Dagora: Um Mercado Virtual Baseado em *Side-Chains*

Gustavo Inácio[1], Luciana Rech[1], Ray Neiheiser[2], Joni Fraga[2]

[1] Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)

[2]Departamento de Automação e Sistemas – Universidade Federal de Santa Catarina (UFSC)

Abstract. *The price increase resulting from the growing popularity of the blockchain technology, created numerous issues for decentralized applications. An increasing popular solution is running side-chains next to a popular blockchain, offering vast cost advantages. While the security impacts of these solutions are well understood, the complexity of moving an application to a side-chain and the cost advantages have received little attention in the literature. In this paper we analyze both the complexity as well as the cost advantages using a decentralized market application called Dagora Market as an example. We show that it is relatively simple to move an application to a side-chain resulting in a cost advantage of up to 6 orders of magnitude compared to running it on the main-chain.*

Resumo. *Com o recente crescimento dos preços das criptomoedas, o uso de blockchains em diversas aplicações com contratos inteligentes tem se tornado impeditivo devido aos custos dos serviços. Uma das soluções que vem sendo muito aceita é o uso de side-chains para a execução dos smart contracts, ficando uma blockchain principal (main-chain) para a sincronização dos resultados. Os benefícios apontados na literatura indicam grandes vantagens. Porém, a complexidade em mover uma aplicação para um side-chain e o ganho em termos de custos não são examinados com o devido rigor na literatura. Neste artigo, apresentamos o Dagora Market, um mercado descentralizado de compra e venda de mercadorias construído a partir de contratos inteligentes em side-chain. Além de apresentar os aspectos arquiteturais, demonstramos os ganhos nesta abordagem de side-chain, principalmente no referente ao custo econômico se comparado com execuções de smart-contracts em main-chain.*

1. Introdução

A grande aceitação de *ICOs* (*Initial Coin Offerings*) [Fenu et al. 2018] nos últimos anos, deu origem a uma quantidade surpreendente de aplicações baseadas em *blockchains* ao redor do mundo. Estes projetos estiveram sempre na promoção da descentralização, com o objetivo de colocar o controle das operações bem próximo aos usuários. Deste modo, afastando sempre que possível o controle das *Big Techs* (como *Amazon*, *Microsoft*, etc.).

O uso do conceito de contratos inteligentes (*smart contract*) contribui muito para o controle descentralizado (*smart contracts*[1] [Szabo 1997]) disponíveis em *blockchains*

[1]Um *smart contract* [Szabo 1997] nada mais é do que a formação de contratos digitais firmados por partes que expressam a vontade de realizar um determinado negócio. Este contrato é igual a contratos do mundo real, com a diferença de ser digital. Os *smart contracts* são auto-executáveis por fazerem uso de *blockchain*. Um contrato inteligente pode ser descrito como uma máquina de estado replicada e disposta em todos os nós de um *ledger* distribuído, podendo ser invocada por clientes e mesmo outros *smart contracts*.

e usados em muitas aplicações. São exemplos: o *ERC-20 token* fornece uma espécie de criptomoeda, permitindo qualquer pessoa criar a sua criptomoeda ao lançar seu *token* em uma *blockchain* [eth 2021a]; o *Uniswap* é uma bolsa de criptomoedas descentralizada que permite a troca de diferentes tipos de *tokens* (criptomoedas) [Adams et al. 2021] e o *Kleros* é um sistema para resolver disputas de forma descentralizada [Lesaege et al. 2020].

O *ledger Ethereum*, é um dos ecossistemas de *blockchain* mais valiosos [coi 2021]. Entretanto, o crescimento de interesse nestes *ledgers* distribuídos gerou como consequência um forte aumento no preço das criptomoedas dos mesmos, tornando seu uso mais restritivo. No *Ethereum* [Buterin 2013], cada operação gera um custo em *"gas"*. Neste caso, cada transação tem que conter *gas* suficiente para cobrir o gasto de sua execução. O preço do *gas* depende da carga da rede (*ledger*) e é medido em *Gwei* [eth 2021a]. Altos custos provocam muitas vezes a inviabilidade de muitas aplicações, principalmente as menores e menos lucrativas [Foxley 2021].

Para resolver o problema na execução de contratos inteligentes, muitas soluções foram propostas na literatura. Estas soluções têm sido identificadas como *layer-2-solutions*, ficando o *ledger* principal (o *Ethereum*, por exemplo) denotado como *layer-1*. Em síntese, as soluções de nível 2 são construídas em um servidor centralizado ou em outra *blockchain* (*side-chain*) para a execução de contratos inteligentes. Neste caso, uma transação é recebida em nível 2, sendo então validada e executada no contrato inteligente. Depois, o *layer-2* pode fazer persistir os resultados de um *batch* de transações no *layer-1* podendo oferecer as mesmas garantias que a *blockchain* principal [eth 2021a].

A execução dos códigos de contratos em nível 2 usando servidores centralizados quase sempre diminui o custo significantemente quando comparado com execuções em qualquer *blockchain* de *layer-1*. Porém, para isso funcionar, é necessário um certo nível de responsabilidade do desenvolvedor de contratos. Um exemplo de contratos inteligentes em nível 2 baseado em servidor é apresentado em [Neiheiser et al. 2020] descentralizando concursos públicos. Neste caso, todas as informações são publicadas pelo servidor em *blockchain* (nível 1) para que os envolvidos possam confirmar a regularidade e a transparência do processo. Na ocorrência de irregularidades, a instituição pode ser processada. Porém, este tipo de abordagem não é adequado para todos os casos possíveis de aplicações. Em alguns casos, como o de um mercado fornecedor de bens onde as compras podem estar espalhadas pelo mundo inteiro, a centralização da camada de nível 2 em um servidor se torna inviável. Em qualquer malversação do vendedor fora da jurisdição do país do comprador impossibilitaria possíveis ações jurídicas.

Por outro lado, as soluções de *side-chains* colocam as execuções de nível 2 dos contratos inteligentes em uma outra *blockchain* (a *side-chain*). Os resultados são geralmente sincronizados, e tornam-se persistentes na *blockchain* principal (*main-chain*). Este tipo de solução, comparada a do servidor, apresenta transparência e maior segurança, podendo ser aplicada em vários tipos de aplicação [eth 2021a].

