^* \neq \theta_i$. Mas como $\theta_i \in \Theta_{i+1}$, para o novo bloco fazer parte da blockchain de consenso é preciso mudar essa informação em Θ_{i+1} e achar um novo $\mathbf{y}_{i+1}^*$ correspondente ao novo Θ_{i+1}^*. Como resultado, teremos $h(\Theta_{i+1}^*, \mathbf{y}_{i+1}^*) = \theta_{i+1}^* \neq \theta_{i+1}$. Como $\theta_{i+1} \in \Theta_{i+2}$ o processo precisaria ser repetido novamente e de novo para cada bloco $j > i$.

Portanto, para manipular um bloco já existente na versão de consenso da *blockchain*, é preciso refazer todos os blocos subsequentes construindo efetivamente uma nova versão da *blockchain*. Mas se os demais nós continuarem seguindo a estratégia $\mathcal{S}_L$, o agente oportunista só obterá sucesso se ganhar uma corrida contra o restante da rede na qual começou atrás.

[29] No caso da Ethereum, os blocos carregam também uma cópia do estado prévio, S_0, usando técnicas computacionais e criptográficas para reduzir a carga informacional desse dado.

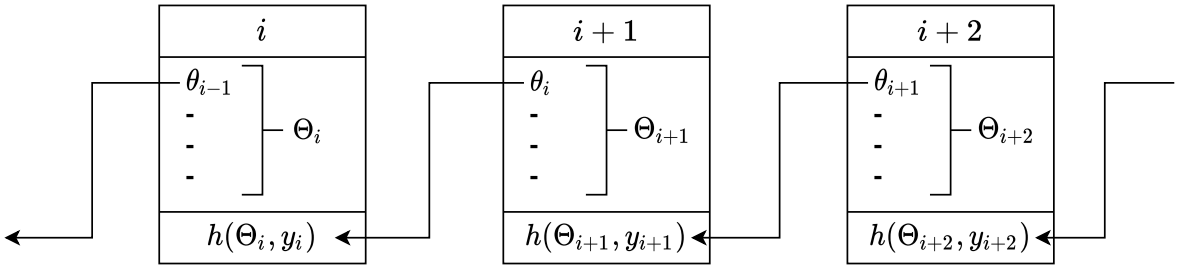


Figura 6 – Encadeamento de blocos

Descrição: Cada bloco contém o *hash* (θ) do bloco antecessor na sua informação (Θ).

Suponha que um agente oportunista decida alterar o bloco i. Se z é o último bloco no momento em que o agente começa a executar seu plano e $z - i = d$, a probabilidade de sucesso de sua estratégia é decrescente em d, dado seu poder computacional, assim como o do restante da rede. A figura 7 ilustra essa dinâmica.

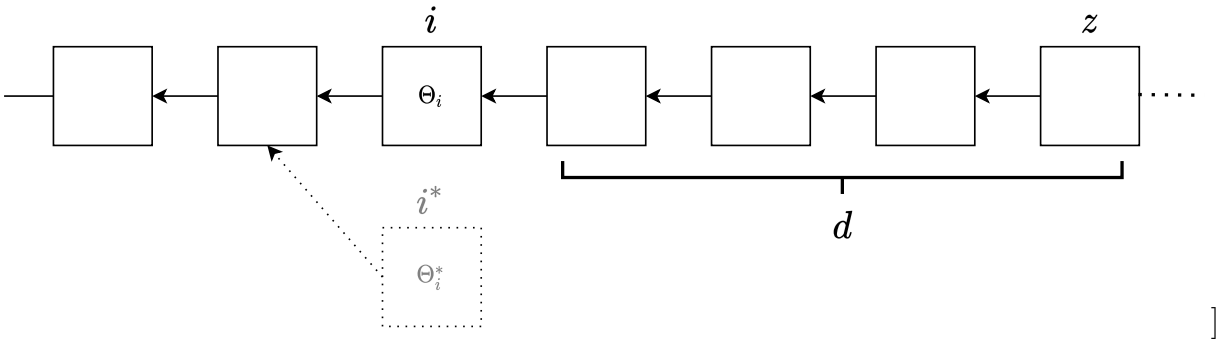


Figura 7 – Manipulação oportunista do histórico de blocos

Descrição: Para modificar a informação contida no bloco i o agente precisa criar um novo bloco a partir do bloco $i - 1$. d representa a "distância" corrente até o final da cadeia.

A partir do momento em que o agente constrói um bloco concorrente ao bloco i, ele efetivamente bifurca a *blockchain*, no entanto ele não pode comunicar seu bloco com os blocos subsequentes (figura 8a), restando a ele construir novos blocos tentando eliminar a diferença d enquanto a cadeia original continua crescendo. A figura 8b esquematiza o processo: m representa o números de blocos que o restante da rede agregou após o bloco i; enquanto l representa a quantidade de blocos que o agente oportunista agregou após o bloco concorrente i^*; e, dessa vez, $d = m - l$. A estratégia só será bem sucedida quando $d = -1$ (condição necessária). Para isso ele precisa crescer sua cadeia mais rapidamente do que o restante da rede.

Por consequência dessa dinâmica, é prudente esperar que n blocos sejam agregados após o bloco (no mesmo ramo) que contenha a transação de interesse para considerá-la como confirmada. Cada bloco agregado é uma "confirmação" de que a transação continua sendo consenso para a rede, considerando que a maioria dos nós segue a estratégia $\mathcal{S}_L$. A partir de certo valor de n, a probabilidade de que a versão de consenso da blockchain não

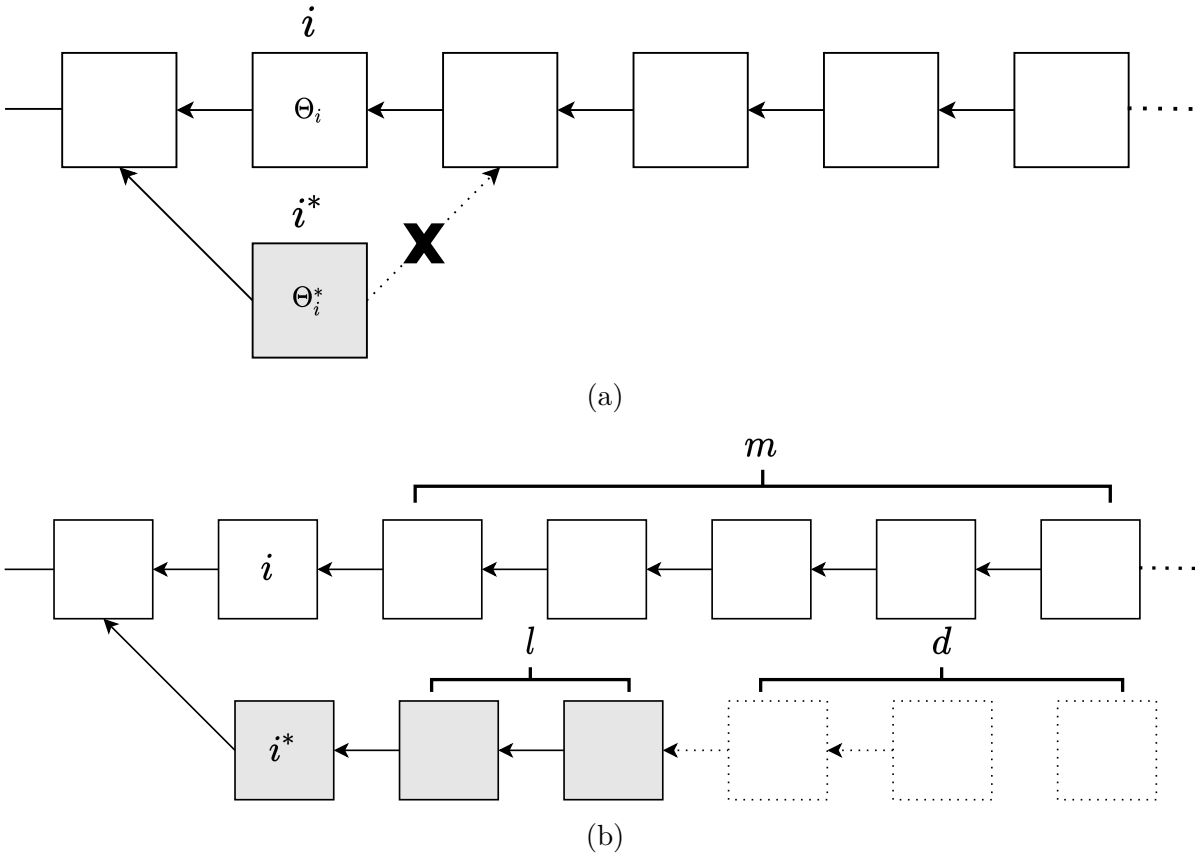


Figura 8 – Manipulação oportunista do histórico de blocos (II)

Descrição: (a) após criar um novo bloco, o agente não pode conectá-lo a nenhum dos blocos seguintes ao bloco concorrente que já foram criados; (b) resta agregar novos blocos ao bloco recém criado, criando uma nova versão da cadeia.

contenha a transação de interesse se torna desprezível.[30]

(ROSENFELD, 2014) modela um ataque *double spending* como uma cadeia de Markov em tempo discreto e estima a sua probabilidade de sucesso. O gráfico da Figura 9 mostra a probabilidade de sucesso do ataque como função do poder computacional controlado pelo agente oportunista $(1 - \rho)$ para diferentes valores de n. Note como, para n pequeno, a probabilidade (eixo vertical) converge para 1 mais rapidamente conforme o poder computacional do ataque (eixo horizontal) aumenta, confirmando o que foi apontado acima.

O mecanismo acima cria, portanto, um critério de consenso para a rede concordar em relação ao estado da blockchain através da estratégia $\mathcal{S}_L$ (o que resolve o Problema 2), ao mesmo tempo que, ao agregar o algoritmo *proof-of-work*, cria barreiras a ataques (Problema 3) provendo segurança ao protocolo e incentivos para que a estratégia $\mathcal{S}_L$ seja seguida, de forma que a crença que os demais nós seguirão essa estratégia seja defensável.

[30] Sendo essa probabilidade decrescente em n, *ceteris patibus*.

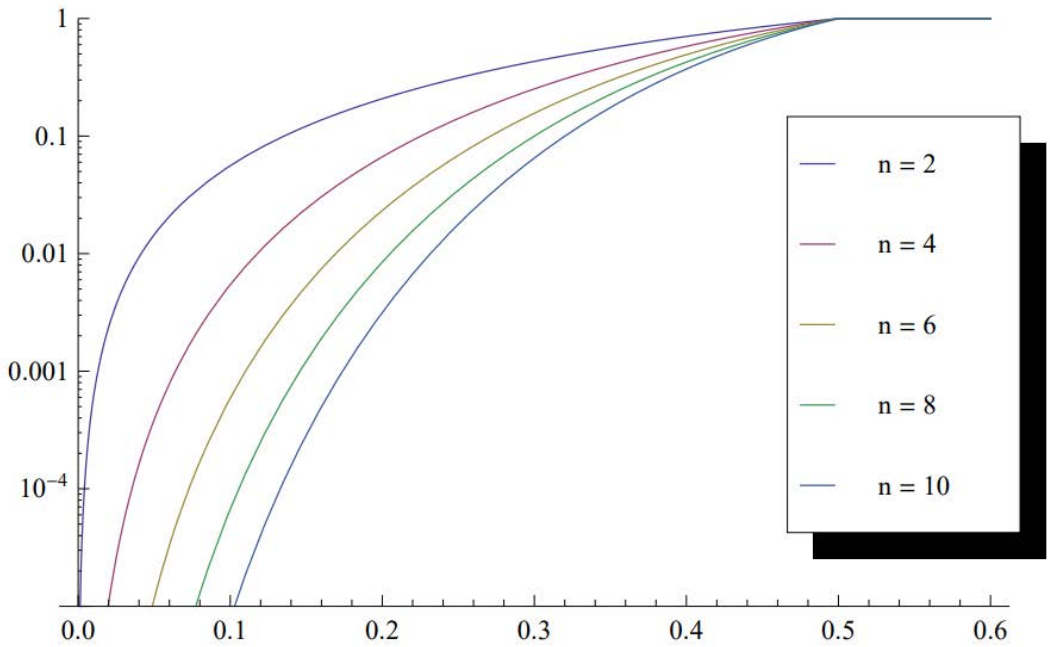


Figura 9 – Probabilidade de sucesso de um gasto duplo

Descrição: O gráfico acima ilustra a probabilidade de sucesso de um ataque de gasto duplo em função da fração do poder computacional da rede em controle do autor da fraude, para diferentes quantidades de confirmações, n. *Fonte:* (ROSENFELD, 2014).

Solução 2 *O mecanismo de consenso do* ***proof-of-work*** *dificulta ataques oportunistas que afetem o consenso da blockchain e induz a rede de nós a concordarem qual versão da blockchain deve ser aceita.*

O *proof-of-work* dificulta ataques e manipulações oportunistas aos registros da blockchain ao tornar o processo custoso, a exemplo do que o mecanismo do gas faz com ataques DoS. Existe, contudo, um tradeoff entre segurança e eficiência sendo feito. Quanto mais nós estão tentando achar uma solução à equação 4.3 para adicionar seu bloco, mais segura fica a blockchain e, ao mesmo tempo, mais dispendiosa. Uma quantidade enorme e crescente de energia é consumida pelas redes de nós que atualizam as blockchains que funcionam com este mecanismo.[31] A seguir veremos uma alternativa a ele que visa, entre outros objetivos, atacar também essa ineficiência.

4.4.2 *proof-of-stake*

Apesar de ser eficaz em atingir seu objetivo principal de induzir consenso do tipo 2 de forma relativamente segura, o mecanismo PoW é muito criticado como sendo energeticamente ineficiente. Com o crescimento de uso de plataformas que usam esse tipo de

[31] Ver, por exemplo, *Bitcoin Energy Consumption Index* <https://digiconomist.net/bitcoin-energy-consumption/>.

mecanismo, mais notadamente a Bitcoin, a quantidade de energia consumida por elas se tornou bem relevante em um contexto global, rivalizando o consumo de algumas nações. Essa característica fica ainda mais preocupante quando é considerado o potencial de adoção global desses tipos de plataformas, algo que não se concretizou até o momento. Um efeito direto dessa característica é que a atividade de mineração acaba sendo concentrada geograficamente em lugares onde existe acesso à energia barata, o que pode representar um foco de centralização à rede.

Além de ser energeticamente intenso, o mecanismo PoW tem sérias limitações no que diz respeito a escalabilidade da rede.[32] O processo de mineração é inerentemente custoso, não só em energia e processamento, mas em tempo.

Outro aspecto desse processo comumente apontado como negativo é que, em alguns casos, ele causa uma espécie de corrida tecnológica para fabricar o chip mais especificamente otimizado para ele. Isso pode resultar em um alto (e indesejável) nível de influência dos fabricantes desses equipamentos em relação à rede.[33] Além do mais, a rede pode ser limitada por gargalos na cadeia produtiva dos chips e vulnerável a potenciais choques de oferta. E, finalmente, dada escassez de oferta desse tipo de equipamento de uso específico, existem grandes oportunidades para economias de escala de grandes players, deixando o mercado pouco acessível para pequenos entrantes e contribuindo para a centralização da rede de mineradores.[34]

Dadas essas vulnerabilidades, algumas alternativas ao modelo PoW foram propostas desde a implementação do protocolo da Bitcoin. As mais notáveis dessas alternativas são as variações de um modelo cunhado de *proof-of-stake* (PoS).

Como vimos acima, a requisição de uma prova de trabalho para participar do processo de atualização da blockchain serve o propósito de gerar,

I. ***Critério de consenso***: agentes concordam qual atualização deve ser aceita como válida (visto que haverão múltiplas propostas);

II. ***Sistema de incentivos***: os agentes são recompensados se propuserem boas atualizações e penalizados se propuserem más atualizações.

Dado seu caráter descentralizado, é possível enxergar a atualização da blockchain como um processo democrático. Conforme os diversos nós propõem novos blocos, apontando para blocos já existentes, eles estão efetivamente "votando" em qual versão da blockchain deve ser considerada válida. Não é possível, no entanto, que este processo seja "puramente democrático"—i.e., um voto por pessoa—pois *pessoas* não são reconhecidas pelo

[32] *Escalabilidade* será tratada em mais detalhe mais adiante.
[33] Isso pode representar um *foco de centralização*.
[34] Esses efeitos são mitigados em alguns protocolos, e.g. Ethereum, que fazem o processo não compatível com chips de aplicação específica (ASIC's), sendo mais propício o uso de chips genéricos (GPU's).

protocolo, apenas endereços (chaves públicas). Seria possível, por exemplo, um agente oportunista criar uma quantidade suficiente de endereços para tomar conta do processo decisório da plataforma.[35] Se a plataforma fosse composta por N nós, cada um com um endereço, bastaria o agente criar $N + 1$ endereços para decidir o estado da plataforma. Consequentemente, um agente racional esperaria esse comportamento do restante da rede e tentaria reproduzir o máximo de nós possíveis, restrito apenas pela sua capacidade computacional.[36]

Portanto, não é possível resolver I de forma estritamente democrática. É necessário haver um critério para ponderar o voto de cada nó que também resolva II. A alternativa mais saliente é considerar a quantidade de recursos alocados ao processo. Agentes que alocam grandes quantidades de recursos correm maior risco de incorrerem em grandes prejuízos. Um bom mecanismo deve usar esse risco para alinhar os interesses dos responsáveis pela execução do processo com o objetivo geral de obter um bom funcionamento da plataforma.

Como vimos, o mecanismo baseado no *proof-of-work* usa, como proxy para o critério acima, trabalho computacional. Este último, por sua vez, demanda investimentos massivos em unidades de processamento, sistemas de refrigeração, eletricidade, espaço físico, entre outros. É conferido alto custo ao processo recorrendo-se a bens escassos.[37] Mecanismos do formato *proof-of-stake* visam reproduzir este mesmo efeito sem recorrer a bens materiais externos ao sistema.

Para isso eles consideram a quantidade monetária de cada um dentro da plataforma, sob as hipóteses que (i) unidades de moeda não podem ser reproduzidas e que (ii) agentes que possuem quantias substanciais devem prezar pelo bom funcionamento do sistema para proteger seu patrimônio. Em contraste com o mecanismo anterior, existe a exigência que agentes tenham quantias da moeda nativa à plataforma em seu portfólio para ter participação no processo.[38]

Hipótese 2 *O payoff de ataques adversários à rede é inversamente proporcional à quantia monetária em posse do atacante dentro da plataforma.*[39]

---

[35] Prática conhecida como *ataque Sybil*.

[36] De certa forma, o mecanismo *proof-of-work* antecipa essa dinâmica e faz os nós competirem com seu poder de processamento dentro das regras do processo, de forma organizada.

[37] Conceito cunhado de *unforjable costliness* por Nick Szabo em *Shelling Out: The Origins of Money* <https://nakamotoinstitute.org/shelling-out/>.

[38] No mecanismo *proof-of-work* o requisito para participar do processo é apenas investimento na infraestrutura necessária para rodar o algoritmo. Em teoria, um agente pode dominar a rede sem possuir nenhuma quantia da moeda nativa, desde que possua a infraestrutura necessária.

[39] No contexto desse trabalho, podemos considerar *ataques adversários* como sendo estratégias que visam dominar uma fração ρ do poder decisório da rede, suficiente para determinar o estado da plataforma em favor do atacante. ρ representa fração mínima necessária para burlar as hipóteses de segurança particulares ao protocolo. Constituem em um exemplo de *ataque racional sofisticado* contra a rede (ver seção 2.2.6).

Para explorar essa hipótese, o poder de voto é feito proporcional ao poder aquisitivo de cada um na plataforma, ou seja, proporcional ao *risco de prejuízo* (*stake*) que cada um tem, caso existam disrupções ao funcionamento da plataforma (consensos 1 e 2) que afetem negativamente o valor da moeda nativa (consenso 3). Nós são sorteados pelo protocolo e ganham o direito de agregar um bloco à cadeia com probabilidade proporcional à quantia de moedas que possuem. Este processo tem um custo computacional quase desprezível se comparado com o mecanismo anterior. Isso leva a (i) um grau de eficiência mais alto, (ii) um custo de entrada mais baixo e (iii) dá maior margem para se reduzir a recompensa por bloco e, consequentemente, a taxa de crescimento da oferta monetária. Esse último item se dá pelo baixo custo relativo de se produzir blocos, sendo similar ao custo de rodar um nó. Esse fato, no entanto, é a semente dos problemas que ilustraremos mais adiante.

O primeiro passo do processo deve ser a seleção de nós para criarem e registrarem os blocos à cadeia. Como todo processo no protocolo deve ser determinístico (todos os nós devem achar o mesmo resultado), é necessária alguma forma de simular aleatoriedade ao processo. Uma solução simples é ter nós rodando uma função $g(\cdot)$, de baixo custo computacional, com comportamento similar à função $h(\cdot)$ da seção anterior.[40] A função pode receber como argumentos alguma informação arbitrária contida na blockchain, a marca de tempo do bloco, t, e a chave secreta do nó que tenta agregar um novo bloco, κ^s. Para ganhar direito de agregar o bloco, a seguinte condição deve ser atendida[41]:

$$g(X, \kappa^s, t) < \mu \cdot \delta, \quad s.a. \quad t > t_0 \tag{4.4}$$

sendo X a informação da blockchain usada para gerar aleatoriedade (também chamada de entropia); μ a quantidade de moedas do nó que roda a função; δ o parâmetro de dificuldade; e t_0 a marca de tempo do bloco imediato (bloco "pai")[42]. Uma segunda possibilidade é usar a condição a seguir,[43]

$$g(X, c, t) < \delta, \quad s.a. \quad t > t_0 \tag{4.5}$$

onde c é o identificador de uma unidade monetária em posse do nó. O nó deve então rodar a função para cada moeda que possui até que a condição seja atendida. Considerando que cada moeda c no sistema tem chances iguais de atender 4.5, a probabilidade do nó ganhar o direito de agregar o próximo bloco deve crescer linearmente em relação à quantidade de moedas que possui, a exemplo da condição 4.4.

[40] Ela funciona de forma aparentemente randômica.

[41] Uma condição semelhante é usada por (KING; NADAL, 2012) e (VASIN, 2014), que foram implementados no protocolo da Peercoin e Blackcoin, respectivamente. Na Peercoin, pioneira do *proof-of-stake*, o parâmetro δ é multiplicado por uma variável cunhada de *coin age* (idade da moeda), que é o somatório dos períodos (medidos em quantidades de blocos) que o nó guardou cada unidade monetária desde o último bloco criado, ou do momento em que a recebeu (o menor entre os dois períodos para cada moeda). (VASIN, 2014) aponta vulnerabilidades resultantes da introdução dessa variável.

[42] I.e., o bloco ao qual se esta vinculando o novo bloco.

[43] Esse formato recebeu a alcunha de *follow-the-satoshi* fazendo referência à unidade básica no sistema da Bitcoin (1 satoshi= 10^{-8} bitcoin).

A escolha da informação X é determinante para a *previsibilidade* do processo, i.e., com quantos blocos de antecedência um nó consegue prever que atenderá à condição. Considere aqui, que X é tal que um bloco só consegue saber que atende à condição no momento em que poderá adicionar o bloco à cadeia, a exemplo dos mecanismos *proof-of-work*.[44]

Em ambos os casos o nó usa sua chave secreta para assinar a criação do bloco. Os demais nós podem verificar a validade do bloco rodando uma função de verificação $v(\cdot)$ que retorna um elemento de $\{0, 1\}$. Essa função serve o propósito de verificar de forma indireta que a condição 4.4 (ou 4.5) foi atendida, assim como verificar que o bloco segue as regras codificadas no protocolo. Para que um bloco B_i seja aceito, é necessário (mas não suficiente) que,

$$v(B_i, t_i, \kappa_i^p) \cdot v(B_{i-1}, t_{i-1}, \kappa_{i-1}^p) = 1 \tag{4.6}$$

sendo κ_j^p a chave pública do nó que criou o bloco B_j. A função acima verifica se o bloco B_j é valido, de acordo com t_j, e se assinatura do bloco foi criada com κ_j^s (no caso da condição 4.5, verifica se κ_j^p é o dono de c_j no bloco B_{j-1}). A equação 4.6 indica que, para um bloco ser aceito, tanto ele quanto seu antecessor precisam ser válidos.

4.4.2.1 Nada em jogo

Conforme já apontamos na presente seção, é comum redes distribuídas funcionarem com um dado nível de *latência*[45]. Isso implica que a defasagem entre o momento em que um bloco é publicado e o momento em que a publicação é percebida por um dado nó é aleatória e possui variância significativa.

Devido à latência da rede e, talvez, a possibilidade de mais de um nó atender a condição 4.5, é possível que ocorram ramificações como descrito no problema 2. Como o custo adicional de se criar um bloco é desprezível, a melhor estratégia para um nó, quando deparado com uma ramificação, deve ser tentar criar blocos em todos os ramos, sob a expectativa de que existe uma probabilidade positiva de que qualquer uma das versões da cadeia seja a futura versão de consenso. Assim, a probabilidade de receber a recompensa é maximizada. Se essa estratégia fosse seguida pela maioria, ramificações nunca se resolveriam, impossibilitando uma solução ao problema 2.

Problema 4 *Como não existe custo adicional associado à criação de blocos, não é custoso desviar da estratégia de consenso.*

[44] Essa propriedade pode ser alcançada, por exemplo, exigindo que X contenha informações do bloco "pai".

[45] O termo *latência* pode ser utilizado para englobar tanto o período referente à comunicação entre os nós quanto o período referente ao desenvolvimento e publicação de blocos (ALASHAIKH, 2021).

