Estimating (Miner) Extractable Value is Hard, Let's Go Shopping!

Aljosha Judmayer[1,2], Nicholas Stifter[1,2], Philipp Schindler[1], and Edgar Weippl[2]

[1] SBA Research, Vienna, Austria
`(firstletterfirstname)(lastname)@sba-research.org`
[2] University of Vienna, Vienna, Austria `{firstname.lastname}@univie.ac.at`

Abstract. The term *miner extractable value* (MEV) has been coined to describe the value which can be extracted by a miner, e.g., from manipulating the order of transactions within a given timeframe. MEV has been deemed an important factor to assess the overall economic stability of a cryptocurrency. This stability also influences the economically rational choice of the security parameter k, by which a merchant defines the number of required confirmation blocks in cryptocurrencies based on Nakamoto consensus. Unfortunately, although being actively discussed within the cryptocurrency community, no exact definition of MEV was given when the term was originally introduced. In this paper, we outline the difficulties in defining different forms of extractable value, informally used throughout the community. We show that there is no globally unique MEV/EV which can readily be determined, and that a narrow definition of MEV fails to capture the extractable value of other actors like *users*, or the probabilistic nature of permissionless cryptocurrencies. We describe an approach to estimate the *minimum* extractable value that would incentivize actors to act maliciously and thus can potentially lead to consensus instability. We further highlight why it is hard, or even impossible, to precisely determine the extractable value of other participants, considering the uncertainties in real world systems. Finally, we outline a peculiar yet straightforward technique for choosing the individual security parameter k, which can act as a workaround to transfer the risk of an insufficiently chosen k to another merchant.

Keywords: Miner Extractable Value · Extractable Value · Expected Extractable Value · Cryptocurrencies · Game Theory

1 Introduction

The term *miner extractable value* was first introduced by Daian et al. [11] to refer to the value which can be extracted by a miner from manipulating the order of transactions within the blocks the respective miner creates. This ability can be used for *front running* attacks [13], which can lead to guaranteed profits through token arbitrage, or other related types of attacks like *back-running* and combinations thereof [33]. Such attacks, which exploit the ability of miners to arbitrarily order transactions within their blocks, are feasible and currently observed in practice [44,46,40,45]. Thus, order fairness evidently poses an issue for prevalent permissionless PoW cryptocurrencies [26]. When the stakes are sufficiently high, MEV can even incentivize blockchain forks and thereby also have consequences not only on transactions in future blocks, but also for the underlying *consensus-layer security*, as also noted by Daian et al. [11] and Zhou et al. [44]. Related attacks aimed in this direction are *undercutting attacks* [9], *time-bandit attacks* [11], or more broadly: Attacks involving economic incentives in general, such as any form of *bribing attack* [5,29,6,28], which have been summarized under the term *algorithmic incentive manipulation* [24].

Given all these attacks and their far-reaching consequences, MEV undoubtedly is an essential concept when reasoning about economic stability aspects and cryptocurrency security under economical considerations. Recently, a related term called *blockchain extractable value* [33] (BEV) was introduced. BEV refers to the value extractable by different forms of front running attacks which are not necessarily performed by miners, but *users*. Unfortunately, in the case of MEV as well as in the case of BEV, no exact definition was given by the authors when the term was originally introduced [3].

As the concept of MEV/BEV is tied to the economic incentives of whether or not to fork a certain block/chain [44], this question also relates to economic considerations regarding the choice of the personal security parameter k of merchants. An accurate estimation of the overall MEV/BEV value, would allow to adapt and increase k accordingly in periods of high overall MEV/BEV. The choice of the security parameter k, which determines the number of required confirmation blocks until a payment can safely be considered confirmed, has been studied in a variety of works. Rosenfeld [35] showed that, although waiting for more confirmations exponentially decreases the probability of successful attacks, no amount of confirmations will reduce the success rate of attacks to 0 in the probabilistic security model of PoW, and that there is nothing special about the often-cited figure of $k = 6$ confirmations. Sompolinsky and Zohar [38] defined different acceptance policies with different error probabilities and use-cases. According to [38] an acceptance policy that is resilient to a double spend anywhere in the chain cannot rely on a static parameter k, but has to be logarithmic in the chain's current length. Garay, Kiayias and Leonardos [18] defined the security parameter k, after which a transaction can be considered part of the *common prefix*, as a function of the security parameter of the hash function (κ), the typical number of consecutive rounds for which a statement would hold, and the probability of at least one honest party finding a valid PoW in a round. In spite of these well-founded theoretical results under honest majority assumptions, in practice there is no global and agreed-upon security parameter k for prevalent PoW cryptocurrencies. Instead merchants choose their individual k on a best-practice basis, taking their individual economical risk into account. Since the likelihood of potential forks increases when sufficiently large extraction

[3] Concurrent work [3,31], also attempts to initially define different forms of *extractable value* and is compared to the paper at hand in Appendix C and D.

opportunities for miners arise, the question on how to define and estimate the extractable value of a block (or a chain) also relates to the choice of the personal security parameter k. This has also been studied in the context of bribing, especially in the *whale attack* [28], where large fees are offered on competing chains, as well as in context of incentives for miners to fork high value blocks when the majority for the block reward comes in form of transaction fees [9,41].

Contribution: In this paper we *describe different forms of extractable value* and how they relate to each other. Furthermore, we outline a series of observations which highlight the *difficulties in defining these different forms of extractable value* and why a generic and thorough definition of it (and thus its precise calculation) is *impossible for permissionless cryptocurrencies* without assuming bounds regarding all available resources (e.g., other cryptocurrencies) that are, or could be of relevance for economically rational players. We also describe a way to estimate the minimum extractable value, measured in multiples of normalized block rewards of a reference resource, to incentivize adversarial behaviour of participants which can lead to consensus instability[4]. In the end, we propose a peculiar yet straightforward technique for choosing the personal security parameters k regardless of extractable value opportunities.

2 Economic Rationality and Extractable Value

Rationality depends on the criteria which should be optimized. This could be reward in terms of cryptocurrency units in some PoW cryptocurrency, or a more abstract criteria such as the overall robustness of the cryptocurrency ecosystem. We start out with a simplified definition of *economic rationality* to model the preferences of *actors*, or *parties*[5] within the system. Hereby, actors are divided into two disjoint sets: *miners* ($\mathcal{M}$) and *users* ($\mathcal{U}$), where $\mathcal{M} \cup \mathcal{U} = \mathcal{P}$. Compared to miners, users do not having any direct voting power in the system (e.g., hashrate). Whenever, we refer to rational within this work, we refer to the definition of *economically rational in* $\mathcal{R}$:

Definition 1 (Economically rational in $\mathcal{R}$). *An actor (i) is* ***economically rational****, with respect to a finite non-empty set of resources $\mathcal{R}_i \coloneqq \{R_0, R_1, \dots\}$, when it is his single aim to maximize his profits measured in these resources. To also map the individual preferences of an actor regarding this set of resources, the quantities of all resources from that actor are converted into* ***value units****. Therefore, from the perspective of an actor i each resource R is a tuple $\langle r, e\rangle$ consisting of: The quantity r the actor i holds in the respective resource; and the individual exchange rate e for the conversion in value units which reflect the individual preference of that actor.*

A quantity of a certain resource $\langle r, e\rangle$, which is optimized by a rational actor (indexed by i), is denoted as $f_i(r, e)$. This function returns the *value units*, also

[4] All source code and artefacts can be found on GitHub `https://github.com/kernoelpanic/estimatingMEVishard_artefacts`

[5] Also the term *players* is commonly used to refer to the involved parties.

referred to as *funds*, actor i has in r, calculated using his individual exchange rate e for r. If the exchange rates are the same for every party, or clear from the context, they can also be omitted. For practical purposes and to aid comparability, we will use *normalized block rewards* of a reference resource (e.g., Bitcoin block rewards including average fees) as *value unit* in which all funds are denoted. Therefore, all exchange rates of other resources convert their respective quantities into normalized block rewards of the reference resource.

In other words, value units can also be thought of as fiat currency, which in turn can be received in exchange for cryptocurrency units. In this paper, value units are measured in multiples of the block reward of a reference cryptocurrency, which is the first resource in $\mathcal{R}_i$ (i.e., R_0). A resource can be anything of value to an actor e.g., a cryptocurrency, a token within a cryptocurrency, or a fiat currency. If not stated differently, all parties care about the same set of resources and the exchange rate for each resource is globally defined (e.g., by an exchange service) and thus the same for every actor. This means, for the simplest case where all parties only care about one resource and have the same global exchange rate (e_{global}), we have: $\forall i \in \mathcal{P}\,(|\mathcal{R}_i| = 1 \wedge e \in R_0 \wedge e = e_{global})$, where the valuation of an actor i is given by $f_i(r, e_{global})$, or abbreviated just $f_i(r)$. We start our evaluation with such a scenario.

Summing up: By our definition of economic rationality actors want to maximize their overall funds (valuation) from all their resources, which are measured in value units, i.e., $f_i(\mathcal{R}_i) = \sum_{\langle r_j, e_j \rangle \in \mathcal{R}} f_i(r_j, e_j)$.