A escolha de uma *blockchain* como *main-chain* deve ser guiada por objetivos bem definidos. Como existem muitas *blockchains* diferentes, numa aplicação de mercado, o desenvolvedor da aplicação se apoiará na vantagem de usar *blockchains* consagradas e bem difundidas. Pois, terá disponível a comunidade (possivelmente grande) já existente na plataforma, além de ter garantias que a *blockchain* continuará efetiva muitos anos.

Neste artigo é apresentado um mercado descentralizado, o *Dagora Market*, onde suas funções são implementadas a partir de contratos inteligentes de *side-chain.* Esta aplicação foi também objeto de estudo com o objetivo de verificar a complexidade em mover contratos para uma *side-chain* e analisar o ganho com relação aos custos neste tipo de solução. O *Ethereum* foi escolhido como *main-chain* para o *Dagora Market*. Os contratos inteligentes foram executados em uma *side-chain* (*Polygon* [Kanani et al. 2019]).

Na quantificação de nossos resultados, analisamos os custos de execução dos contratos inteligentes tanto no uso do *side-chain Polygon*, como a execução direta no *Ethereum*. Seguindo isso, discutimos a complexidade de se aplicar estes contratos no *Polygon* em relação à implementação destes diretamente no *Ethereum*. Ou seja, tratamos de dar uma medida de impacto da aplicação ao utilizar *side-chains*.

O artigo apresenta uma revisão de trabalhos relacionados na seção 2, apontando uma classificação que considera a distribuição da *layer-2* e as abordagens usadas na implementação desta camada. Na seção 3, a funcionalidade do *Dagora Market* é descrita. Na seção 4, são mostradas as análises dos resultados dos contratos *Dagora* sendo executados em *side-chain* e em *main-chain*. Por fim, a seção 5 apresenta as considerações gerais dos resultados e comparações com a literatura.

2. Trabalhos relacionados

O *Bitcoin* [Nakamoto 2008] foi introduzido com a intenção de criar uma moeda eletrônica *peer-to-peer* sem passar pela coordenação ou centralização de uma instituição financeira. No núcleo desta rede *P2P*, está uma *blockchain* que corresponde a um armazenamento imutável. A *blockchain* é uma cadeia de blocos replicada em diferentes nós da rede, onde cada bloco contém o *hash* do bloco anterior. Desta forma, um novo bloco ao ser inserido torna persistente o bloco anterior [Nakamoto 2008].

Para a consistência desta rede, considerando a replicação da cadeia de blocos, é necessário um protocolo de consenso que garanta a sincronização das réplicas (ou seja, réplicas com o mesmo estado). No caso do *Bitcoin*, esta sincronização é feita pelo algoritmo *Proof of Work* (PoW) onde cada nó para colocar um bloco na rede tem que resolver um problema matemático bem definido. O primeiro nó que consegue pode propor o seu bloco como o seguinte na *blockchain*. Os outros participantes verificam se este bloco é válido, e só então o mesmo é tornado persistente [Nakamoto 2008]. A combinação da *blockchain* com o protocolo de consenso é chamado de *ledger* (ou *ledger* distribuído). Além do *Bitcoin*, foram criados centenas de diversos *ledgers* [Bach et al. 2018].

Com essa tecnologia em mente, foi proposto por *Vitalik Buterin* [Buterin 2013], a utilização de *ledgers* distribuídos como plataforma para aplicações descentralizadas permitindo a execução de códigos, dando origem ao conceito dos contratos inteligentes. Este conceito também norteou o projeto do *Ethereum*. No caso deste *ledger*, para garantir que todos os nós na rede obtenham o mesmo resultado de códigos replicados, é usada a *Ethereum Virtual Machine* (EVM) [eth 2021a] e a linguagem *Solidity* [sol 2021]. Com ajuda de contratos inteligentes foram criadas também moedas adicionais na rede *Ethereum*, dando início a muitos projetos. Executar aplicações no *Ethereum* tem vantagens como a de ser um considerável ecossistema com um grande potencial de usuários. Mas, com o crescente uso do *Ethereum*, torna-se muito dispendioso [Rozen 2021].

2.1. Disposição e Abordagens para Execuções de Camada-2.

Soluções de *layer-2* foram propostas para contornar o problema destes custos de escalabilidade. Muitas destas soluções são baseadas em servidores externos que executam as transações de clientes e depois publicam os respectivos resultados na *blockchain*. Desta maneira, qualquer interessado pode verificar a validade dos resultados [eth 2021a]. Um requisito para estas soluções é a existência de um depósito de segurança por parte dos operadores do servidor de modo a ressarcir clientes que tenham sido prejudicados por alguma ação errônea ou desonesta do serviço prestado. Esta abordagem é usada quando os agentes do serviço envolvem grandes empresas ou órgãos públicos. [Neiheiser et al. 2020].

Entidades com recursos não tão significativos (pequenos empreendedores ou clientes comuns) normalmente fazem uso de plataformas distribuídas para execuções de nível 2. Como isto, o uso de *side-chains* na sincronização em *main-chains* vem se tornando uma alternativa muito consistente em relação a servidores centralizados. *Side-chains* começaram a ser usadas com a intenção de executar *smart contracts* em nível 2, deixando a *main-chain* para a persistências dos resultados [Singh et al. 2020]. As *side-chains* podem apresentar seus próprios modelos de consenso e, portanto, serem independentes da *main-chain* (o *Ethereum*, por exemplo) [eth 2021a]. A *Polygon* tem sido uma destas *side-chains*, usando a mesma maquina virtual como o *Ethereum* (EVM) e desta forma oferecendo suporte para contratos inteligentes.

Além deste aspecto da forma de disposição de soluções de *layer-2* (seja em servidores ou em outros *ledgers*), surgiram abordagens de como executar estas funções de camada-2. Muitas das execuções adotam as abordagens identificadas como *rollups* que se apresentam em duas categorias: os *rollups* otimistas e os *zero knowledge (zkrollups)*.