Em outras palavras, dado o problema 4 acima, teoriza-se que tentar prolongar todos os ramos da blockchain é uma estratégia fracamente dominante para cada nó, individualmente. O problema acima ficou conhecido como *nothing at stake* (ou "nada em jogo") e é considerado na maioria das análises desse tipo de mecanismo. Essa vulnerabilidade pode ainda facilitar estratégias similares à descrita no problema 3 ao aumentar a probabilidade que bifurcações fraudulentas sejam seguidas por uma fração da rede.[46]

Definição 8 *Seja $\mathcal{S}_N$ a estratégia na qual, quando deparado com uma ramificação, um nó tenta agregar blocos a todos os ramos da blockchain.*

(SALEH, 2021) no entanto, modela o mecanismo e encontra um equilíbrio em que todos os agentes tentam seguir a estratégia de consenso (a cadeia mais longa), induzidos apenas pela hipótese 2. Ele modela a hipótese definindo o valor real da moeda como sendo estritamente decrescente no período (medido em blocos) necessário para se obter consenso, a partir do momento em que acontece uma bifurcação. Dessa forma, qualquer estratégia que adie a obtenção do consenso pela rede tem um efeito negativo sobre o payoff do agente, que possui moedas no sistema. É condição suficiente para induzir esse equilíbrio, que exista uma cota inferior à quantia de moedas necessária para participar no processo, que seja suficientemente grande em relação à recompensa por bloco. Logo, se α é a quantia mínima necessária para se criar blocos, pode-se usar a condição 4.4, sujeita à $\mu \geq \alpha$, tal que $\alpha \gg r$.[47]

O problema 4, contudo, é amlpamente visto como uma ameaça ao funcionamento de protocolos baseados em proof-of-stake. O leitor mais atento deve ter notado que a hipótese usada por Saleh é mais forte que a hipótese 2. Além do mais, a condição achada no trabalho é dependente no fator de desconto dos agentes, que é mais uma hipótese que precisa ser tomada para produzir essa condição.[48] O autor considera, ainda, que o payoff da criação de blocos é fixo, desconsiderando as taxas de transação.[49] Será considerado, então, de forma mais cautelosa, que é possível que $\mathcal{S}_N$ seja considerada ótima por uma fração da rede, o que representaria fonte de instabilidade ao sistema e potencialmente impossibilitaria o consenso em relação ao estado da blockchain.[50]

No nosso exemplo, por enquanto, estamos considerando que a aleatoriedade é gerada imediatamente antes da proposição do novo bloco. Assim, em caso de ramificações,

[46] Veja, por exemplo, (BROWN-COHEN et al., 2019).

[47] O efeito sendo gerado por essa condição é que, se $\alpha \gg r$, o efeito desvalorização domina o efeito recompensa no payoff esperado do agente que utiliza $\mathcal{S}_N$. A condição exata encontrada é $\alpha \geq \frac{r}{\beta^k(1-\beta)^2}$, sendo β o fator de desconto do agente e k o número de blocos de vantagem que um ramo precisar ter sob os concorrentes para ser considerado o consenso. Essa condição é suficiente para induzir todos os nós a seguirem $\mathcal{S}_L$. O resultado pode ser visto na *proposição 2* do trabalho (SALEH, 2021);

[48] Veja nota de rodapé 47.

[49] Faremos uma simplificação similar neste trabalho. Veja nota de rodapé 52.

[50] (BUTERIN, 2014) argumenta que o valor da moeda proveniente da integridade do protocolo é um bem público e, como tal, seria sub-ofertado.

os diferentes ramos irão eleger nós independentemente um do outro. Isso pode implicar em um cenário, caso uma fração dos nós usem a estratégia $\mathcal{S}_N$, em que os ramos crescem de forma independente um do outro.

Existe um outro cenário que pode ocorrer. Alguns protocolos introduzem previsibilidade ao processo. Uma forma de fazer isso é usar como X informação contida em um bloco a uma distância de n blocos do ponto da blockchain sendo prolongado. Dessa forma, nós conseguem prolongar todos os ramos de uma bifurcação no ponto onde a distância n do bloco que contém X é igual. Ramos com um comprimento menor que n blocos não são mais alongados de forma independente. Nesse cenário, nós que usem a estratégia $\mathcal{S}_N$ conseguem incluir blocos em todos os ramos, que podem crescer indefinidamente caso a estratégia seja adotada por uma fração significativa da rede. Continuemos, por agora, a considerar que existe previsibilidade mínima ao processo ($n = 1$).

Uma solução simples para lidar com a ameaça representada pela adesão de nós à estratégia $\mathcal{S}_N$ é atrelar um custo à ela. Se o custo for alto o suficiente, nós devem ser dissuadidos de utilizá-la. Como no nosso exemplo os diferentes ramos elegem nós de forma independente, esse custo deve ser associado ao ato de prolongar o ramo errado—i.e., um ramo que não será o consenso. Ou seja, nós que escolherem o ramo errado devem incorrer em um custo por isso. Isso pode ser feito aplicando uma penalização a esses nós. Corre-se o risco de penalizar nós "honestos"[51], mas o efeito negativo sobre o payoff do uso da estratégia $\mathcal{S}_N$ é criado.

A forma trivial de executar a penalização é confiscar parte das moedas dos nós que desviarem do consenso. No entanto, como os diferentes ramos da blockchain não se comunicam entre si, pois são independentes, não é trivial identificar, a partir do ramo do consenso, que um determinado nó desviou dele, agregando bloco em outro ramo. È necessário introduzir alguma forma de garantir que essa informação seja provida. Uma alternativa é dar um incentivo para que outros nós sejam pró-ativos e façam essa provisão.

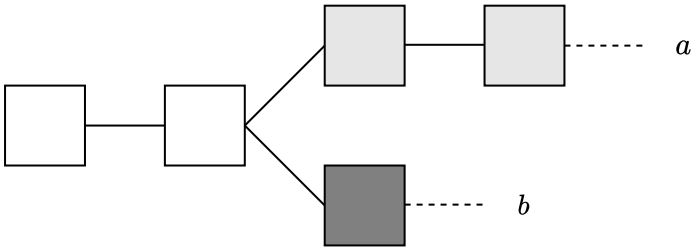
O mecanismo pode funcionar da seguinte forma: quando um nó agrega um novo bloco à blockchain ele recebe a devida recompensa, r, no próprio bloco; em algum bloco contido em um ramo independente ao ramo ao qual ele agregou o bloco, ele recebe uma penalização $-s$; caso o ramo no qual ele agregou o bloco seja o consenso o seu payoff é r, caso contrário o seu payoff é $-s$.[52] Se um agente acha que determinado ramo tem probabilidade p de virar o consenso, ele apenas prolonga o ramo caso $pr \geq (1-p)s$. Ramos com probabilidade baixa então devem ser abandonados. Se o agente espera que

[51] Nesse contexto nós *honestos* são nós que tentam seguir a estratégia de consenso. Nesse cenário, mesmo que sejam honestos, nós podem escolher o ramo errado. Isso poderia ocorrer por falta de informação ou simplesmente por azar. Penalização similar ocorre no mecanismo proof-of-work sob a forma de consumo de energia entre outros recursos.

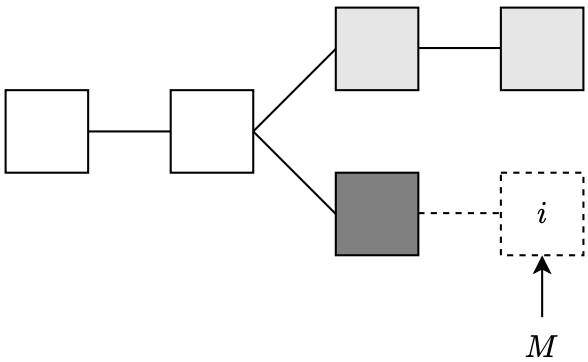
[52] Na verdade, caso o ramo que o nó prolongou vire consenso, o payoff dele seria $\tau = r + \sum pg$ conforme definido no algoritmo de validação na seção 4.3. Será considerado aqui $\tau = r$ com fins de simplificar a explicação.

os demais nós sigam a estratégia $\mathcal{S}_L$, conforme um ramo ganha vantagem sob os demais, sua probabilidade de virar consenso deve se aproximar de 1, e a melhor resposta deve ser abandonar os ramos nos quais $p < (1-p)\frac{s}{r}$, se estabelecendo um critério de convergência para o maior ramo, de forma a oferecer uma solução ao problema 2. Para que a penalização seja feita, porém, é preciso que algum nó acuse o desvio no ramo de consenso. Para isso, o protocolo pode recompensar o nó responsável pela acusação com um valor v.

Uma ilustração de como o mecanismo funcionaria na prática seria: (i) em $t = 0$ o nó M se depara com uma ramificação da blockchain, composta por dois ramos, a e b; (ii) em $t = t_1 > 0$, M tenta estender ambos os ramos, obtendo sucesso apenas em b, agregando o bloco i, incluindo sua recompensa r; (iii) em $t = t_2 > t_1$, o nó N percebe que o ramo b não é tido como consenso, e envia uma transação acusando a criação de i; (iv) caso a acusação seja válida (e inédita), a transação é validada incluindo $-s$ ao saldo de M e v ao saldo de N. As figuras 10 e 11 esquematizam o processo.



(a) $t = 0$



(b) $t = t_1$

Figura 10 – Nothing at Stake

Descrição: (a) em $t = 0$, a blockchain contém uma bifurcação se dividindo em dois ramos a e b; (b) em $t = t_1$, o nó M, utilizando a estratégia $\mathcal{S}_N$, consegue vincular o bloco i ao ramo b.

Para que o mecanismo acima funcione é necessário que haja tempo suficiente para que a acusação seja feita, de forma que nós que desviem do consenso não possam, uma vez que perceberem que serão penalizados, vender suas moedas retirando seus fundos da plataforma e ficando efetivamente impunes ao desvio.

Para driblar essa possibilidade, o protocolo pode exigir que os nós depositem suas moedas (ou parte delas) em um contrato, C_1, de forma que a custódia dessas moedas

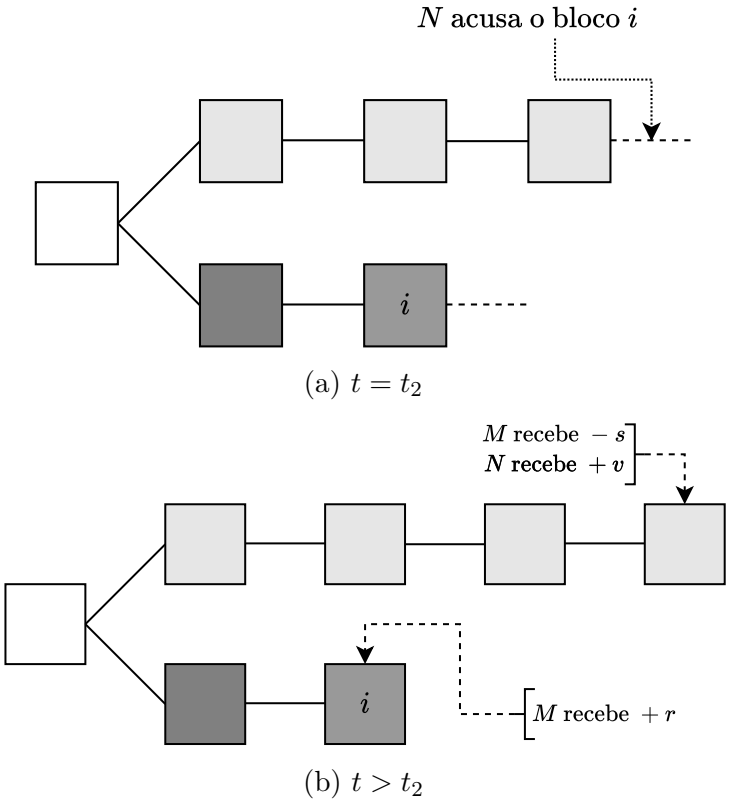


Figura 11 – Penalização em PoS

Descrição: (a) em $t = t_2$, o nó N acusa que o bloco i foi criado fora do consenso, apresentando a devida prova; (b) após a acusação ser processada, a penalização é registrada, assim como a devida recompensa.

não mais seja de cada um do nós. A condição 4.4 deve considerar, então, μ como sendo apenas a quantia depositada em C_1. Para reaver essa quantia (ou parte dela), o nó deve enviar uma transação solicitando que ela seja enviada a outro contrato, C_2, onde ela deve ficar parada durante um período $q > t_2$, suficiente para que um potencial desvio seja identificado e a devida penalização aplicada. Durante o período q, a quantia transferida para C_2 não contribui para o poder de voto do nó.

Entretanto, o valor da penalização, s não pode ser tão alto em relação à recompensa r. Caso contrário, nós podem ser dissuadidos de participarem desse processo, dada a possibilidade de serem penalizados mesmo que não tentem deliberadamente desviar do consenso. Ademais, é possível que resoluções a simples ramificações sejam dificultadas. Antecipando uma potencial penalização, poderia ser ótimo para um nó esperar que uma ramificação se resolva antes de tentar prolongar qualquer um dos ramos (figura 12). Ele apenas agiria quando a probabilidade p de determinado ramo fosse alta o suficiente para compensar o tamanho da relação s/r. Isso poderia atrasar a obtenção de consenso na rede e prejudicar sua funcionalidade. Por outro lado, penalizações pequenas implicam

custos pequenos a tentativas de fraudes como a do problema 3.[53]

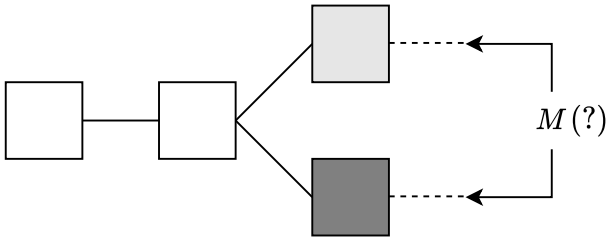


Figura 12 – Bifurcação em PoS

Descrição: Em $t = 0$ o nó M se depara com uma ramificação. Se M acredita que determinado ramo tem probabilidade $p \geq 1/2$ de virar consenso, ele prolonga o ramo apenas se $pr \geq (1-p)s$, caso contrário ele espera.

Vamos considerar agora o caso em que existe previsibilidade de n blocos no processo. Nesse caso, estratégia $\mathcal{S}_N$ se manifestará através do prolongamento de múltiplos ramos simultaneamente (figura 13). Nós deveriam ser penalizados, então, por proporem blocos em múltiplos ramos. Não se corre o risco de penalizar nós que escolherem o ramo errado por acaso, apenas nós que deliberadamente utilizarem $\mathcal{S}_N$, podendo assim ser utilizada uma penalização mais severa. O mecanismo de penalização pode ser similar ao anterior, mas dessa vez o nó N acusa o desvio quando nota que um mesmo nó agregou blocos em ramos concorrentes. Nesse formato, um nó maximiza seu payoff —salvo em casos de tentativa de ataque—seguindo o ramo com maior probabilidade, sem correr o risco de sofrer penalizações.

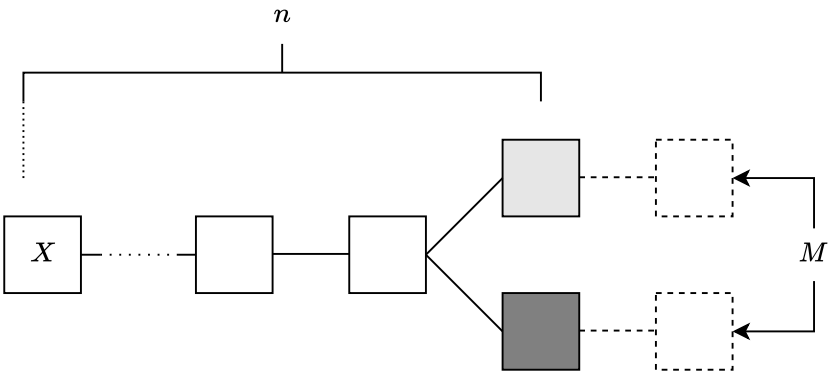


Figura 13 – Nothing at Stake com previsibilidade n.

Descrição: Com previsibilidade n, o nó M executa a estratégia $\mathcal{S}_N$ incluindo um bloco em cada ramo.

Previsibilidade, no entanto, facilita fraudes de gasto duplo e abre portas para coordenação entre maus atores, com possibilidade de fazer subornos aos eleitos para propor no-

[53] Protocolos sem previsibilidade também são mais vulneráveis ao que é chamado de *grinding attack*. Nesse tipo de estratégia, nós tentam manipular a aleatoriedade do processo para obter vantagem no sorteio dos blocos subsequentes.

vos blocos. O tradeoff da introdução de previsibilidade é analisado em (BROWN-COHEN et al., 2019) e (BAGARIA et al., 2019). Propostas de modelos que usam mecanismos com previsibilidade incluem (BENTOV; PASS; SHI, 2016), (KIAYIAS et al., 2017) e (BAGARIA et al., 2019).

4.4.2.2 Simulação gratuita

Um outro problema derivado do baixo custo de se produzir blocos surge do que (POELSTRA, 2015) chama de *costless simulation* (simulação gratuita). Como já foi apontado, o principal incentivo para agentes respeitarem o consenso da rede é a potencial perda do valor depositado na plataforma. No entanto, uma vez que esse risco não existe mais esse incentivo desaparece. Se um agente decide vender suas moedas em algum momento, retirando seus fundos da plataforma, sua chave secreta continua sendo capaz de recriar blocos criados por ele no passado. Alguém em posse de chaves que não possuem mais moedas vinculadas na plataforma em determinado momento no tempo, poderia tentar simular sua própria versão da blockchain, sem precisar adquirir uma fração significativa do total de moedas depositadas.

Essa estratégia é possível pois não existe um rastreio do tempo sendo feito pela blockchain. Qualquer agente pode, em teoria, bifurcar a cadeia em algum ponto no "passado" (medido em blocos) e tentar recriar a história. Dado o baixo custo de simulação, implementações *ingênuas* de mecanismos proof-of-stake falham em inviabilizar tal estratégia.

Problema 5 *Dado o baixo custo adicional de produzir novos blocos, simulações do histórico são viáveis para agentes sem fundos depositados na plataforma.*

Uma resposta é requerer que moedas fiquem depositadas durante longos períodos (medidos em blocos). Essas medidas forçam que, para executar tal estratégia, seja preciso reescrever longos trechos do histórico. *Ataques de longo alcance*[54], contudo, continuam sendo viáveis.

Mas em uma cadeia consensual, i.e., formada majoritariamente por nós honestos, longos trechos da cadeia correspondem a relativamente longos trechos de tempo. Uma simulação de um longo trecho da cadeia seria, então, facilmente identificável pelos nós como ilegitima e, consequentemente, seria rejeitada pela rede. Existem dois casos, todavia, em que nós honestos seriam vulneráveis à ataques de longo alcance: (a) quado estivessem entrando na rede pela primeira vez e (b) quando ficassem fora do ar por longos períodos, suficientes para que tais ataques fossem realizados. Como nesses casos o nó não teve acesso

[54] Veja (DEIRMENTZOGLOU; PAPAKYRIAKOPOULOS; PATSAKIS, 2019) para uma análise de *long range attacks*.

a todo ou significante parte do registro histórico da cadeia até o momento da entrada na rede, ele não tem referência para diferenciar a cadeia válida de uma simulação barata.

Em ambos casos, o problema pode ser contornado da seguinte forma: seja M um nó no cenário (a) ou (b) e q a carência para um nó reaver moedas depositadas,[55]

(1) M verifica um bloco j, com q blocos descendentes consecutivos, que faz parte do consenso;

(2) M descarta qualquer bloco não descendente de j;

(3) a partir de j, M segue $\mathcal{S}_L$.

O histórico válido, até j, deve ser o que o produz, que é único. 2 previne que M seja vítima de ataques de longo alcance: quando um nó conseguisse reaver suas moedas e retirar seus fundos da plataforma, seria tarde de mais para executar o ataque. Em qualquer momento antes, ele estaria sujeito à penalização. Em qualquer ponto após j, M pode estar seguro que não será vítima desses ataques. A partir desse ponto, ele depende, como todos os demais, da segurança do mecanismo em induzir $\mathcal{S}_L$ para chegar ao estado atual.

Para obter 1, no entanto, M precisa confiar em alguma fonte externa para lhe prover j. Esse ponto quebra a objetividade característica do modelo *proof-of-work*. Nos protocolos que usam proof-of-work, M conseguiria determinar a versão válida da blockchain usando um critério puramente objetivo. Isso permite o protocolo a funcionar praticamente sem nenhuma necessidade de confiança entre os membros da rede.

Modelos *proof-of-stake* que recorrem aos métodos apresentados, necessariamente introduzem um nível de subjetividade ao critério, ao exigir que agentes confiem em terceiros para proverem a informação apontada em 1. Como essa subjetividade só ocorre nos casos apontados acima, ela recebeu a alcunha de *subjetividade fraca* por (BUTERIN, 2014).

4.4.3 Considerações

Percebe-se, portanto, que existe um grande tradeoff sendo feito pelo ganho de eficiência introduzido pelo modelo proof-of-stake. A aparente ineficiência do modelo *proof-of-work* é, na verdade, o que garante que os protocolos atinjam um nível de segurança funcional através de um mecanismo extremamente simples. Os dois problemas que abordamos acima são introduzidos pelo ganho de eficiência do proof-of-stake, que tenta se livrar do custo para seguir o consenso e atrelar, por outro lado, um custo para desviar dele.

[55] q é medido em quantidade de blocos desde o último bloco criado pelo nó.

Essa tentativa, no entanto, abre portas para vulnerabilidades particulares a esse modelo que não existem no paradigma de prova de trabalho. Para lidar com essas vulnerabilidades são introduzidos altos níveis de complexidade—que não são abordados neste trabalho—e essa complexidade provê, em muitos casos, superfície para diferentes tipos de ataques à rede. Uma sumarização de alguns tipos de ataques pode ser vista em (BAGARIA et al., 2019).

Existe um outro grupo de protocolos que são baseados na teoria *Byzantine Fault Tolerance* (BFT) de sistemas computacionais distribuídos. Essas propostas visam atacar outra característica indesejável de blockchains, a falta de finalidade absoluta às transações. Elas não usam variações de $\mathcal{S}_L$ para atingir consenso. Contudo, elas continuam vulneráveis à subjetividade que apontamos acima, caso se baseiem na hipótese 2 para atingir segurança no sistema. Exemplos mais notáveis incluem a implementação do Ethereum 2.0 (BUTERIN; GRIFFITH, 2017; ZAMFIR, 2017) e do Algorand (GILAD et al., 2017; CHEN; MICALI, 2019).

O propósito dessa seção não foi explorar a fundo esses tipos de protocolos, mas apresentar o propósito *comum* às diferentes implementações, assim como os problemas mais fundamentais que são introduzidos e os mecanismos mais simples para lidar com eles. O trabalhos citados nessa seção são indicados para o leitor interessado no assunto.

O modelo PoS representa, portanto, um importante avanço no desenvolvimento da tecnologia blockchain, principalmente no que diz respeito à eficiência energética dos processos baseados nela. No entanto, ele introduz dificuldades ao alcance de consenso, por parte da rede, de solução nada trivial e repleta de tradeoffs. Essas dificuldades são provenientes, em grande parte, do fato que esses protocolos utilizam apenas recursos internos ao sistema para prover incentivos e, por isso, dependem do próprio histórico que estão tentando formar para impor custos a desvios dele (POELSTRA, 2015).

4.5 *blockchain* e a camada 1

Blockchain é, atualmente, um termo usado para definir uma série de sistemas diferentes em vários aspectos. Característica comum entre a maioria deles, é que o seu principal propósito é manter um registro compartilhado por uma rede, que pode ser facilmente consultado e auditado. Esse registro é, tipicamente, protegido por criptografia e organizado em blocos, que reúnem as informações relevantes ao seu estado alimentadas ao sistema pela rede de nós. Essa organização é realizada através da interação de um subconjunto de membros da rede com o protocolo que rege o processo de atualização e validação das informações alimentadas pela rede. Esse protocolo, por sua vez, é formado por um conjunto de regras acordadas pelos membros da rede e codificadas em uma série de algoritmos. As

características particulares às diferentes implementações de blockchain devem variar de acordo com a solução prática que visam oferecer.

Conforme já foi apontado anteriormente, é objetivo deste trabalho focar em implementações que são,

i. **sem permissão**, i.e., qualquer um pode participar do sistema em qualquer capacidade, desde que tenha acesso à infraestrutura necessária;

ii. **públicas**, i.e., os registros estão disponíveis online para o acesso de todos;

iii. **descentralizadas**, i.e., o protocolo é construído sob a hipótese que uma grande quantidade de agentes irá desempenhar os papéis necessários para manter o seu funcionamento.

Para atingir funcionalidade com essas três propriedades, blockchains normalmente utilizam as seguintes ferramentas (ANTONOPOULOS; WOOD, 2018):

(1) uma rede *peer-to-peer* conectando e transmitindo mensagens entre os membros;

(2) mensagens, no formato de transações, que representam transições de estado do registro;

(3) um conjunto de regras de consenso que definem o que representa uma transição válida;

(4) um mecanismo de consenso que induz agentes independentes a cooperarem na imposição das regras acordadas para validar novas transações.

Como vimos, a última ferramenta acima, conceitualmente a mais complexa, deve envolver (a) um algoritmo que permita que a validação de transações seja feita de forma descentralizada; e (b) um esquema de incentivos para induzir agentes a seguirem as regras no processo de validação. Todas as propostas que envolvem implementações dos conceitos acima são representativas do que podemos chamar de *camada 1* (*layer one*) de uma blockchain.

4.5.1 O trilema

Das três características citadas acima, as duas primeiras—i.e., *sem permissão* e *pública*—podem ser postas sob a mesma rubrica de *aberta*. *Abertura* é uma propriedade facilmente reproduzível. Basta que as especificações da camada 1 da blockchain prevejam que essas

propriedades sejam devidamente atendidas.[56] Já a terceira, *descentralização*, envolve um nível de complexidade mais alto.