2.1 Miner Extractable Value

To calculate the gain or profit an actor has made within a chain of blocks c, with their sequence of transactions τ, it is essential to estimate the costs as well as the extractable value for the respective actor. This has been done in previous works [11,33] mostly by analyzing past Ethereum blocks with their associated transactions, while looking for profitable trades on the blockchain in retrospect. The gathered data was then also used when analyzing how to automatically detect and exploit such trading situations [44,46]. In this context, the term *miner extractable value* (first introduced by Daian et al. [11]) was informally used to describe the value which can be extracted by a miner by including a certain transaction in terms of fees, or guaranteed profits through token arbitrage. We now provide a definition within the context of our model and in accordance to the literature. Therefore, we first focus on a scenario where we assume that there is only one resource, as well as one exchange rate, which is the same for every actor.

Definition 2 (Miner Extractable Value MEV$_i(c)$). *The miner-extractable value* $\mathrm{MEV}_i(c)$*, describes the total value (denominated in value units), transferred to a* ***miner*** *i from a sequence of transactions τ, included and thus mined in the respective chain of blocks c, which is part of the main chain.*

This definition focuses on identifying the extractable value in retrospect and is thus suited for empirically analyzing the MEV of historic blocks. We extend this to a more forward looking definition later in Section 2.3, where we also take the probabilistic nature of most cryptocurrencies into account.

Regardless of the time when the MEV is determined, there are couple of characteristics all types of MEV have in common. First, the total extractable value depends on the type of optimization the miner i is performing. If transactions are only ordered by fee, the miner extractable value can be expressed as the "usual" mining reward from fees and block rewards i.e., $\text{MEV}_i(c) \coloneqq \textsc{Fee}(c) + \textsc{BlockReward}(c)$. If there is a value gain from received transactions, or rewards from performed token arbitrage and order optimizations, then the respective income opportunities, e.g., income from received transactions and attacks, have to be taken into account as well. In other words, a general definition of MEV describes the value (i.e., the reward) which can be extracted by a miner, extended by additional revenue opportunities originating from the capabilities to interact with the system the miner is tasked to validate. This leads to the question: What *possibilities* to extract value are available to miners?

This question already outlines why the concrete amount of MEV is difficult to generalize for all miners, as it can be different for every individual miner. The MEV of a given miner i depends on the type of value extraction optimization the respective miner is capable and willing to perform. Hereby, the possibilities reach from simple fee optimization techniques, like selecting the transactions which provide the highest fees, over order optimization (for example, attempting to maximize gas consumption in smart contracts[6]), to participating in sophisticated front running attacks. Moreover the MEV is affected by the information available to the miner (e.g., her view of the transaction pool). Therefore, we can make the following observation:

Observeration 1. The miner extractable value (MEV) is different for every miner depending on the optimization techniques the miner is capable and willing to perform, as well as other transactions which directly affect her individual revenue during the respective period (e.g., received funds).

In other words, there may exist a miner m_1 that has a higher MEV than a miner m_2 because he has received a large incoming payment transaction in the same sequence of transactions $\tau \in c$, i.e., $\text{MEV}_{m_1}(c) > \text{MEV}_{m_2}(c)$. Therefore, miner m_2 may be more willing to participate in an attack which changes the past blocks c than the miner m_1.

2.2 Extractable Value

Since not all attacks (or more generally, ways to maximize profit) necessarily require the capabilities of a miner, the given definition of MEV does not capture

[6] Note that, a naive algorithm for finding the optimal ordering of all transactions is factorial in the number of transactions, which is computationally infeasible even for most current Ethereum blocks which have more than 200 transactions.

these. front running attacks for example, can be performed by actors which do not necessarily have to be miners themselves, but can be *users* instead. From their perspective the definition of MEV does not apply, as they are not capable of mining a block on their own (as they have zero hashrate).

Observeration 2. The definition of MEV is focused on *miners* and does not capture opportunities for *users* to gain (more) value units from a certain sequence of transactions τ than from another sequence of transactions τ'.

In contrast to MEV, *blockchain extractable value* (BEV) [33], was previously described in a broader context and thus also refers to the value extractable by different forms of front running attacks which are not necessarily performed by miners. As recent analysis [11,33,44,46,40] show, front running is performed by bots, which bid for an early slot in a block by raising the miner extractable transaction fee[7]. However, these earlier discussions regarding BEV [33] omit a precise definition, which we provide in the general form of *extractable value* (EV_i), for the amount that is transferred to any actor i from a given blockchain c, or sequence of transactions τ.

Definition 3 (Extractable Value $\mathbf{EV}_i(\cdot)$). *The extractable value* $\mathrm{EV}_i(\cdot)$*, describes the total value, which is transferred to actor i from a transaction, or sequence of transactions τ, if it is included and thus mined in the respective chain of blocks c, which is part of the main chain.*

Using this definition of extractable value, the miner extractable value can also be defined as $\mathrm{EV}_i(c)$ of any miner $i \in \mathcal{M}$. As with miner extractable value, the extractable value of different actors i and j for the same chain of blocks c can also be different. Again $\mathrm{EV}_i(c)$ and $\mathrm{EV}_j(c)$ depend on whether they have received, or sent, transactions within this chain or not.

In other words, observation 2 shows that MEV is just a way to extract value which can be executed by a subset of actors i.e., miners. Since we also know from observation 1 that there is no globally unique MEV, which is the same for all miners, the same argument can also be extended to EV. So there also cannot be a globally unique EV, that is the same for all actors. The question is if the EV can be meaningfully estimated, or bounded? Assume we have two parties i and j, where j wants to estimate the extractable value of i for a certain chain c. If j wants to estimate $\mathrm{EV}_i(c)$, then this means j has to attribute all transactions generating value for i in c correctly. If the respective actor i pseudonymously performs and receives transactions under various different addresses (or in other privacy preserving ways like shielded transactions in Zcash [25,20]), this hampers the correct estimation of $\mathrm{EV}_i(c)$ for any third party j which does not know which transactions belong to a certain actor.

[7] In Ethereuem the extractable fee is a combination of `gasPrice` multiplied by `gasUsed`.

Observeration 3. If there are transactions in c that are not uniquely attributable to other parties from the perspective of actor j, then the upper bound of the $\mathrm{EV}_i(c)$ for a certain actor i is the total value transferred by those non-attributable transactions.

This means, for known finite chains of blocks in certain cryptocurrencies where the overall value that has been transferred is observable, the extractable value can be upper-bounded in retrospect as soon as the respective chain is known.

Note that, even if all transactions can be correctly attributed to an actor, the exact effect of an outgoing transaction on the EV in a smart contract capable cryptocurrency is still difficult to measure. Generally, all outgoing transactions reduce the EV, as the miner loses funds. Although, if the outgoing transaction is a guaranteed profit token arbitrage, or other profitable type of front running, the EV might very well increase. We omit the details of analyzing the EV of a particular transaction in a smart contract capable cryptocurrency and refer to a strain of research dealing with this topic [11,44,46,40,45].

2.3 Expected Extractable Value

So far we have only considered the extractable value of past blocks in retrospect ($\mathrm{MEV}_i(c)$), or blocks and transactions under the assumption that they eventually will make it into the main chain ($\mathrm{EV}_i(c)$, or $\mathrm{EV}_i(\tau)$). Hereby, we did not account for the probability with which the estimated value can be realized. As mining in prevalent cryptocurrencies is a stochastic process, getting a certain chain accepted into the common-prefix depends on several factors - one of which being the hashrate that supports a given chain[8]. Therefore, it is more appropriate to refer to the *expected extractable value* (EEV) when comparing potential/future rewards of mining strategies, pending blocks, or forks.

Definition 4 (Expected Extractable Value $\mathbf{EEV}_i(\cdot)$). *The expected extractable value* $\mathrm{EEV}_i(\cdot)$*, describes the total value in value units, which is transferred to actor i on expectation using a certain strategy which produces a transaction, sequence of transactions (τ), or blocks (c) that later become part of the main chain with some probability.*

To maximize the EEV, actors will pick an according strategy. Hereby, the probability space depends on the concrete model under which a given strategy is analyzed.

Observeration 4. In permissionless PoW cryptocurrencies, the number of participants $|\mathcal{P}|$ is theoretically unbounded, as miners can join and leave at any time. Thus the sample space of possible events is theoretically unbounded as well.

Therefore, certain assumptions e.g., regarding the available resources of (potential) players are required, to bound their influence (i.e., action space) on a given cryptocurrency.

[8] Another one being propagation times, but we will ignore that for now.

We now describe a simple idealized system model and strategy and use it as a reference to compare deviating strategies and changes in the model against this setting. We therefore, assume that there is only one resource of interest and no further actors join the cryptocurrency.