Os *rollups* otimistas correspondem a serviços de segundo nível (*layer-2*) com uma certa quantia de criptomoedas em um contrato inteligente na *main-chain* (*layer-1*) [eth 2021a]. Estes serviços, possuem "agregadores" que executem as transações de seus usuários e publicam *batches* dos resultados compactados em *smart contract* na *layer-1*. Um *batch* inserido na *main-chain* pode ser verificado por um cliente ou verificador externo. A verificação envolve a reexecução das transações do *batch* comparando os resultados com os que persistiram na *main-chain*. Se não houver um *match*, o verificador correspondente denuncia o agregador via o contrato inteligente na *layer-1*. Se o *batch* se confirmar errôneo, o agregador perde o depósito de segurança e o verificador recebe uma parte pela detecção da malversação. Em resumo, estes *optimistic rollups* tornam, no melhor caso, as transações menos custosas para serem executadas na *layer-2*. Porém, existe um *trade-off* entre a latência na finalização e o custo de transações.

Os *rollups* identificados como *Zero knowledge (zkrollups)* são similares ao caso anterior, funcionando em *layer-2*, acumulando transações executadas fora da *main-chain* e publicando na camada 1 a correspondente prova criptográfica (*SNARKs* [Bowe et al. 2018]). Estas *ZK proofs* comparam o estado (*snapshot*) da *blockchain* antes das execuções das transações com o estado da *blockchain* após as execuções (ou seja, os valores antes e depois das execuções) e relata apenas as mudanças para a *main-chain* em um *hash* verificável. Os *zkrollups* cumprem a finalidade de rápido desempenho nas verificações, porém necessitam de poder de processamento maior.

Uma outra abordagem presente na literatura, identificada como *State-Channels*,

faz uso de recursos de clientes na execução de camada-2 [sta 2021]. Neste caso, dois clientes, ou um cliente e uma aplicação se comunicam por fora da *blockchain*. Com as trocas finalizadas, ambos mandam a história das trocas (os resultados e transações) para um *smart contract* na camada 1 para a persistência dos mesmos. Desta forma, a rede do *Ethereum* fica reduzida a um mínimo de operações e o cliente só paga taxas de transferência.

Side-chains conectadas a uma *main-chain*, constituem outro tipo de abordagem para a execução da *layer-2*. Contratos inteligentes determinam as trocas fáceis de ativos entre das duas *blockchains*. *Checkpoints* regulares das execuções de nível 2 são publicados regularmente com o *hash* do *Merkel Tree Root* de *Batches* destas transações na *main-chain*. Estas cadeias laterais devem ter o seu próprio modelo de confiança, pois os fundos transferidos para a *side-chain* usualmente não são protegidos pelo modelo de confiança do *Ethereum*. *Side-chains* que usam provas de fraudes, são identificados como *Plasma*. A rede *Polygon* é um *side-chain* que implementa este tipo de modelo.

A tabela 1 resume as discussões sobre as abordagens e como são dispostas as funcionalidades com contratos em nível 2, citando nas entradas projetos exemplos.

Abordagens de *Layer-2*		Posicionamento da *Layer-2*		
		Servidor centralizado	Decentralizados	Recurso de Cliente
Rollups	Optimistic	Arbitrum [arb 2021]	Optimism [opt 2021]	
	Zero Knowledge	Loopring [loo 2021]	zkSync	
Side-chain	Plasma		Polygon	
	Puro		Skale	
State-channels				kChannels [kch 2021]

Tabela 1. Abordagens e Disposições de Contratos em Nível 2

2.2. Mercados Virtuais de Bens

No contexto de mercados virtuais, existem diversos trabalhos focando a troca de bens (digitais). São exemplos de mercados *Non Fungible Tokens* (NFT) [wha 2021] o *OpenSea* [ope 2021]. Todos os citados fazem uso da descentralização baseados em contratos do *Ethereum* e sem adotar a *layer-2*. Uma alternativa para produtos da vida real é o *Origin-Protocol* [ori 2021] que tambem usa *smart contracts* do *Ethereum* e funciona como um *backend* descentralizado para a criação de *webshops* pessoais.

Em relação a trabalhos acadêmicos, recentemente surgiram diversos *market places* fazendo uso de *blockchains*. O DESEMA [Klems et al. 2017] é um destes trabalhos onde um protótipo de um mercado descentralizado é parcialmente desenvolvido. No DESEMA, todos os serviços são baseados em *smart contracts*, porém não foram implementados por completo. Portanto, seus custos em uma *blockchain* não têm como serem avaliados.

O trabalho em [Ranganthan et al. 2018] propõe um aplicativo descentralizado de mercado virtual construído sobre o *Ethereum*. Este mercado é baseado somente em contratos inteligentes do *Ethereum* (não usa a definição de *layer-2*). Este trabalho faz um estudo comparativo de suas proposições com mercados estabelecidos que não usam *blockchains*. A base deste estudo são os custos da aplicação proposta na *blockchain* usada. Além disto, este trabalho descreve as margens de lucro de um vendedor que tem seus produtos no *eBay* e o *Sotheby's*, comparando estas margens com as obtidas quando o mesmo vendedor apresenta seus produtos no mercado proposto sobre o *Ethereum*. O *eBay* e o