Para uma blockchain ser segura sob a hipótese de descentralização, é necessário haver uma quantidade relativamente grande de agentes independentes participando dos processos relevantes ao seu funcionamento. Simplificadamente, podemos dizer que uma blockchain é "de-facto descentralizada" caso essa quantidade de agentes seja suficientemente grande, o que pode variar a depender da fonte. Essa quantidade é função de várias variáveis como, por exemplo, adesão à rede e uso das plataformas que ela comporta.

Para a blockchain comportar um aumento no seu uso, todavia, ela precisa ser "escalável", o que significa que ela precisa ser capaz de atender o aumento na demanda por transações sem ter sua performance prejudicada. Essa performance pode ser medida na quantidade de transações que ela consegue processar em determinado período de tempo. Um aumento na carga de transações alimentada à blockchain implica em um aumento na carga informacional que precisa ser processada, comunicada e validada por todos os nós da rede, que precisam também entrar em consenso em relação ao histórico, que cresce a uma taxa maior. Esse aumento implica, *ceteris paribus*, em um aumento no tempo necessário para que essas atividades sejam realizadas. Quanto mais nós precisam se comunicar e concordar em relação ao estado do registro, maior deve ser esse tempo. Quanto maior esse tempo, mais vulnerável fica a blockchain a bifurcações (problema 2). Consequentemente mais instável fica o processo de consenso e menos resiliente fica a rede a tentativas de ataques que tentem tirar proveito dessa instabilidade (problema 3). Ou seja, a quantidade de recursos necessária para executar tais ataques diminui. Esse relacionamento representa um canal de tradeoff entre duas de três propriedades importantes, que formam uma espécie de trilema, como veremos a seguir.

Para isso, vamos definir por,[57]

(1) *descentrabilidade*, a capacidade do sistema rodar com todos os nós utilizando até R em recursos;

(2) *segurança*, a resiliência do sistema à ataques que alocam até S em recursos;

(3) *escalabilidade*, a capacidade do sistema de processar, em média, até T transações por período de tempo;

[56] Existe espaço para nuance nesse argumento. *Na prática*, para uma blockchain ser realmente *sem permissão*, ela precisa ser á prova de censura, o que é uma propriedade mais difícil de se atingir. Para ter essa última característica pode ser necessário envolver um conjunto de regras e mecanismos propícios, assim como um determinado grau de descentralização. Para os méritos do argumento acima, considere que, *na teoria*, a propriedade *sem permissão* é uma função apenas das especificações do protocolo.

[57] *Ethereum Wiki*, <https://eth.wiki/sharding/Sharding-FAQs>.

No parágrafo anterior ilustramos como acontece o tradeoff entre 2 e 3. Segue dele que existe um feedback negativo entre T e S. Tomemos como hipótese, que R é inversamente proporcional a quantidade de nós que compõem a rede. Logo, conforme a teoria apresentada nessa seção, segue que deve existir um feedback negativo entre R e S. R, por outro lado, deve ser um fator limitador à T, pois todas as transações precisam ser processadas e todo o registro armazenado por todos os nós da rede, individualmente e de forma a respeitar ainda as hipóteses de segurança.[58] Fecha-se aqui um triângulo de tradeoffs que precisa ser abordado por todas as implementações de camada 1 de blockchains abertas.

Primeiro vamos nos atentar a primeira variável. No nosso modelo, R deve ser inversamente proporcional à quantidade de nós que compõem a rede.[59] Portanto, para obter uma blockchain com um alto nível de *de-facto* descentralização queremos minimizar R. Mas T deve ser fator limitador dessa descentralização, pois existe uma cota superior à quantidade de nós que conseguem processar T transações por segundo, sob um parâmetro R de recursos, respeitando as hipóteses de segurança que mantenham S constante.

T, por sua vez, deve ser inversamente proporcional ao custo marginal de se enviar transações e, consequentemente, diretamente proporcional, *ceteris paribus*, ao número de usuários e ao valor agregado às plataformas. Para atrair um grande número de usuários, queremos maximizar T, sujeito a S e R. Porém, como foi apontado acima, os últimos serão limitadores ao nível T que poderá ser atingido. Portanto, o principal problema que implementações de camada 1 visam solucionar é encontrar R e T que maximizem a utilidade gerada pela plataforma, para um determinado parâmetro de segurança S. Vejamos alguns exemplos práticos desse trilema em ação.

A partir de 2015, cerca de 4 anos após o lançamento da Bitcoin, a comunidade mais ativa no entorno da plataforma começou a discutir propostas de aumentar o tamanho máximo de cada bloco, que até então era 1MB, com fins de aumentar T, i.e., escalar a plataforma. Uma fração significativa dessa comunidade, no entanto, se opôs às propostas pelo efeito reverso que a mudança teria na de-facto descentralização da plataforma, através do aumento de R. Essa divergência culminou na divisão da plataforma em 2017, em duas plataformas independentes, cada uma seguindo uma direção nesse tradeoff.[60]

No mesmo ano, cerca de 2 anos após seu lançamento, a Ethereum começou a sofrer com congestionamento da plataforma, resultando em períodos de confirmação e

[58] Na maioria das implementações existe uma modalidade de nó, comumente chamado de *lightweight*, que não armazena todo o registro da blockchain, terceirizando esse armazenamento para nós *completos*. Com fins de simplificação, será considerado aqui que tais nós não existem. As conclusões seguirão as mesmas.

[59] R é diretamente proporcional ao custo de se rodar um nó, ergo, inversamente proporcional, *ceteris paribus*, à quantidade de agentes dispostos a realizar essa atividade.

[60] Esse episódio ilustra como o processo de coordenação para obter consenso sobre as regras pode acontecer em plataformas de governança externa em casos de alta controvérsia. Vide seção 4.4 (consenso tipo 1) e nota de rodapé 17.

taxas de transação inconvenientemente altos. Dada alta demanda da Ethereum, várias plataformas de camada 1 começaram a surgir oferecendo taxas de transação menores em troca de um menor grau de de-facto descentralização. Essas blockchains atraíram grande movimento, principalmente de usuários que não podiam pagar as altas taxas de transação da Ethereum, assim como de plataformas de games desenhadas em blockchain, que se beneficiam muito de um aumento na agilidade e diminuição no custo de transações. Essas blockchains têm um custo R de se rodar um nó consideravelmente mais alto e apresentam níveis de descentralização muito mais baixos que a Ethereum. Um exemplo notável é a Binance Smart Chain, com quantidade de nós duas ordem de grandezas menores que a Ethereum.[61]

Uma abordagem alternativa para alcançar maiores níveis de T é investir na comunicação entre diferentes plataformas de camada 1 e assumir que cada uma terá uma gama de aplicações. Essa estratégia, porém, apesar de eficaz em relação à escalabilidade, sacrifica o nível de segurança do ecossistema. Um fator N de aumento na escalabilidade implica em uma diminuição na segurança do ecossistema na mesma proporção pois uma quantidade de recursos $\sim 1/N$ será necessária para atacar uma plataforma particular.[62]

A proporção desse tradeoff é definida, é claro, pelo protocolo utilizado pela plataforma, podendo ser mudado através de mudanças nesse último. Essas potenciais mudanças devem ser acordadas através do consenso social definido na seção 4.4(consenso tipo 1). Pode se dizer que os processos relevantes a esse tipo de consenso formam a *camada 0* da plataforma, e são particulares a cada comunidade formada no entrono de cada plataforma.

Logo, se $T = w(R)$, sendo $w(\cdot)$ representativa da tecnologia utilizada no protocolo da plataforma, existe demanda por soluções que redefinam w para relaxar o tradeoff existente entre os dois parâmetros. Na divisão da plataforma Bitcoin mencionada acima, um lado da divisão mitigou o problema de escalabilidade explorando o tradeoff existente entre R e T, enquanto o outro lado decidiu introduzir mudanças em w para relaxar esse tradeoff e permitir um nível de escalabilidade moderadamente mais alto.[63]

Outro exemplo de solução no nível protocolar, é o mecanismo de consenso. Uma das vantagens significantes do PoS em relação ao PoW é exatamente sua maior escalabilidade. Plataformas que visam atender demanda por escalabilidade comumente utilizam variações de mecanismos baseados em PoS. Ainda outro exemplo importante de técnicas aplicadas aos protocolos é tecnologia *sharding*. Essa técnica efetivamente divide a blockchain em várias sub-cadeias e requer que nós verifiquem apenas a sub-cadeia circunstancialmente relevante a ele, diminuindo assim a carga informacional do processo e

[61] <https://docs.binance.org/smart-chain/validator/overview.html> ; <https://ethernodes.org/>, acessado em 5/4/2022.

[62] Vide nota de rodapé 57

[63] Em particular, a mudança introduzida foi a tecnologia Segregated Witness, ou *Segwit* (ANTONOPOULOS, 2017, Appendix D).

permitindo que blocos sejam registrados paralelamente sem dependerem um do outro. O exemplo mais notável de implementação dessa tecnologia é a proposta da Ethereum 2.0, que até o momento ainda encontra-se em fase de desenvolvimento. Essa solução apresenta, é claro, tradeoffs particulares e uma exploração detalhada deles foge do escopo deste trabalho.[64]

4.5.2 Anonimato

Outra propriedade importante que algumas implementações de camada 1 procuram oferecer é *anonimato*. Nas plataformas mais populares, como Bitcoin e Ethereum, é possível, através de técnicas sofisticadas de rastreamento de transações, associar diferentes endereços a um único dono, e eventualmente identificar esse dono (SPAGNUOLO; MAGGI; ZANERO, 2014; LIAO et al., 2016). São exemplos de plataformas consideradas *pseudônimas* (ALASHAIKH, 2021). Plataformas como Monero[65] e Zcash[66], exemplos de plataformas consideradas *anônimas*, visam solucionar essa vulnerabilidade ao anonimato dos agentes através de técnicas de criptografia como *ring signatures*[67] e *zero-knowledge proofs*[68]. Uma descrição do funcionamento dessas plataformas foge do escopo deste trabalho.

4.5.3 Camada 2

Uma outra abordagem para as soluções apresentadas acima são as soluções de *camada 2*. Tratam-se de protocolos (smart contracts) desenhados em cima das plataformas de camada 1, usando o mecanismo de consenso dessa última, mas processando transações independentemente ao protocolo da camada 1, com intuito de alcançar propriedades que não podem ser reproduzidas por essa última.

Aplicações que visam alcançar escalabilidade incluem principalmente: (i) *state channels*, exemplo mais notável desse sendo a Lighning Network na Bitcoin; (ii) *plasma*, e.g. Polygon na Ethereum; e (iii) *rollups*, e.g. Optimism (*optimistic* rollup) e Zksync (*zero-knowledge* rollup) na Ethereum. Uma descrição das característica e tradeoffs apresentados por essas soluções pode ser vista em (SGUANCI; SPATAFORA; VERGANI, 2021).

Aplicações que visam simular anonimidade às transações incluem *mixers* (ou *tumblers*) e protocolos baseados na tecnologia zero-knowledge proofs. Descrições dessas solu-

[64] Ver (HAFID; HAFID; SAMIH, 2020).
[65] <https://www.getmonero.org/>.
[66] <https://z.cash/>.
[67] Tecnologia que permite que a identidade de autores de transações sejam ofuscadas, usando um algoritmo que agrupa múltiplas assinaturas de agentes diferentes e produz uma única assinatura que não pode ser revertida.
[68] Essa tecnologia, por sua vez, permite que um agente valide uma informação, de acordo com um critério previamente estabelecido, sem ter acesso ao conteúdo dessa informação.

ções podem ser vistas em (FENG et al., 2019).

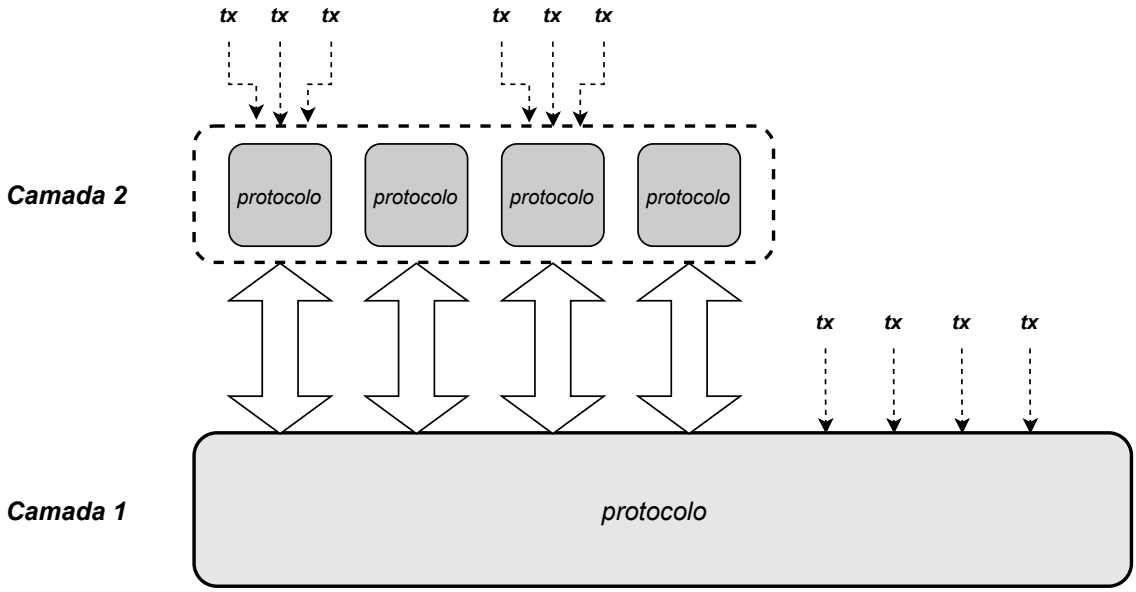


Figura 14 – Camada 2

Descrição: Protocolos de camada 2 processam transações e comunicam o estado para a camada 1.

5

Camada de Aplicações

Dada a possibilidade de se desenvolver programas computacionais (*smart contracts*) que utilizam o sistema de transição de estados representado pela camada 1, várias aplicações podem ser implementadas em cima dela formando uma camada de plataformas dependentes dessa primeira (figura 15). Essas aplicações são comumente chamadas de *Dapps*, uma abreviação para "decentralized applications". Esse termo será utilizado aqui para se referir a estes exemplos de plataformas desenvolvidas em cima das plataformas de camada 1, tipicamente através do uso de smart contracts, mas ele pode ser usado de outras formas por outros autores.

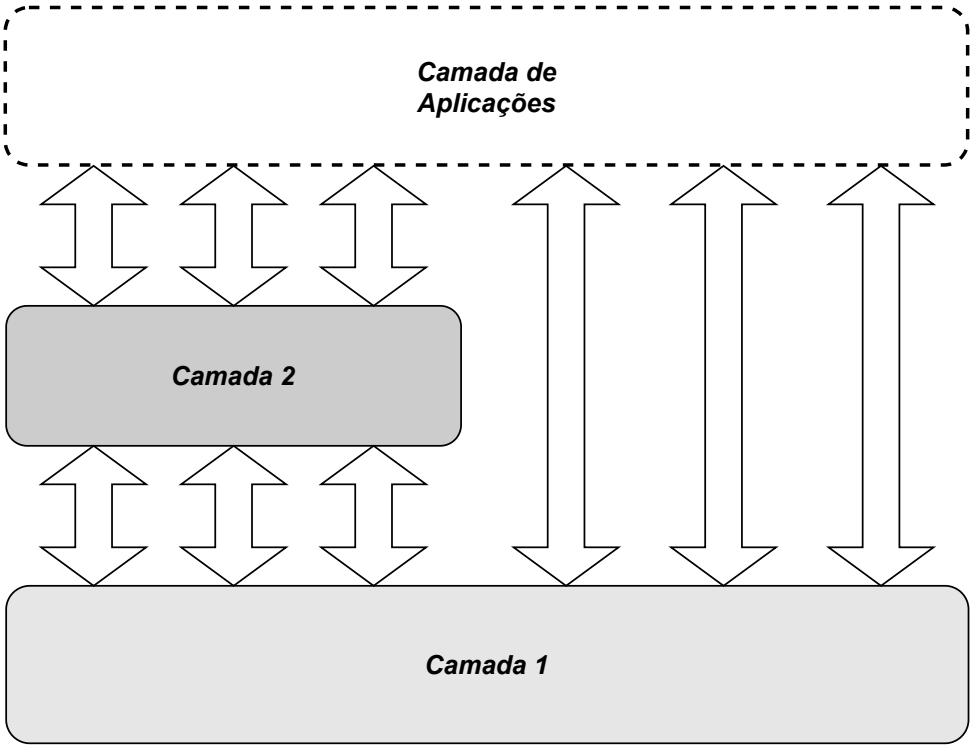


Figura 15 – Camada de aplicações

Descrição: As diferentes camadas da Web3.

Já existem vários casos de uso desse tipo de aplicação, propostos e implementados, que podem variar de registros de propriedade (imóveis, arte, etc.) a mídias sociais e plataformas de e-comerce. A área mais impactante até agora é sem dúvida a de instrumentos financeiros, formando o subgrupo de plataformas conhecido como DeFi (Decentralized Finance). Nessa seção abordaremos duas categorias de aplicações que estão entre os casos de uso mais significativos: *tokens* e *ativos sintéticos*.

5.1 Tokens

Tokens podem ser considerados como a unidade atômica do novo ecossistema da internet formado por aplicações descentralizadas, conhecido como *Web3* (VOSHMGIR, 2020). Eles normalmente funcionam como um *certificado ao portador* e podem ter várias aplicações. Alguns descrevem tokens como um método para inserir propriedade no meio digital. Outros como uma forma de simular escassez. Ambas descrições tem seu mérito, porém elas não fazem jus ao quão consequente essa ferramenta pode ser, na humilde opinião deste autor.

Criptomoedas são exemplos de implementação de tokens em plataformas descentralizadas, tipicamente o token *nativo* da plataforma de camada 1 na qual ele é emitido e transacionado, e.g. BTC na Bitcoin e ETH na Ethereum. No entanto outros tipos de tokens podem ser emitidos via smart contracts aproveitando o protocolo de transição de estado que elas oferecem.

O conceito utilizado por essa ferramenta é antigo e pode ser rastreado aos primeiros registros feitos por civilizações babilônicas (FERGUSON, 2008, p. 28). Civilizações tão antigas quanto os Sumérios (4500 BC - 1900 BC) usavam tabletes de argila para registrar informações como transações e dívidas que poderiam ser apresentadas como comprovantes em determinadas situações. A própria manifestação digital do conceito é completamente independente da tecnologia blockchain (e.g., unidades de saldo numa conta de banco). Contudo, o desenvolvimento dessa tecnologia, em especial plataformas de smart contracts de propósito geral, tornou implementações de tokens digitais uma atividade quase trivial.

Comumente, tokens tem uma natureza tal que eles não podem ser facilmente criados ou reproduzidos. Isso possibilita que eles retenham valor e funcionalidade. Historicamente, materiais raros, como metais preciosos, foram usados para cumprir as funções que hoje atribuímos ao conceito de *dinheiro*. Previamente ao desenvolvimento da blockchain, essa propriedade era atingida digitalmente através da centralização da emissão de tokens e do controle de transações envolvendo eles. Já em implementações que utilizam plataformas descentralizadas, desenvolvidas através de smart contracts, o protocolo da camada 1 aliado ao código do contrato do token são usados para reproduzir essa característica.

Eles podem oferecer várias funcionalidades como certificados de propriedade (e.g. imóveis, arte, etc.), permissões de acesso (e.g. ingressos de shows, identificação de sócio), direitos de voto, entre outras. Propriedades importantes de serem consideradas incluem *durabilidade*, *transferibilidade*, *divisibilidade*, *oferta*, *fluxo* e *fungibilidade*. A importância de cada uma vai depender da aplicação e propósito que o token visa atender.

Durabilidade indica a capacidade do token de manter sua funcionalidade durante longos períodos de tempo (VOSHMGIR, 2020). No contexto de blockchains essa característica é dependente da infraestrutura usada pela rede de nós para manter seu funcionamento. Dependendo de como definirmos essa característica, pode-se dizer também que ela depende da habilidade da(s) plataforma(s) ligadas ao uso desse token de manter valor agregado à eles através da geração de utilidade. Uma vez que o token não tem mais valor agregado, na maioria dos casos, ele perde funcionalidade. Tokens ligados a plataformas que são capazes de gerar utilidade aos usuários durante longos período, além de serem seguras contra ataques, são mais propícias a ter uma alta durabilidade, sob essa perspectiva.

Transferibilidade indica a possibilidade do portador do token transferir a posse deste para outrem. É uma característica binária. Moedas e cerificados de propriedade em geral são exemplos de tokens transferíveis, enquanto pontos em programas de milhas ou um diploma que certifique uma titulação são exemplos de tokens não transferíveis.

Divisibilidade indica a ordem de gradeza sobre a qual é possível fracionar uma unidade do token. Um BTC, por exemplo, é divísivel até uma ordem de 10^8, i.e., a menor denominação de BTC que alguém pode transacionar é 0.00000001 BTC. Essa propriedade é particularmente importante para tokens transacionáveis que atingem um valor de mercado relativamente alto, como é o caso do BTC atualmente. A ordem 10^n de divisibilidade de um token é determinada no seu contrato (protocolo) e nada impede, em teoria, que n seja arbitrariamente grande. No entanto, o benefício marginal de acréscimos em n é decrescente e a partir de determinado valor de n esse benefício passa a ser negativo. Isso acontece porque o esforço computacional envolvido em uma operação deve ser crescente na quantidade de casas decimais dos números. Além do mais, valores muito altos de n podem levar a um nível alto de poeira[1] no sistema, o que pode não ser eficiente.

O sistema que determina a *oferta* de um token pode ser implementado de múltiplas formas. Ele pode ou não fixar previamente um valor máximo de tokens que podem entrar em circulação (conceito conhecido como *cap*). Ele pode emitir tokens em uma única ocasião, em lotes, ou programaticamente. A Bitcoin é um exemplo de sistema que funciona com um *cap* fixo ($2.1 \cdot 10^7$ BTC), e emissão programática. Várias plataformas *dapps* usaram um modelo com cap e emissão em lotes com o maior lote sendo executado

[1] Poeira nesse contexto refere-se a quantias que são tão pequenas que não viabilizam qualquer tipo de transação, pois são menores que as taxas necessárias para movê-las.

no início do projeto para arrecadar fundos para financiar o desenvolvimento, prática que ficou conhecida como ICO (Initial Coin Offering).

O *fluxo* do token caracteriza come se dá a vida-útil de uma unidade dele. Ele pode ser completamente *circular*, i.e., uma vez que é emitido ele circula no sistema eternamente. Por outro lado, ele pode ser completamente *linear*, ou seja, é emitido a um determinado portador com um propósito específico e, após cumprir esse propósito, deixa de existir, ou é "destruído". Pode também ser um meio termo entre esses dois paradigmas.[2]

Fungibilidade é uma das principais característica que que definem a natureza do token e a sua funcionalidade. Um token é considerado *fungível* se a troca de uma unidade por outra não implica em nenhum impacto sobre o portador, salvo em algumas situações extraordinárias. Moedas são exemplos claros de tokens fungíveis. Em contrapartida, um token não é considerado fungível caso cada unidade possua alguma característica fundamental que a diferencie das demais. Um ingresso a um concerto que especifique o dia, o horário, o local e o assento referentes, é um exemplo de token *não fungível*.

5.1.1 Tokens fungíveis

No dicionário podemos encontrar a seguinte definição para o adjetivo 'fungível':

"[aquilo] que pode ser substituído por outro do mesmo gênero, da mesma qualidade ou quantidade." [3]

Essa característica não deve ser totalmente inerente ao objeto, mas sim sujeita a um certo nível de subjetividade e contexto dentro do qual ele é considerado. É possível, por exemplo, uma pessoa conferir um valor sentimental a um objeto considerado fungível pela maioria.

Imagine que várias pessoas acreditem que um simples pincel, aparentemente ordinário e facilmente substituível, pertenceu a Pablo Picasso. Além disso, é mantido ainda que ele usou esse pincel para pintar uma de suas mais famosas obras. Muitas pessoas que apreciam o trabalho do artista e acreditam nessa proposição iriam atribuir um valor desproporcional ao objeto, ao mesmo tempo que descartariam pincéis aparentemente idênticos como velhos e sem utilidade. Essa situação não deveria afetar a maneira com

[2] O DAI, token da plataforma MakerDAO é um exemplo de meio termo entre essas duas caracterizações de fluxo. Ele é emitido com um propósito: alcançar estabilidade de valor contra determinado lastro com volatilidade significativa. Ele pode circular como uma moeda, e alcançar funcionalidades que criptomoedas não podem, dada sua estabilidade de valor. No entanto, ele pode ser usado para resgatar o lastro, ocasião na qual ele é queimado e deixa de existir. Veja a seção 5.2.1 para uma descrição básica de como esse sistema funciona.

[3] "fungível", in Dicionário Priberam da Língua Portuguesa [em linha], 2008-2021, <https://dicionario.priberam.org/fung%C3%ADvel> [consultado em 12-02-2022].

a qual a maioria das pessoas, incluindo todos os fãs de Picasso, tratam todos os outros objetos de características semelhantes.

