Multidimensional Blockchain Fees are (Essentially) Optimal

Guillermo Angeris* gangeris@baincapital.com
Theo Diamandis tdiamand@mit.edu
Ciamac Moallemi ciamac@gsb.columbia.edu

February 2024

Abstract

In this paper we show that, using only mild assumptions, previously proposed multidimensional blockchain fee markets proposed in are essentially optimal, even against worst-case adversaries. In particular, we show that the average welfare gap between the following two scenarios is at most $O(1/\sqrt{T})$, where T is the length of the time horizon considered. In the first scenario, the designer knows all future actions by users and is allowed to fix the optimal prices of resources ahead of time, based on the designer's oracular knowledge of those actions. In the second, the prices are updated by a very simple algorithm that does not have this oracular knowledge, a special case of which is similar to EIP-1559, the base fee mechanism used by the Ethereum blockchain. Roughly speaking, this means that, on average, over a reasonable timescale, there is no difference in welfare between 'correctly' fixing the prices, with oracular knowledge of the future, when compared to the proposed algorithm. We show a matching lower bound of $\Omega(1/\sqrt{T})$ for any implementable algorithm and also separately consider the case where the adversary is known to be stochastic.

1 Introduction

Public blockchains allow any user to submit a transaction that modifies the shared state of the network. These transactions are independently verified and executed by a decentralized network of full nodes, who must each re-execute the transaction to verify the correct updated state of the blockchain. Each transaction requires some amount of *resources*, such as compute or storage, to be executed. Because full nodes have finite resources, blockchains must limit the total computational resources that can be consumed per unit of time, usually measured per-block. As user demand may fluctuate, most blockchains implement a transaction fee mechanism in order to allocate finite resources among competing transactions.

*The authors are listed in alphabetical order

Transaction fees. Transaction fee mechanisms in practice often use a combination of a *base fee*, which is the minimum cost of entry for a particular block, and a *priority fee*, which is an additional fee that a user may pay so that a particular contentious transaction is included. In this paper, we concentrate on the base fee for transaction inclusion. These fees are computed algorithmically as part of the blockchain's consensus mechanism; any transaction not paying the computed base fee is invalid. Thus, the algorithm for computing these base fees is a crucial part of the blockchain's design, as it ultimately determines which transactions can be included in the blockchain and the blockchain's robustness to adversarial behavior, such as spam or denial-of-service attacks.

Dynamic base fees. Because user demand fluctuates, many blockchains have implemented or proposed a dynamic base fee mechanism. These mechanisms update the base fee based on current demand for resources, measured by the resource consumption of the transactions in previous blocks. Most famously, EIP-1559 [BCD+19] for the Ethereum blockchain introduced a base fee that adjusts to track a target block usage. Many of these mechanisms can be interpreted as applying gradient descent on the dual of a particular optimization problem [DECA23].

Multidimensional fees. Calculating transaction fees through a single, joint unit of account fixes the relative prices of all resources. These fixed relative prices inhibit granular price discovery and, ultimately, reduce the throughput of the blockchain. To mitigate this behavior, some blockchains have proposed *multidimensional fees*, where different resources are priced separately. EIP-4844 [BD21], to be implemented in the upcoming Dencun update, allows users to submit special pieces of data called blobs [Adl19a, Adl19b], which have an independent fee mechanism. In essence, EIP-4844 creates a two-dimensional fee market. SIMD-110 [YZ24] in the Solana ecosystem proposes a local fee controller for each piece of state on the Solana virtual machine. During periods of high user demand, highly contested pieces of state would become more expensive without affecting the price of other state. Similar multidimensional mechanisms have been proposed for the Ethereum ecosystem [Adl22, But22] and other blockchains, such as Avalanche [O'G23] and Penumbra [Pen23], have also implemented multidimensional blockchain fees. Like dynamic base fees, these multidimensional mechanisms can be designed and interpreted by considering the optimization problem that they attempt to solve.

Optimality. While the design and incentives of particular existing transaction fee mechanisms have been extensively studied, the overall design space has not been well-characterized. In particular, it is unclear what properties we can hope for from a 'good' fee mechanism, the tradeoffs present within the space of 'good' mechanisms, and what 'good' even means in this context, where adversarial behavior may preclude the existence of 'true' market-clearing prices for resources. These open questions are important for the many blockchains currently working on updated fee mechanisms, especially because many traditional results do not apply in the adversarial blockchain environment.