Definition 5 (The strategy Honest). *We define the strategy* HONEST *for miners in a cryptocurrency R, as the process of always extending the currently known longest (heaviest) chain and immediately publishing and forwarding every found block and transaction.*

If every miner plays HONEST and has a constant hashrate, then this results in an infinitely repeated game, in which every miner receives exactly the reward that is proportional to his hashrate. Therefore, HONEST satisfies *ideal chain quality* [16], as the percentage of blocks in the blockchain of every actor is exactly proportional to their individual hashing power[9]. We assume that this is the desired ideal state in which the system should be and thus the goal of the mechanism design. Moreover, it is more-or-less the empirically observed behavior of miners as serious deviations are rarely observed in mainstream cryptocurrencies[10].

Assume a miner i that does not receive or send any transactions (apart from collecting rewards from mined blocks), and that the extractable value is given in normalized block rewards (including fees) as a value unit. Then if everybody plays HONEST, the strategy HONEST for a sequence of n blocks, would have the EEV depicted in equation 1. The strategy would be profitable if the mining costs ($costs_{mining}$) for mining the respective number of blocks is lower than the EEV. This also assumes that the hashrate (p_i) of actor i is common knowledge and static for the duration of the evaluation.

$$\mathrm{EEV}_i\left(R, \textsc{Honest}, n\right) \coloneqq n \cdot p_i \tag{1}$$

$$\rho_i \coloneqq \mathrm{EEV}_i\left(R, \textsc{Honest}\right) - costs_{mining} \cdot n \tag{2}$$

If the respective actor i also performs and receives transactions, and all of them are uniquely attributable to i, then expected extractable value has to be extended by the extractable value from those transactions. As we are in a scenario where all actors act HONEST, no malicious forks[11] will happen.

Apart from such simple toy examples, estimating the expected extractable value becomes more involved, as soon as different attacks and their probabilities and consequences should be captured. As there are plenty of possible attacks, we

[9] Note that, in a model with constant hashrate and difficulty, deviations like selfish mining [14], only increase the relative reward of an actor compared to others and not the absolute reward over time [37,30]. So in a constant difficulty model, selfish mining would not be more profitable over time than ordinary mining. This observation also holds in a model with variable difficulty until the difficulty is adjusted. In Bitcoin for example, this happens roughly every two weeks (2016 blocks).

[10] As an analysis of Bitcoin shows [21], miners more-or-less stick to the rules despite preferring transactions with higher fees and smaller blocks for faster propagation

[11] With the simplifying assumption that no blocks are found concurrently.

cannot cover them all in this paper and refer to the related research on estimating the success probability in such cases [35,38,46,19,28]. For the rest of the paper we focus on the potential economic consequences of large scale attacks and their relation to the minimum EEV (given in normalized block rewards) required to incentivize deviating strategies that could lead to consensus instability while accounting for *all* future rewards.

3 Estimation of the min. EEV in the Context of Attacks

We now look at the question how much EEV for a participant can lead to consensus instability. Thus we have to investigate the EEV in presence of attacks and especially their economic consequences. Therefore, we view the cryptocurrency R from a game theoretic standpoint and model it as an infinitely repeated game. But first we describe how the EEV can be used to compare different strategies against each other. The question whether any attack strategy is profitable for some actor i, can be summarized by comparing the EEV as well as the costs of the attack against the behavior intended by the protocol designer, i.e., the strategy Honest, for that actor.

$$\mathrm{EEV}_i\left(R, \textsc{Attack}\right) - costs_{\textsc{Attack}} > \mathrm{EEV}_i\left(R, \textsc{Honest}\right) - costs_{\textsc{Honest}} \quad (3)$$

In other words, if a deviation form the Honest strategy is more profitable, then this strategy is economically rational. Here the costs can also incorporate potential losses of value of already accumulated resources due to negative consequences of the attack on the exchange rate of those resources. The security and incentive compatibility of a cryptocurrency, against an attack strategy Attack, can thus be ensured if the following condition in Formula 3 holds at any time for all actors. Formula 3 can also be described as a version of the formula provided by Böhme in a presentation [2].

Observeration 5. The expected extractable value, as well as the utility of an actor, describe the achievable gain from a certain strategy. Thus the utility of an actor as well as the EEV are equivalent and can be used as synonyms.

In our model a side-payment, or “bribe”, can be expressed as part of the EEV e.g., as an incoming payment that is only valid on the chain desired by the attacker (hence conditional). This illustrates that (side-)payments can influence the incentives of actors. If the EEV of an attack is large enough to overcompensate for the induced costs, an attacker can use a portion of his profit to bribe other miners to convince them to mine on the attack chain. Using side-payments any economically rational actor can be incentivized to support an attack. The question directly related to EEV is: How large does such a side-payment have to be to incentivize illicit activity of other actors. To address this question we also have to take into account potential future EEV (or payoffs/rewards in terms of game theory) and the reduction of such, in case of an event that reduces the exchange rate for the attacked resource, i.e., a value loss.

3.1 Single Resource (R)

We now compare potential future EEV and thus the overall payoff for different strategies. From a game-theoretic point of view, we model a cryptocurrency as an infinitely repeated game with discounting[12]. Therefore, we have to define a *discount factor* $\delta \in (0, 1)$, which specifies the preference of either immediate or future rewards. If δ is close to 0 immediate rewards are preferred. If δ is close to 1 future rewards are almost as good as immediate rewards. If $\delta = 0$ we would have a single-shot game as there would not be any future reward. To account for mining shares and δ, we have to extend our definition of a resource:

Definition 6 (A resource R). *From the perspective of an actor i each resource R is a quadruple $\langle r, e, p, \delta \rangle$ consisting of: The quantity r the actor i holds in the respective resource. The exchange rate e for the conversion in value units which reflect the individual preference of that actor. A parameter p which represents the power (e.g., hashrate) of that actor in this resource, which is used together with a discount factor δ to denote expected future rewards in that resource.*

As the payoffs in an infinite game create a geometric series $(p + p \cdot \delta + p \cdot \delta^2 + p \cdot \delta^3 \dots)$, the payoff for the first n rounds can be written as:

$$r_n := \frac{p_i \cdot (1 - \delta^n)}{1 - \delta} \tag{4}$$

This can be rewritten as a closed form formula for the infinite case since δ^n goes to 0 as n goes to infinity. Thus the EEV for a single actor i with hashrate $p_i \leq 1$ in our infinite game, where every actor plays Honest, can be approximated by:

$$\mathrm{EEV}_i(R, \textsc{Honest}, \infty) := \frac{p_i}{1 - \delta} \tag{5}$$

This estimation again denotes the EEV in normalized block rewards as a value unit (with $e = 1$) and assumes that the hashrate of actor i remains static in relation to the hashrates of all other actors.

We now compare this payoff to another strategy which requires a different (attack) action once and then falls back to the original honest behavior, but with a potential negative consequence on future rewards as the exchange rate has dropped. This is comparable to a *grim trigger* strategy in infinitely repeated games, although in our case the environment executes the grim trigger strategy by devaluing the global exchange rate ($e < 1$).

In our scenario, ε is the one-time side-payment to motivate the deviation and e is the value loss in terms of a drop in exchange rate, which of course also has the same negative impact on future EEV and thus must also be accounted for in all potential future mining rewards if the loss is (in the worst case) permanent.

$$\mathrm{EEV}_i(R, \textsc{Attack}, \infty) := \varepsilon + \frac{\delta \cdot p_i \cdot e}{1 - \delta} \tag{6}$$

[12] Pass et al. [32] pointed out that PoW blockchains cannot stop without becoming insecure, so they have to run infinitely long.

As we are only interested in an approximation, we abstract the particular success probability calculations to evaluate the likelihood of a single attack being successful. Furthermore, we omit the loss of blocks a miner potentially faces if the chain he contributed to becomes stale. If known, this value can be included by adding it to the required bribe ε.

We now estimate how high this one-time side-payment ε has to be to incentivize a one-time deviation form the Honest strategy with permanent consequence on e for a mainstream cryptocurrency. Therefore, we first have to define some plausible range for the discount factor δ miners might have in practice. Figure 1 shows the normalized block reward after a certain number of passed blocks for different values of δ and a hashrate of $p = 0.1$. It can be observed that a relatively high value $\delta = 0.99995$ is needed already to approximate (within a 5% margin) the average income in normalized block rewards after one Bitcoin difficulty period (2016 blocks). For a far sighted miner that has a one to two year interest in Bitcoin a $\delta = 0.999999$ would suffice to be within a margin of 5% of the average number of normalized block rewards after two years.

Now that we have picked some plausible values for δ, we can approximate the required total side-payment ε that would be required to change the incentives of participating miners with different hashrates. Therefore, we compare the EEV of honest behavior with the EEV of the attack. Assuming the costs of mining in both cases are identical, the EEV of the attack has to be more profitable than the honest behavior, for the attack to be realistic.