O importante a ser considerado aqui é como a grande maioria das pessoas tratam objetos de determinada categoria, essa última definida tanto por características materiais quanto pela funcionalidade do objeto. Tokens são usado para representar algo externo a ele próprio—e.g. valor, direitos, qualificações, etc. Esse propósito, portanto, deve ser determinante em relação à fungibilidade do objeto para a maioria das pessoas. Se o mesmo propósito pode ser alcançado com duas unidades diferentes daquele token, essas unidades devem ser completamente substituíveis. No contexto de tokens, características cruciais para que ele seja fungível incluem,

A) diferentes unidades do token tem características idênticas, pelo menos no que diz respeito a sua funcionalidade;

B) quaisquer quantidades podem ser adicionadas formando maiores quantidades;

C) quaisquer quantidades (maiores que a unidade fundamental[4]) podem ser divididas em quantidades menores;

A principal implicação das características acima é que só a quantidade é importante para usuários do token. O exemplo mais típico para ilustrar esse conceito é o dinheiro. Moedas são tokens utilizados para representar unidades de dinheiro. Idealmente, unidades monetárias deveriam ser indistinguíveis entre si. No entanto isso não é completamente verdade na maioria dos casos.

O papel moeda emitido por bancos centrais ao redor do mundo normalmente tem características, particulares a cada cédula, que podem levar algumas a se tornar itens colecionáveis, com um valor de mercado bem superior ao seu valor nominal. O número de série é uma dessas características e pode ser usado por autoridades que para rastrear cédulas que foram usadas para facilitar atividades ilícitas. Nesses casos, essas cédulas não são mais fungíveis como todo o resto. Caso um montante de notas tenha sido usada em práticas ilegais, um agente teria justificativa de recusar recebê-las como pagamento, caso estivesse ciente do fato. O mesmo pode ser dito em relação a implementações digitais de moedas, onde transações podem ser rastreadas ainda mais facilmente, especialmente no caso de criptomoedas como BTC, onde todo o histórico de transação da moeda, desde sua emissão, pode ser acessado publicamente.

Mesmo assim é possível dizer que moedas são fungíveis, pois na maioria dos casos são tratadas dessa forma.[5]Ordinariamente, só quantidade é importante. Qualquer outra característica do token não é levada em conta na sua utilização.

[4] I.e., a menor unidade a partir da qual o token não pode ser mais divido.

[5] Outros objetos como barras de ouro, lingotes de alumínio, sacos de farinha, etc. são *ainda mais*

A implementação de tokens fungíveis através de smart contracts é fundamentalmente bem simples. A exemplo de moedas digitais e criptomoedas, tokens não são manifestados através de arquivos digitais, mas sim como entradas em um livro-razão que mapeia saldos a identificações representativas dos respectivos portadores (VOSHMGIR, 2020). Um token representa então uma unidade de saldo dentro do contrato. Assim, um contrato de token *fungível* é fundamentalmente um registro de saldos, i.e., um livro razão.

No caso mais comum, o de tokens *transferíveis*, os componentes fundamentais do contrato serão: (i) um mapeamento entre endereços e saldos, e (ii) funções que permitem transferir unidades de saldo de um endereço para outro. (i) pode ser interpretado como o estado do livro-razão[6]e (ii) como uma função de transição de estados, análoga a função 4.1 abordada na seção 4.2.

A figura 16 ilustra o uma transferência executada através de um contrato deste tipo. O contrato possui um mapeamento $s : \mathcal{E} \mapsto \mathbb{R}_+$, sendo $\mathcal{E}$ o conjunto universo de endereços. O contrato especifica ainda uma função f_t que recebe uma transferência $x \in X \equiv \{(v, i, j) | v \in \mathbb{R}_+; i, j \in \mathcal{E}; i \neq j\}$, um mapeamento s e retorna um mapeamento s', que após a transação será considerado no estado corrente do contrato. No exemplo, é ilustrada uma transação que interage com o contrato #a33 e chama a função $f_t(\cdot)$ para executar uma transferência de 5 unidades de Alice (A) para Bob (B).

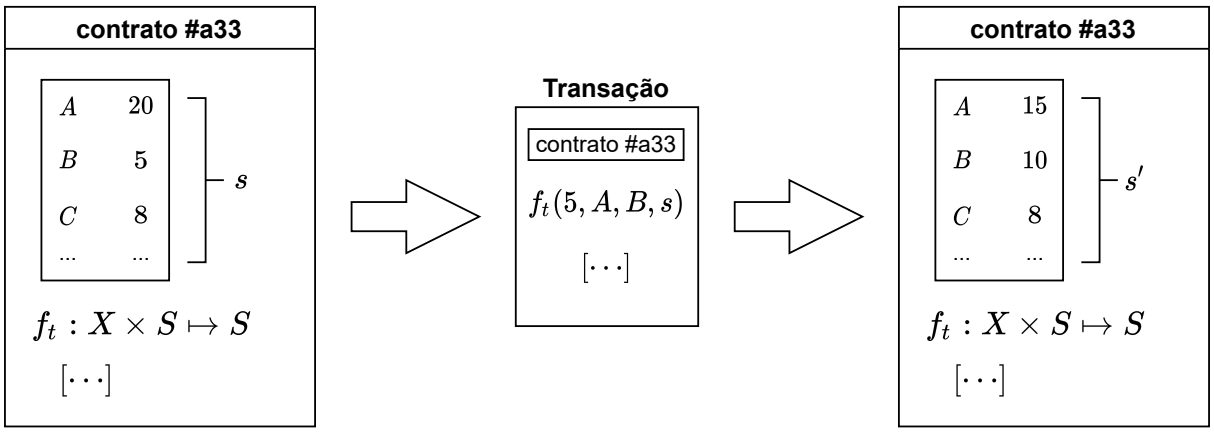


Figura 16 – Transição de estados de um contrato de tokens

Um terceiro componente possível, é uma função $f_s : \mathcal{E} \times S \mapsto \mathbb{R}_+$ que, a partir de um mapeamento $s \in S$ e um endereço $E \in \mathcal{E}$, retorna um saldo $s(E)$. Ou seja $f_s(s, E) = s(E)$. Se E é representativo do autor da transação, existem duas condições necessárias para que a transação acima (representada na figura 16) ocorra:

fungíveis, pois é mais difícil distinguir uma unidade da outra. *Tokens de privacidade*—i.e., tokens que podem ser transacionados sem que informações sobre os envolvidos sejam reveladas—também podem ser considerados mais fungíveis do que papel moeda ou BTC.

[6] O estado do *contrato* pode ser composto por outras informações que podem variar de acordo com a complexidade dele. O saldo de cada endereço dentro do contrato é uma informação de estado *fundamental* presente em todos os contratos desse tipo.

(1) $E = A$;

(2) $f_s(s, A) \geq 5$.

Caso alguma das condições acima não seja atendida a alteração não é feita e a função retorna o mapeamento s inalterado. Caso contrário, teríamos $s'(A) = s(A) - 5 = 15$ e $s'(B) = s(B) + 5 = 10$ devidamente computados no novo estado do contrato.

Essas funções básicas—entre outras seis funções—são implementadas no padrão ERC20 utilizado na plataforma Ethereum.[7] A grande maioria dos contratos de tokens *fungíveis* e *transferíveis* usam as funções básicas especificadas nesse padrão, assim como a linguagem usada nele (nomes, especificação lógica, etc.). Essa padronização da especificação básica dos contratos tem a vantagem de tornar os tokens desenvolvidos em cima dela interoperáveis com todos os protocolos/contratos que utilizam esse padrão na sua programação, que representam a grande maioria deles.

Outra função importante implementada nesse padrão possibilita um endereço a dar permissão para que outro endereço transfira um montante m do seu saldo. Um vez que essa função é chamada, o contrato verifica se o endereço que chamou a função é igual ao endereço portador dos tokens. Caso contrário, a permissão não é concedida. Isso garante que, para todo $E \in \mathcal{E}$, E só pode dar permissão referente ao próprio saldo. A função deve receber como argumento o montante m e um endereço, e.g. A, e produzir um estado do contrato que especifica que A pode transferir até m do saldo de E.

Suponha então que E dê permissão para A transferir m de seu saldo. Em seguida, A decide transferir um montante v do saldo de E para B. Para tal, A deve usar uma quarta função, de *transação indireta*. Caso $v > m$, a função não executa nenhuma alteração. Caso, $v < m$, será produzido um estado do contrato que, além das atualizações nos respectivos saldos, terá especificado que A tem permissão para transferir $m - v$ do saldo de E. Uma vez que um total de m do saldo de E tenha sido transferido por A, a permissão se esgota (o que inclui o caso em que $v = m$).

Essa duas funções possibilitam que os tokens sejam utilizados em transações intermediadas por contratos, oferecendo mais segurança aos envolvidos. Um contrato de *exchange*, por exemplo, pode executar uma transferência bilateral entre terceiros, uma vez que tenha as permissões necessárias, com apenas dois comandos: transfira v_1 de A para B; transfira v_2 B para A. Caso A pretenda usar o contrato de exchange repetidas vezes, ela pode fixar um valor m arbitrariamente alto para poupar o valor de `gas` necessário para executar repetidas permissões. A figura 17 ilustra o processo de execução dessa transação.

[7] (OPEN ZEPPELIN, a).

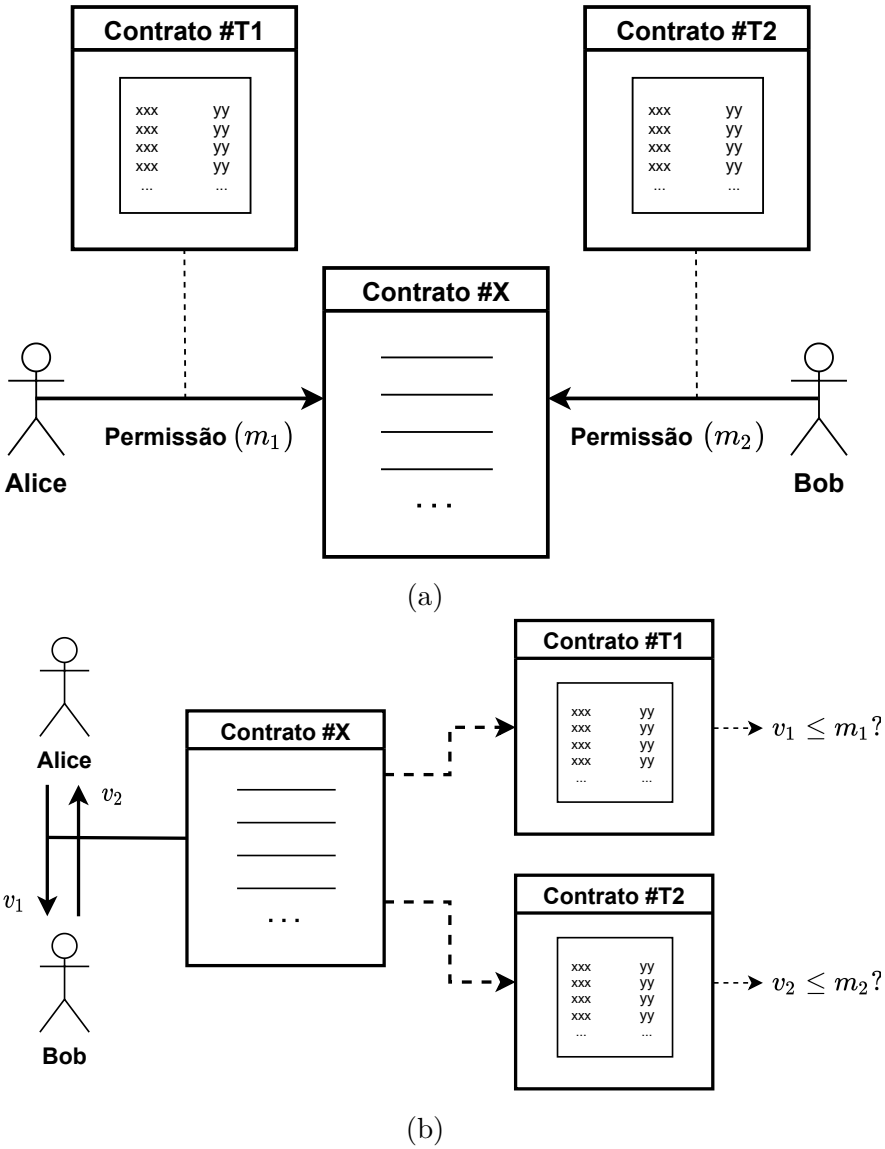


Figura 17 – Permissão e transferência bilateral

Descrição: (a) A concede permissão a #`X` para transferir m_1 de seu saldo com #`T1` e B concede permissão a #`X` para transferir m_2 de seu saldo com #`T2`; (b) #`X` se comunica com #`T1` e #`T2` que executam as transferências fazendo as devidas atualizações nos saldos de A e B.

5.1.1.1 Exemplo: fundo de investimento

Através desse funcionamento básico tokens fungíveis podem ter uma gama de aplicações e propósitos. Na maioria dos casos as diferentes funcionalidades de um token vão emergir da interação dele com os contratos da plataforma que o utiliza para determinado propósito. Essas funcionalidades, assim como algumas propriedades, emergem portanto da interação entre diferentes contratos, incluindo o ilustrado acima, que normalmente executa apenas uma função de contabilidade.

Suponha, por exemplo, que queremos criar um token que represente uma parcela em um fundo de investimento. Para tal, usaríamos um token fungível e transferível, que poderia então ser livremente transacionado no mercado secundário. O primeiro passo seria criar um contrato similar ao ilustrado acima, que podemos chamar de contrato #1, para executar a contabilidade do fundo investimento, i.e., registrar quem investiu e quanto cada um investiu no fundo. Em seguida poderíamos criar um outro contrato, o contrato #2, que seria responsável por receber os investimentos e emitir os tokens representando a parcela do total do fundo referente a eles. Além disso esse contrato seria responsável por redimir os tokens do fundo em valor referente à parcela da qual os tokens representam, assim como alocar os valores depositados nos ativos previstos no fundo.

A plataforma poderia funcionar da seguinte forma: Alice (A) enviaria uma transação ao contrato #2 transferindo o valor que ela deseja investir no fundo. O contrato #2 verifica o valor sendo transferido e calcula o montante de tokens, y, referente ao valor depositado e comunica ao contrato #1 que atualize o saldo de A, adicionando y. Para tal, o contrato #1 deveria ter especificado que o contrato #2 tem permissão para executar essa operação. Para simplificar esse processo o contrato #2 poderia ser programado para receber apenas a moeda nativa da plataforma de camada 1 sendo usada, e.g. ETH.[8]A figura 18 ilustra esse processo.

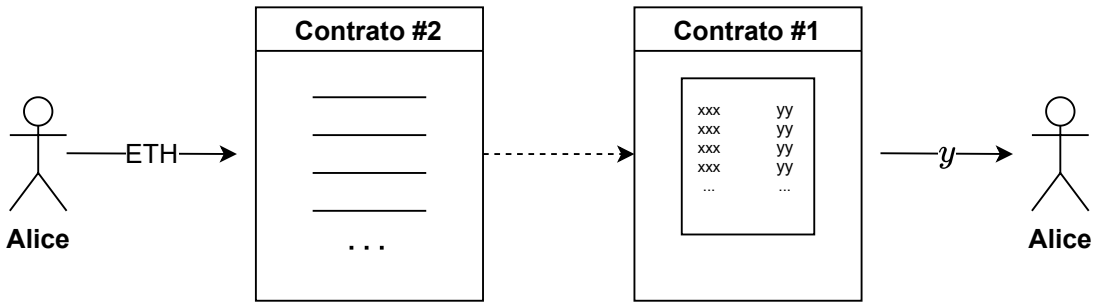


Figura 18 – Caso de uso de contrato de tokens: fundo de investimento

Descriçao: O contrato #2 recebe o valor depositado por Alice e se comunica com o contrato #1 para atualizar o saldo de Alice.

[8] Seria possível que o contrato recebesse um conjunto de tokens previamente estabelecido, mas para

O contrato #2 precisaria de algum algoritmo para determinar o valor total V, denominado em ETH, gerido por ele a qualquer momento. Suponha então que A deposita v no fundo. Seja Y_0 o total de tokens previamente emitido pelo contrato e V_0 o valor total do fundo, no momento imediatamente antes da transação. O valor de y seria calculado como segue,

$$\frac{y}{y+Y_0} = \frac{v}{v+V_0}$$
$$yv + yV_0 = yv + vY_0$$

$$y = v\frac{Y_0}{V_0} \tag{5.1}$$

Depois de emitir y em tokens para A, o contrato #2 usa o valor recebido de Alice para comprar os ativos previstos no contrato seguindo algum tipo de algoritmo.

Após ter seu saldo ou adicionado de y no contrato #1, Alice pode agora transacionar esses tokens livremente. Ela poderia, por exemplo, efetuar um pagamento a Bob usando um montante w desses tokens. Bob poderia verificar no contrato #2, através do valor corrente do fundo, V_t, o quanto ele receberia se liquidasse esse montante w. O valor líquido (l) de w seria calculado pela equação[9],

$$l = w\frac{V_t}{Y_t} \tag{5.2}$$

Se Bob, após receber o pagamento de Alice, decidisse liquidar w para receber l em ETH, o contrato #2 comunicaria ao contrato #1 que subtraísse w do saldo de Bob. Esse montante w seria efetivamente "destruído", pois não estaria sendo transferido para ninguém. Em seguida, o contrato #2 liquidaria a quantidade de ativos necessária para levantar l e transferiria-o para o endereço de Bob.[10] A figura 19 ilustra essa operação.

Vemos então um exemplo de como o simples formato de contrato apresentado (contrato #1) pode ser usado para manifestar um objeto transacionável que tem um fluxo *semi-circular*. Isso é alcançado através da interação entre os dois contratos, #1 e #2, um representativo da contabilidade e o outro da utilidade (ou propósito) dos tokens, respectivamente. Poderíamos ainda, por exemplo, adicionar um processo no qual os portadores dos tokens tem direito de votar, proporcionalmente aos montantes de tokens que possuem,

isso seria necessário envolver um processo para definir a cotação do token em relação ao numerário que o contrato usa para rastrear o valor do investimento. Esse processo poderia ser feito vendendo os tokens diretamente em alguma plataforma de exchange ou através `oráculos`.

[9] É importante ressaltar que ambas as equações, 5.1 e 5.2, não são exatamente realistas pois ignoram as taxas de transação associadas a ambas as operações. No entanto, a inclusão de quaisquer taxas nos cálculos acima não alteram fundamentalmente o conteúdo teórico que esta sendo proposto.

[10] Essa liquidação não é uma operação trivial e precisaria de um mecanismo complexo apenas para lidar com ela. Uma possibilidade, usada por alguns protocolos é executar um formato de leilão para tentar otimizar o preço obtido pelos ativos.

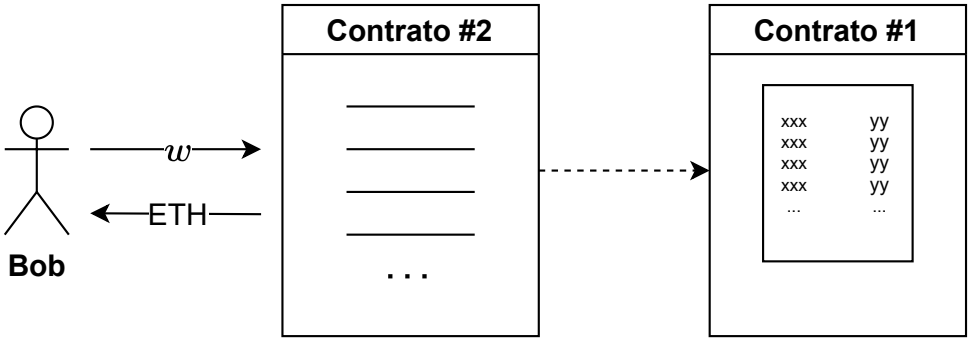


Figura 19 – Caso de uso de contrato de tokens: liquidação de investimento

Descrição: Bob envia w para o contrato #2 que comunica ao contrato #1 que subtraia w do saldo de Bob. Em seguida o contrato #2 transfere a Bob o valor levantado pela liquidação dos ativos referentes a w.

na alocação dos fundos geridos pelo contrato, ou em eventuais atualizações do protocolo. O token teria ainda, nesse caso, utilidades de gestão e governança do projeto.

Além dessas características, o token do exemplo acima teria oferta variável (com ou sem cap) dependente da demanda que o fundo atrairia, além de durabilidade diretamente relacionada ao tempo de vida do projeto. O token poderia, a princípio, ter um grau de divisibilidade arbitrário, conforme fosse conveniente para sua utilidade.

5.1.2 Tokens não fungíveis

Uma vez que definimos fungibilidade no contexto de tokens, fica fácil definirmos essa categioria de tokens que ficou conhecida pela sigla NTF (*Non Fungible Tokens*). Se por um lado fungibilidade implica substitutibilidade entre unidades (ou quantidades iguais), NFT's são caracterizadas por serem insubstituíveis entre si. Eles representam objetos que possuem propriedades únicas que não podem ser reproduzidas em outro objeto similar.

Essa classificação, é claro, será sujeita ao contexto em que ele é aplicado e não ao objeto de representação do token em si. De fato, NFT's podem ser usadas para representar objetos que seriam completamente substituíveis na maioria dos contextos. Por exemplo, para rastrear o caminho de um lote de sacos de cimento na cadeia produtiva.[11] A princípio, um consumidor não se importa em substituir um saco por outro, dada a uniformidade do produto final, mas ele pode se preocupar com a procedência do produto. Se existe algum tipo de prática que ele não aprova em algum estágio da cadeia—e.g., trabalho escravo, agressões ao meio ambiente, etc.—ele pode decidir em não contribuir para essa prática

[11] A produção de cimento é um processo razoavelmente complexo que utiliza múltiplas matérias-primas como calcário, argila, óxido de ferro e alumínio, além do coque do petróleo, gasolina e gás natural para geração de energia. Todos esses inputs da cadeia produtiva poderiam ser registrados por um serviço especializado para possibilitar que o rastreio do produto final fosse feito e devidamente mapeado nessa cadeia.

consumindo o produto final dessa cadeia. Nesse contexto existe espaço para diferenciação entre produtos quase idênticos, e a aplicação da tecnologia permite que essa diferenciação seja feita.

NFT's ganharam notoriedade considerável nos últimos dois anos pela sua aplicação nos mercados digitais de colecionáveis e video games. O conjunto potencial de casos de uso dessa ferramenta, contudo, é grande e pouco explorado até o momento. Além do exemplo dado acima, NFT's podem ser usadas para representar qualquer tipo de propriedade—imóveis, automóveis, arte, etc.—com a vantagem que a propriedade pode ser comprovada instantaneamente, assim como a transferência dela feita com uma simples transação na blockchain. Além do mais, o sistema está sempre disponível para executar essas atividades (24 horas por dia, 7 dias por semana). O custo de transação se torna quase desprezível em relação ao que é atualmente para determinados ativos[12], o que pode prover mais liquidez para eles. Outra vantagem que sistemas que usarem essa ferramenta herdam é a segurança contra manipulação de dados oferecida pelo protocolo da camada 1. Plataformas e-commerce, são outros exemplos de atividades que podem se beneficiar da aplicação dessa tecnologia.

O funcionamento básico de um contrato de NFT é bem simples, a exemplo do formato apresentado anteriormente. Agora, ao invés de fazer um mapeamento entre *endereços* e *saldos*, será feito um mapeamento entre endereços e *identificadores de tokens* (id's). Caso alguém queira verificar propriedade de determinado token, basta fornecer o id ao contrato e verificar o endereço registrado como portador do token. Os componentes fundamentais do contrato serão, então: (i) um mapeamento entre id's e endereços; (ii) uma função que verifica o portador de determinado token; (iii) uma função que faz a transferência do token.[13]

Seja então $\mathbb{T} = \{t_i | i \in \mathbb{N}\}$ o conjunto de id's, $\mathcal{E}$ e S os universos de endereços e mapeamentos, respectivamente. A função $f_v : \mathbb{T} \times S \mapsto \mathcal{E}$, entrega um endereço relacionado a um id em determinado mapeamento, ambos os últimos recebidos como argumento. Ou seja, caso Alice (A) seja a portadora do token t_1 no estado corrente do contrato, com mapeamento s, segue que $f_v(t_1, s) = s(t_1) = A$. A função $f_t : X \times S \mapsto S$ recebe uma transferência $x \in X \equiv \{(t_i, j, k) | t_i \in \mathbb{T}; j, k \in \mathcal{E}\}$, um mapeamento s, no qual t_i é registrado a j, e retorna um mapeamento s' com t_i registrado a k. Dessa vez, se $x = (t_i, j, k)$ e E é o endereço autor da transação, são condições necessárias para que $f_t(x, s) = s'$:

(i) $E = j$

(ii) $f_v(t_i, s) = E$

[12] Transferências de propriedade, por exemplo, exigem idas ao cartório para homologação de documentos, atividades com uma taxa atreladas a elas. Determinados casos demandam contratação de advogados para intermediar o processo.

[13] Caso o token seja *não-transferível*, esse último componente não existe.

Caso alguma das condições acima não seja atendida, $f_t(x, s) = s$. A figura 20 ilustra uma transferência na qual Alice (A), portadora do token t_1, transfere a posse do token para Bob (B). Como A foi executor da transação, (i) é atendida. Como A é o portador do token t_1 no mapeamento s, (ii) é atendida. Consequentemente a função entrega um novo estado do contrato com o mapeamento s', no qual $s'(t_1) = B$.