This paper. In this paper, we seek to characterize the optimality of the algorithm for setting multidimensional blockchain fees proposed in [DECA23]. To judge the optimality of this algorithm, we rely on the notion of *regret* from the online convex optimization literature [SS+12, H+16]. This quantity measures the difference between two scenarios: in the first, we dynamically update the resource prices as in [DECA23, §3.4]; in the second, we assume that the algorithm knows all future transactions and chooses the best fixed vector of prices given this knowledge. (The second scenario is similar to the current implementation of gas in Ethereum, but, of course, the fixed gas prices of individual resources are not known to be optimal.) We show that the average welfare gap between the two scenarios is at most $O(1/\sqrt{T})$, where T is the length of the time horizon considered, and we construct a corresponding lower bound. Taken together our results show that the proposed algorithm is essentially optimal according to this metric, even under arbitrary adversarial behavior that eludes traditional game theoretic analysis. We also show that, if the adversary is stochastic, the prices do converge to a market clearing price and give an explicit rate of convergence. We point to several additional directions for future work, where we suspect additional tools from online optimization may be used to improve upon the results presented here.

1.1 Related work

There are a number of papers proposing new mechanisms for pricing resources or analyzing mechanisms currently implemented in practice; we review some of the relevant literature below.

Blockchain resource pricing. In the specific case of blockchain resource pricing, these papers can be roughly broken down into three buckets, which we describe next. The first set analyzes past or current fee mechanisms or variations thereof. These papers include some of the original economic analyses of EIP-1559 [Rou21], empirical studies [But18, LLN+22], and clean theoretical models [LRMP23]. All of these papers focus on the single-dimensional resource case, usually under some simplifying assumptions such as, *e.g.*, certain stochastic limits in the case of [LRMP23]. The second set of papers proposes simple extensions or characterizations of mechanisms for pricing single resources in a variety of settings [FMPS21, BGR23, BEOS23], including some impossibility results [CS23]. The final set proposes multidimensional fee mechanisms for a variety of settings [But22, DECA23, CMW23]. Perhaps the closest in spirit to this particular work is [CMW23], which proposes a stochastic demand model of resources with stochastic observations. They apply linear-quadratic control theory to set prices that minimize a composite loss function that seeks to balance matching the target block size, while smoothly changing prices. This work, on the other hand, shows that, even in an adversarial setting, a very simple update rule suffices and has low regret relative to optimally setting the prices. We also show that, in the 'easier' stochastic setting, we recover a market-clearing price and provide a rate of convergence.

Tatonnement processes. The design of transaction fee mechanisms via gradient descent on an appropriately-chosen dual problem is also related to the literature on tatonnement processes [Uza60, CF08]. Many works (*e.g.*, [BDHN19] and [RSUW20]) use gradient descent-like algorithms to develop dynamic pricing strategies for various types of markets. These algorithms use a dynamic step size to ensure convergence to optimal prices. More similar to our work, [ALQS22] uses a fixed step size, and their algorithm, as a result, does not necessarily converge to any single price.

2 Problem set up

In this section, we briefly introduce the blockchain resource allocation problem of [DECA23]. We review how the dual problem leads to a notion of resource prices, which, when set correctly, ensure that block builders include the 'optimal' transactions, *i.e.*, those that maximize the utility of the transaction producers, minus the loss incurred by the blockchain for including those transactions.

2.1 The block building problem

We will fix some notation that we use for the remainder of the paper in the paragraphs that follow and describe the problem we are looking to solve.

Transactions and resources. First, let $t = 1, \ldots, T$ denote the current blockheight. At blockheight $t-1$, we assume that there are n_t transactions from which we may pick a subset to be included in block t. (This list of transactions that we are allowed to choose from is often called the *mempool*.) Each of these $j = 1, \ldots, n_t$ transactions has some associated utility $(q_t)_j \in \mathbf{R}_+$ which is gained if it is included in the block. Additionally, transaction j, if included in the block, will consume some amount of the m available resources, which we denote as an m-vector $a_j \in \mathbf{R}^m$. (The ith entry of this vector, $(a_j)_i$, denotes how much of resource i is consumed by transaction j.) Finally, the vector $x_t \in \{0,1\}^{n_t}$ will denote the transactions that were included in block t: the jth entry $(x_t)_j = 1$ if transaction j is included in block t and is 0, otherwise.