So far we have not taken into account that miners could also hold funds ($f_i(r)$), which are distinct from hashrate (which describes future gains given out as currency units). Since a successful attack will lead to a potential drop in the exchange rate, we have to consider this for all future rewards, as well as all funds the miner is currently holding. Equation 9 compares the two strategies under the assumption that there is only one resource (r) the respective miner cares about. Hereby, the hashrate is viewed as some share in the protocol which provides future rewards in the respective cryptocurrency (in r) proportional to the size of the share.

$$\mathrm{EEV}_i(R, \textsc{Honest}, \infty) := \frac{p}{1-\delta} + f_i(r) \tag{7}$$

$$\mathrm{EEV}_i(R, \textsc{Attack}, \infty) := \varepsilon + \frac{\delta \cdot p \cdot e}{1-\delta} + f_i(r, e) \tag{8}$$

$$\mathrm{EEV}_i(R, \textsc{Honest}, \infty) < \mathrm{EEV}_i(R, \textsc{Attack}, \infty) \tag{9}$$

Solving equation 9 for ε, we can calculate the required side-payment for the following example: To compensate a five percent value drop ($e = 0.95$) for a miner with zero funds ($f_i(r) = 0$) and 10% hashrate ($p = 0.1$), a side-payment in approximately the size of 500 times the normalized block reward is needed (if $\delta = 0.99999$). If the side-payment itself is performed in r, and thus subject to the same value drop of 5% as well, then ≈ 527 times the normalized block reward is required as a side-payment.

Although theoretically possible, such high bribes in the size of hundreds of normalized block rewards appear unlikely in practice from a current stand point. Moreover, for an attack to be economically viable for an attacker, he would have to perform a double-spend of a transaction which is much larger than the required overall side-payments. Ideally the attacker himself (as well as the victim) does not possess any hashrate in the targeted cryptocurrency, such that his personal future income will not be negatively affected by the consequences of the attack, i.e., drop in exchange rate. Moreover, the attacker is advised to use all his funds in r in the double-spend transaction to further minimize the negative effects on the exchange rate. The leftovers from the double-spend, after subtracting the required side-payments to incentivize a sufficient portion of the hashrate to support the attack chain, could be viewed as profit for the attacker. So for such an attack to work, funds would have to be unevenly distributed amongst actors and an individual payment must only be limited by the available overall supply of the respective resource. Then the amount of a double-spend can theoretically be high enough that the excess profit of the attacker can be used to bribe a majority of miners to support an attack chain.

Observeration 6. In a scenario where there is only one cryptocurrency participants care about, the side-payment necessary to incentivize a deviation has to account for all current and future losses.

3.2 Multiple Resources ($\mathcal{R}$)

The analysis so far assumed that all actors only care about the same single resource, i.e., cryptocurrency, and express their extractable value in normalized block rewards of this resource. The resulting question is, what if there are multiple resources and not all actors necessarily care about the same set of resources to a comparable degree?

To approach this question, we modify equation 9 to estimate the EEV for the honest strategy, as well as for the attack strategy, by accounting for all resources (e.g., cryptocurrencies) a player i cares about. Hereby, we model the individual hashrates (p) as a part of each resource which defines the share of future rewards an actor will receive in the respective resource. Additionally there is a set of δ values for each resource. Moreover, in this scenario a bribe does not necessarily have to be paid in the resource where the attack action should happen, thus several bribes are possible $\{\varepsilon_0, \varepsilon_1, \varepsilon_2, \dots\}$.

$$\mathrm{EEV}_i(\mathcal{R}, \textsc{Honest}, \infty) := \sum_{j=0}^{|\mathcal{R}|} \left(\frac{p_j}{1-\delta_j} \right) + \sum_{j=0}^{|\mathcal{R}|} f_i(r_j) \tag{10}$$

$$\mathrm{EEV}_i(\mathcal{R}, \textsc{Attack}, \infty) := f_i(r_0 + \varepsilon_0, e_0) + \frac{\delta_0 \cdot p_0 \cdot e_0}{1-\delta_0} \tag{11}$$

$$+ \sum_{j=1}^{|\mathcal{R}|} \left(\frac{p_j \cdot e_j}{1-\delta_j} \right) + \sum_{j=1}^{|\mathcal{R}|} f_i(r_j + \varepsilon_j, e_j) \tag{12}$$

Compared to the single resource case in Section 3.1 the multi-resource case now allows actors to escape certain negative consequences on the exchange rate e_0, e.g., by moving their hashrate to another permissionless PoW cryptocurrency (like for example R_1). This of course only works if the miner's hardware can also be efficiently used in the other cryptocurrency and e_1 is not affected to the same degree, or generally much higher to begin with ($e_1 \geq e_0$ and $\delta_1 \geq \delta_0$). If also current holdings in resources can be transferred to other resources through exchange services, then in the worst case it may be possible to evade all negative consequences from attacks on a certain target resource R_0. To which degree such negative consequences can be evaded depends on several factors: The availability of adequate alternatives, the type of the attack, as well as how fast resources can be moved and exchange rates adopt (cf. [4]).

If such evasion techniques are possible, this raises an interesting question from a game theoretic point of view. The previously infinitely repeated game, now becomes a finite game, as actors can leave the system at will. Therefore, the option to defect in the (personally) last round of the (now) finite game suddenly becomes an economically rational strategy.

Observeration 7. If appropriate alternative resources exist, parties can evade negative attack consequences on their overall EEV, by moving their assets to another less, or even positively affected resource. Thereby, the once infinite game becomes finite from their perspective.

We provide some visual examples for this multi-resource model in Appendix E, by comparing the EEV before a certain event with the EEV after the event. These examples further illustrate, that miners who are tied to a cryptocurrency due to their specific mining hardware, have a higher incentive not to risk negative consequences on the exchange rate of that cryptocurrency. If switching to another equally, or more profitable alternative is possible though, attacks become more attractive. This highlights, that in an environment in which multiple cryptocurrencies co-exist and represent alternative resources to each other, the EEV cannot be estimated by looking at a single resource/cryptocurrency alone. Especially since cryptocurrencies can be created at any point in time, for example through forks, which change the overall cryptocurrency landscape and potentially affect the exchange rates of existing cryptocurrencies in one, or the other way.

Observeration 8. In the multi-cryptocurrency environment where new resources can be created, the EEV can be influenced by these new resources, since the set of available resources for actors changes. Thereby, providing new alternatives, or modifying existing exchange rates and discount factors.

This problem of considering out-of-band income streams in economic security models of permissionless PoW cryptocurrencies, is also nicely illustrated by various bribing, or algorithmic incentive manipulation attacks which utilize out-of-band payments [5,39,29,23,24] as well as other economic arguments regarding the incentive structure of such systems [15,8].

4 Discussion

The presented observations highlight that accurately estimating the EEV of a particular miner is impossible, even when knowing all transactions belonging to this miner, as well as all current preferences regarding cryptocurrencies and resources the respective miner cares about and to which degree. There are two major reasons for this: First, it is not possible to predict the actions taken by other actors interacting with the system by issuing transactions which either directly, or indirectly affect the respective miner i, or the exchange rate of a resource. Second, if miner i is open for accepting payments or new resources (e.g., validator roles which provide future incomes), then the fact that new cryptocurrencies can be forked, or created at any point in time (with a free choice of rules and distribution of funds) provides new possibilities of income to i. As even the sheer existence of new cryptocurrencies can have a negative affect on the valuation of existing cryptocurrencies, this can also influence the EEV of i. Even more so, if the rules of the newly created cryptocurrencies are designed in a way that actively harm existing ones, as outlined in [22].

In other words, precisely calculating the EEV of a miner i is impossible even with perfect information on the current global state. Nevertheless, it may still be possible to approximate the EEV, if the number of possibly available resources $\mathcal{R}$, the computational capabilities of actors, as well as their overall number, can be meaningfully bounded. As cryptocurrencies provide the possibility to virtually create new resources at any point in time, this can technically not be prevented in practice. It is therefore questionable, if economic security models of permissionless cryptocurrencies that take the interplay between multiple resources into account, will be able to produce satisfactory security guarantees, compared to the high standards regarding security proofs that we are used to, for our cryptographic primitives, formally verified smart contracts, or classical Byzantine fault tolerant consensus systems.

This leaves us with the open question on how to best include economic considerations into the choice of the security parameter k determining the number of required confirmation blocks. In Section 4.1 we show a simple workaround that relieves us from the burden of correctly determining the right value for k.

4.1 The Let's Go Shopping Defense

We now describe a simple defensive strategy a merchant M can use to transfer the risk of choosing an insufficient security parameter k_M, to another merchant W. In our scenario we, assume that merchant M offers some quantity v of resource r for sale to a customer C. For this technique to work, we need to assume that there exists a merchant W with an already defined security parameter k_W. Furthermore, merchant W offers some easily tradable good/resource r' that is purchasable in arbitrary quantity and also stable in price. Then merchant M can now choose his k_M such that $k_M > k_W$, and immediately use all received funds directly to acquire r'. Therefore, M has to create a transaction tx_M that immediately uses all funds that were used by the customer to purchase r. In other words the transaction of M builds up on the transaction of C, i.e., $tx_C \rightarrow tx_M$. In an UTXO model cryptocurrency this can be achieved by using the respective UTXO as input, whereas in an account based model a dedicated account can be created to handle the purchase Technically, in most prevalent cryptocurrencies tx_C and tx_M can even be part of the same block if included in the right order and if M broadcasts tx_M immediately after observing tx_C in the P2P network.