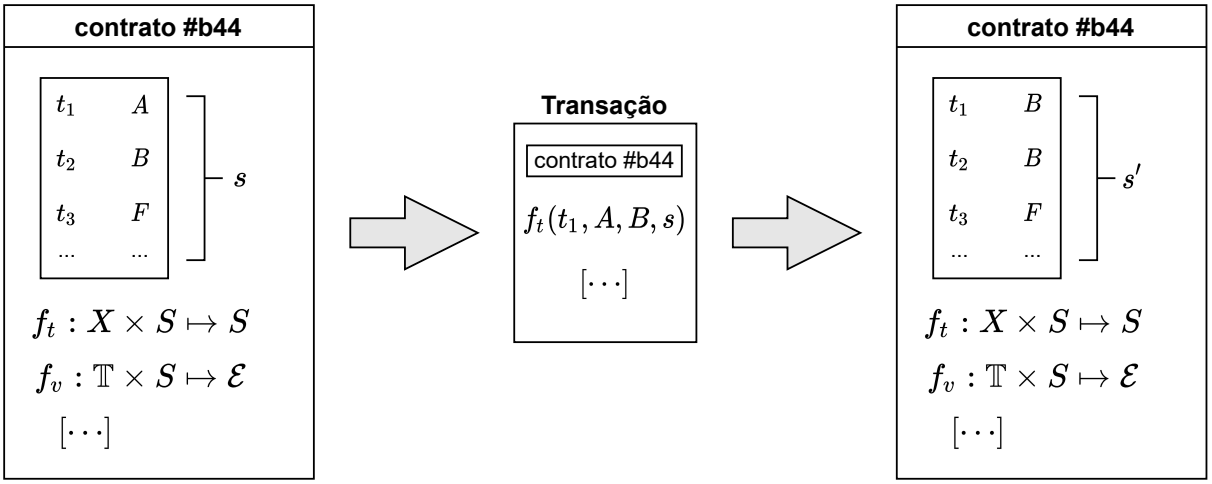


Figura 20 – Transferência de uma NFT

Descrição: Transferência de propriedade de t_1 de A para B.

Outra função fundamental não apontada ainda é a função que emite um token novo. A função de emissão, $f_e : \mathcal{E} \times S \mapsto S$, recebe um endereço, um mapeamento e produz um mapeamento no qual é adicionado o registro de um novo token ao endereço recebido. Dessa forma, suponha que, em s, temos $t_j \in \mathbb{T}$, tal que $j \geq i$ para todo i tal que $t_i \in \mathbb{T}$. Segue que, se $f_e(E, s) = s'$, a diferença entre s' e s é que $s'(t_{j+1}) = E$ e $s(t_{j+1}) = 0$, i.e., em s o token t_{j+1} não pertence a ninguém, enquanto em s' ele pertence a E.

O padrão ERC721 é o formato mais usado para NFT na Ethereum. A exemplo do formato anterior, ele possui uma função de *permissão* e uma função de *transferência indireta*.[14] Novamente, essas funções possibilitam que transferências bilaterais sejam feitas com maior nível de segurança. Permissões são dadas especificamente para cada token, de forma que a função recebe como argumento o endereço que recebe a permissão, e.g. E, e o id do token, t_i, e produz um estado em que E pode transferir t_i para um terceiro (ou para o próprio E).

[14] A função de permissão recebe o nome `approve`. A função de transferência, *direta* e *indireta* são implementadas pela `transferFrom`. Nessa última a função considera como condição necessária para executar uma transferência que, (i) o autor da transação seja o portador do token *ou* que (ii) o autor da transação possua permissão do portador do token. A função de verificação é implementada pela `ownerOf` e a função de emissão pela `_mint`. A especificação do padrão pode ser vista em (OPEN ZEPPELIN, b).

A partir desse funcionamento básico contratos de NFT podem ser usados para qualquer tipo de aplicação que exija que certificados de bens não fungíveis sejam registrados a portadores, esses últimos podendo reivindicar propriedade ou direitos atrelados a esses bens.

5.1.2.1 Exemplo: segurança de automóveis

Suponha então, que um fabricante de automóveis deseja utilizar NFT's para representar propriedade de seus produtos assim como aumentar a segurança de seus clientes. Esse fabricante poderia programar nos computadores dos carros que, para abrir as portas ou acionar o motor, fosse necessário fazer uma verificação em um contrato NFT. O sistema do carro, que deveria ter amplo acesso à internet, se comunicaria com um contrato, c#1, que faria a verificação no contrato de NFT, c#x, que o solicitante é o portador do token especificado, ou possui permissão do portador. Quando alguém quisesse solicitar acesso, ele acionaria um comando c_* em c#1 que, por sua vez, acionaria a função de verificação, f_v, em c#x. Se E é o endereço do solicitante, t_1 o token especificado, e s o mapeamento no estado corrente de c#x, c_* retornaria:

$$c_* = \begin{cases} 1 & \text{, caso } f_v(s, t_1) = E \text{ ou } E \text{ possui permissão;} \\ 0 & \text{, caso contrário.} \end{cases}$$

Caso $c_* = 1$, o sistema do carro abriria as portas, ou acionaria o motor, dependendo de qual função foi acionada em c#1. Caso contrário o acesso seria negado. Mapeamentos de permissões de acesso (abertura de portas, acionamento do motor, etc.) deveriam ser adicionados em um dos dois contratos, assim como funções que o atualizassem. Á medida que novos carros fossem fabricados, novos tokens, t_i, seriam emitidos em c#x e especificados em c#i, o contrato no qual o carro é programado para se comunicar.

A utilização desse sistema implicaria em duas importantes vantagens: primeiro, transferências de propriedade do carro seriam feitas por simples transações na blockchain, com finalização quase instantânea[15]; segundo, existiriam menos incentivos para roubar um carro pois o acesso à funcionalidade dele não estaria atrelado a um objeto físico facilmente portável (chaves de carro), mas a uma informação que, na maioria dos casos o próprio dono não tem conhecimento (chaves privadas na blockchain). O carro poderia oferecer a alternativa de uso de chaves físicas para os casos eventuais de pane no sistema, porém esse casos seriam esporádicos e essas chaves poderiam ser guardadas em um local seguro como um cofre.

[15] Conforme mencionado anteriormente na seção 4.4, alguns protocolos não oferecem finalidade completa às transações, mas sim finalidade de acordo com as hipóteses de segurança do mecanismo. Dependendo das hipóteses usadas, assim como do funcionamento mecanismo, o tempo mínimo necessário para considerar uma transação finalizada com uma margem de segurança razoável pode variar. Entretanto essa variação deve acontecer de poucos minutos até uma hora (certamente não dias, ou dias úteis).

A desvantagem clara desse sistema é a necessidade do carro ter acesso à internet a todo momento. No presente estado das coisas, esse requisito provavelmente inviabiliza esse tipo de implementação. No entanto, a total globalização do alcance da internet, com avanços na tecnologia de provisão via satélite, parece não estar tão distante.

Caso o custo transacional para executar as funções acima fosse relevante, isso seria uma outra desvantagem que poderia inviabilizar esse sistema. O aumento na segurança talvez não justificasse ter que pagar uma taxa sempre que quisesse usar o seu próprio carro. No entanto, alguns tipos de verificação de informação na rede, como é feita para ligar ou abrir o carro no exemplo acima, não incorre em cobranças de taxas de transações (i.e., contabilização de `gas`) em algumas plataformas, e.g. a Ethereum.

Uma outra *vantagem* seria uma redução no custo transacional quando o carro é usado como colateral para algum tipo de acordo, como um contrato de crédito por exemplo. O contrato poderia ser programado de forma que, caso a dívida não fosse paga no prazo estabelecido, a transferência de propriedade, e consequentemente do direito de acesso, seria automaticamente executada em favor do credor.[16] Só seria necessário que, no momento do acordo, o dono do carro desse autorização de transferência ao contrato de crédito, que apenas executaria a transferência na situação prevista no acordo. Consequentemente, serviços de confisco do veículo não seriam necessários. Recursos custosos à justiça e outros métodos de adjudicação também seriam evitados em vários casos.

Esse exemplo mostra como, a partir da simples implementação de contrato ilustrada acima, aliada a estrutura provida pela camada 1 e a tecnologia embutida no veículo, pode-se criar um objeto que serve os propósitos de *registro de propriedade* e *controle de acesso*. A durabilidade do token seria inteiramente dependente da vida útil do carro. O token seria transferível para possibilitar compra e venda do veículo. Seria ainda não divisível, como é o caso para todos os tipos de NFT. O fluxo seria circular: o token não precisaria ser destruído, pois sua única funcionalidade seria permitir as atividades atreladas ao carro. Uma vez que o carro parasse de funcionar o token perderia qualquer tipo de relevância.

5.2 Ativos sintéticos

Através da implementação de tokens, na maior parte fungíveis, várias aplicações foram criadas nos últimos anos implementando instrumentos e serviços financeiros diversos, formando o ecossistema conhecido por DeFi. Um caso de uso comum em DeFi é usar tokens para representar ativos reais externos à blockchain. Essa implementação possibilita agen-

[16] Esse exemplo do carro no contexto de smart contracts é utilizado por (SZABO, 1997) para ilustrar essa possibilidade.

tes a serem expostos a alguns ativos tradicionais enquanto aproveitam as vantagens de transacionar usando a tecnologia provida pelos protocolos das diferentes plataformas descentralizadas.

O formato mais trivial de se reproduzir essa solução é através do que podemos chamar de tokens de *lastro direto*. Nesse formato, para cada unidade de token emitido na blockchain deve existir uma unidade do ativo correspondente, disponível para ser redimida a qualquer momento. A forma mais segura de prover esse formato de lastro é ter uma entidade centralizada responsável, tanto pela emissão do token, quanto pela custódia dos ativos correspondentes externamente à blockchain.

Os exemplos mais notáveis deste tipo de aplicação são usados para reproduzir tokens lastreados no dólar americano (USD). Tokens lastreados em moedas soberanas, mais notadamente o USD, recebem o nome de *stablecoins* (moedas estáveis) pelo comportamento relativamente estável de seu preço em comparação com a maioria das criptomoedas. As implementações mais usadas incluem as stablecoins da Tether (USDT), Circle (USDC) e Binance (BUSD). Além do dólar, esse formato pode ser usado para lastrear tokens em qualquer tipo de ativo, incluindo commodities como ouro, prata, urânio, etc.

Conceitualmente, essa prática não passa de uma "tokenização" de ativos reais, usada para aliar os benefícios de se expor a esses ativos de alta demanda com as vantagens de utilizar a tecnologia das plataformas descentralizadas para transacioná-los. No entanto, ela introduz a clara desvantagem da centralização do processo (emissão/custódia), tornando-o inteiramente dependente de uma única entidade centralizada. Além de introduzir um ponto central de falha, existe ainda a necessidade de monitorar a entidade central para garantir que ela respeite o acordo—i.e., manter a relação entre a quantidade de ativos em custódia e tokens em 1:1.[17]

Existe, porém, um formato alternativo de oferecer, em parte, a solução proposta acima sem introduzir centralização ao processo. Ele é introduzido pelos *ativos sintéticos*. Ativos sintéticos utilizam o conceito de *lastro indireto*, ou lastro sintético (*soft peg*). Ao contrário do formato com lastro direto (*hard peg*), um token que utiliza lastro indireto não pode ser redimido no ativo correspondente pois a custódia deste último não esta sendo feita por ninguém. O que estes tokens procuram fazer é rastrear o preço de um ativo sem oferecer exposição real a ele, ou seja, não são feitas a aquisição e custódia do ativo e não são incorridos os custos correspondentes a essas atividades. Esse processo visa permitir que agentes sejam expostos à valorização de ativos sem precisar adquiri-los. Esses tokens são comumente referidos pelo termo *derivativos*[18], pois derivam seu valor de um ativo

[17] A Tether, por exemplo, tem um péssimo histórico em relação à cumprir esse acordo. Depois de anos falhando em prover um processo de auditoria aberto e imparcial de seus fundos, eles finalmente, no início de 2021 após o envolvimento da SEC (Comissão de Valores Mobiliários dos EUA), abriram suas contas para uma auditoria que acabou produzindo um relatório que mostrava que eles tinham menos de 3% de seus fundos denominados em USD.

[18] Um tipo de instrumento financeiro tradicional com um mercado global atual de, literalmente, 1 qua-

subjacente, mas não representam posse dele.

Á exemplo de tokens com lastro direto, ativos sintéticos podem ser usados para reproduzir stablecoins, commodities, ou ativos diversos incluindo ações de empresas. Vamos ver, então, de forma simplificada, como plataformas implementam esse tipo de aplicação.

5.2.1 O modelo de Posição Colaterizada

Como já foi mencionado, a inovação introduzida pelos protocolos de ativos sintéticos é o conceito de lastro indireto. O objetivo dessa prática é reproduzir o efeito de valorização de um ativo externo à blockchain usando um ativo interno à blockchain como lastro. O primeiro modelo que será mostrado usa um conceito que recebeu o nome de *posição de dívida colateralizada* (*Colateralized Debt Position*, ou CDP). Conforme sugerido pelo nome, nesse modelo agentes abrem posições de dívida com o protocolo depositando criptoativos como colateral. Podem ser utilizadas, em alguns casos, como plataformas de empréstimo, de alavancagem ou de posições vendidas.

Os protocolos dessas plataformas são tipicamente compostos por múltiplos contratos, cada um responsável por desempenhar diferentes funções cruciais ao seu funcionamento. Na exposição abaixo (assim como nas subsequentes), contudo, não será dado foco à arquitetura do protocolo, e sim à sua proposta e funcionamento básico. Portanto, o protocolo do modelo abaixo (assim como os subsequentes) será largamente tratado como uma unidade. Primeiro será apresentado o nível mais básico de funcionamento do modelo. Em seguida, serão introduzidas algumas complexidades passo a passo.

5.2.1.1 Criação/Destruição de sintéticos.

Um montante s de um determinado ativo sintético S pode ser criado através da abertura de uma CDP. Para isso, uma transação deve ser enviada a um contrato do protocolo, na qual uma transferência de um montante c de um criptoativo colateral C é feita, acionando a devida função de abertura da posição. Suponha que A é o ativo que S visa reproduzir. Uma vez que a CDP é aberta, o agente ganha o direito de solicitar a emissão de s, tal que,

$$s \leq \frac{1}{\lambda} c P_C \tag{5.3}$$

onde $\lambda > 1$ é o fator de colateralização do ativo C para a emissão do sintético $S(A)$ e P_C é o preço de C em relação a A no momento da abertura da CDP, alimentado ao processo através de um oráculo[19].

drilhão de USD.

[19] Ver seção 2.2.9.

Por exemplo: suponha que, no momento da abertura da CDP, 1 ETH compre aproximadamente 4000 USD; suponha ainda que, para o par ETH/USD, tem-se $\lambda = 2$. Ao abrir uma CDP com 1 ETH, sob $P_C = 4000$, ganha-se o direito de emitir $s \leq 2000$ S(USD). Uma outra forma de se enxergar essa dinâmica é: para emitir 2000 S(USD), é preciso transferir, no mínimo, 1 ETH para a abertura da CDP, sob o preço de 4000 USD. Assim, a CDP deve sempre ser criada de forma *sobre*-colateralizada, pois tem-se $\lambda > 1$. No exemplo acima, cada 1 S(USD) deve ser colateralizado com o equivalente a 2 USD denominado em ETH.

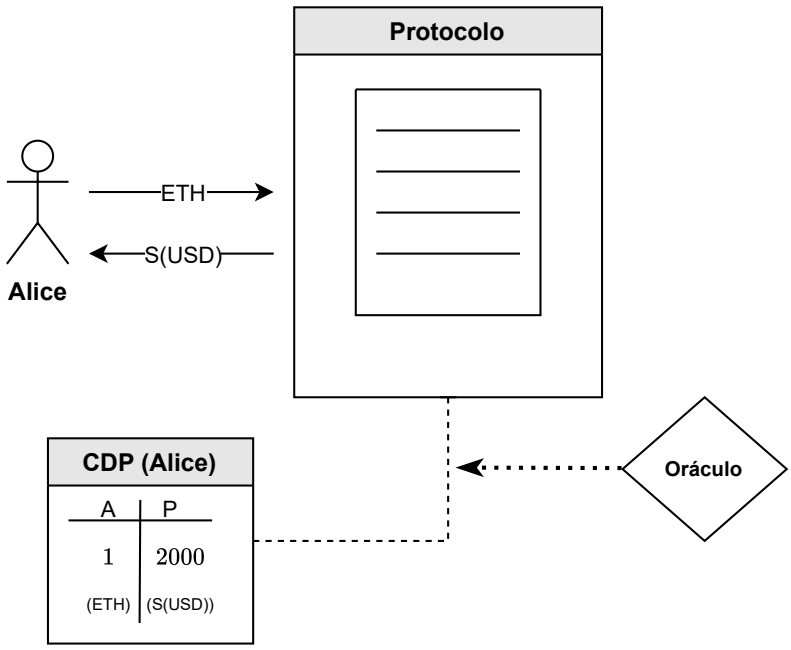


Figura 21 – Criação de uma CDP

Descrição: Alice cria uma CDP, depositando 1 ETH e emitindo 2000 S(USD).

Esse processo pode ser visto, então, como a criação de uma dívida, sendo s o valor nominal da dívida, e $cP_C(1 - 1/\lambda)$ uma margem de segurança para proteger o credor (plataforma) de oscilações no valor usado para colateralizar a dívida. Uma vez aberta a CDP e emitido o montante s, o agente que executou o processo recebe a custódia de s e é livre para transacioná-lo no mercado. Para fechar a CDP basta o agente depositar um montante s de sintético no contrato numa transação solicitando o fechamento da posição. Uma vez executada essa operação o montante s é destruído pelo protocolo e o montante c é transferido de volta ao dono da CDP fechada (Figura 22).

5.2.1.2 Preço de S(A).

Conforme CDP's são abertas e quantias de $S(A)$ são emitidas e transacionadas no mercado secundário, o seu preço pode flutuar. O ativo sintético é garantido pelo colateral que fica depositado no contrato do protocolo. Esse valor representa um piso sobre o valor do

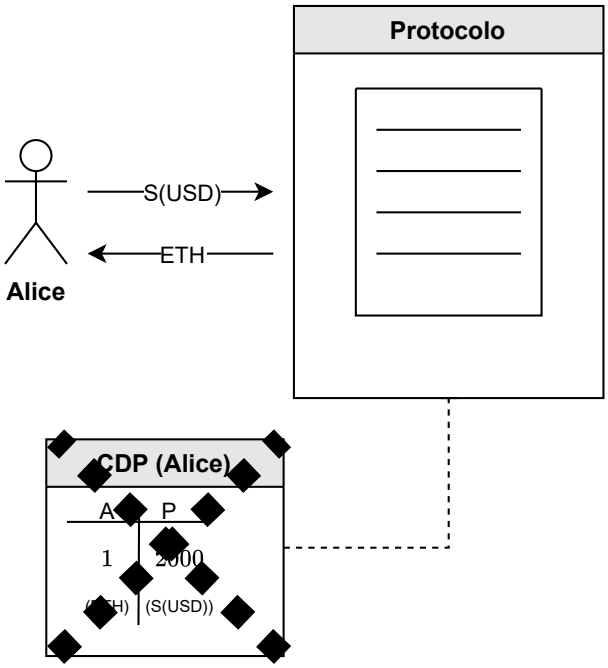


Figura 22 – Fechamento de uma CDP

Descrição: Para fechar a CDP, Alice deposita 2000 S(USD) e recebe 1 ETH de volta.

sintético, pois sempre haverão agentes interessados em resgatar o colateral caso o ativo sintético se desvalorize. Em contrapartida, esse mesmo valor pode ser visto como um teto à valorização do sintético, pois sobre valorizações deste último em relação ao primeiro abrem oportunidades claras de arbitragem.

Suponha que, dado uma diminuição na demanda por $S(A)$, ele comece a ser transacionado a um valor inferior a A no mercado secundário. Nesse caso agentes são incentivados a comprar $S(A)$ barato no mercado e fechar suas posições. A compra dos sintéticos no mercado, assim como o fechamento das posições—que contrai a oferta de $S(A)$—ambos fazem pressão para valorizar $S(A)$.

Se, por outro lado, dado um aumento na demanda por $S(A)$, ele esteja sendo transacionado a um valor superior a A, agentes são incentivados a abrir uma posição e vender $S(A)$ caro no mercado. A abertura de posições—expandindo a oferta—e a venda subsequente ambos fazem pressão para desvalorizar $S(A)$.

Vamos usar o exemplo anterior para ilustrar essa dinâmica. Temos 1 ETH $\simeq$ 4000 USD, $\lambda = 2$. Suponha que Alice tem uma posição com 1 ETH de colateral e 2000 S(USD) de dívida em aberto. Em $t = 0$, Alice percebe que S(USD) está sendo transacionado por .90 USD. Suponha ainda que, previamente a $t = 0$, Alice tenha vendido seus 2000 S(USD), emitidos na CDP, pelo equivalente a 2000 USD. Em $t = 0$, Alice pode comprar 2000 S(USD) pelo equivalente a 1800 USD e fechar sua posição, resgatando 1 ETH, obtendo lucro equivalente a 200 USD.

Se, por outro lado, Alice ainda estivesse exposta aos 2000 S(USD) em $t = 0$, ela poderia executar uma estratégia alternativa. Se Alice espera que agentes que estejam em uma posição similar a apontada acima executem a estratégia anterior, ela espera que existirá um movimento de valorização do S(USD) resultante tanto do aumento sobre a demanda (agentes comprando S(USD) para fechar suas posições) quanto da contração sobre a oferta do sintético (agente fechando suas posições). Em $t = 0$ Alice fecha sua posição e compra 4444.44 S(USD) por 1 ETH. Em $t = 1$, se o preço de S(USD) voltar a 1 USD, Alice pode vender seus sintéticos lucrando o equivalente a 444.44 USD com essa operação. Em qualquer momento anterior a $t = 1$ Alice poderia vender seus sintéticos uma vez que determinado nível de valorização se concretizasse obtendo um lucro menor.

Ambas as estratégias tem um efeito inicial similar sobre o preço do sintético, fazendo pressão para valorizá-lo. Se a posição de Alice em $t = 0$ fosse intermediária às duas ilustradas acima, i.e., se ela tivesse uma quantia $0 < x < 2000$ de S(USD) em seu portfólio, ela poderia usar ambas as estratégias caso acreditasse que o preço do sintético iria se apreciar.

Se, por outro lado, em $t = 0$ o preço de S(USD) está acima de 1 USD, Alice mantém sua posição caso espere que o preço retorne a um nível inferior ao corrente, já que a sua dívida está denominada em S(USD) e não USD. Caso Alice tenha custódia de S(USD) em $t = 0$, ela pode vender a quantia emitida ao preço caro do mercado, na expectativa que o preço irá se reduzir em um segundo momento permitindo que ela lucre com a diferença. Enquanto o preço do S(USD) estiver superior ao USD existe espaço para arbitragem de agentes que esperem que essa estratégia será seguida pelo mercado.

Normalizando o preço do ativo A (P_A=1), seja P_C o preço relativo do cripto ativo C e $P_{S(A)}$ o preço corrente do sintético $S(A)$ sendo transacionado no mercado. Suponha que, em $t = 0$, Alice possui uma posição com c em C colateralizado; s em dívida aberta denominada em $S(A)$; e possui em sua custódia x, denominado em $S(A)$, e o equivalente a $y = s - x$, denominado em outros cripto ativos estáveis (chamemos de Y, com $P_Y = 1$). Suponha que, em $t = 1$, Alice percebe que $S(A)$ está sendo negociado com um fator de variação α em relação ao preço de A ($P_{S(A)} = \alpha P_A$). Primeiramente, suponha que $\alpha < 1$ —i.e., $S(A)$ sofreu uma desvalorização. Eis a estratégia que Alice deve seguir para lucrar com a desvalorização[20]:

(1) usa αy em Y para comprar $(s - x)$ em $S(A)$: nesse momento Alice tem custódia de s denominado em $S(A)$ e $(1 - \alpha)y$ denominado em Y;

(2) usa s para fechar a posição: nesse momento Alice tem custódia de um equivalente a $cP_C + (1 - \alpha)(s - x)$, podendo encerrar a operação com um lucro $l = (1 - \alpha)(s - x)$;

[20] Com fins de simplificação, está sendo considerado que não existem taxas associadas às transações.

(3) caso Alice acredite que $S(A)$ irá apreciar, ela usa c em C e l em Y para comprar $x' = \frac{cP_C+l}{\alpha}$ em $S(A)$;

(4) suponha que, em $t = 2$, $P_{S(A)} = \beta > \alpha$: Alice vende x' por $(cP_C + l)\frac{\beta}{\alpha}$ fechando com um lucro $l^* = cP_C(\frac{\beta}{\alpha} - 1) + l(\frac{\beta}{\alpha} + 1)$;

(5) caso $\beta > 1$, inicia a estratégia abaixo.

Alternativamente, suponha que $\alpha > 1$. Nesse caso, se Bob, que não possui posição aberta, acredita que existirá um movimento do mercado para arbitrar em cima da variação de preço, ele recorre a seguinte estratégia:

(1) deposita c' e emite s' em $S(A)$;[21]

(2) vende s' arrecadando o equivalente a $\alpha s'$ em cripto ativos estáveis;

(3) se em, $t = 2$, tivermos $P_{S(A)} = \beta < \alpha$: usa $\beta s'$ da quantia acima para comprar s' em $S(A)$;

(4) fecha a posição, lucrando o equivalente a $(\alpha - \beta)s'$;

Caso esperasse que essa última estratégia fosse seguida por outros, como Bob, Alice poderia executar uma estratégia similar: vendendo x pelo equivalente a αx em $t = 1$; e, em $t = 2$, comprando sintéticos para fechar sua posição pelo equivalente a βs, fechando com um lucro equivalente a $x(\alpha - 1) + s(1 - \beta)$.[22]

Vamos usar $\mathcal{S}_A$ para identificar as estratégias ilustradas acima. Se uma quantidade relevante de agentes seguirem $\mathcal{S}_A$, o preço de $S(A)$ deve convergir ao preço de A, pelo menos no médio a longo prazo (isso será abordado mais adiante). No entanto, é necessário ressaltar que duas hipóteses estão implícitas na exposição acima:

Hipótese 3 *Existe ampla liquidez de $S(A)$ no mercado secundário.*

Hipótese 4 *Agentes acreditam que o preço de $S(A)$ deve se manter em P_A.*

A Hipótese 3 é bem razoável e não causa grandes problemas à teoria objeto dessa análise. Já a Hipótese 4 merece ser analisada com um pouco mais de cuidado. O leitor mais atento percebeu que na análise das estratégias acima, partimos de um cenário (em $t = 0$) onde $P_{S(A)} = P_A$. Ou seja, antes de existir a variação no valor de $S(A)$, ele era transacionado ao mesmo nível de preços que A. Isso não é um dado, visto que $S(A)$

[21] Lembrando que aqui o preço usado pelo oráculo para abrir a CDP será 1, i.e., o preço de A.