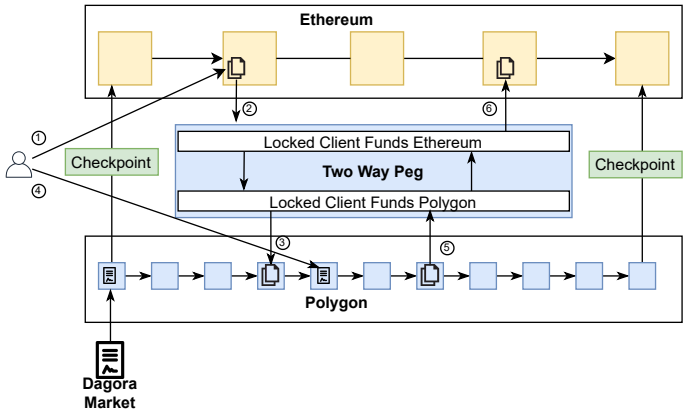
Sotheby's são sites que não fazem uso de *blockchains* e cobram dos vendedores taxas sobre os produtos vendidos que se aproximam de 10% do valor obtido. O texto mostra que se usado mercado baseado no *Ethereum*, o vendedor teria mais receita em produtos mais caros e o custo para o comprador aumentaria em alguns centavos. Com isto, os autores reiteram que por ter uma margem maior de lucro, produtos com valores altos poderiam ser vendidos mais baratos no *blockchain*. Entretanto, não é falado no texto que tudo isso depende do preço de *gas* e do valor do *token* (considerando o *Ethereum*). Comparado com a época em que o artigo foi escrito, o custo de execução hoje em dia seria 75 vezes maior (de 0.86 centavos no artigo para 64.50$ nos dias de hoje) tornando a proposta, independente do produto, não competitiva.

3. Dagora Market

Neste trabalho, usamos a aplicação de um *marketplace* descentralizado, denominado *Dagora Market* para explorar os custos de uso de uma composição *side-chain/main-chain*. O *Dagora* conecta diretamente de forma descentralizado vendedores a compradores, utilizando criptomoedas como meio de pagamento e contratos inteligentes nestas conexões. Desta forma é possível anunciar produtos assim como administrar anúncios existentes através de contrato inteligente no *Polygon* referente ao *Dagora Market*. Informações sobre anúncios existentes podem ser obtidas com *queries* ao histórico da *blockchain Polygon*. O *Dagora Market* é apresentado nos seus aspectos arquiteturais e funcionais nesta seção.

3.1. Arquitetura

Figura 1. System Archicecture



A arquitetura do sistema proposto é mostrada na Figura 1. O *Polygon* é apresentado como cadeia lateral e o *Ethereum* como a cadeia principal. Portanto, a aplicação de mercado é implementada como *layer-2* no *Polygon*, na forma de contratos inteligentes. Neste arranjo *side-chain/main-chain*, o *Polygon* e o *Ethereum* apresentam um procedimento identificado como *two-way peg* (Figura 1) para as trocas seguras de fundos (criptomoedas) e pontos de verificação regulares (*checkpointing*). O *two-way peg* envolve as seguintes trocas:

1. Um cliente envia fundos para um endereço especifico no *Ethereum*;
2. Os fundos ficam então bloqueados neste contrato do *Ethereum*;

3. Os fundos são criados (*minted*) na cadeia lateral *Polygon*;
4. O cliente interage com o mercado no *Polygon* despendendo numa compra, além do custo do produto, um valor pela transação significativamente reduzido. E, uma vez terminada a transação, os fundos restantes podem ser transferidos de volta para a cadeia principal, bloqueando os mesmos em um contrato inteligente no *Polygon*;
5. Os fundos restantes são transferidos e desbloqueados no Ethereum. Estes fundos são destruídos na *side-chain Polygon*.

A entidade do *Dagora* que recebe fundos do *Ethereum* via o *Polygon* é um contrato inteligente e foi nomeada Gerente de Fundos. No passo 4 do procedimento acima, a interação do cliente acontece com o contrato inteligente que chamamos de Gerente de Compras.

O *Polygon* oferece duas maneiras de manter o *two-way peg* confiável que diferem significativamente ao retirar *tokens* para o *Ethereum*. A primeira é através de um quórum *PoS* de validadores do *Polygon*, permitindo então que o contrato inteligente no *Ethereum* possa liberar os *tokens* (os fundos) ao cliente em aproximadamente 10-30 minutos. A segunda maneira de retirar *tokens* do *Polygon* e desbloqueá-los no *Ethereum* é através de uma prova de fraude (*Plasma*) que indica que os fundos foram enviados. Esta prova é enviada para um contrato no *Ethereum*, a ser verificado em até 7 dias. Após este período os *tokens* estarão desbloqueados no *Ethereum*. Embora a solução de quórum *PoS* seja mais rápida, ela é menos segura uma vez que requer confiança nos validadores do *Polygon*.

Os *checkpoints* mostrados na Figura 1 contém *Merkel Root Hash* de todos os blocos desde o último *checkpoint* bem como *hash* de um *batch* de transações. As transferências destes *checkpoints* são dirigidas a um contrato no *Ethereum* que garantem a imutabilidade do estado do *Polygon*. Estes *checkpoints*, portanto, são garantias do aplicativo de mercado aos vendedores e compradores contra os possíveis riscos em suas transações.

3.2. Funcionamento

A honestidade dos participantes é garantida por incentivos baseados na teoria de jogos[2] [Liu et al. 2019]. Para esse fim, utilizamos *tokens ERC-20* [eth 2021a] que denotamos no texto como *DGR*. Um vendedor para anunciar um produto necessita adquirir *tokens DGRs* e fazer um depósito de segurança (*stake*) de uma quantidade mínima[3] no contrato do mercado (Gerente de Compras). Esta quantia pode ser considerada um depósito de segurança para garantir a honestidade do mesmo.

Com o *stake* realizado, os vendedores podem anunciar um produto. Compradores poderão buscar produtos na plataforma usando palavras-chave, categorias, etc. A ordem em que os anúncios são mostrados é proporcional à quantidade que possuem de *tokens* que estão em seus *stakes*, quanto mais *DGRs*, maior a chance do anúncio aparecer no topo da página de busca. Desta forma, vendedores que investem em um maior depósito de segurança, podem ser considerados mais confiáveis. O Gerente de Fundos emite *tokens DGR* que são usados como principal meio de troca entre compradores e vendedores. Uma característica importante desses *tokens* (ERC-20) é que, além de permitir a realização de

[2]A teoria de jogos, no contexto de *blockchain*, trata de incentivos monetárias com o fim de criar um ambiente onde a participação honesta de um usuário seja a mais vantajosa.

[3]Esta quantitade mínima poderia ser fixa, definida anteriormente, ou pode ser ajustada por uma organização autônoma descentralizada (*DAO*). Porém este processo está fora do escopo deste trabalho

transferências de fundos entre participantes, estes também autorizam contratos inteligentes a realizarem transferências de uma quantidade pré-definida de seus fundos.