Using this technique, merchant M can be sure to receive $v_{r'}$ before he has to hand out v_r. If now transaction tx_C is double-spent, or otherwise is invalidated so will be tx_M, but at that point either M has not yet handed out v_r, or already received $v_{r'}$. In both cases M does not face any direct damage from an attack.

5 Conclusion

We have shown that MEV is a special form of value extractable by miners, and that there is a difference between the *extractable value* that is computed in retrospect and *expected extractable value* (EEV) that is also forward looking and takes the probability of events influencing it into account. Further we have shown that estimating the EEV of any actor i is hard or even impossible in practice as we have to deal with imperfect information and possibly multiple cryptographically interlinked cryptocurrencies. The EEV depends on several factors: Transactions affecting i, which reach from incoming and outgoing regular payments, over bribes, to front running and arbitrage opportunities, as well as all consequences of actions affecting the valuation of assets in different resources actor i cares about. The difficulty to accurately estimate the EEV is further amplified by the fact that new cryptocurrencies might pop up, increasing the set of resources actors care about, or putting pressure on existing cryptocurrencies. Although, theoretically workable, a rather unsatisfactory workaround is described to transfer the risk of choosing an insufficiently large security parameter k to another merchant. In the wake of more and more attacks that exploit aspects of the economic rationally of actors (like for example front running), a better understanding of the economic interplay between such actors, as well as cryptocurrency systems as a whole, is desperately required to more accurately model the security guarantees of prevalent permissionless cryptocurrencies under such economical considerations and attacks. If the lack of descriptive models (which take practical economic considerations into account) persists, we have to ask ourselves if economic incentives in permissionless cryptocurrencies can ever produce satisfactory security grantees, our just occasionally worked “better in practice, than in theory” for a while.

References

1. How the winner got fomo3d prize - a detailed explanation. medium, 2018. accessed: 2020-09-15.
2. Talk: A primer on economics for cryptocurrencies. School of Blocks, Blockchain summer school at TU Wien, 2019. accessed: 2020-09-15.
3. K. Babel, P. Daian, M. Kelkar, and A. Juels. Clockwork finance: Automated analysis of economic security in smart contracts. *CoRR*, abs/2109.04347, 2021.
4. G. Bissias, R. Böhme, D. Thibodeau, and B. N. Levine. Pricing Security in Proof-of-Work Systems. *arXiv preprint arXiv:2012.03706*, 2020.
5. J. Bonneau. Why buy when you can rent? Bribery attacks on Bitcoin consensus. In *BITCOIN '16: Proceedings of the 3rd Workshop on Bitcoin and Blockchain Research*, Feb. 2016.
6. J. Bonneau. Hostile blockchain takeovers (short paper). In *5th Workshop on Bitcoin and Blockchain Research, Financial Cryptography and Data Security 18 (FC). Springer*, 2018.
7. J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. In *IEEE Symposium on Security and Privacy*, 2015.
8. E. Budish. The economic limits of bitcoin and the blockchain. Technical report, National Bureau of Economic Research, 2018.
9. M. Carlsten, H. Kalodner, S. M. Weinberg, and A. Narayanan. On the instability of bitcoin without the block reward. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 154–167. ACM, 2016.
10. P. Daian, S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, and A. Juels. *Flash Boys 2.0: Frontrunning, Transaction Reordering, and Consensus Instability in Decentralized Exchanges.* 2019. Published: arXiv preprint arXiv:1904.05234.
11. P. Daian, S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, and A. Juels. Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 910–927. IEEE, 2020.
12. A. Dembo, S. Kannan, E. N. Tas, D. Tse, P. Viswanath, X. Wang, and O. Zeitouni. Everything is a Race and Nakamoto Always Wins. Technical Report 601, 2020.
13. S. Eskandari, S. Moosavi, and J. Clark. SoK: Transparent Dishonesty: front-running attacks on Blockchain. In *arXiv preprint arXiv:1902.05164*, 2019.
14. I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *Financial Cryptography and Data Security*, pages 436–454. Springer, 2014.
15. B. Ford and R. Böhme. *Rationality is Self-Defeating in Permissionless Systems.* 2019. _eprint: arXiv:1910.08820.
16. J. Garay, A. Kiayias, and N. Leonardos. *The Bitcoin Backbone Protocol: Analysis and Applications.* 2014. Published: Cryptology ePrint Archive, Report 2014/765.
17. J. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In *Advances in Cryptology-EUROCRYPT 2015*, pages 281–310. Springer, 2015.
18. J. A. Garay, A. Kiayias, and N. Leonardos. *The Bitcoin Backbone Protocol: Analysis and Applications.* 2020. Publication Title: IACR Cryptology ePrint Archive, Report 2014/765.

19. A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdo rf, and S. Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC*, pages 3–16. ACM, 2016.
20. D. Hopwood, S. Bowe, T. Hornby, and N. Wilcox. Zcash protocol specification, 2021. accessed: 2021-09-06.
21. B. Hou and F. Chen. A Study on Nine Years of Bitcoin Transactions: Understanding Real-world Behaviors of Bitcoin Miners and Users. 2021.
22. A. Judmayer, N. Stifter, P. Schindler, and E. Weippl. Pitchforks in Cryptocurrencies: Enforcing rule changes through offensive forking- and consensus techniques (Short Paper). In *CBT'18: Proceedings of the International Workshop on Cryptocurrencies and Blockchain Technology*, Sept. 2018.
23. A. Judmayer, N. Stifter, A. Zamyatin, I. Tsabary, I. Eyal, P. Gaži, S. Meiklejohn, and E. Weippl. Pay-to-win: Cheap, crowdfundable, cross-chain algorithmic incentive manipulation attacks on pow cryptocurrencies. Cryptology ePrint Archive, Report 2019/775, 2019.
24. A. Judmayer, N. Stifter, A. Zamyatin, I. Tsabary, I. Eyal, P. Gaži, S. Meiklejohn, and E. Weippl. Sok: Algorithmic incentive manipulation attacks on permissionless pow cryptocurrencies. Cryptology ePrint Archive, Report 2019/775, 2020.
25. G. Kappos, H. Yousaf, M. Maller, and S. Meiklejohn. An empirical analysis of anonymity in zcash. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 463–477, 2018.
26. M. Kelkar, F. Zhang, S. Goldfeder, and A. Juels. Order-fairness for byzantine consensus. In D. Micciancio and T. Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 451–480. Springer, 2020.
27. J. A. Kroll, I. C. Davey, and E. W. Felten. The economics of Bitcoin mining, or Bitcoin in the presence of adversaries. In *Proceedings of WEIS*, volume 2013, page 11, 2013.
28. K. Liao and J. Katz. Incentivizing blockchain forks via whale transactions. In *International Conference on Financial Cryptography and Data Security*, pages 264–279. Springer, 2017.
29. P. McCorry, A. Hicks, and S. Meiklejohn. Smart Contracts for Bribing Miners. In *5th Workshop on Bitcoin and Blockchain Research, Financial Cryptography and Data Security 18 (FC). Springer*, 2018.
30. K. Nayak, S. Kumar, A. Miller, and E. Shi. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. In *1st IEEE European Symposium on Security and Privacy, 2016*. IEEE, 2016.
31. A. Obadia, A. Salles, L. Sankar, T. Chitra, V. Chellani, and P. Daian. Unity is strength: A formalization of cross-domain maximal extractable value, 2021.
32. R. Pass and E. Shi. *Hybrid Consensus: Scalable Permissionless Consensus.* Sept. 2016.
33. K. Qin, L. Zhou, and A. Gervais. Quantifying Blockchain Extractable Value: How dark is the forest? *arXiv preprint arXiv:2101.05511*, 2021.
34. M. Rosenfeld. *Overview of colored coins.* 2012. Publication Title: White paper, bitcoil. co. il.
35. M. Rosenfeld. *Analysis of Hashrate-Based Double Spending*, volume abs/1402.2009. 2014. Publication Title: CoRR.
36. G. Rosu. K: A semantic framework for programming languages and formal analysis tools. *Dependable Software Systems Engineering*, 50:186, 2017.

37. A. Sapirshtein, Y. Sompolinsky, and A. Zohar. *Optimal selfish mining strategies in Bitcoin*. 2015. Publication Title: arXiv preprint arXiv:1507.06183.

38. Y. Sompolinsky and A. Zohar. Bitcoin's Security Model Revisited. *arXiv preprint arXiv:1605.09193*, 2016.

39. J. Teutsch, S. Jain, and P. Saxena. When cryptocurrencies mine their own business. In *Financial Cryptography and Data Security (FC 2016)*, Feb. 2016.

40. C. F. Torres, A. K. Iannillo, A. Gervais, and R. State. *The Eye of Horus: Spotting and Analyzing Attacks on Ethereum Smart Contracts*. 2021. _eprint: 2101.06204.

41. I. Tsabary and I. Eyal. The gap game. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 713–728. ACM, 2018.

42. M. Vukolić. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. In *International Workshop on Open Problems in Network Security*, pages 112–125. Springer, 2015.

43. F. Winzer, B. Herd, and S. Faust. *Temporary Censorship Attacks in the Presence of Rational Miners*. 2019. Published: Cryptology ePrint Archive, Report 2019/748.