[22] A condição para que essa operação seja lucrativa é $\beta < \frac{x(\alpha-1)+1}{s}$.

é apenas um instrumento com objetivo de representar A que não tem qualquer vínculo direto com o ativo real A.

Como, então, a Hipótese 4 pode ser justificada? Para termos $P_{S(A)} = P_A$ devem existir agentes dispostos a pagar esse preço pelo sintético no mercado secundário. Para resolver essa questão precisamos expandir o nosso modelo.

É fácil notar como, por enquanto, consideramos apenas variações em $P_{S(A)}$, mantendo os preços de C e A constantes. Pode-se dizer que essa análise cobre o curtíssimo prazo: uma frequência maior que a frequência de variações em P_C e P_A. Operações similares às contidas em $\mathcal{S}_A$ são tipicamente realizadas de forma automatizadas por algoritmos executados computacionalmente através de *bots*, analogamente à operações de *high-frequency trading* em mercados de ativos convencionais. A seguir vamos, então, expandir o modelo para acomodar um cenário onde há possibilidade de variações nos preços de C e A.

5.2.1.3 Estratégia de posição vendida.

Considere, por enquanto, a hipótese 4 como dada. Considere um cenário inicial onde todas CDP's estão colateralizadas de acordo com a condição 5.3. Primeiro, se P_C aumentar em valor o protocolo se mantém sobrecolateralizado, com uma margem de segurança ainda maior do que a inicial. A dívida de agentes com posições abertas ficou menor em relação ao seu colateral. Eles poderiam resgatar parte de seu colateral ou emitir mais dívida, desde que mantivessem $cP_C/s \geq \lambda$.

Analisemos o cenário onde ocorre uma desvalorização de A no mercado. Se, em $t = 1$, A sofre uma desvalorização de α (i.e., $P'_A = (1-\alpha)P_A$, tal que $0 < \alpha < 1$) e em $t = 0$ o grau de colateralização da CDP de Alice era $\lambda = cP_C/s$, em $t = 1$ o novo grau de colateralização será

$$\lambda' = \frac{c}{s} \cdot \frac{P_C}{(1-\alpha)} > \lambda.$$

Alice poderia resgatar $\alpha \cdot c$ de colateral de forma a rebalancear a colateralização de volta para λ. Ela possui o mesmo valor nominal de dívida, porém com uma exposição menor, pois tem menos colateral depositado. Suponha que Alice havia vendido seus sintéticos em $t = 0$ pelo equivalente a s. Em $t = 1$ ela poderia fechar sua posição *pelo equivalente a* $(1-\alpha)s$, lucrando o equivalente a $\alpha \cdot s$ com a operação.

Fica claro que agentes dispostos a abrir um CDP, devem esperar uma desvalorização do sintético, o que implica, caso considerem a hipótese 4, uma desvalorização do ativo A (ou uma oportunidade de arbitragem conforme mostrado em $\mathcal{S}_A$). Trata-se de uma *estratégia de posição vendida*. Para que ela funcione, no entanto, é necessário que exista agentes dispostos a comprar os sintéticos pelo preço de A em $t = 0$.

5.2.1.4 Processo de liquidação

Suponha agora que P_C diminuiu devido a uma desvalorização de C no mercado. Existe o risco de posições se tornarem insolventes. Para lidar com essa possibilidade primeiro introduzimos um parâmetro de colateralização mínima μ. Se P_C cair suficientemente de forma que tenhamos, para uma determinada CDP,

$$P_C < \frac{\mu s}{c} \tag{5.4}$$

pode ser aberto um *processo de liquidação* dessa CDP com objetivo de usar c para cobrir a dívida pendente. Devemos ter $1 < \mu \leq \lambda$, de forma que $\lambda - \mu$ funciona como uma margem para evitar processos de liquidação (caso $\mu = \lambda$ essa margem não existe).

Uma vez que a condição 5.4 seja atendida, qualquer agente que perceba que a posição está liquidável pode iniciar o processo de liquidação da CDP enviando uma transação que acione a liquidação dela no protocolo. Após enviada tal transação, o protocolo verifica, através do preço corrente alimentado pelo oráculo, se a condição foi atendida. Em caso afirmativo, o protocolo inicia um algoritmo que executa um **leilão de colateral** que tem como objetivos (em ordem de prioridade): (i) levantar um valor fixo para cobrir a dívida da CDP (s) e (ii) minimizar a fração de c vendida para alcançar (i). Uma vez que o leilão foi bem sucedido, o protocolo destrói s, efetivamente saldando a dívida, e transfere o restante do colateral não vendido para o dono da posição.

A possibilidade de liquidação implica que sempre haverá demanda pelos sintéticos para fechar as posições, seja por parte dos donos das CDP's que querem evitar ser liquidados, ou por parte do protocolo no processo de liquidação. Isso significa que agentes que esperam que A se aprecie no futuro próximo, devem estar dispostos a comprar os sintéticos pelo preço de A no presente, pois poderão vendê-los a um prêmio quando a apreciação se concretizar. Trata-se de uma *estratégia de posição comprada*. Se existe oportunidade de lucro com essa estratégia, devem existir agentes dispostos a pagar o preço P_A pelo sintético, dando a base que precisávamos para justificar a hipótese 4.

Essencialmente, a plataforma forma um mercado de agentes com posições vendidas e agentes com posições compradas, sendo um grupo a contraparte do outro. **A plataforma provê a estrutura para que agentes dispostos a assumir uma posição vendida emitam os sintéticos e transacionem com agentes dispostos a assumir uma posição comprada no mercado secundário.**

Existe, no entanto, uma vulnerabilidade que ainda precisa ser tratada. Se o protocolo não alcançar o objetivo (i) no leilão de colateral, parte da dívida ficará descoberta. Isso pode acontecer, por exemplo, em casos de movimentações bruscas em P_C, que levem posições a um nível de *sub*-colaterização antes que os processos de liquidação sejam realizados.

Problema 6 *Movimentações bruscas no nível de preço relativo do colateral (P_C) podem levar posições a se tornarem insolventes.*

Esse problema é de extrema relevância considerando o histórico de volatilidade das principais criptomoedas. Vamos introduzir então mais ferramentas para lidar com essa vulnerabilidade.

5.2.1.5 O Token de Governança.

Como já deve ter ficado claro, os parâmetros λ e μ são calibráveis e podem demandar recalibrações devido a movimentação de variáveis macro relevantes ao funcionamento do protocolo. Um método comumente utilizado para possibilitar a implementação de um processo decisório, tanto dessas quanto de outras potenciais mudanças no protocolo, sem introduzir centralização ao processo, é a criação de um *token de governança* que permita que agentes votem nas decisões finais proporcionalmente à quantia de tokens que possuem.[23] O propósito fundamental dessa ferramenta é possiblitar que decisões importantes ao funcionamento do protocolo sejam tomadas de forma descentralizada alinhando os interesses dos responsáveis por essas decisões com o objetivo de obter um funcionamento pleno da plataforma.

Esse token (chamemos ele de G) deve ser fungível e transferível. Primeiro, para permitir que essa comunidade de governança seja maleável e aberta. Segundo, para permitir que os votos dos membros da comunidade sejam proporcionais ao capital que cada um alocou ao processo. Essas características permitem ainda que esses agentes possam abandonar suas posições de membros da governança, transacionando seus tokens para outros que estejam mais interessados em participar do processo ou aumentar sua influência nele. A ideia central do token de governança é atrair agentes que estejam interessados em utilizar a plataforma, assim como fomentar seu funcionamento, a investir nesses tokens.

Outra decisão importante tomada nesse processo é a dos conjuntos de cripto ativos aceitos como colateral, $\mathbb{C} \equiv \{C, C', C'', ...\}$, e de ativos que terão sintéticos reproduzidos no protocolo, $\mathbb{A} \equiv \{A, A', A'', ...\}$.

Idealmente, os tokens podem ser acompanhados de determinadas recompensas para criar um incentivo ainda mais forte para agentes fazerem provisões necessárias para alcançar uma alta eficiência da plataforma. Penalizações atreladas à problemas de performance do protocolo podem acompanhar (essas penalizações podem ser manifestadas na forma de uma diminuição das recompensas, ou de formas mais diretas).

[23] Esse é um exemplo de sistema de governança (consenso tipo 1) para plataformas Dapps. Ver seção 4.4.

Qualquer tipo de coordenação entre a comunidade envolvida neste processo que precise ser feita externamente ao protocolo pode ser efetuada por canais de comunicação alternativos (fóruns, blogs, mídias sociais, etc.) que, idealmente, devem ser acessíveis por qualquer um.

5.2.1.6 Reserva do protocolo.

Para implementar mecanismos de incentivo para a governança da plataforma é conveniente existir algum tipo de arrecadação de fundos para compor uma reserva que fique sob controle do protocolo. Essa reserva pode ainda servir o propósito de cobrir eventuais posições insolventes, oferecendo uma solução inicial ao problema 6.

A forma mais direta de realizar essa arrecadação de fundos é atrelar uma taxa de serviço à utilização da plataforma. Essa taxa pode ser vista como uma taxa de juros sobre a dívida contraída com a abertura de uma posição. No nosso modelo essa cobrança será feita na forma de uma taxa simples em cima do valor nominal da dívida que está sendo saldado. O valor desse parâmetro deve ser estabelecido/alterado pelo processo governança apontado acima.

Seja τ o valor da taxa: para fechar uma posição agora o agente deve depositar $(1+\tau)s$ para reaver o colateral c. O valor acima do valor nominal da dívida, τs, deve ser necessariamente adquirido no mercado secundário. Do valor depositado, s será destruído e τs será transferido para a reserva do protocolo. Essa taxa afetará diretamente a lucratividade da estratégia $\mathcal{S}_A$, não podendo ser fixado em um valor muito alto. Além disso, uma taxa muito alta dissuadirá usuários em geral de usar a plataforma com estratégias diversas. Membros da governança devem ter incentivos de fixar uma taxa que maximize o uso da plataforma e proveja robustez aos mecanismos relevantes ao seu funcionamento.

É possível exigir que a taxa seja paga na mesma denominação da dívida ou, alternativamente, em G, o token de governança. Ao exigir que a taxa seja paga em G, o protocolo cria uma pressão positiva sobre a demanda de G, aumentando seu valor de mercado. Por outro lado, é preciso envolver um oráculo em mais um processo, i.e, indicar a cotação de G em relação a A para que seja calculado o valor x, em G, tal que $x \simeq \tau s$, em $S(A)$. Para evitar isso vamos manter a cobrança na mesma denominação da dívida. Usando o mesmo exemplo usado anteriormente, caso $\tau = .03$, seria necessário depositar 2060 S(USD) para fechar a dívida de 2000 S(USD) e reaver o 1 ETH usado de colateral. Após o fechamento, o protocolo destruiria 2000 S(USD), cancelando a dívida, e transferiria 60 S(USD) para a reserva de protocolo.

Adaptando o processo de liquidação, podemos estabelecer uma taxa de penalização $\sigma \geq \tau$. Uma taxa de penalização superior à taxa de serviço ajuda a incentivar usuários a se manterem vigilantes em relação às suas posições, ajudando a diminuir o risco de

sub-colateralização do protocolo. Agora o leilão de colateral tem como objetivo principal levantar um valor $(1+\sigma)s$. Uma vez que o leilão é bem sucedido, o protocolo destrói s, transfere σs para a reserva e retorna o restante de colateral não vendido, c^l, para o dono da posição. Mesmo que o dono da CDP tenha sua posição liquidada, ele ainda mantém s e c^l.

Vamos nos voltar agora para o caso em que C tem uma desvalorização tal que o leilão não consegue levantar fundos suficientes para cobrir a dívida. Suponha que s^l é o valor levantado pelo leilão. O valor $s^r = s - s^l$ é retirado das reservas e destruído junto com s^l, efetivamente anulando a dívida. Nesse caso, a taxa de penalização/serviço, não é absorvida pela reserva do protocolo.

Vamos estender novamente o exemplo que está sendo usado ao longo desta seção, i.e., com 1 ETH $\simeq$ 4000 USD usado para abir uma CDP emitindo 2000 S(USD) ($\lambda = 2$). Suponha agora que $\mu = 1.5$ e $\sigma = .13$. Alice é a dona da posição descrita acima. Suponha que Alice se descuide e falhe em perceber que o preço do ETH sofreu uma brusca desvalorização chegando a 3000 USD. Bob, em busca de oportunidades de lucro, está monitorando o protocolo em busca de posições liquidáveis. Ele percebe que, dado o valor do ETH sendo negociado no mercado, a posição de Alice poderia ser liquidada. Ele aciona então o processo de liquidação da CDP de Alice, que é iniciado pelo protocolo após ser feita a confirmação através do feedback do oráculo, iniciando então o leilão de colateral. A partir desse momento, agentes interessados em comprar ETH em desconto começam a realizar propostas na forma de preço $\times$ montante. Desde que o preço pago pelo ETH seja menor que o preço negociado no mercado, existe uma oportunidade de arbitragem para esses participantes.

Suponha que o leilão consegue levantar os 2260 S(USD)[24] vendendo .77 ETH, obtendo um preço médio efetivo de 2935.06 S(USD)/ETH. Após destruir 2000 S(USD) para cancelar a dívida e fechar a CDP de Alice, o protocolo envia 260 S(USD) para o contrato de reserva e .23 ETH para Alice. Após o fechamento da CDP, Alice ainda tem o equivalente a 2690 USD, fechando em um prejuízo equivalente a 1310 USD. Além disso, o protocolo arrecadou 260 S(USD) e o(s) ganhador(es) do leilão, cerca de 50 USD vendendo a quantia de ETH no mercado.

Alternativamente, suponha que, devido a circunstâncias extraordinárias, o preço do ETH despenque cerca 50%, chegando próximo aos 2000 USD, antes que Alice consiga fechar sua posição e antes que algúem inicie um processo de liquidação da CDP de Alice. A partir desse momento é muito provável que a CDP tenha se tornado insolvente. Suponha que, após iniciado o processo de liquidação da CDP, o leilão consiga levantar apenas 1960 S(USD) vendendo 1 ETH do colateral. Após terminado o leilão, o protocolo retira 40 S(USD) da reserva e destrói $1960 + 40 = 2000$ S(USD), efetivamente cancelando a dívida.

[24] $(1+\sigma)\times$ Valor nominal da dívida.

Alice nesse caso fechou com o equivalente a 2000 USD de prejuízo e o protocolo perdeu 40 S(USD) de suas reservas. O(s) ganhador(es) do leilão ainda pode(m) levantar cerca de 40 USD liquidando seus ganhos.

5.2.1.7 Compradores de última instância.

A criação da reserva e o uso dela para cobrir dívidas descobertas é uma boa solução ao problema 6, porém é preciso abordar o eventual caso onde a reserva do protocolo não possui fundos suficientes para cobrir a dívida assumida por ele.

Primeiro vale ressaltar que, no nosso modelo generalizado, a reserva do protocolo é composta por várias sub-reservas, cada uma em uma denominação diferente, das quais dívidas foram emitidas via CDP (e.g., $S(A)$, $S(A')$, $S(A'')$,...). Se, por exemplo, a reserva denominada em $S(A'')$ não fosse suficiente para cobrir a dívida assumida pelo protocolo em $S(A'')$ em determinado momento, o protocolo poderia recorrer a um leilão de reserva com objetivo de levantar essa diferença vendendo conteúdo das outras sub-reservas.

Mas e se o total de reserva não for suficiente para cobrir a dívida do protocolo? Nesse caso podemos recorrer aos portadores de G. O protocolo pode, através da emissão de novas unidades de G, realizar um leilão com o objetivo de levantar um valor fixo, em $S(A'')$, para cobrir a dívida, minimizando o montante de G vendido. Assim, os portadores de G acabam cobrindo a dívida do protocolo através de um imposto inflacionário. Uma vez concluído o leilão, o valor levantado é destruído para anular a dívida do protocolo.

Solução 3 *Através da emissão de G, o protocolo garante que sempre haverá liquidez para cobrir posições insolventes.*

A reserva deve ser o primeiro recurso do protocolo para cobrir essas eventuais dívidas descobertas, tornando a solução acima como um recurso de última instância, evitando que a oferta de G seja inflada. É importante que G sustente valor de mercado para que a solução 3 seja eficaz. Para dar resiliência ao preço de mercado de G o protocolo pode estabelecer recompensas aos portadores desse token, e.g.: (i) remunerar depósitos de G através da reserva, quando esta última alcançar uma cota mínima estabelecida pela governança; (ii) comprar G através de leilões usando fundos da reserva (quando acima da cota mínima), destruindo a quantia levantada e contraindo a oferta de G; (iii) remunerar depósitos de G via expansão da oferta. Cada solução terá seus próprios tradeoffs e sua eficiência dependerá das particularidades de cada plataforma. Esses processos podem ser mudados pela governança do protocolo, de acordo com sua eficácia em determinado momento.

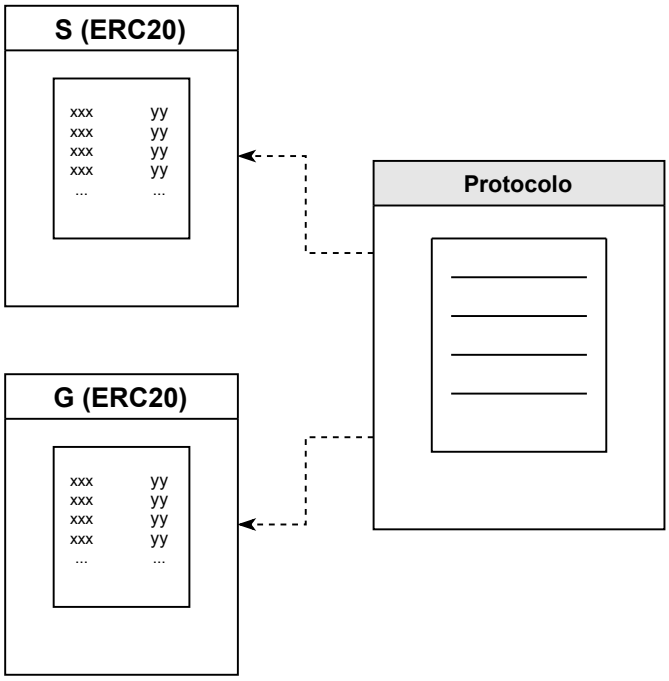


Figura 23 – Gestão da oferta dos tokens S e G

Descrição: O protocolo se comunica com contratos dos tokens G e S, com permissão para emitir e destruir unidades de cada um.

5.2.1.8 Posição vendida/comprada

Vamos revisitar os conceitos de posição vendida/comprada provendo um exemplo para ilustrá-las melhor. Suponha que Alice espera que o ativo A se depreciará no mercado em determinado período. Ao abrir uma posição, Alice está efetivamente assumindo uma dívida denominada em A, assumindo que o preço de mercado de $S(A)$ convirja ao preço de mercado de A no médio a longo prazo. Existe portanto uma oportunidade de lucrar com a desvalorização de A. Caso ela espere que o valor de C se mantenha estável (ou aprecie), ela pode abrir uma posição usando um montante C de colateral e vender a quantia de sintético emitida por uma quantia de C (ou outro ativo estável). Uma vez que a desvalorização se concretizar, o valor real da dívida de Alice diminuiu e ela pode lucrar com a diferença fechando sua posição. Se, por outro lado, Bob acha que o ativo A irá apreciar, ele compra a quantia de Alice e, caso a valorização se concretizar, vende a quantia a um prêmio para Alice fechar/rebalancear sua posição, ou para o protocolo liquidá-la, dependendo do cenário.

Vamos ilustrar usando um exemplo. Em $t = 0$, Alice abre uma posição com 1 ETH, emitindo 10000 S(BRL) ($P_C = 20000$ e $\lambda = 2$). Imediatamente Alice vende essa quantia pelo equivalente a 10000 BRL no ativo estável X. Em $t = 1$, Alice percebe que BRL sofreu uma desvalorização de 15% no mercado, enquanto o ativo X se manteve no mesmo nível. Nesse momento, Alice pode fechar sua posição usando 85% da quantia em

sua custódia, e embolsar o resto, fechando a operação com um equivalente a $1,075$ ETH. Se, por outro lado, BRL tivesse valorizado 15% em $t = 1$, Alice estaria olhando para um prejuízo corrente de cerca de .075 ETH. Bob, por outro lado, caso tivesse comprado a quantia de 10000 S(BRL) de Alice em $t = 0$, poderia vendê-la ao preço valorizado, lucrando o equivalente a 1500 BRL. No primeiro cenário Bob estaria olhando para um prejuízo corrente de 1500 BRL.

5.2.1.9 Posição n-alavancada

Uma forma de potencializar os lucros da estratégia de posição vendida é assumir uma *posição alavancada*. Similarmente, suponha que Alice acha que P_C irá subir de valor em determinado período. Alice pode abrir uma CDP com c em C, e usar o valor s em $S(A)$ para comprar mais C. Se ela emitir em dívida o valor máximo permitido pelo protocolo, após esse primeiro passo, Alice estará exposta à até $(1 + 1/\lambda)c$. Até esse momento a estratégia é similar a ilustrada acima. Caso Alice esteja realmente confiante, ela pode se alavancar ainda mais, usando a quantia adquirida em C como colateral para aumentar sua dívida e comprar mais C com a quantia de sintéticos emitida. Nesse momento, Alice estaria exposta à cerca de $(1 + 1/\lambda + 1/\lambda^2)c$. A cada passo de alavancagem que Alice dá, ela aumenta sua exposição ao ativo C por uma parcela $1/\lambda^n$, sendo n o número de vezes que ela realizou a alavancagem. Ao final de n passos de alavancagem, Alice estará exposta à um montante l_n, denominado em C, conforme indicado abaixo:

$$l_n = \frac{c(1/\lambda^{n+1} - 1)}{1/\lambda - 1}.$$

Caso P_C sofra uma variação α, a partir de seu preço inicial P_{C_0} ($P_C = (1 + \alpha)P_{C_0}$), Alice estará exposta a um lucro líquido[25],

$$l^* = [\alpha l_n - \tau(l_n - c)]\, P_{C_0},$$

se, no eventual caso de $\alpha < 0$, ele não seja suficiente para que a posição seja liquidada. Para aumentar sua proteção contra uma potencial liquidação, após executar os n passos de alavancagem, Alice pode depositar o valor c/λ^n, comprado em C no último passo de alavancagem, em sua CDP para aumentar o colateral de sua dívida e diminuir o *preço de liquidação* da sua posição, que, nesse caso, poderia ser calculado por,

$$P_L = \frac{\mu(l_n - c)}{l_n}.$$

Nesse momento o nível de colateralização de sua CDP seria $l_n/(l_n - c) > \lambda$. Outra forma de se proteger contra possíveis desvalorizações em C seria usar o montante emitido em

[25] Ignorando taxas de transação.

sintético no último passo de alavancagem para comprar um ativo sem uma correlação relevante com C.

Vamos ver como essa estratégia funcionaria usando o mesmo exemplo usado ao longo da seção: Alice tem uma CDP com 1 ETH de colateral, em $t = 0$, quando 1 ETH $\simeq$ 4000 USD. Além disso temos $\lambda = 2$ e $\mu = 1.5$. Alice está confiante que o preço do ETH irá se valorizar em relação ao USD, então ela emite 2000 em S(USD), compra .5 ETH, e deposita em sua CDP. Alice pode manter sua posição e ficar exposta à 3/2 da valorização de seu montante inicial de ETH. Como está muito confiante, Alice emite mais 1000 S(USD), compra .25 ETH e deposita na CDP, aumentando seu grau de exposição à valorização do seu capital inicial a 7/4. Nesse momento o grau de colateralização efetivo e o preço de liquidação da CDP de Alice são, respectivamente, $\lambda^e = 2.33$ e $P_L = 2571,43$ USD/ETH. Ou seja, para Alice ser liquidada o preço do ETH tem que desvalorizar cerca de 36%. Suponha que, em $t = 1$, o preço do ETH valorize 15%. Caso Alice fechar sua posição nesse momento, ela encerra a operação com um lucro equivalente a 960 USD após pagar a taxa de serviço ($\tau = .03$). Se não executasse essa estratégia, ela ganharia o equivalente a 600 USD com a valorização do ativo.[26]

Suponha, alternativamente, que em $t = 1$ o preço do ETH desvalorizou 15%. Nesse momento Alice tem um prejuízo líquido corrente equivalente a 1140 USD (pós-taxa). Suponha que Alice decida manter sua posição e, em $t = 2$, o preço do ETH sofra mais uma desvalorização de cerca de 25% chegando a 2571 USD/ETH. Nesse momento a posição de Alice está liquidável. No melhor dos cenários, ela consegue recuperar o equivalente a 1109.25 USD, após paga a taxa de penalização $\sigma = .13$, fechando com um prejuízo equivalente a 2890.75 USD.

5.2.1.10 MakerDAO.

Uma das primeiras implementações de plataforma a usarem uma variação de modelo CDP foi a (MAKERDAO, a)[27], que funciona na camada 1 da Ethereum. Essa plataforma usa um formato semelhante ao apresentado—com algumas diferenças—para implementar uma stablecoin que rastreia o USD, chamada DAI. Além da estrutura básica, que se assemelha à apresentada acima, são implementadas outras ferramentas que ajudam a induzir a convergência do sintético ao preço do USD. Um exemplo notável é a Dai Savings Rate (DAS), um retorno sobre os depósitos de DAI, pago através da reserva, que pode ser ajustado como resposta a variações na demanda pelo sintético. Além da DAS, a taxa de serviço da plataforma, que funciona como uma taxa de juros composta capitalizada por bloco—diferentemente da da taxa fixa usada acima—também é adaptável às movimenta-

[26] O valor do lucro líquido com a estratégia não é exatamente 1.75 vezes maior que o valor da valorização devido ao pagamento da taxa de serviço, que não é feito caso a CDP não seja aberta.