3.2.1. Interações entre os Participantes de uma transação comercial

A Figura 2 mostra o exemplo de uma compra. Dois tipos de interações são mostrados na figura. No primeiro, temos as setas contínuas que são usadas nas identificações de chamadas de operações nos contratos gerentes. Estas chamadas são interceptadas pelo quórum de validadores do *Polygon* para as verificações de procedência e de validade, sendo depois executadas nos contratos de destino. As setas tracejadas na figura 2, correspondem a obtenção de notificações via detecção na blockchain *Polygon*.

Figura 2. Fluxo otimista

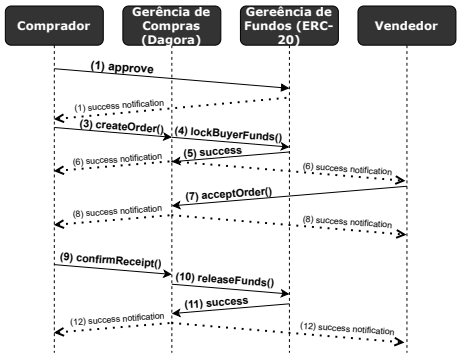
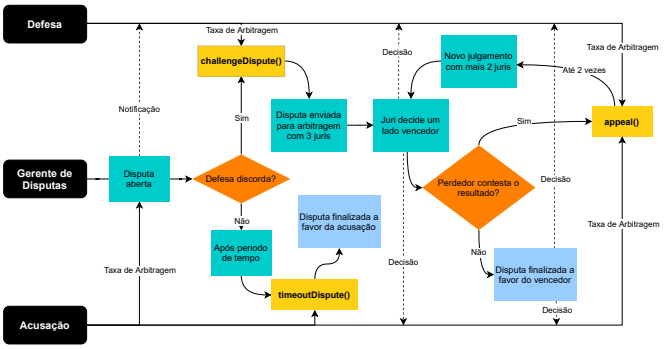


Figura 3. Fluxo de Disputa



O fluxo de interações mostrado na Figura 2 começa inicialmente com um comprador autorizando (através de chamada) o Gerente de Fundos a usar quantia em *tokens DGR* (passo 1). O Gerente de fundos então insere a operação executada (*approve()*) e informações do resultado na *blockchain* (passo 2). Ao detectar (na *blockchain*) a disposição de fundos, o comprador executa a chamada *createOrder()*, mas desta vez dirigida ao contrato Gerente de Compras (passo 3). Na execução desta operação, este contrato faz a chamada *lockBuyerFunds()* no Gerente de Fundos para bloquear fundos do comprador (passo 4). A resposta positiva ou negativa do Gerente de Fundos aparece imediatamente no Gerente de Compras por ser uma chamada entre contratos (passo 5).

Tendo fundos suficientes, o Gerente de Compras coloca a operação *createOrder()* com suas informações na *blockchain* (passo 6). O comprador e o vendedor detectam esta ordem de compra na *blockchain*. O vendedor estando de acordo com a venda faz a chamada *acceptOrder()* no Gerente de Compras (passo 7) que, por sua vez, publica na *blockchain* a operação (*acceptOrder()*) e informações correspondentes (passo 8). Ambos, comprador e vendedor detectam na *blockchain* a ordem aceita e então devem trocar detalhes da entrega do produto por fora do *Dagora* no sentido de manter suas privacidades. O comprador ao receber o produto deve na sequência executar a chamada *confirmReceipt()* no Gerente de Compras (passo 9). Como consequência este contrato invoca *releaseFunds()* no Gerente de Fundos (passo 10). Este último gerente então publica a operação de transferência de fundos para a conta do vendedor na *blockchain*. A partir deste momento os participantes sabem que a transação está concluída.

3.2.2. Resolução de Disputas

Disputas são uma parte importante da plataforma, neste sentido, são definidos meios para resolver conflitos. No *Dagora Market* existem dois tipos de conflitos: denúncia de anúncios irregulares e disputas entre clientes e vendedores. Se um produto anunciado violar as regras, o seu proponente pode ser denunciado por qualquer participante do *Dagora* desde que este último possua uma quantidade mínima de *stake* na plataforma. O anúncio correspondente é então pausado e uma disputa é criada. No segundo tipo de conflito, caso o comprador não receba o produto em período de tempo pré-acordado ou entenda que o produto é não satisfatório, ele pode negociar um reembolso com o vendedor. Se não for atendido, o comprador pode abrir uma disputa. Se a devolução for solicitada, o vendedor tem o mesmo período de confirmação para aceitar a devolução ou, sentindo-se prejudicado, pode abrir uma disputa. Todas as situações de disputas citadas acima são levadas a um júri que irá julgar baseando-se nas regras pré-definidas no *Dagora*, na descrição do produto e nas evidências colocadas pelas duas partes (a acusação e a defesa).

Uma disputa é iniciada por um dos participantes (comprador, vendedor ou terceiros) através do envio da taxa de arbitragem para o contrato inteligente do *Dagora* que chamamos de Gerente de Disputa. O fluxo de disputas é mostrado na Figura 3. Este diagrama de fluxo é autoexplicativo, diante disto nos limitamos a pequenos comentários. No teste "Defesa discorda?", é oferecida a possibilidade da defesa assumir a culpa pela saída do arco "Não", ao esperar o *time-out*. Como resultado a acusação ganha a causa e recebe a devolução da sua taxa[4]. Se a defesa discorda então ela paga uma taxa de arbitragem e um júri de três participantes é montado. Este júri decide e no teste "Perdedor contesta o resultado?" no arco "Sim" existe a possibilidade de apelo sobre a decisão onde são requisitados mais dois juízes para arbitrar a disputa. A seleção do júri é feita de forma aleatória sobre participantes da plataforma de disputas, porém a chance de escolha é fortemente influenciada pela quantidade de *tokens* em *stake* dos participantes. O Gerente de Disputa oferece certos incentivos aos jurados de modo que estes façam um julgamento honesto. Um jurado que decida diferente da maioria pode, por exemplo, perder *tokens* na plataforma. Esta seleção não precisa de um júri altamente qualificado, pois existem *guidelines* previamente definidos. Do mesmo modo, verificar o rastreamento de um produto também pode ser considerado uma tarefa simples. Para Gerente de Disputa do *Dagora* foi usada a plataforma Kleros [Lesaege et al. 2020].