Total consumption. This set up lets us write the total resource consumption of all transactions included in block t in a convenient way. Define the m-vector

$$y_t = \sum_{j=1}^{n} a_j (x_t)_j = A_t x_t,$$

where the matrix $A_t \in \mathbf{R}_+^{m \times n_t}$ has jth column a_j denoting the resources consumed by transaction j in the mempool. We then have that $(y_t)_i$ denotes the total amount of resource i is consumed by the transactions included in block t.

Allowable transactions. We denote the set of allowable transactions in a block by $S_t \subseteq \{0,1\}^{n_t}$, which may include resource limits (*e.g.*, if $x \in S_t$, then $Ax \leq b$), and interactions among transactions. Examples may include complementarity constraints such as, if $x \in S_t$, then $x_j = 1$ implies that $x_{j'} = 0$ for some $j' \neq j$, among many other, potentially very complicated, constraints.

Loss function. Finally, we denote the network's 'unhappiness' with the resource usage at block t as the loss function $\ell_t : \mathbf{R}^m_+ \to \mathbf{R} \cup \{\infty\}$. This function maps the resource usage at block t, which is always a nonnegative vector, to some number $\ell_t(y_t)$. Higher values denote less desirable resource usage. We assume this function, which is chosen by the blockchain designer, is convex and lower semicontinuous for the remainder of the paper.

Block building problem. We now define the *block building problem* as the problem of optimizing the utility of the included transactions, minus the loss incurred by the network, subject to the transactions being in the allowable transaction set:

$$\begin{array}{ll} \text{maximize} & q_t^T x - \ell_t(y_t) \\ \text{subject to} & y_t = A_t x_t \\ & x_t \in \mathbf{conv}(S_t). \end{array} \tag{1}$$

Here, the variables are the included transactions $x_t \in \mathbf{R}^{n_t}$ and the total resource consumption of the included transactions, $y_t \in \mathbf{R}^m$. Additionally, $\mathbf{conv}(S_t)$ denotes the convex hull of the set S_t. Replacing S_t with its convex hull allows 'fractional' transactions, such as $x_j = 1/3$. These values of $x_j \in (0,1)$ can be interpreted as 'spreading out' transaction j over the next $1/x_j$ blocks. We denote the optimal value of problem (1) by $s^\star$.

Bounds. In general, it is a good idea to have maximum per-block resource consumption bounds, which we write as $b \in \mathbf{R}^m_+$, so we will assume that

$$S_t \subseteq \{x \in \{0,1\}^{n_t} \mid A_t x \leq b\},$$

so any allowable choice of transactions $x_t \in S_t$ will satisfy

$$0 \leq A_t x_t \leq b,$$

as both A_t and x_t are nonnegative. In other words, we assume that the maximum amount of resource i that any set of acceptable transactions may consume is no more than b_i. Note that this also implies

$$0 \leq y_t \leq b,$$

for any y_t feasible for (1) since $y_t = A_t x_t$. (For why such limits are necessary, see the discussion in [DECA23, §1].) Since $0 \leq y_t \leq b$ for any feasible points, we assume, without loss of generality, that $\ell_t(y) = \infty$ whenever $y \not\leq b$ or $y \not\geq 0$ for convenience.

Discussion. We may view this block building problem as the following 'idealized' scenario: if we, as the network, knew the welfare generated by each transaction, along with all possible constraints for all transactions then solving problem (1) would give us a way to include transactions as to maximize welfare minus the loss incurred by the network. Of course, in general, almost all of these things are not known (or, potentially even knowable) to the network, so we will show how to approximately solve this problem by, instead, correctly setting the price of the consumed resources for each block.

2.2 The dual problem

In this subsection, we construct a particular dual problem, whose variables are the prices of the m possible resources; correctly setting these prices would solve our original problem (1). We then show that its gradients can be efficiently evaluated, without knowledge of things like the allowable transactions S_t nor the welfare q_t for each block t. This suggests a heuristic in which we use the gradient at each step to update the prices of the resources, which we prove in a later section is (essentially) optimal.

The dual function. One possible relaxation of (1), for each block $t = 1, \ldots, T$, can be constructed by relaxing the equality constraint in (1) to a penalty, parametrized by a vector $p_t \in \mathbf{R}^m$, which we will call the *prices*:

$$\begin{array}{ll} \text{maximize} & q_t^T x - \ell_t(y_t) + p_t^T(y_t - A_t x_t) \\ \text{subject to} & x_t \in \mathbf{conv}(S_t), \end{array}$$

where the variables are the same as those of problem (1). If we denote the optimal value of this problem, which is parametrized by the vector p_t, as $f_t(p_t)$, then, since the objective is separable over x_t and y_t, we can write the optimal value of this problem as

$$f_t(p_t) = \sup_y \left(p_t^T y - \ell_t(y)\right) + \sup_{x \in \mathbf{conv}(S_t)} (q_t - A_t^T p_t)^T x, \tag{2}$$

We note that this function f_t, which we will call the *dual function*, is convex since it is the pointwise supremum of a family of functions linear in p_t.