44. L. Zhou, K. Qin, A. Cully, B. Livshits, and A. Gervais. *On the Just-In-Time Discovery of Profit-Generating Transactions in DeFi Protocols*. 2021. _eprint: 2103.02228.

45. L. Zhou, K. Qin, and A. Gervais. A2MM: Mitigating Frontrunning, Transaction Reordering and Consensus Instability in Decentralized Exchanges. *arXiv:2106.07371 [cs]*, June 2021. arXiv: 2106.07371.

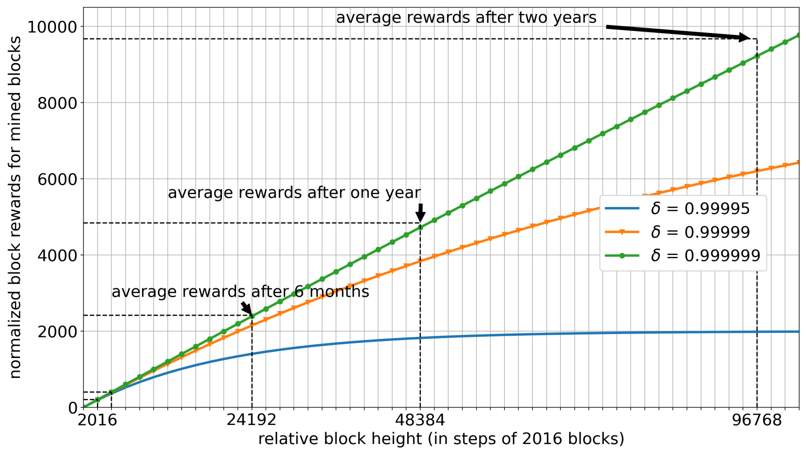
46. L. Zhou, K. Qin, C. F. Torres, D. V. Le, and A. Gervais. High-Frequency Trading on Decentralized On-Chain Exchanges. *arXiv preprint arXiv:2009.14021*, 2020.

Acknowledgements

This material is based upon work partially supported by (1) the Christian-Doppler-Laboratory for Security and Quality Improvement in the Production System Lifecycle; The financial support by the Austrian Federal Ministry for Digital and Economic Affairs, the Nation Foundation for Research, Technology and Development and University of Vienna, Faculty of Computer Science, Security & Privacy Group is gratefully acknowledged; (2) SBA Research; the competence center SBA Research (SBA-K1) funded within the framework of COMET Competence Centers for Excellent Technologies by BMVIT, BMDW, and the federal state of Vienna, managed by the FFG; (3) the FFG Industrial PhD projects 878835 and 878736. (4) the FFG ICT of the Future project 874019 dIdentity & dApps. (5) the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078 (FeatureCloud). We would also like to thank our anonymous reviewers for their valuable feedback and suggestions.

A Figure to Approximate δ

Fig. 1: Figure to approximate δ by comparing the average block rewards received by a miner with $p = 0.1$, to the expected infinite game rewards for the same miner with different discount factors δ. All rewards are given in normalized block rewards.



B Methods to Optimize Extractable Value

In this section, we want to classify the different ways in which the (expected) extractable value can be optimized by users as well as miners. Thereby, we build upon previous classifications [13,33,24]. For our classification, we divide all attempts into two major categories, namely *intrusive methods* and *unobtrusive methods*:

Unobtrusive methods , which do not interfere with consensus, i.e., require *no forks* and thus have also been termed *no-fork attacks* [24]. Some of these attacks can therefore be executed by miners as well as users. This category can be further separated into:

- **Passive no-fork attack**, approaches which do not require the creation of transactions. There attacks require hashrate, i.e., can only be executed by miners. Examples are:
  - *Passive no-fork order optimization* [10], in which the ordering which provides the highest value (e.g., in terms of fees) is selected without creating new transactions. Another possibility, would be to intentionally order transactions such that they consume the most gas in Ethereum.
  - *No-fork exclusion attacks* [29,23,43], in which a transaction is not included into a block because there is a revenue opportunity for the miner in this case e.g., due to a side payment (bribe or AIM attack), or because he himself is then able to profit directly e.g., by winning an auction.
- **Active no-fork attack**, approaches which do require the creation of additional transactions. Examples are:
  - *Active no-fork order optimization* [10], in which not only the ordering which provides the highest value in terms of fees is selected, but also additional transactions are created to collect value if needed, e.g., a guaranteed profit opportunity through arbitrage is observed, or an *any-on-can-spend* output is observed. These attacks can only be executed by miners.
  - *Active no-fork exclusion attacks*, in which other transactions are excluded by broadcasting sufficiently many high priority transactions (e.g., with high fee) to fill all available space in blocks. An example would be the Fomo3D exploitation [1]. This approach has also been referred to as *clogging* [33], or transaction *triggering* [24]. Attacks in this category can be executed by every actors that can created or incentivize sufficiently many transactions.

Intrusive methods , which require interference with consensus i.e., a fork. These attacks require the active participation of actors with hashrate (i.e., miners). These can be further separated into:

- **Deep-fork attacks**, where a fork with depth of at least ℓ exceeding a security parameter k_V of some victim V is necessary (i.e., $\ell > k_V$). The victim defines k_V [17,38] and it refers to its required number of confirmation blocks for accepting transactions[13]. In other words, the victim indirectly defines the required minimum fork length ℓ by his choice of k_V.
  - *Double-Spending attack* [35,12], in which a transaction is revised and the amount is transferred back to the original sender s.t., the sender can spend the same amount twice.
- **Near-fork attacks**, where the required fork depth is *not* dependent on k_V, but forks might be required. In other words, the attacker defines the gap k_{gap} (which can be smaller than k_V) he wants to overcome.[14] Examples are:
  - *Near-fork Undercutting attack* [9], in which the miner forks the latest block or blocks to mine the included transactions himself.
  - *Near-fork exclusion attack* [29,23,43], which is basically the same as the no-fork exclusion attack, but in this case the extractable value for excluding a transaction is large enough to incentivize a near-fork. An early variant of this has also been termed *feather forking* [7].
  - *Near-fork time bandit attack* [10], in which the attacker re-orders transactions which have already been included in a block in retrospect. Since the point in time when this reordering happens in independent of the individual security parameter (k_V) of a given transaction, this type of attack can be categorized as a near-fork attack.

[13] We emphasize that each transaction has a recipient (and thus a potential victim with an individual k_V), in practice there is no global security parameter k which holds for all transactions.

[14] The length of k_{gap} also depends on the attacker's resources and willingness to succeed (e.g., to exclude a certain block).

C Related Work: Clockwork Finance

In concurrent work [3] a formal definition of MEV was provided, within the *clockwork finance framework* (CFF), which uses "*formal verification techniques to reason about the profit extractable from the system by a participant*" [3]. The goal of CFF is it to prove bounds on the economic security of certain contracts without manually coding of adversarial strategies, which is defined as the "*maximum amount adversaries can extract from them*" [3]. Therefore, they use the K Framework [36], an executable semantic framework in which they modeled certain interesting DeFi contracts and reasoned about the maximum extractable value they offer if deployed together on the same system. They validated the modeled contracts empirically in a very elegant way, by utilizing the capabilities of K to use concrete inputs to programs to be evaluated. Therefore, they where able to replay previously collected real-world data of interactions whith the respective contracts, to the modeled contracts and compare the final states. Furthermore, they provide scripts to download, process and validate this data for each analyzed protocol.

Clearly, this work is of practical importance for quickly discovering and analyzing attacks on deployed and new DeFi contracts and protocols, which result in situation where attackers can extract high amounts of value if certain contracts are deployed/used in combination. This property is referred to as *composability* of contracts in [3]. However, the chosen approach and model also faces certain limitations and challenges which we will discuss in more detail after summing up the original paper.

In the paper [3] *extractable value* (EV) is defined as the "*maximum value, expressed in terms of the primary token, that can be extracted by a given player from a valid sequence of blocks that extends the current chain.*". More formally, the extractable value for a player P, with accounts A_P, starting from a state s and given a valid block sequence $\mathcal{B}$ of length k is defined by the maximum achievable balance in the primary token on all his accounts after k blocks. For a summary of relevant variables and symbols according to the model from [3] is given in Table 1.

$$\mathtt{EV}(P, \mathcal{B}, s) = \max_{(B_1,\ldots,B_k)\in\mathcal{B}} \left(\sum_{a\in A_P} \mathtt{balance}_k(a)[0] - \mathtt{balance}_0(a)[0] \right) \tag{13}$$

From this definition of EV, they define the *miner extractable value* (MEV or k-MEV) if P is a miner as the EV from *all* valid blocks that can be created by P in state s:

$$\mathtt{MEV}(P, s) = \mathtt{EV}\big(P, \mathtt{validBlocks}_1(P, s), s\big) \tag{14}$$

$$k\text{-}\mathtt{MEV}(P, s) = \mathtt{EV}\big(P, \mathtt{validBlocks}_k(P, s), s\big) \tag{15}$$

In other words, the miner extractable value of player P, is the maximum value that can be extracted from all valid blocks that can be created by P in state s. Note that these definitions assume that P is going to mine the next or the next k blocks, but so far the definition does not account for the probability, or the difficulty for this to happen. To capture the probability to mine the exactly k consecutive blocks, the concept of *weighted* MEV (WMEV) is defined in the appendix [3]:

$$\texttt{WMEV}(P, s) = \sum_{k=1}^{\infty} p_k \cdot k\text{-}\texttt{MEV}(P, s) \tag{16}$$

With the concept of MEV the paper [3] then goes ahead and defines *DeFi Composability* as follows:

(DeFi Composability): Consider a state s and a player P, A DeFi instrument C' is ε-composable under (P, s) if

$$\texttt{MEV}(P, s') \leq (1 + \varepsilon)\texttt{MEV}(P, s) \tag{17}$$

Here s' refers to the state of the system if contract C' is added to state s. In other words, this definition states that a certain contract C' is DeFi composable with a system in state s, for a given player P, if this player cannot gain significantly more value out of the system when C' is deployed on s.