4. Avaliação dos Resultados

Nós criamos os *smart contract* da aplicação, em *Solidity*, com aproximadamente 1100 linhas de código, utilizando *Truffle Suite* [tru 2021], onde implementamos funções necessárias para executar todos os possíveis fluxos de interação entre compradores e vendedores. Entre as funções estão: *stake* e *unstake* de *tokens*, *create, read, update, delete* (CRUD) de anúncios, a criação, execução e finalização de ordens, levantamento e resolução de disputas e denúncia de anúncios.

Estes fluxos, subdividimos em 4 categorias: Fluxo "Bem-Sucedido" onde compras são finalizadas e o dinheiro é enviado para o vendedor; Fluxo "Mal Sucedido" onde o

[4]Uma taxa de arbitragem é cobrada para pagar o processo de disputa, a taxa do lado vencedor é devolvida no fim do processo (resultando no pagamento do processo por parte do perdedor)

dinheiro é devolvido ao comprador; Fluxo “Disputa” onde ocorre algum problema e uma das partes sinaliza uma discordância e o Fluxo de “Denúncia” onde alguém emite um anúncio de comportamento ilegal segundo as regras definidas.

Para nossa avaliação, a *main-chain* é a *Ethereum* que utiliza como *token* principal o *Ether* (ETH) e executa *smart-contracts* através da *Ethereum Virtual Machine* (EVM). A blockchain *Polygon*, considerada como a maior *side-chain* do Ethereum, é capaz de executar contratos inteligentes. Além disso, também usa a *EVM* nas execuções de seus contratos, ou seja, é fácil fazer a portabilidade de código do *Ethereum* para o *Polygon* e vice-versa. Portanto, fazer uma comparação entre estas blockchains nas execuções é praticamente imediata. A *Polygon* utiliza *MATIC* como *token* principal para pagar taxas na rede. Devemos ressaltar que podem existir possíveis custos ao depositar *tokens* do *Ethereum* para a *Polygon*, porém esses custos não foram mapeados, pois, como descrito na seção anterior, podem existir facilidades de venda desses *tokens* por cartão de crédito, bem como o depósito de corretoras de criptomoedas diretamente em *side-chains*.

Figura 4. Uso de *gas* médio por subgrupo de fluxos

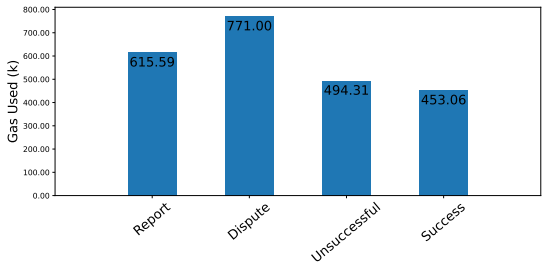
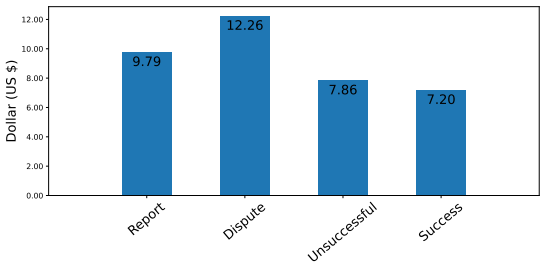


Figura 5. Custo em dólares no *Ethereum*



O custo de execução em *blockchain* depende do total de informação que foi enviada e salva na *blockchain*. Tendo isso em vista, para avaliação de custos foram executados todos os fluxos de interação 10 vezes no *Ganache*[tru 2021] (uma instância de *blockchain* local do *Ethereum*) com diferentes parâmetros de anúncios para calcular o custo médio de execução em *gas* de cada fluxo. Resultando no custo médio de execução para cada subcategoria como representado na Figura 4.

Considerando que ambos, *Ethereum* e o *Polygon* utilizam a *EVM* para execução dos contratos, o custo de *gas* é igual. Devido a isso, o custo final depende de dois fatores: o preço do *gas*, para saber o custo em *tokens*, e o preço do *token*, para saber o custo em moeda fiduciária. Utilizamos a seguinte fórmula para conseguir o preço em dólares:

(custo em *gas* * preço do *gas* em *tokens* * preço do *token* em dólares) / 10^{18}.[5]

Para obter o preço de execução no *Ethereum* em dólares, buscamos o preço do *gas* mediano dos últimos 200 dias[6] resultando em 94.27 GWEI [eth 2021b] ($94.27 * 10^9 wei$). Para buscar o preço em *Ether*, a fim de evitar flutuações, utilizamos a média dos últimos 25 dias chegando ao valor US$1686.31[7]. Obtemos com isso os resultados na Figura 5.

Similarmente, para calcular o preço de execução na *Polygon* em dólares, utilizamos o preço do *gas* como 1 *GWEI* ($1 * 10^9 wei$) que é considerado pela própria rede um

[5]Convertido para a unidade básica do *token* (1 Ether = 10^{18} *wei*)
[6]Preço de *gas* obtido dia 05/03/2021
[7]Preços de *tokens* obtidos dia 17/03/2021

Figura 6. Custo em dolares por subgrupo de fluxos na *Polygon*

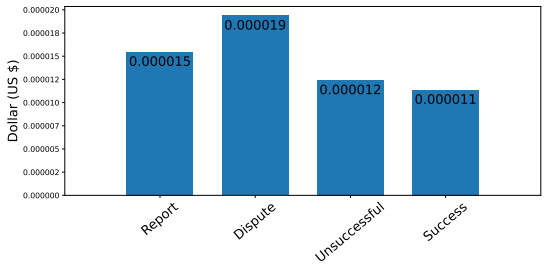
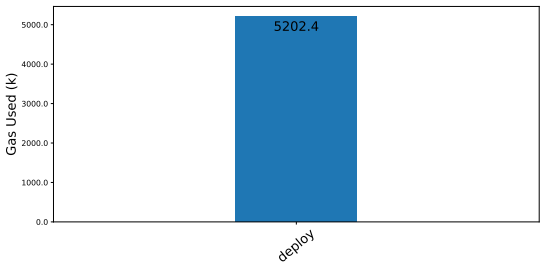


Figura 7. Custo de implantação. *Ethereum* - US$827.02 — *Polygon* - US$0.001308



preço padrão [mat 2021] levando em consideração as últimas 500 transações. Para buscar o preço em *MATIC*, a fim de evitar flutuações, utilizamos a média dos últimos 25 dias que resultou em US$0.25149. A Figura 6 demonstra os valores finais obtidos.