Fenchel conjugate. The first term of (2) can be recognized as the Fenchel conjugate of ℓ [BV04, §3.3] evaluated at p_t, which we write as $\ell_t^*(p_t)$. This function can be interpreted as choosing the resource allocation y that maximizes the network's profit minus the loss as a result of resource consumption y. We note that the Fenchel conjugate has a closed-form expression for many commonly used loss functions.

Block building problem. The second term in (2) is the optimal value of the following problem:

$$\begin{array}{ll} \text{maximize} & (q_t - A_t^T p_t)^T x \\ \text{subject to} & x \in \mathbf{conv}(S_t), \end{array}$$

with variable $x \in \mathbf{R}^n$. We can interpret this problem as the transaction producers' problem of creating and choosing transactions to be included in a block in order to maximize the utility of the included transactions ($q_t^T x$), after netting the fees paid to the network ($(A_t^T p_t)^T x$). (This problem is sometimes called the *block building problem* or *packing problem* that is solved by block producers. These 'block producers' include both users and validators.) Since the objective of this problem is linear, it has the same optimal value as the nonconvex problem

$$\begin{array}{ll} \text{maximize} & (q_t - A_t^T p_t)^T x \\ \text{subject to} & x \in S_t, \end{array} \tag{3}$$

with variable $x \in \mathbf{R}^n$. Note that the packing problem problem depends on the problem data associated with block t: the utilities q_t, the resources required by the submitted transactions, A_t, and the associated constraints S_t for a particular block at time t. We will denote the optimal value of the packing problem as $h_t(p)$, where the index t indicates that the problem data is associated with block t.

Gradient. With this notation, the dual function can be written as

$$f_t(p) = \ell_t^*(p) + h_t(p).$$

We note, for future use, that, when f_t is differentiable at p,

$$\nabla f_t(p) = y_t^\star(p) - A_t x_t^\star(p), \tag{4}$$

where $y_t^\star(p)$ denotes the maximizer for the Fenchel conjugate of ℓ_t, at price p, while $x_t^\star(p)$ denotes the optimal transactions to be included for the block building problem (3) at price p. If f_t is not differentiable at p, then any solution to each subproblem suffices. Note that both of these quantities are easy to compute if price p is set at blockheight $t-1$ and block t is submitted with these resource prices p: the former term, $y^\star(p)$, is the solution to a basic problem that often has a closed form, while the latter term, $Ax^\star(p)$, is exactly the total resource consumption of the transactions included in block t.

Bounds on the gradient. From the discussion in §2.1, we have that the allowable transactions in S_t can only consume up to some resource limit b, which means that:

$$0 \leq A_t x_t^\star(p) \leq b,$$

from problem (3). Additionally, since $\ell_t(y) = \infty$ unless $0 \leq y \leq b$, as noted in §2.1, then, it is not hard to show that

$$0 \leq y_t^\star(p) \leq b,$$

so we receive the simple bound on the gradient of f_t at p:

$$-b \leq y_t^\star(p) - A_t x_t^\star(p) \leq b.$$

This implies the gradient bound

$$\|\nabla f_t(p)\|_s \leq \|b\|_s$$

for any s-norm.

2.3 The aggregated dual problem

If the dual function f_t is differentiable, then, since it is convex, finding a minimizer of the dual function corresponds exactly to finding a point $p^\star$ satisfying

$$\nabla f_t(p^\star) = 0.$$

Using (4), this is the same as finding a set of prices $p^\star$ such that

$$A_t x_t^\star(p^\star) = y_t^\star(p^\star).$$

We can interpret the left term as the demand for resources at price $p^\star$ (from users and nodes) and the right term as the supply of resources at price $p^\star$ (from the network). Setting the prices correctly balances the demand and supply of the available resources. One can also show that, when f_t is differentiable and this condition is satisfied, the transactions $x_t^\star(p^\star)$ and $y_t^\star(p^\star)$ exactly solve the original problem (1).