Variable	Description
P	A player
$\mathcal{B}$	Set of valid block sequences of length k
k	Length of specified block sequence $\mathcal{B}$
s	State of the system
$B_1, \ldots, B_k$	Blocks in $\mathcal{B}$
A_P	Set of accounts belonging to player P
$a \in A_P$	Individual account of player P
$\texttt{balance}_x(a)[0]$	Function that returns the balance of a given account a at height x measured in the primary token ([0]), e.g., ether.
$\texttt{data}_x(a)$	Function returning the data of a given account a at height x
$\texttt{validBlocks}_k(P, s)$	Denotes the set of *all* valid blocks (or block sequences length k) that can be created by player P in state s if it could work as a miner.
p_k	Probability that P mines exactly k consecutive blocks
s'	State of the system, when contract C' is added to s

Table 1: Overview of important notation and variables used in the model presented in [3].

C.1 Challenges, Open Problems and the Relation to this Paper

In this section, we discuss the differences in the definitions of EV and MEV given in the paper [3] to the paper at hand. Furthermore, we comment on some challenges and open problems that arise from the definitions of EV and MEV introduced in the paper [3] and their relation to the paper at hand. The discussion is continued in Section D.1, after summarizing a follow-up work [31] in Section D which extends upon certain aspects of the model presented in [3] and also provides some open problems in this space.

Definitions: The main differences of the chosen definition of EV in the paper [3] to the paper at hand, are the explicit definition of the *current state* (s) as a parameter of the function, whereas the state in our definitions (2,3,4) is defined implicitly by the entire history of the blockchain, i.e., all its transactions which represent the individual state transitions.

The other major difference lies in the definition of EV as the *maximum* extractable value by a certain participant which is implicitly assumed to be oneself. From the perspective of the paper [3] this make sense as they aim to operate on unconfirmed transactions since no forks are considered. Although, this makes this definition not directly applicable when talking about the extractable value of already settled blocks, although this could be augmented.

Furthermore, it should be noticed that identifying the *maximum* EV of future blocks requires perfect information regarding the set of available transactions. As this set is dependent on the transactions other players make, it cannot readily be predicted, especially for multiples blocks into the future. This indicates that, although searching and approximating the maximum achievable extractable value from makes sense from a personal profit oriented perspective, defining EV and MEV as *the maximum value* that could be extracted is not optimal given all the computational and practical limitations, as well as for reasoning about potential consensus instability issues.

Combinatorial explosion and required information: As already noted in [3], the number of possible orderings of transactions grows factorial in the number of transaction in the mempool. Given that blocks in Ethereum approximately contain 200-250 transactions, reasoning over all possible orderings is computationally infeasible. In the paper [3] this problem is tackled by only considering transactions that are relevant (i.e., interact) with the contracts under investigation, therefore the set of transactions that was considered per block is reduced to 7 to 10 transactions. This is a reasonable approach, as current block sizes and intervals in Ethereum with around 15 transaction per second (TPS) lead to such a relatively low number of relevant interactions with certain smart contracts (or set of smart contracts). This restriction however may not apply to future versions of Ethereum which aim to scale up the number of transaction per second. Moreover, such restrictions might not be applicable to current high-throughput alternatives like Solana [15] or polygon [16] which claim to handle thousands of transactions per second. So identifying the *maximum* value that

[15] cf. `https://solana.com/`.
[16] cf. `https://polygon.technology/`.

can be extracted form a certain set of transactions is a computational problem that is difficult to solve in practise if systems and TPS grow larger.

Potential dependencies between certain transactions might on-first sight reduce the overall search space, but complicated transaction dependencies might still increase the overall computational costs for identifying a valid profitable ordering. As not all reordering are valid by the protocol rules, the validity of each ordering has to be checked as well, which further increases the overall computational costs as it potentially requires the evaluation of the entire ordering. Although this can be augmented, the validity of profitable orderings identified with CFF is not automatically checked at the moment.

Personal MEV, and MEV of other players: In the paper [3] MEV is mostly viewed from the perspective of a single player, which is implicitly assumed to be one-self. This is reasonable, and makes sense if someone wants to discover lucrative arbitrage/value extraction opportunities, which has been demonstrated to be possible with CFF. However, this analysis method cannot prove the absence of MEV exploitation vectors for any other player in the system, as their capabilities as well as their accounts might not be known to a third party. Therefore, reasoning about consensus instability requires further adaption of the chosen approach.

DeFi composability does not imply the security of contracts: The introduced notion of DeFi composability in [3] is only concerned with the value that can be extracted by a given actor P. On the one hand, this definition of DeFi composability does not consider the state of contracts, which leads to a situation in which the deployment of another contract C' can trigger/set an existing contract C into a different state, or even self-destruct, such that C can no longer operate. This would not be a problem under the notion of DeFi composability. If the value of the extractable funds does not increase significatly, C' and C are still DeFi composable, even if every player would lose significant amounts of funds if C' is deployed.

On the other hand, DeFi composability can be instantly violated by deploying a contract C' which hands out funds x, where $x > \varepsilon$, to the first player who calls the respective contract. In this case DeFi composabillity is violated for contract C' and every player P, although only one player can collect the promised (bribing) funds.

Model does not consider forks: As also noted in the paper [3] the model currently does not consider the likelihood and potential impact of forks in forkable blockchains.

Scope limited to a single system/domain/base resource: Currently the model presented in the paper [3] considers only balances of the primary asset and tokes/contracts based within the same system/domain i.e., smart contract cryptocurrency. The `balance()[T]` function is used to provide the balance of the respective asset T, conversions and exchange rate to the primary currency/asset are not discussed. Therefore, no cross-chain interactions and value extraction opportunities are modelled. This topic was addressed by another recent publication [31] which will be summarized and discussed in Section D.

D Related Work: Unity is Strength

In concurrent work [31] the authors address the concept of *cross-domain (maximal) extractable value* and outline a series of open questions in relation to this concept. Therefore, they adopt the definitions of EV,MEV/k-MEV given in [3] and extend them to a cross-domain i.e., cross-chain context. The authors of [31] define extractable value as follows:

"*Extractable value is the value between one or more blocks accessible to any user in a domain, given any arbitrary re-ordering, insertion and censorship of pending or existing transactions.*"

More formally, they define the extractable value from the perspective of a user/player P, in a specific domain/system i as the difference in his overall balance after a sequence of actions $a_1, \ldots, a_n$ is executed on an initial state s. For a summary of relevant variables and symbols introduced in the respective paper [31], see Table 2.

$$ev_i(P, s, a_1, \ldots, a_n) = b_i(s', P) - b_i(s, P) \tag{18}$$

Note that this definition, is more abstract in a sense that is does not talk about blocks but actions and thus might be applicable to a wider range of systems compared to [3]. Based on this definition of extractable value, the concept of *maximal extractable value* (MEV) is defined as follows:

"*Maximal Extractable Value (MEV) is the maximal value extractable between one or more blocks, given any arbitrary re-ordering, insertion or censorship of pending or existing transactions.*"

$$mev_i^j(P, s) = \max_{a_1,\ldots,a_n \in A_j} \big(ev_i(P, s, a_1, \ldots, a_n)\big) \tag{19}$$

Here j is the domain in which the actions are taken, and i is the domain in which the balance change occurs. In a one-domain context $i = j$. Again, as in [3] this concept refers to the *maximum value* that can be extracted, by a certain sequence of actions. The question though is how this sequence of actions can be defined and then explored in practise? See the Section D.1 regarding a discussion on this subject. The given definition of *mev* then can be extended to a cross-domain setting by adding domain. Here is an example of a two-domain setting:

$$mev_{i,k}^{i,j}(P,s) = \max_{a_1,\ldots,a_n \in A_i \cup A_j} \Big(ev_i(P,s,a_1,\ldots,a_n) + p_{i\to j}\big(ev_j(P,s,a_1,\ldots,a_n)\big)\Big) \tag{20}$$

This can of course be further extended by including more domains, a generalization for n domains is given in [31]. At the end the paper [31] closes with a series of open questions, which we will also discuss in Section D.1:

I How do we best define the action space?
II Aside from cross-domain arbitrage, what are other forms of cross-domain MEV?
III What does a protocol for sequencer collusion look like and what are its desirable properties?
IV How can we identify and quantify cross-chain MEV extraction taking place?
V What can we learn from existing distributed and parallel programming literature?