Com os resultados finais em dólares das duas abordagens, pode-se perceber que o custo de executar contratos inteligentes na *Polygon* é cerca de 6 ordens de magnitude menor. Também foram realizados testes para obter o custo da implantação do contrato (Figura 7). Considerando o custo de implantação no *Ethereum* e *Polygon*, o custo de um centavo de dólares no *Polygon* é ínfimo comparado ao custo de US$827 no *Ethereum*.

5. Considerações Gerais

Como mostramos na seção 2.2, existem poucos trabalhos sobre mercados decentralizados baseados em *blockchains*. Dos citados, todos executam diretamente contratos inteligentes no *Ethereum*, sem o uso de camada-2. Poucos destes trabalhos apresentam análise dos custos de uso da *blockchain Ethereum* que consideramos insipientes. Alguns trabalhos acadêmicos [Klems et al. 2017, Ranganthan et al. 2018] apresentam alguma forma de avaliação, porém é importante salientar que estes não foram implementados por completo e as medições apresentadas estão fundamentadas em avaliações já defasadas no tempo, considerando-se os custos do *Ethereum* atualmente.

Neste artigo, apresentamos nosso trabalho de mercado virtual baseado em *side-chain*, um dos objetivos de nossas análises é verificar execuções em *side-chain* com comparações das mesmas execuções no *Ethereum*, pois com isto estaremos relacionando o *Dagora* com os trabalhos de mercado apresentados na literatura que desconsideram processamentos de camada-2).

Existem diversos trabalhos que não tratam mercados virtuais, mas apresentam análise comparativa entre execuções de contratos da aplicação em *side-chains* com execuções realizadas diretamente em *main-chains*, misturando as camadas 1 e 2. Porém estes estudos se limitam em análises qualitativas como o apresentado em [Singh et al. 2020]. Na verdade, uma análise quantitativa e de complexidade de portabilidade da aplicação entre as blockchains é necessária para assegurar os possíveis ganhos nas escolhas de abordagens.

A aplicação de mercado descentralizado apresenta um caso de uso interessante, atualmente grandes mercados online como *Amazon*, *eBay*, MercadoLivre são sites que

centralizam as buscas de compradores e vendedores anunciam seus produtos. Para os vendedores, manter seus anúncios por sites próprios não dá a mesma visibilidade, por esta razão, vendedores profissionais são coagidos a anunciar seus produtos nestas plataformas, sendo que estes *marketplaces* centralizados cobram taxas entre 10% e 15%.

Além disso, em muitas das plataformas, para poder competir com outros vendedores profissionais, os mesmos devem pagar taxas adicionais para ganhar visibilidade através de planos *premium*, e/ou pagar por pacotes para impulsionar seus anúncios. Também existem taxas cobradas pelos sistemas de pagamento, normalmente mediado pela própria plataforma, estes valores variam entre 1% e 5%, dependendo do prazo para o vendedor receber o dinheiro e a forma de pagamento. A união destes fatores citados resulta em quase 20% do preço do produto, sem considerar os impostos. Dessa maneira, perdem ambos, consumidores pagando um preço maior, e vendedores com reduzida faixa de lucro. Todos esses custos poderiam ser reduzidos ao custo de processamento de pagamentos, através de uma plataforma descentralizada, onde vendedores e compradores podem se encontrar sem precisar de uma empresa intermediária.

Com uma possível adoção de mercados descentralizados, alguns problemas são evidentes. Por exemplo, o uso de *tokens* para pagamentos substituindo moeda fiduciária. Hoje, o processo de conversão acontece através de *exchanges* e funciona da seguinte forma: a moeda é depositada na conta da corretora, convertida, e por fim transferida para uma carteira própria na rede escolhida. Além disso, existem alguns serviços, como o *Moonpay*[moo 2021] e *Wyre*[wyr 2021], que permitem o pagamento por cartão de crédito para depósito direto em carteiras de criptomoedas, porém resultando em taxas maiores.

Atualmente não existem corretoras e/ou serviços que depositam diretamente em *side-chains* como a *Polygon*. Resultando em um passo adicional para se obter *tokens* na carteira na *side-chain*. Sendo necessário primeiramente depositar no *Ethereum* para depois transferir para o *Polygon*. Como já mencionado, executar uma transferência no *Ethereum* exige um custo alto, apesar de ser significantemente menor do que realizar a execução do contrato completo no mesmo. Este custo adicional não foi considerado na avaliação, visto que, como mencionado anteriormente, o criador do *marketplace* pode oferecer um *gateway* de transferência de dinheiro (*paypal*, cartão de crédito, etc) para *tokens* DGR no *Polygon*.

Com os dados obtidos, é possível comparar o uso em mercados centralizados e descentralizados. Considerando as taxas cobradas (10% a 20% do preço do produto) nos mercados centralizados executando a compra diretamente no *Ethereum*, percebe-se que qualquer produto acima de US$72.85 resulta em taxas menores do que se estivesse utilizando mercados centralizados. Executando o mesmo processo na *Polygon*, o custo se torna insignificante. Analisando todos os testes realizados, comprovou-se uma grande vantagem ao executar transações na *Polygon*. Para produtos com valores elevados, pode ser vantajoso executar essas transações em uma instância do *Dagora Market* no *Ethereum*, devido ao maior grau de segurança.