The (aggregated) dual problem. As blockchain designers, we would like to set prices that, at least on average, render the supply and demand of resources (approximately) equal. That is, we would like to find a set of prices that solves the problem:

$$\text{minimize} \quad \sum_{t=1}^{T} f_t(p), \tag{5}$$

over $p \in \mathbf{R}^m$. Finding a price $p^\star$ that minimizes this objective means that, using (1), over the T blocks, we have

$$\sum_{t=1}^{T} A_t x_t^\star(p^\star) = \sum_{t=1}^{T} y_t^\star(p^\star),$$

i.e., that the demand and supply for resources over the T blocks are both equal. Or, in other words, that the market, over all T blocks, clears on aggregate when the prices p are correctly set. (The case where the objective is not differentiable over $p^\star$, but is subdifferentiable, means that there is at least one solution to each subproblem, at optimal prices $p^\star$, for which supply and demand are both equal.)

A basic heuristic. Since the prices p for the blockchain must be set before the blocks are observed, it would be hard to solve (5) without oracular knowledge of the functions f_t. A simple heuristic is then the following: start with some prices p_t, set at blockheight $t-1$, observe what happens at block t with these prices, and update the prices slightly (by, say $\eta > 0$) according to the gradient information:

$$p_{t+1} = p_t - \eta \nabla f_t(p_t). \tag{6}$$

(If f_t is not differentiable at p_t, then replace the gradient with a subgradient.) From before, we know that the gradient at block t is observable at the prices set in the previous block,

p_t. This heuristic has roughly the 'right behavior': if our prices are too low for a certain resource, say i, then the demand at these prices exceeds the expected supply, so

$$(y_t(p_t))_i > (A_t x_t^\star(p_t))_i.$$

From the update step, this would increase the price $(p_t)_i$ of resource i proportional to this amount. (The opposite is true if the prices at time $t-1$ were set too high.) See [DECA23] for additional discussion.

A multiplicative update. Of course, we can also imagine any number of other update rules. For example, we may use a multiplicative update:

$$p_{t+1} = p_t \circ \exp(-\eta \nabla f_t(p_t)), \tag{7}$$

where $\circ$ denotes the elementwise (Hadamard) product. This update ensures that the prices are always nonnegative; though the previous update (6) can be modified slightly to accommodate this as well. We note that this update rule, when there is one resource, corresponds exactly to EIP-1559 as implemented as in the EIP-4844 update Ethereum today.

Discussion. Of course, the next natural question is: just how well do these heuristics perform? We will show in the remainder of the paper that the average gap between the objective values provided by this heuristic and those given by setting the optimal prices $p^\star$ (which clear the market on average; *i.e.*, those that solve problem (5)) tends to zero as the number of blocks in question, T, grows:

$$\frac{1}{T}\left(\sum_{t=1}^{T} f_t(p_t) - \sum_{t=1}^{T} f_t(p^\star)\right) \leq \frac{C}{\sqrt{T}}, \tag{8}$$

for any (reasonable) choice of p. Perhaps the most surprising part is that we will show that this is true *even when the allowable transactions S_t, resource utilizations A_t, and the welfare q_t are controlled by an adversary.* We will also show that these algorithms are essentially optimal: no better rate, in terms of T, is possible for any algorithm with such an adversary. The fact that this is possible, along with the original bounds for such problems, come from a long line of work in online convex optimization [SS+12], which we present (and adapt to our specific scenario) in the next section.

3 Bounding the gap

From §2, the main quantity we wish to bound is the difference between the algorithm's performance (*i.e.*, the aggregate performance of the p_t) against the performance of the aggregate market clearing price (*i.e.*, the price $p^\star$ that minimizes (5)):

$$R = \sum_{i=1}^{T} f_t(p_t) - \sum_{i=1}^{T} f_t(p^\star). \tag{9}$$

This quantity R is called the *regret* of the algorithm. Bounding this regret for a variety of algorithms for choosing p_t is the main goal of the field of online algorithms, for which there are many great resources, *cf.*, [SS$^+$12, H$^+$16]. A bound on this quantity R would then also give us a bound of the form of (8) given by R/T, which is called the *average regret*. We will give a very short and self-contained exposition of online convex optimization, tailored to our special case, in the what remains of this section and in the next section.