[27] Ver também (MAKERDAO, b).

ções do mercado para manter o preço alvo do sintético. Esses ajustes são decididos em um processo decisório formado por portadores do token de governança MKR. No entanto, a plataforma é limitada a apenas um tipo de sintético, a stablecoin DAI.

5.2.1.11 Mirror.

O (MIRROR PROTOCOL,) usa um modelo CDP pra implementar sintéticos de uma gama de ativos, em cima das camadas 1 da Terra e Ethereum. Até o momento em que este trabalho é escrito a plataforma oferece sintéticos de 34 ativos diferentes incluindo ações de empresas (e.g. AAPL, AMZN, GOOGL, TSLA), índices (SPX, VIX, SLV, USO)[28] e critomoedas (BTC, ETH, DOT). O token de governança da plataforma é o MIR.

Vamos ver a seguir mais dois exemplos de plataformas que reproduzem propósitos semelhantes, porém de forma mais simplificada.

5.2.2 O modelo de Compensação

Stablecoins cumprem um papel crucial no ecossistema DeFi. Dado o histórico de volatilidade das criptomoedas tradicionais, assim como dos tokens das maiores aplicações, existe um grande empecilho em usar esses cripto ativos como denominação para fixar contratos, efetuar pagamentos programados, fixar preços, etc. Cripto ativos com valor atrelado a moedas soberanas, mais notadamente o USD, foram criados para cumprir esses papéis que cripto ativos voláteis falham em cumprir de maneira eficiente. Stablecoins podem ser usadas ainda para se proteger da volatilidade sem precisar retirar fundos da blockchain através de exchanges reguladas (como a Binance, Coinbase, etc.), um processo com custos atrelados, tanto na forma de taxas como na forma de prazos. De fato, stablecoins como a USDT, da Theter, são amplamente usadas por traders que desejam liquidar suas posições, realizando seus lucros ou se protegendo de subseqüentes prejuízos. Soluções como a USDT, no entanto, são mau vistas por grande parte dos adeptos das tecnologia descentralizadas por introduzirem centralização ao seu processo, como já foi apontado.

Stablecoins descentralizadas como a DAI oferecem uma alternativa aos que não quiserem se expor à centralização das stables de lastro direto. Contudo, devido à sua proposta de funcionamento, plataformas que usam modelos CDP possuem um gargalo no que diz respeito à sua escalabilidade. Dado que precisa ser *sobre*-colateralizada a todo momento, a oferta total de DAI, assim como das demais stables baseadas num modelo CDP, possui uma cota superior proporcional a uma fração dos cripto ativos que estão

[28] Standard & Poor's 500, ProSahres VIX short-term futures, iShares Silver Trust, United States Oil Fund.

disponíveis a serem usados como colateral nessas plataformas. A oferta de colateral será sempre uma fatia relativamente pequena da economia, dado que o conjunto de cripto ativos aceitos pelos protocolos para cumprir esse papel é seleto, e essa seleção é crucial à integridade e segurança dessas plataformas. Esse limite à escalabilidade da oferta dessas stables é um grande empecilho à adesão delas como alternativa principal para suprir a demanda por cripto ativos estáveis.[29]

Como resultado, o montante transacionado diariamente de stables *centralizadas*, assim como as suas ofertas, são atualmente ordens de grandeza maiores que os de stables descentralizadas.[30]

O modelo exposto a seguir é uma alternativa ao modelo CDP que não sofre da mesma limitação deste último. Trata-se de uma plataforma camada 1, com suporte para smart contracts, que tem uma funcionalidade voltada para implementação de stablecoins embutida no protocolo principal. O fato de o próprio protocolo da camada 1 oferecer essa funcionalidade permite que o mecanismo seja consideravelmente simplificado.

O mecanismo proposto por esse modelo é baseado primariamente em dois componentes principais:

A) *o token nativo da plataforma*, L, que, além de ser o meio de pagamento de taxas associadas à plataforma, é usado como ponderador/recompensa no processo de mineração, que utiliza uma variação de mecanismo PoS;

B) *um sistema descentralizado de oráculos* responsável por fornecer a cotação das moedas que a plataforma visa reproduzir.[31]

A partir dessas duas ferramentas o mecanismo é simples: qualquer usuário pode, a qualquer momento, vender para o protocolo um montante de L pela moeda sintética de acordo com a cotação fornecida pelo oráculo. Se um usuário deseja adquirir sintéticos

[29] A MakerDAO chegou a introduzir uma mudança no protocolo, que passou a aceitar algumas stables centralizadas, como a USDT, como colateral com parâmetro $\lambda = 1$. Nesse novo módulo chamado PSM, não existe risco de liquidação e usuários podem trocar USDT por DAI com câmbio de um pra um. Essa mudança, no entanto, foi alvo de críticas por introduzir centralização ao processo, já que agora o protocolo aceita uma stable centralizada como garantia <https://makerdao.world/en/learn/governance/module-psm/>.

[30] Fonte: <https://coinmarketcap.com/>, acessado 24/3/2022.

[31] No protocolo da Terra, a principal implementação do modelo acima, o sistema de oráculos é formado pelos próprios mineradores, que votam no valor da cotação e recebem recompensas ou penalizações caso seus votos estejam dentro ou fora de 1 desvio-padrão da mediana ponderada. A escolha de usar os próprios mineradores para formar este sistema visa mitigar a possibilidade de colusão para fraudar esse processo, colocando os agentes que seriam os principais afetados por uma fraude no sistema como responsáveis pela sua integridade.

do USD (sUSD)[32], por exemplo, ele deposita um montante x de L no protocolo e recebe em troca $y = xP_D$ de sUSD, sendo P_D a cotação corrente do L em USD. Por outro lado, qualquer usuário pode a qualquer momento redimir seus dólares sintéticos em L, depositando um montante y' de sUSD e recebendo em troca $x' = y'(1/P_D)$. A figura 24 esquematiza o processo.

A cada vez que essas transações são feitas, as quantias de tokens usadas na transação são inteiramente criadas/destruídas pelo protocolo. Assim, os preços dos sintéticos podem ser mantidos via estratégia de arbitragem similar a $\mathcal{S}_A$ abordada na exposição do modelo CDP. Quando o sintético se desvaloriza em relação à moeda referente, existe oportunidade de lucro em comprar sintético pelo preço de mercado e redimi-los em L pelo preço do oráculo. Por outro lado, quando o sintético está sendo transacionado a um preço superior ao da moeda, exite oportunidade de lucro em comprar sintético no protocolo, pelo preço do oráculo, e vendê-lo no mercado pelo preço sobrevalorizado. Essas estratégias devem fazer a oferta do sintético expandir e contrair de acordo com o desvio do seu preço em relação ao preço da moeda que ele reproduz, fazendo o preço do sintético convergir ao preço alvo. A oferta do sintético se aproxima da elasticidade perfeita em relação à sua demanda no mercado, proporcionalmente ao capital aplicado a essas operações.

Não é mais preciso aqui utilizar a hipótese 4. É necessário apenas conceder que o token L terá valor de mercado. Esse é o motivo principal para implementar o modelo em uma camada 1 independente: é a forma mais segura de garantir que L reterá valor, pois é necessário para acessar qualquer tipo de aplicação no ecossistema endógeno à essa plataforma, assim como receber as recompensas associadas ao processo de mineração. Pode-se dizer que, através do mecanismo acima, a volatilidade dos sintéticos é absorvida por L (KEREIAKES; KWON; PLATIAS, 2019).

Esse token precisa, portanto, ser resiliente a ciclos econômicos e choques de mercado que podem afetar o seu valor. Essa é, provavelmente, a principal vulnerabilidade do modelo acima. Em um potencial cenário de choque negativo sobre a atividade econômica na plataforma, e uma subsequente contração na demanda por sintético, seguiria um choque positivo na oferta de L (que seriam redimidos em troca de sintéticos), o que teria um efeito negativo no seu valor, ceteris paribus. Dependendo desse cenário de queda, é plausível que uma situação de corrida para redimir sintéticos e liquidar L no mercado seguisse, formando um ciclo com feedback negativo para o valor de L e potencialmente uma espiral negativa para a economia endógena da plataforma.

Um efeito secundário e extremamente importante de um cenário como esses seria o choque sobre a renda dos mineradores. Se os mineradores recebessem em L, eles estariam

[32] Está sendo usada uma notação diferente para representar o sintético, em relação ao modelo CDP, para enfatizar que esses sintéticos são completamente independente um do outro e não precisam apresentar nenhum tipo de correlação independente à que possuem com seu ativo subjacente.

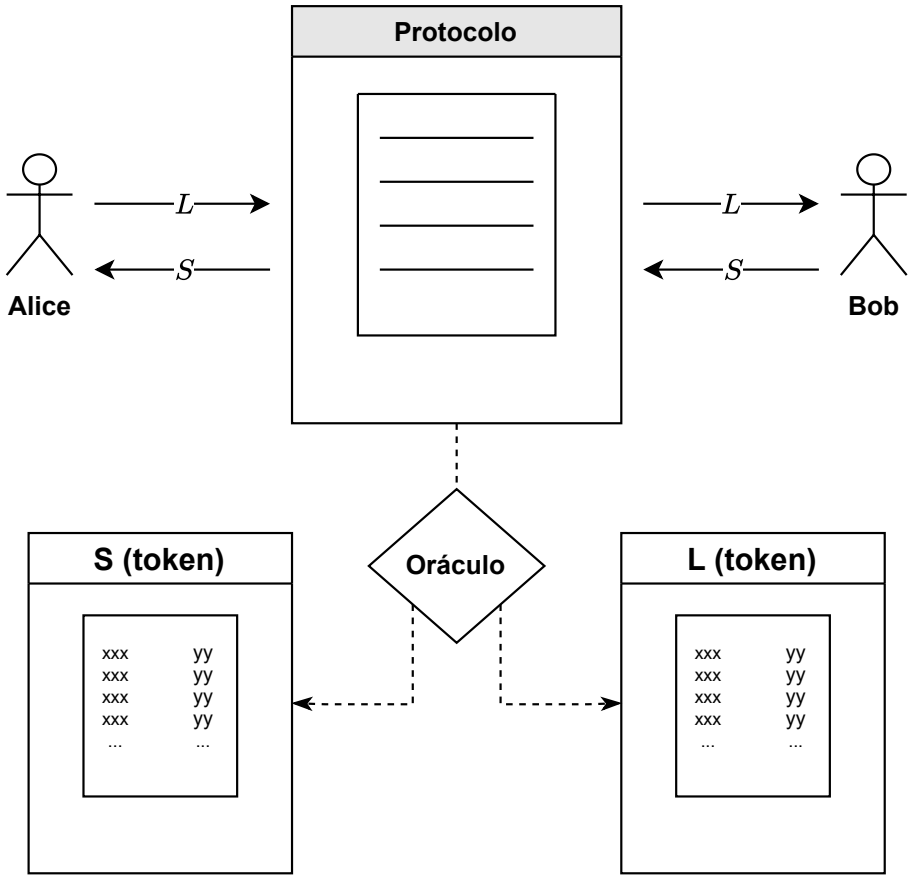


Figura 24 – O modelo de Compensação

Descrição: Alice transaciona com protocolo depositando em L e recebendo o sintético S. Já Bob, deposita em S e recebe em L. O protocolo ajusta os saldos de ambos nos contrato dos tokens de acordo com cotação provida pelo oráculo.

completamente expostos à esses choques no valor de L, e isso poderia ter um feedback negativo sobre a oferta desse serviço nesses cenários, o que seria problemático para a segurança da plataforma.

5.2.2.1 Implementação Terra

A plataforma (TERRA,) implementa um modelo similar ao modelo de compensação. A proposta é fundamentalmente sustentar valor ao cripto ativo L (LUNA) através do fomento ao desenvolvimento de vários tipos de aplicações, formando um ecossistema de serviços. Se espera que, dessa forma, alto nível de utilidade seja atrelado ao token L e robustez provida à economia endógena da plataforma. Uma das formas usadas para incentivar criação de aplicações na plataforma é através de arrecadações via cobrança de taxas e aportes decididos através de processos de governança.[33] Investimentos de organizações

[33] Originalmente a arrecadação era feita através de senhoriagem: a cada operação de emissão de sintéticos, o protocolo destruía apenas uma fração do montanto de L recebido e guardava o restante em uma reserva. Essa proposta pode ser vista em (KEREIAKES; KWON; PLATIAS, 2019).

centralizadas com capital aportado na plataforma (incluindo a Luna Foundation[34]), são outra alternativa de financiamento.

Para diminuir a exposição dos mineradores a choques no valor de L, a plataforma utiliza um mecanismo de ajuste de uma *taxa de estabilidade* cobrada em cima de transações na plataforma. Essa taxa é ajustada de acordo com a variação na receita da mineiração de forma inversamente proporcional, i.e., variações negativas na receita dos mineradores levam a variações positivas no valor dessa taxa (e vice-versa). Através desse mecanismo, espera-se que a renda dos mineradores possa ser suavizada frente a ciclos econômicos. Para evitar que grandes choques negativos sobre a atividade econômica leve a grandes choques positivos sobre o custo de transação, é estabelecida uma cota superior sobre essa taxa.

Essa proposta pode ser vista em (KEREIAKES; KWON; PLATIAS, 2019). No trabalho os autores mostram resultados de simulações nas quais a economia da plataforma sofre grandes choques de demanda e o mecanismo de ajuste é bem sucedido em suavizar a receita dos mineradores.[35] No entanto, não são providos detalhes sobre o modelo, e.g. se é considerada a clara endogeneidade da variável *demanda*.

A plataforma Terra foi criada em meados de 2019 e já atraiu bastante capital. A LUNA (o token nativo) já possui um mercado superior a U$32 bilhões, com a maior parte do seu crescimento acontecendo a partir de agosto de 2021. A UST, o USD sintético da plataforma, já possui uma oferta de quase 16 bi UST, ultrapassando o DAI por um fator de aproximadamente 1.65. Os dois representam a maior parte do capital alocado à plataforma da Terra. Além disso, tiveram o crescimento de seus mercados acontecendo praticamente todo durante o mesmo período.[36] Ambos os fatos sugerem que a UST foi o principal fator de crescimento à adesão da plataforma Terra.

No entanto, ainda existe incerteza, pelos motivos expostos acima, em relação à capacidade da plataforma em resistir grandes choques de demanda, ao contrário de outras implementações que já apresentaram um nível de resiliência considerável frente ao mercado altamente volátil de cripto ativos.[37] Ademais, o crescimento na adoção à plataforma ocorrido nos últimos meses tem sido em grande parte sustentado por subsídios[38] feitos em cima de depósitos de UST, que atualmente rendem cerca de 20% a.a. em aplicações de empréstimos na plataforma.

[ATUALIZAÇÃO] No meio do ano de 2022, antes da publicação dessa dissertação,

Atualmente não existe senhoriagem (<https://docs.terra.money/docs/develop/module-specifications/spec-market.html#seigniorage>, acessado 25/3/2022).

[34] Fundação ligada à Terraform Labs, organização responsável pelo desenvolvimento do protocolo.

[35] Esse modelo ainda prevê senhoriagem do protocolo.

[36] Fonte: <https://coinmarketcap.com/>, acessado em 25/3/2022.

[37] Ver, por exemplo, a "black thursday" (quinta feira negra) da MakerDAO (<https://insights.glassnode.com/what-really-happened-to-makerdao/>).

[38] Subsídios feitos pela Luna Foundation.

o cenário de espiral negativa abordado acima se concretizou derrubando completamente a plataforma Terra.

5.2.3 Modelo Poço de Dívida

No Modelo de Posição Colateralizada vimos como uma plataforma descentralizada pode ser construída para permitir agentes a emitirem sintéticos assumindo *posições vendidas* em relação ao ativo subjacente. Veremos abaixo um formato de plataforma que permite que agentes emitam sintéticos em ambas posições, i.e., vendidas ou compradas.

5.2.3.1 Criação/destruição de sintéticos

O primeiro componente do mecanismo será o cripto ativo usado como colateral, K. O protocolo permite a emissão de sintéticos a partir do depósito de quantias de K. Suponha que Alice é uma nova usuária da plataforma. Após fazer um depósito de uma quantia k de K no protocolo, Alice pode solicitar a abertura de uma dívida, denominada em um ativo numerário A_1, do qual normalizaremos o preço para um, i.e., $P_1 = 1$. O valor d da dívida, no ato da emissão, deve ser tal que,

$$d \leq \lambda \cdot kP_K \tag{5.5}$$

onde λ[39] é o parâmetro de colateralização para a emissão de sintéticos. A exemplo dos modelos anteriores, a cotação P_K é alimentada por um sistema oráculo. Após completado esse passo, Alice recebe custódia de uma quantia s_1 de sintéticos $S(A_1)$, tal que, $s_1 = d$. A partir desse momento Alice possui s_1 em $S(A_1)$ e uma dívida equivalente a d.

Suponha que $\mathbb{A} \equiv \{A_1, A_2, ..., A_n\}$ é o conjunto de ativos dos quais a plataforma emite sintéticos. Se Alice desejar se expor à valorização do ativo A_2, por exemplo, ela pode transacionar sua quantia s_1 com o protocolo e receber uma quantia s_2, tal que,

$$s_2 = \frac{s_1}{P_2}$$

onde P_2 é a cotação do ativo A_2 fornecido pelo oráculo. Quando essa transação é feita, o protocolo destrói s_1 de $S(A_1)$ e emite s_2 de $S(A_2)$, transferindo essa última quantia para Alice. Alice pode transacionar entre quaisquer sintéticos com protocolo, que realizará a compensação acima, similarmente ao *Modelo de Compensação* visto anteriormente. Esse sistema de compensação entre sintéticos permite que a plataforma ofereça liquidez ilimitada de cada sintético, já que a contraparte de cada operação é o próprio protocolo, que controla a oferta desses tokens. Para fechar sua dívida Alice deve depositar o valor

[39] Não tem nenhuma relação com λ do modelo CDP, apesar de ter um propósito similar.

correspondente à sua dívida em sintéticos para receber de volta a quantia k depositada em colateral no primeiro passo.

Esse é o funcionamento básico do modelo sob uma perspectiva estática. Ele, de certa forma, alia características dos dois modelos anteriores. Uma característica nova que contrasta com o modelo CDP é que, apesar serem abertas dívidas colateralizadas, a dívida total do protocolo é compartilhada por todos os usuários. Cada emissão de dívida por cada usuário contribui para aumentar o *poço de dívida* do protocolo.

5.2.3.2 A dinâmica contábil

Antes de introduzir dinâmica ao modelo, vamos formalizar a notação que será usada. Denominaremos por,

- S_i o sintético do ativo A_i;
- s_i^j, a quantia do sintético S_i sob custódia do usuário j;
- p^j, o valor da posição de j, denominado em A_1, i.e., $p_j = \sum_i s_i^j P_i$;
- d^j, o valor da dívida do usuário j, denominada em A_1;
- d^T, a dívida total do protocolo, denominada em A_1;
- k^j a quantia, em K, que j tem depositado no protocolo;
- k^T, a quantia total de K depositada no protocolo;
- $\Delta(X)$, a variação da variável X, i.e. $\Delta(X) = X' - X$.

Vamos começar supondo que $P_{S_i} = P_i$, para *todo* i, para ilustrar a dinâmica de funcionamento do protocolo para variações de P_i para *algum* i. Em seguida, mostraremos como P_{S_i} converge para P_i, para qualquer i, via estratégias de arbitragem.

Suponha novamente que Alice não possui dívida ou depósitos no protocolo. Em $t = 0$, Alice deposita k^a no protocolo e emite d^a em dívida, sujeita a condição 5.5. Ato contínuo, o protocolo entra em um mapeamento que ele mantém, duas variáveis relacionadas ao endereço de Alice: k^a e q^a, sendo esta última a fração da dívida total referente a Alice, com valor dado por,

$$q^a = \frac{d^a}{d_0^T + d^a},$$

sendo d_0^T a dívida total previamente à operação realizada por Alice e $\Delta(d^T) = d^a$. Após a computação ser feita, o valor d^T pode ser atualizado para incluir d^a, i.e., $d^T = d_0^T + \Delta(d^T)$.

Após Alice receber $s_1^a = d^a$ em S_1, Alice usa x dessa quantia para transacionar com o protocolo em troca de S_2, de forma que agora a posição de Alice é dada por,

$$\begin{aligned} s_1^a &= d^a - x \ , \\ s_2^a &= x/P_2 \ , \\ s_i^a &= 0 \ , \quad \forall i \notin \{1,2\}. \end{aligned}$$

Suponha que, em $t = 1$, P_2 sofre uma variação $\alpha > -1$, ou seja, $P_2' = (1+\alpha)P_2$, enquanto os demais ativos não variaram em preço. Como resultado dessa variação no preço de S_2, a dívida total do protocolo foi adicionada do mesmo valor absoluto que a posição de Alice, ou seja,

$$\Delta(p^a) = \Delta(d^T) = s_2^a \cdot \alpha P_2 = \alpha x.$$

A fração q da dívida de cada usuário, no entanto, permanece constante. A variação da dívida de Alice, portanto, será dada por,

$$\begin{aligned} \Delta(d^a) &= q^a \left(d^T + \Delta(d^T)\right) - q^a d^T \\ &= q^a \cdot \Delta(d^T) \\ &= q^a \cdot \alpha x. \end{aligned}$$

O lucro corrente de Alice, l^a, pode ser calculado subtraindo o seu ganho em dívida do ganho da sua posição, ou seja,

$$l^a = (1-q) \cdot \alpha x.$$

Obviamente, $\alpha < 0$ implica $l^a < 0$, significando prejuízo para Alice. Nesse exemplo, o que aconteceu essencialmente foi que os demais usuários financiaram (arrecadaram) o ganho (a perda) de Alice, já que suas posições se desvalorizaram (valorizaram) em relação à posição dela.

O valor corrente de d^j para cada usuário j, portanto, é atualizado através de q^j e do valor total da dívida d^T, i.e., $d^j = q^j \cdot d^T$. A dívida total, por sua vez, é dada por,

$$d^T = \sum_j \sum_i s_i^j P_i = \sum_j p^j,$$

para $i \in \{1, 2, 3, ..., n\}$ e $j \in \mathbb{U} \equiv \{a, b, c, ..., \}$, sendo $\mathbb{U}$ representativo do conjunto de usuários da plataforma. Variações em d^T podem ser calculadas por,

$$\Delta(d^T) = \sum_j \Delta(p^j),$$

enquanto variações no valor das posições de cada um são dadas por,

$$\Delta(p^j) = \sum_i s_i^j \Delta(P_i).$$

Esse valores são, portanto, contabilizados a qualquer momento através dos valores q^j e s_i^j, para todo i, j. Para liquidar sua posição, qualquer j deve pagar o equivalente a d^j em

sintéticos para receber de volta k^j em K. No momento que essa operação é executada, a quantia de sintéticos paga é destruída, e a dívida com o protocolo é saldada.

Vamos usar um exemplo de dois usuário e dois ativos. Suponha que $\lambda = 4$, $P_K =$ 100 USD e $P_{btc} = 1000$ USD. Em $t = 0$, Alice deposita 12 K de colateral no protocolo e emite 300 S_{usd} em dívida. Bob deposita 8 K, emite $200S_{usd}$ e imediatamente troca por .2 S_{btc}. Nesse momento temos $d^T = 500$ USD, $q^a = .6$, $q^b = .4$, $s^a_{usd} = 300$, $s^b_{usd} = 0$ e $s^b_{btc} = .2$. Em $t = 1$, P_{btc} aumenta em 15%. Após a valorização temos $\Delta(d^T) = \Delta(p^b) = 30$, enquanto $\Delta(p^a) = 0$. Como resultado, o estado contábil da plataforma será,

$$d^{T'} = 500 + 30 = 530,$$

$$p^{b'} = 200 + 30 = 230,$$

$$p^{a'} = p^a = 300,$$

$$d^{b'} = .4 \cdot d^{T'} = 212,$$

$$d^{a'} = .6 \cdot d^{T'} = 318.$$

Nesse momento, Bob pode fechar sua dívida depositando cerca de .1843 S_{btc} e lucrar o equivalente a 18 USD. Alice teve o valor da sua posição mantido constante porém sua dívida aumentou em exatos 18 USD. Para saldar sua dívida com o protocolo, Alice precisa comprar os 18 USD em sintéticos que Bob obteve de lucro.

5.2.3.3 Posição vendida

Por enquanto mostramos como agentes assumem posições compradas na plataforma. Como o valor nominal da dívida de cada um varia de acordo com a flutuação do preço dos ativos no mercado, não é possível assumir posições vendidas emitindo sintéticos e os vendendo no mercado como feito no modelo CDP. Para possibilitar estratégias de posição vendida, introduzimos mais um tipo de sintético:

- Seja S_{-i} o simétrico de S_i.

Assim como o objetivo do sintético é capturar variações no preço do ativo subjacente, o objetivo do *sintético simétrico* será capturar o valor simétrico de variações no preço do ativo. Ou seja,

$$P'_i = (1 + \alpha)P_i \Longrightarrow P'_{S_{-i}} = (1 - \alpha)P_{S_{-i}}.$$

A contabilidade feita pelo protocolo para variações do preço desses sintéticos segue o mesmo raciocínio feito acima.