6. Conclusões

Este artigo apresenta uma solução prática e completa para a criação de mercados descentralizados na rede *Ethereum* e também na *side-chain* como o *Polygon* com o *Ethereum* como *main-chain*. A partir destes exemplos mostramos que existe uma grande oportu-

nidade de redução de custos (de até 6 ordens de magnitude) em se executar contratos inteligentes de aplicativos em *side-chain*.

As complexidades do desenvolvimento e *deployment* do mercado descentralizado em esquemas de *side-chain* ou diretamente em *main-chain* são idênticas no nosso caso, pois as duas *blockchains* usadas possuem a mesma máquina virtual para execuções de contratos. Com relação a complexidade adicional para o usuário (vendedores e compradores) que não podem depositar e sacar diretamente na *side-chain*, é possível ocultá-la oferecendo um serviço de pagamentos.

Portanto, conforme é possível verificar em nossas avaliações e é importante enfatizar que independente do preço do produto, sempre existe uma grande vantagem tanto para vendedores como para compradores na utilização de um mercado descentralizado.

A complexidade das chamadas dos contratos inteligentes, assim como a interação entre cliente e vendedor podem ser facilmente escondidas nas interfaces de usuários como discutido em [Neiheiser et al. 2020, Neiheiser et al. 2019]).

Agradecimentos Este trabalho teve auxílio financeiro da CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior).

Referências

(2021). Arbitrum. https://offchainlabs.com/. Acesso em 2021-04.

(2021). Ethereum - #2 market cap. https://coinmarketcap.com/pt-br/currencies/ethereum/. Acesso em 2021-03.

(2021a). Ethereum doc. https://ethereum.org/en/developers/docs. Acesso em 2021-03.

(2021b). Ethereum gas price chart. https://etherscan.io/chart/gasprice. Acesso em 2021-03.

(2021). kchannels. https://www.kchannels.io/. Acesso em 2021-04.

(2021). Loopring. https://loopring.org/. Acesso em 2021-04.

(2021). Matic gas station. https://docs.matic.network/docs/develop/tools/matic-gas-station. Acesso em 2021-03.

(2021). Moonpay. https://www.moonpay.com/. Acesso em 2021-04.

(2021). Non-fungible tokens. https://ethereum.org/en/nft/. Acesso em 2021-03.

(2021). Opensea. https://opensea.io/. Acesso em 2021-03.

(2021). Optimism. https://optimism.io/. Acesso em 2021-04.

(2021). Origin protocol. https://www.originprotocol.com/. Acesso em 2021-03.

(2021). Solidity programming language. https://soliditylang.org/. Acesso em 2021-03.

(2021). State channels. https://docs.ethhub.io/ethereum-roadmap/layer-2-scaling/state-channels/. Acesso em 2021-03.

(2021). Truffle suite. https://www.trufflesuite.com/docs. Acesso em 2021-03.

(2021). Wyre. https://www.sendwyre.com/. Acesso em 2021-04.

Adams, H., Zinsmeister, N., Salem, M., Keefer, R., and Robinson, D. (2021). Uniswap v3 core. https://uniswap.org/whitepaper-v3.pdf. Acesso em 2021-03.

Bach, L. M., Mihaljevic, B., and Zagar, M. (2018). Comparative analysis of blockchain consensus algorithms. In *2018 MIPRO*.

Bowe, S., Gabizon, A., and Green, M. D. (2018). A multi-party protocol for constructing the public parameters of the pinocchio zk-snark. In *International Conference on Financial Cryptography and Data Security*, pages 64–77. Springer.

Buterin, V. (2013). Ethereum whitepaper. https://ethereum.org/en/whitepaper/.

Fenu, G., Marchesi, L., Marchesi, M., and Tonelli, R. (2018). The ico phenomenon and its relationships with ethereum smart contract environment. In *2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, pages 26–32.

Foxley, W. (2021). Ethereum's top dapps are increasingly turning to 'rollups': Here's why. https://www.coindesk.com/ethereum-dapps-rollups-heres-why. Acesso em 2021-04.

Kanani, J., Nailwal, S., and Arjun, A. (2019). Matic whitepaper. https://github.com/maticnetwork/whitepaper. Acesso em 2021-03.

Klems, M., Eberhardt, J., Tai, S., Härtlein, S., Buchholz, S., Tidjani, A., Vallecillo, A., Wang, J., and Oriol, M. (2017). Trustless intermediation in blockchain-based decentralized service marketplaces. In *Service-Oriented Computing*.

Lesaege, C., George, W., and Ast, F. (2020). Kleros. https://kleros.io/static/whitepaper_en-8bd3a0480b45c39899787e17049ded26.pdf.

Liu, Z., Luong, N. C., Wang, W., Niyato, D., Wang, P., Liang, Y., and Kim, D. I. (2019). A survey on applications of game theory in blockchain. *CoRR*, abs/1902.10865.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. http://www.bitcoin.org/bitcoin.pdf.

Neiheiser, R., Inácio, G., Rech, L., and Fraga, J. (2020). Hrm smart contracts on the blockchain: emulated vs native. *Cluster Computing*.

Neiheiser, R., Inácio, G., Rech, L., and Fraga, J. (2019). Hrm smart contracts on the blockchain. In *2019 IEEE Symposium on Computers and Communications (ISCC)*.

Ranganthan, V. P., Dantu, R., Paul, A., Mears, P., and Morozov, K. (2018). A decentralized marketplace application on the ethereum blockchain. In *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pages 90–97.

Rozen, J. (2021). The ridiculously high cost of gas on ethereum. https://coingeek.com/the-ridiculously-high-cost-of-gas-on-ethereum/. Acesso em 2021-04.

Singh, A., Click, K., Parizi, R. M., Zhang, Q., Dehghantanha, A., and Choo, K.-K. R. (2020). Sidechain technologies in blockchain networks: An examination and state-of-the-art review. *Journal of Network and Computer Applications*, 149:102471.

Szabo, N. (1997). Formalizing and securing relationships on public networks. *First Monday Journal*.