Variable	Description
P	A player
s	A system state
$A_P(s)$	Set of all actions available to player P in state s
$a_1,\ldots,a_n \in A_P$	Series of n actions of player P
A	Set of all actions of all players in the protocol
i,j	Two different domains / systems
$b_i(s,P)$	Functions returning the balance of a player
$p_{j\to i}$	Pricing function, indicating the balance in domain j is expressed in units of the native asset of domain i

Table 2: Overview of important notation and variables used in the model presented in [31].

D.1 Challenges, Open Problems and the Relation to this Paper

In this section, we outline the different definitions of MEV given in the paper [31] and their relation to the paper at hand. In this context, we discuss some challenges and open problems that arise from the definitions of extractable value introduced in the paper [31].

Definitions: In resemblance to the definitions given in paper [3], the definition in [31] also use the player and the current state as parameters to the function EV. In contrast to [**?**] the other parameter is actions instead of blocks or transactions. This makes this definition more abstract and thus more broader applicable. MEV is defined as the *maximum* extractable value from a sequence of actions, while the action space is not yet well defined and left to future work. Furthermore, cross-domain scenarios are considered which would roughly resemble a multi resources settings in the model of the paper at hand. The analysis of the implications of such a cross-domain model are left to future work.

Sets of players and actions: The definition of the available actions space is stated as an open question in the paper [31]. The probabilistic nature of the interactions in Nakamoto-Style systems was already mentioned in the paper [31] as a factor that makes the definition of the available action space a challenging tasks.

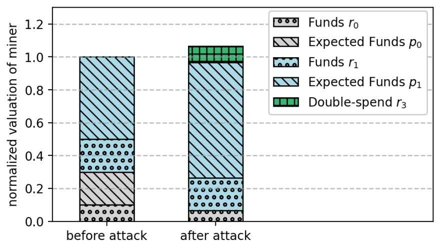
We now want to add some more aspects, which introduce additional challenges. The combinatorial explosion of possible orderings of transactions as well as the required information was already discussed in Section C.1, and can also be considered challenges in defining the action space. Furthermore, at least in theory, current permissionless [42] Nakamoto-Style systems allow players (especially miners) to join and leave at any time. Therefore, the set of players is theoretically infinite. As a result, also the overall action space of players is theoretically unbounded as the number of players is unbounded.

In practise, given a finite world, the set of players is still challenging to define as it might change drastically over time. This in turn, also changes the available action space over time.

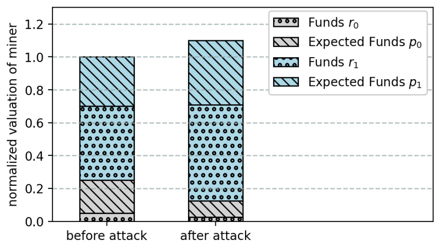
Balance and Pricing of Assets: In the paper [31] a pricing function was assumed to model the exchange of units in different domains into the native asset of the base domain. Also more complex models which depend on the supply and demand, or the quantity exchanged are mentioned. In addition to this discussion on how to define the price of different assets on different domains, we would like to add the following challenge: The price, or utility of a given asset is highly dependent on a specific player and his investment and interest in different systems/domains. Therefore, the price of the very same asset might be different for two different players due to the domains they are invested in, or things like colored coins [34]. The pricing function, also implicitly assumes perfect information regarding the exchange rates.

E Illustration of Different Events and their Consequences

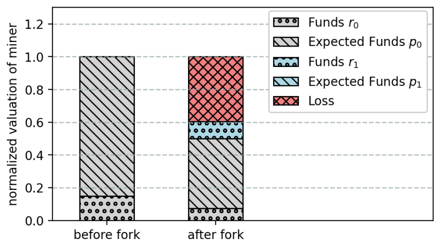
Fig. 2: Visual illustration of different events and their consequences on a participant with $\mathcal{R} := \{R_0, R_1\}$. The total valuation of a participant *before* the respective event is normalized to 1. This means that the values for the initial exchange rates are $e_{0,1} = 1$ (s.t. $f(r_{0,1}, 1) = r_{0,1}$), and that δ is static and thus ignored ($\delta_{0,1} = 0$). In other words expected future rewards where already accounted for in the relation between $p_{0,1}$ and $r_{0,1}$, s.t. $p_0 + r_0 + p_1 + r_1 = 1$.



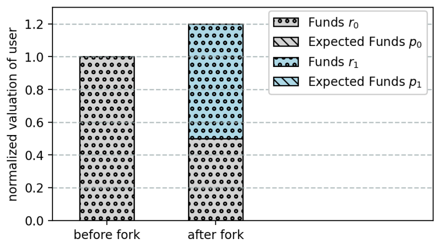
(a) A double-spend attack on a cryptocurrency R_0 from the perspective of a miner. *Before*: The total expected future income from mining is $p_0 = 0.2$ and $p_1 = 0.5$, in relation to the currently held funds in $r_0 = 0.1$ and $r_1 = 0.2$. *After*: A double-spend of all available funds r_0, leads to an additional gain of $r_3 = 0.1$ through the double-spend, but also to negative consequences on the exchange rate $e'_0 = 0.65$. This drop is evaded by moving all hashrate to R_1, where the exchange rate remains constant $e'_1 = 1$. This leads to a gain of exactly the exchange rate in R_0 as the double-spend funds cannot be moved without losses: $r_0 \cdot e'_0 + (p_0 + p_1 + r_1) \cdot e'_1 + r_3 = 1.065$



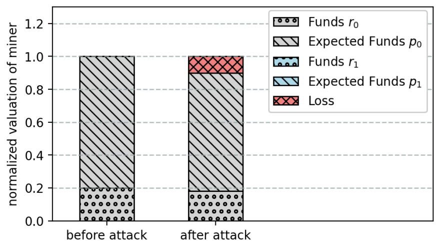
(b) A Goldfinger attack [27,29,6] on a cryptocurrency R_0 from the perspective of a miner. *Before*: The total expected future income from mining is $p_0 = 0.2$ and $p_1 = 0.3$, in relation to the currently held funds in $r_0 = 0.05$ and $r_1 = 0.45$.
After: The Goldfinger attack leads to a drop in the exchnage rate in R_0 of 50% ($e'_0 = 0.5$), while the exchange rate in R_1 increses by 30% ($e'_1 = 1.3$). Due to the distribution of his current holdings and expeceted future income in those two resources R_1 and R_2, at the end of the day m profits more from the increase than he loses from the decrease:
$(p_0 + r_0) \cdot e'_0 + (p_1 + r_1) \cdot e'_1 = 1.1$



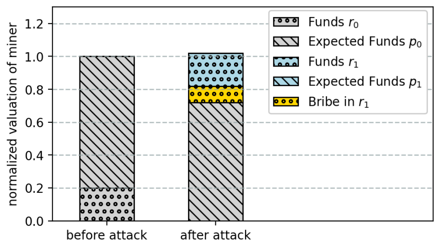
(c) A fork of R_0 into R_0 and R_1 from the perspective of a miner. *Before*: The total expected future mining rewards are $p_0 = 0.85$ while $r_0 = 0.15$ come from current holdings. *After*: The fork changes the exchange rate to $e'_0 = 0.5$, thus cutting the previous total valuation in half and at the same time adds the funds from the new resource $r_1 = r_0 = 0.15$ with an exchange rate $e'_1 = 0.7$. This leads to an overall loss for the miner: $(p_0 + r_0) \cdot e'_0 + r_1 \cdot e'_1 = 0.605$



(d) The same fork as in Figure 2c from the perspective of a user. *Before*: There is no expected future income $p_0 = 0$, thus 100% come from current holdings in R_0 ($r_0 = 1$). *After*: The forks changes the exchange rate to $e'_0 = 0.5$, thus cutting the previous total valuation in half, but at the same time adds the funds from the new resource $r_1 = r_0 = 1$ with an exchange rate $e_1 = 0.7$. This leads to a surplus of 0.2 for the user in this case: $(p_0 + r_0) \cdot e'_0 + r_1 \cdot e'_1 = 1.2$



(e) An attack on a R_0 from the perspective of a miner. *Before*: The total expected future income from mining is $p_0 = 0.8$, while $r_0 = 0.2$ come from current holdings. *After*: An attack with negative consequences on the exchange rate $e'_0 = 0.9$ is depicted leading to a loss for the miner:
$(p_0 + r_0) \cdot e'_0 = 0.9$



(f) The same attack as Figure 2e, but in this case the funds $r_0 = 0.2$ can be transferred to an alternative cryptocurrency R_1 in which no value loss occures ($e'_1 = 1$). Furthermore, a bribe $\varepsilon = 0.1$ is payed in r_1 to the miner to accommodate for the induced losses. In this case the miner would have surplus, although his hashrate p_0 is non-transferable. $(p_0+r_0)\cdot e'_0+(r_1+\varepsilon)\cdot e'_1 = 1.02$