5.2.3.4 Preço de S_i

Vamos ver agora como, dado o funcionamento ilustrado acima, estratégias de arbitragem contribuem para fazer o preço de mercado dos sintéticos convergir para o preço dos ativos subjacentes. Assim como nos modelos anteriores, os sintéticos emitidos aqui podem ser livremente transacionados no mercado secundário com ativos diversos, independentes da nossa plataforma. Agentes que não possuem depósitos na plataforma podem adquirir esses sintéticos no mercado secundário e inclusive podem transacionar com a funcionalidade de compensação no protocolo para trocar um tipo de sintético por outro. Para existirem agentes dispostos a adquirir esses tokens, no entanto, é crucial que exista confiança que o preço deles irá convergir para o preço alvo, caso contrário o risco dos tokens repentinamente perderem valor torna-se alto demais.

Continuando a usar $P_1 = 1$, suponha que, em dado momento, $P_{S_1} = \beta$. Se $\beta < 1$, qualquer usuário j pode comprar uma quantia x de S_1 no mercado pelo equivalente a βx e usar essa quantia para saldar o equivalente a x em dívida que ele possui no protocolo, obtendo com essa operação um lucro líquido equivalente a $(1-\beta)x$. Essa operação causou uma contração na oferta de S_1 na mesma quantidade, x. Se por outro lado, $\beta > 1$, qualquer usuário pode emitir x em dívida no protocolo e vender a quantia pelo equivalente a βx, obtendo um lucro equivalente a $(\beta - 1)x$. Essa operação causou uma expansão de x na oferta de S_1. A contração/expansão na oferta dos sintéticos resultante das estratégias acima deverá ter o mesmo efeito sobre o seu preço de mercado mencionado anteriormente. Enquanto existe diferença entre o P_1 e P_{S_1} existe oportunidade de lucro nas estratégias acima. O mesmo raciocínio segue para todo S_i, tal que $i \in \{1, 2, ..., n\}$.

5.2.3.5 Risco de sub-colateralização.

A exemplo do que acontece no modelo CDP, flutuações nos preços de mercado dos ativos, assim como de K, levam a flutuações no nível de colateralização efetivo do protocolo, que é dado por,

$$\kappa^T = \frac{k^T P_K}{d^T}.$$

A primeira implicação clara disso é que o protocolo pode se tornar subcolateralizado, de forma que o valor depositado nele não seja mais suficiente para cobrir o total de dívidas (nesse caso teríamos $\kappa^T < 1$). Uma situação similar ao problema 6.

Problema 7 *Flutuações nos preços dos ativos podem levar a plataforma a ficar subcolateralizada.*

Nesse caso, os sintéticos perderiam seu mecanismo de lastro já que as estratégias de arbitragem descritas acima poderiam perder seu potencial de lucro: uma vez que

agentes percebessem que a plataforma está sub-colateralizada, ele perceberiam que o valor depositado no protocolo não é suficiente para sustentar demanda pelo sintético através da estratégia de arbitragem, ou seja, agentes que possuem depósitos abririam mão de seus depósitos pois o valor deles não justificaria demandar os sintéticos por um preço próximo ao preço alvo. Nesse momento a demanda pelos sintéticos cairia e agentes tentariam se livrar deles levando o preço a sofrer uma grande queda. Quando o preço da oferta de sintéticos caísse para um nível inferior ao do colateral, os depositantes poderiam resgatar seu colateral a um prêmio. A esse ponto no entanto, os depositantes potencialmente teriam sofrido perdas consideráveis, assim como o mercado secundário de sintéticos como um todo.

Como resultado desse risco, uma segunda implicação prejudica a eficácia da implementação do mecanismo acima. Dado o problema 7, agentes que emitem dívidas devem incorrer o risco de perder parte de seus fundos, caso esse cenário se concretize (considerando o histórico de volatilidade do mercado de cripto ativos, esse risco deve ser relevante). Visando evitar assumir esse risco, a melhor estratégia para agentes que desejam adquirir sintéticos deve ser obtê-los no mercado secundário, ao invés de emití-los abrindo dívidas e assumindo risco. Se todos seguirem essa estratégia, ninguém emitirá dívidas no protocolo e não haverá sintéticos no mercado.

Problema 8 *O risco associado ao problema 7 pode gerar um equilíbrio onde agentes não emitem sintéticos.*

Considerando que a quantidade de agentes dispostos a assumir o risco da sub-colateralização será pequena, assim como o capital que eles estarão dispostos à alocar, o mercado formado em torno desses instrumentos será menor e com menos oportunidades de ganhos. Como resultado, existirão menos agentes interessados em transacionar esses sintéticos, limitando assim sua liquidez e tornando o investimento inicial ainda menos atraente.

Para atacar o problema 7 diretamente, incentivos devem ser providos para agentes depositarem colateral no protocolo. Isso pode ser feito: (i) recompensando depósitos, ou (ii) penalizando agentes que depositaram pouco. Uma forma de implementar (ii) é estabelecer um processo de liquidação (a exemplo do modelo CDP) usando o grau de colateralização de cada usuário j como variável. O grau de colaterização pode ser calculado por,

$$\kappa^j = \frac{k^j}{d^j}.$$

Para evitar liquidação, j deve garantir que $\kappa^j > \mu$ (sendo μ o parâmetro de colateralização mínima), aumentando k^j (através de depósitos de K), ou diminuindo d^j (através da destruição de S_i). O processo de liquidação deveria usar parte da quantia k^j para redimir uma quantidade suficiente de sintéticos que retornaria κ^j ao nível desejado.

É claro que, mitigando o risco de sub-colateralização, existirá um efeito indireto de mitigação do problema 8. No entanto, deve exsitir um grau de independência entre os dois problemas. Se, por exemplo, implementarmos um sistema de recompensas sobre depósitos, agentes poderiam simplesmente depositar colateral no protocolo e não emitir dívidas, aproveitando as recompensas sem correr nenhum risco. Isso por um lado é bom pois faz pressão para aumentar a colateralização do protocolo. Mas por outro lado, pode falhar em induzir que agentes emitam dívidas e provejam liquidez ao mercado de sintéticos. Portanto, para atacar o problema 8 diretamente, deve existir incentivo direto em emitir dívidas no protocolo.

5.2.3.6 Synthetix

A plataforma (SYNTHETIX, a)[40] implementa um modelo similar ao ilustrado acima na camada 1 da Ethereum. Para lidar com o problema 7, eles remuneram depósitos de K através de expansão da oferta desse token. Para ser elegível à essa remuneração, o usuário j deve garantir que $\kappa^j \geq \lambda$. O valor dessa remuneração pode variar de acordo com o nível de colateralização do protocolo, κ^T: quando κ^T está abaixo de determinado valor, a remuneração pode ser aumentada para induzir um aumento de depósitos; quando κ^T superar determinado valor a remuneração pode ser diminuída. Esse sistema é arriscado pois expande a oferta do token, podendo levar a um efeito inflacionário. No entanto, como ele induz agentes a manterem depósitos, ele também tem um efeito limitador sobre a oferta circulante dos tokens, o que pode mitigar esse potencial de inflação.

Para lidar com o problema 8, primeiro é estabelecido uma taxa sobre a operação de compensação da plataforma. Sempre que agentes realizam uma troca entre tipos de sintéticos, uma taxa τ é cobrada em cima da operação e enviada à uma reserva. Por exemplo, se um usuário transaciona s_1 de S_1 por s_2 de S_2, então $s_2 = s_1(1-\tau)P_2$. A diferença (τs_1) é enviada à reserva. Periodicamente, usuários que possuem dívidas abertas podem reivindicar um montante da reserva proporcional à sua fração q da dívida. Ou seja, se R representa a reserva em dado momento, j tem direito de reivindicar $q^j \cdot R$.

Além de ser feita essa remuneração em cima das dívidas, existe outro sistema de incentivos que utiliza o próprio sistema de remuneração de depósitos mencionado acima para incentivar emissão de dívidas: o protocolo só remunera depósitos suficientes para atingir a taxa de colateralização κ. Qualquer quantidade adicional de colateral depositado não é remunerado. Se um usuário j tem k^j depositado e,

$$P_K \cdot \frac{k^j}{d^j} > \lambda,$$

[40] Ver também (SYNTHETIX, b).

ele tem direito a uma remuneração proporcional apenas ao montante $k^j - \delta$, tal que,

$$P_K \cdot \frac{k^j - \delta}{d^j} = \lambda.$$

Assim, para maximizar a remuneração sobre o seu depósito ele deve emitir adicionalmente em dívida d^*, tal que,

$$d^* = \frac{P_K \delta}{\lambda}.$$

Originalmente, a plataforma contava com os sistemas de incentivos acima. Porém, mesmo com incentivos providos para a realização de depósitos, o risco de sub-colaterização continua uma ameaça saliente, motivo pelo qual a plataforma historicamente usou um valor λ bem grande (variando entre 4 e 7.5).[41] Atualmente, eles contam ainda com um processo de liquidação que será explicado a seguir.

5.2.3.6.1 Processo de liquidação.

Seja μ o parâmetro de colateralização mínima, sobre o qual posições podem ser liquidadas. Se, em $t = 0$, Alice tem k^a e d^a, tal que,

$$\frac{P_K k^a}{d^a} \leq \mu, \tag{5.6}$$

ela está exposta a ser liquidada. Nesse momento Bob, a procura de uma oportunidade de lucro, aponta ao protocolo a situação de Alice. Se confirmada a acusação de Bob, o protocolo começa a contar um prazo t_l para Alice recalibrar sua posição. Quando $t = t_l$, se a condição 5.6 se mantiver, o protocolo abre o processo de liquidação. Nesse momento Bob pode usar s_l em sintéticos para comprar k_l do total de colateral de Alice, tal que,

$$s_l = \frac{\lambda d^a - k^a P_K}{\lambda - 1 + \sigma},$$

$$k_l = s_l(1 + \sigma),$$

onde σ é representativo da taxa de penalização da liquidação. Após a operção acima, a posição de Alice é tal que,

$$\frac{k^{a'} P_K}{d^{a'}} = \lambda.$$

Bob pôde liquidar parte da dívida de Alice, de forma a recuperar o nível de colaterização dela para λ, comprando o colateral de Alice a um desconto estabelecido pela taxa σ.

Esse modelo é provavelmente o mais versátil dos três apresentados acima. Ele oferece uma plataforma de trading com liquidez ilimitada, onde usuários podem assumir tanto posições compradas quanto vendidas, com um sistema de incentivos para agentes

[41] Como referência de comparação, o MakerDAO tem $\lambda = 1.5$ para colateral ETH.

proverem liquidez ao mercado assumindo dívidas colateralizadas com o protocolo. Ela ainda pode ser implementada em múltiplas camadas 1, o que é uma grande vantagem para adoção em relação ao modelo da Terra, por exemplo.

A dependência em relação a manutenção do valor do token colateral[42] é uma vulnerabilidade, especialmente se considerarmos uma potencial política expansionista sobre a sua oferta. A plataforma Synthetix está abrindo a possibilidade de usar outros tipos de colateral, o que deve diminuir essa dependência.[43] Outra vulnerabilidade é a necessidade de manter o protocolo sobrecolateralizado, prejudicando a escalabilidade da oferta de sintéticos conforme apontado na descrição do modelo anterior. Por consequência, esse modelo não deve ser ideal para implementações de stablecoins e outros casos de uso dos quais escalabilidade de oferta seja uma propriedade importante.

[42] O token colateral da plataforma Synthetix é o SNX.
[43] Até o momento apenas o ETH é permitido como alternativa ao SNX.

6

Considerações finais

Espera-se que o conteúdo apresentado ao longo deste (longo) trabalho, sirva de base para subsequentes trabalhos que pretendam desenvolver estudos em cima dos conceitos e aplicações abordados. Procurou-se prover uma definição ao objeto principal de estudo, assim como a contextualização sobre sua importância na atual conjuntura e potencial impacto no futuro próximo. Foi argumentado que as implementações do conceito central ao tema do trabalho têm um caráter disruptivo sobre as principais Instituições formais, conforme as definimos na seção 1. Conforme foi colocado ainda, essa disrupção tem potencial de estabelecer um novo paradigma de relacionamentos econômicos, assim como de instituições com o propósito de regulá-las. Um estudo interessante de algumas dessas potenciais implicações pode ser visto em (WRIGHT; FILIPPI, 2015).

Uma dessas implicações seria, caso tomemos algumas hipóteses feitas acima como dadas, a obsolescência de alguns tipos de instituições tradicionais criadas para regular comportamento social. Em cima desse vácuo institucional, se faz necessário o desenvolvimento de outros tipos de instituições, com maior ênfase em valores sociais e normas informais, assim como algoritmos mecanísticos de coordenação, típicos de processos de governança organizacional, estudados por exemplo sob a rubrica da Economia de Custos de Transação (*Transaction Cost Economics*). Uma prévia de estudo desses tipos de mecanismos pode ser visto em (DIXIT, 2011), que usa como base a hipótese que em alguns países as instituições formais não funcionam como planejado (em alguns casos são até predatórias), fazendo necessária a utilização desses sistemas de coordenação descentralizados.

Procurou-se ainda dar uma base conceitual às essas implementações sob a perspectiva dos relacionamentos transacionais entre indivíduos/organizações, abordando o conceito de contratos, e seu par da era digital, contratos inteligentes. Foram abordadas algumas propriedades de relações contratuais que são impactadas pela introdução de contratos inteligentes, assim como potenciais vantagens dessas implementações que justificariam a sua adoção. Entre as principais vantagens estariam a desintermediação de vários

tipos de transação, o que poderia reduzir fricções e viabilizar transações previamente inviáveis;[1] a possibilidade de proteger informações sensíveis quando a intermediação fosse necessária; assim como a viabilização de transações onde existem dificuldades em relação ao comprometimento de agentes.[2]

A partir das exposições das aplicações de protocolos, procurou-se apresentar a lógica por trás de seus funcionamentos básicos, sobre a lente da Teoria Econômica. Na prática, implementações desses modelos de aplicações devem tocar em vários detalhes e complexidades que não foram abordados acima e que extrapolam o objetivo deste trabalho, que é oferecer uma introdução à teoria sendo aplicada por tais ferramentas. Ademais, não foi pretendido pelo conteúdo acima, atestar pela eficácia ou eficiência desses modelos. Dada a complexidade das aplicações abordadas, existe uma quantidade relevante de variáveis envolvidas em uma análise de performance desses protocolos. Um trabalho empírico, portanto, se faz propício para alcançar esse objetivo. Dada a extensa e acessível oferta de dados presente nas diferentes blockchains, trabalhos de análises quantitativas dessas variáveis representam uma avenida promissora para futuros projetos de pesquisa. Além dessas possibilidades, análises de outros tipos de aplicações sob o prisma teórico, assim como seus impactos sobre a teoria tradicional, terão extrema significância olhando para frente. DAO's, sigla para Organizações Descentralizadas Autônomas representam um caso especial que tocam diretamente no ponto sobre sistemas de governança descentralizados apontados acima, com potencial ainda pouco explorado.

É importante ressaltar, no entanto, que o caráter de descentralização das diferentes aplicações é um tema relativamente controverso. O conceito apresentado na seção 1, deve ser encarado sob o prisma da teoria que ele visa expor. Na prática, existirão espaços para nuances e argumentos sobre focos de centralização, espectros de descentralização e até caracterizações binárias dessas características. Até a Bitcoin, largamente considerada como a plataforma mais de facto descentralizada atualmente, recebe críticas em relação a centralização da oferta de mineração, assim como da oferta dos equipamentos necessários para essa operação. Uma coisa é muito clara, entretanto, quando feita uma análise econômica desses mercados: que descentralização é uma propriedade valorizada. Dado esse fato, cabe a todos nós, pesquisadores interessados nesse assunto, trabalhar para garantir que exista conhecimento disponível que possibilite que essas tecnologias sejam movidas da melhor forma possível.

[1] Ver (HALABURDA; LEVINA; SEMI, 2019).
[2] Ver (EENMAA-DIMITRIEVA; SCHMIDT-KESSEN, 2019).

Referências Bibliográficas

AL-BREIKI, H. et al. Trustworthy blockchain oracles: review, comparison, and open research challenges. *IEEE Access*, IEEE, v. 8, p. 85675–85685, 2020.

ALASHAIKH, L. Blockchain-based software systems: Taxonomy development. In: *2021 IEEE International Conference on Blockchain (Blockchain)*. [S.l.: s.n.], 2021. p. 491–498.

ALLEN, T.; WIDDISON, R. Can computers make contracts? *Harv. JL & Tech.*, HeinOnline, v. 9, p. 25, 1996.

ANTONOPOULOS, A. M. *Mastering Bitcoin: Programming the open blockchain*. [S.l.]: " O'Reilly Media, Inc.", 2017.

ANTONOPOULOS, A. M.; WOOD, G. *Mastering ethereum: building smart contracts and dapps*. [S.l.]: O'reilly Media, 2018.

BABAIOFF, M. et al. On bitcoin and red balloons. In: *Proceedings of the 13th ACM conference on electronic commerce*. [S.l.: s.n.], 2012. p. 56–73.

BAGARIA, V. et al. Proof-of-stake longest chain protocols: Security vs predictability. *arXiv preprint arXiv:1910.02218*, 2019.

BENIICHE, A. A study of blockchain oracles. *arXiv preprint arXiv:2004.07140*, 2020.

BENTOV, I.; PASS, R.; SHI, E. Snow white: Provably secure proofs of stake. *IACR Cryptol. ePrint Arch.*, v. 2016, p. 919, 2016.

BERTOLAI, J. D. P.; OLIVEIRA, V. A. de A. Criptomoedas e teoria monetária: Uma introdução. *Análise Econômica*, v. 38, n. 76, 2020.

BIER, J. *The Blocksize War: The battle over who controls Bitcoin's protocol rules*. [S.l.]: Independente, 2021.

BROWN-COHEN, J. et al. Formal barriers to longest-chain proof-of-stake protocols. In: *Proceedings of the 2019 ACM Conference on Economics and Computation*. [S.l.: s.n.], 2019. p. 459–473.

BUTERIN, V. *Proof of Stake: How I Learned to Love Weak Subjectivity*. 2014. Disponível em: <https://blog.ethereum.org/2014/11/25/proof-stake-learned-love-weak-subjectivity/>.

BUTERIN, V.; GRIFFITH, V. Casper the friendly finality gadget. *arXiv preprint arXiv:1710.09437*, 2017.

BUTERIN, V. et al. A next-generation smart contract and decentralized application platform. *white paper*, v. 3, n. 37, 2014.

CARLSTEN, M. et al. On the instability of bitcoin without the block reward. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. [S.l.: s.n.], 2016. p. 154–167.

CHEN, J.; MICALI, S. Algorand: A secure and efficient distributed ledger. *Theoretical Computer Science*, Elsevier, v. 777, p. 155–183, 2019.

DAIAN, P. et al. Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In: IEEE. *2020 IEEE Symposium on Security and Privacy (SP)*. [S.l.], 2020. p. 910–927.

DAMERON, M. Beigepaper: An ethereum technical specification. *Ethereum Project Beige Paper*, 2018.

DEIRMENTZOGLOU, E.; PAPAKYRIAKOPOULOS, G.; PATSAKIS, C. A survey on long-range attacks for proof of stake protocols. *IEEE Access*, IEEE, v. 7, p. 28712–28725, 2019.

DIXIT, A. K. *Lawlessness and economics*. [S.l.]: Princeton University Press, 2011.

EENMAA-DIMITRIEVA, H.; SCHMIDT-KESSEN, M. J. Creating markets in no-trust environments: The law and economics of smart contracts. *Computer law & security review*, Elsevier, v. 35, n. 1, p. 69–88, 2019.

EYAL, I. The miner's dilemma. In: IEEE. *2015 IEEE Symposium on Security and Privacy*. [S.l.], 2015. p. 89–103.

EYAL, I.; SIRER, E. G. Majority is not enough: Bitcoin mining is vulnerable. In: SPRINGER. *International conference on financial cryptography and data security*. [S.l.], 2014. p. 436–454.

FENG, Q. et al. A survey on privacy protection in blockchain system. *Journal of Network and Computer Applications*, v. 126, p. 45–58, 2019. ISSN 1084-8045. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1084804518303485>.

FERGUSON, N. *The ascent of money: A financial history of the world*. [S.l.]: Penguin, 2008.

GILAD, Y. et al. Algorand: Scaling byzantine agreements for cryptocurrencies. In: *Proceedings of the 26th symposium on operating systems principles*. [S.l.: s.n.], 2017. p. 51–68.

HAFID, A.; HAFID, A. S.; SAMIH, M. Scaling blockchains: A comprehensive survey. *IEEE Access*, v. 8, p. 125244–125262, 2020.

HALABURDA, H.; LEVINA, N.; SEMI, M. Understanding smart contracts as a new option in transaction cost economics. In: *Proceedings of the 40th International Conference on Information Systems, Munich, Germany*. [S.l.: s.n.], 2019.

KEREIAKES, E.; KWON, M. D. M. D.; PLATIAS, N. *Terra money: Stability and adoption*. 2019.

KIAYIAS, A. et al. Ouroboros: A provably secure proof-of-stake blockchain protocol. In: SPRINGER. *Annual International Cryptology Conference*. [S.l.], 2017. p. 357–388.

KING, S.; NADAL, S. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. In: . [S.l.: s.n.], 2012.

KOCHERLAKOTA, N. R. Money is memory. *journal of economic theory*, Elsevier, v. 81, n. 2, p. 232–251, 1998.

KROLL, J. A.; DAVEY, I. C.; FELTEN, E. W. The economics of bitcoin mining, or bitcoin in the presence of adversaries. In: *Proceedings of WEIS*. [S.l.: s.n.], 2013. v. 2013, p. 11.

LIAO, K. et al. Behind closed doors: measurement and analysis of cryptolocker ransoms in bitcoin. In: *2016 APWG Symposium on Electronic Crime Research (eCrime)*. [S.l.: s.n.], 2016. p. 1–13.

MA, J.; GANS, J. S.; TOURKY, R. *Market structure in bitcoin mining*. [S.l.], 2018.

MAKERDAO. *The Maker Protocol: MakerDAO's Multi-Collateral Dai (MCD) System*. Disponível em: <https://makerdao.com/en/whitepaper/>.

MAKERDAO. *MakerDAO Technical Docs*. Disponível em: <https://docs.makerdao.com/>.

MIRROR PROTOCOL. *Mirror Documentation*. Disponível em: <https://docs.mirror.finance>.

NAKAMOTO, S. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, p. 21260, 2008.

NAYAK, K. et al. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. In: IEEE. *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. [S.l.], 2016. p. 305–320.

NORTH, D. C. *Institutions, institutional change and economic performance*. [S.l.]: Cambridge University Press, 1990.

OPEN ZEPPELIN. *ERC20*. Disponível em: <https://docs.openzeppelin.com/contracts/2.x/erc721>.

OPEN ZEPPELIN. *ERC721*. Disponível em: <https://docs.openzeppelin.com/contracts/2.x/tokens>.

POELSTRA, A. On stake and consensus (2015). *URL https://download. wpsoftware. net/bitcoin/pos. pdf*, 2015.

ROSENFELD, M. Analysis of hashrate-based double spending. *arXiv preprint arXiv:1402.2009*, 2014.

RUSSO, C. *The infinite machine: how an army of crypto-hackers is building the next internet with Ethereum*. [S.l.]: Harper Business Nueva York, 2020.

SALEH, F. Blockchain without waste: Proof-of-stake. *The Review of financial studies*, Oxford University Press, v. 34, n. 3, p. 1156–1190, 2021.

SAPIRSHTEIN, A.; SOMPOLINSKY, Y.; ZOHAR, A. Optimal selfish mining strategies in bitcoin. In: SPRINGER. *International Conference on Financial Cryptography and Data Security*. [S.l.], 2016. p. 515–532.

SCHÄR, F. Decentralized finance: On blockchain-and smart contract-based financial markets. *FRB of St. Louis Review*, 2021.

SGUANCI, C.; SPATAFORA, R.; VERGANI, A. M. Layer 2 blockchain scaling: A survey. *arXiv preprint arXiv:2107.10881*, 2021.

SPAGNUOLO, M.; MAGGI, F.; ZANERO, S. Bitiodine: Extracting intelligence from the bitcoin network. In: CHRISTIN, N.; SAFAVI-NAINI, R. (Ed.). *Financial Cryptography and Data Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. p. 457–468. ISBN 978-3-662-45472-5.

SYNTHETIX. *Litepaper*. Disponível em: <https://docs.synthetix.io/litepaper/>.

SYNTHETIX. *Smart Contract Architecture*. Disponível em: <https://docs.synthetix.io/contracts/>.

SZABO, N. Formalizing and securing relationships on public networks. *First monday*, 1997. Disponível em: <https://firstmonday.org/ojs/index.php/fm/article/view/548/469>.

TERRA. *About Terra Protocol*. Disponível em: <https://docs.terra.money/docs/learn/protocol.html>.

VASIN, P. Blackcoin's proof-of-stake protocol v2. *URL: https://blackcoin. co/blackcoin-pos-protocol-v2-whitepaper. pdf*, v. 71, 2014.

VOSHMGIR, S. *Token Economy: How the Web3 reinvents the Internet*. [S.l.]: Token Kitchen, 2020. v. 2.

WILLIAMSON, O. E. Markets and hierarchies: analysis and antitrust implications: a study in the economics of internal organization. *University of Illinois at Urbana-Champaign's Academy for Entrepreneurial Leadership Historical Research Reference in Entrepreneurship*, 1975.

WILLIAMSON, O. E. The new institutional economics: taking stock, looking ahead. *Journal of economic literature*, v. 38, n. 3, p. 595–613, 2000.

WOOD, G. et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, v. 151, n. 2014, p. 1–32, 2014.

WRIGHT, A.; FILIPPI, P. D. Decentralized blockchain technology and the rise of lex cryptographia. *Available at SSRN 2580664*, 2015.

ZAMFIR, V. Casper the friendly ghost: A correct by construction blockchain consensus protocol. *Whitepaper: https://github. com/ethereum/research/blob/master/papers/caspertfg/caspertfg. pdf*, 2017.