Discussion. Note that this comparison is interesting for a few reasons. First, the price $p^\star$ can only be computed if all of the dual functions f_t are known in advance, which would require knowledge that the network does not have. (On the other hand, updating the prices p_t via a rule such as (6) is very easy to implement in practice.) The second observation is that it need not be true that p_t approaches $p^\star$, except in very special cases, even when the regret is low. (One case where this is true is, for example, when the $f_t = f$, for some fixed function f, which is a common simplification in some analyses of dynamic fees, or when the f_t are i.i.d. drawn from some fixed distribution and strongly convex.)

Linearization. Since the functions f_t are convex, then, by definition [BV04, §3.1.3] we have that

$$f_t(p^\star) \geq f_t(p_t) + \nabla f_t(p_t)^T(p^\star - p_t).$$

Rearranging, this gives an upper bound on the difference in objective value at time t,

$$f_t(p_t) - f_t(p^\star) \leq \nabla f_t(p_t)^T(p_t - p^\star).$$

For the remainder of the paper, we will write $g_t = \nabla f_t(p_t)$ for notational convenience. (If f_t is not differentiable at p_t, then any subgradient will suffice.) Summing both sides over t then gives

$$R \leq \sum_{i=1}^{T} g_t^T(p_t - p^\star). \tag{10}$$

In other words, if we can find a reasonable upper bound for the linear term on the right hand side, we can give a bound on the regret R, or the average regret R/T.

Results. The main point of this section will be to use bound (10), along with a simple framework to show that the linear update rule (6), with appropriately chosen step size $\eta > 0$ has average regret bounded from above by

$$\frac{R}{T} \leq \frac{BM}{\sqrt{T}},$$

where $\|b\|_2 \leq B$ is an upper bound on the norm of the resource limits, while $\|p^\star\|_2 \leq M$ is a bound on the optimal price. In the case that the adversary is stochastic (*i.e.*, if the f_t are random functions, uniformly and identically drawn under certain conditions we state below)

there is a much stronger result. Here, the time-averaged price $\bar{p} = (1/T)\sum_{t=1}^{T} p_t$ converges to the optimal value in expectation in that

$$\mathbf{E}\left[f(\bar{p}) - f(p^\star)\right] \leq \frac{B^2 M}{\sqrt{T}},$$

using again rule (6), where $f(p) = \mathbf{E}[f_1(p)]$, for any fixed p. (We can, of course, replace f_1 with any other f_t.) This also implies that, if the f_t are almost surely μ-strongly convex, for some $\mu > 0$, then $\bar{p}$ converges to the optimal price $p^\star$ in expectation. Finally, if the update used is the multiplicative rule of (7), then the average regret can be bounded from above by

$$\frac{R}{T} \leq BM\sqrt{\frac{m\log(M/\varepsilon)}{2T}}.$$

Here, we define $\|p^\star\|_\infty \leq M$, while $B = \|b\|_2$, as before, while $\varepsilon \leq 1$ is a lower bound on the price used in the first iteration of the rule, $p_0 \geq \varepsilon\mathbf{1}$, where $\mathbf{1}$ is the all-ones vector and the inequality is elementwise.

3.1 Choice functions and updates

We will focus on the two heuristics presented in §2.3 to update the prices and show that, for each, the regret is low. To do this, we rewrite the algorithms as a special case of a general update rule, sometimes known as FTRL or Follow The Regularized Leader [SS$^+$12, H$^+$16], which we will then use to find a bound on the regret. The presentation here is a slight variation of the lovely theorem and proof of [KSST09, §2].

Choice function. Given some *choice function* $F : \mathbf{R}^m \to \mathbf{R} \cup \{\infty\}$, we choose the new prices p_{t+1} given the previous gradients $g_1, \ldots, g_t$ by setting

$$p_{t+1} = \nabla F(-\eta(g_1 + \cdots + g_t)), \tag{11}$$

where $\eta > 0$ will be some specific step size to be chosen later. (The gradient can be replaced with any subgradient whenever the function is not differentiable, but is convex.) The one requirement we will have of this function F is that it is 'smooth'; in particular that the Hessian of F (if twice-differentiable) is bounded from above by σI, where $\sigma > 0$, in some suitable domain $D \subseteq \mathbf{R}^n$.

Gradient descent. One choice of functions we will use is the norm-squared,

$$F(z) = \frac{1}{2}\|p_0 + z\|_2^2.$$

Since $\nabla F(z) = p_0 + z$, then

$$p_{t+1} = p_0 - \eta(g_1 + \cdots + g_t) = p_t - \eta g_t,$$

which corresponds exactly to the gradient descent rule proposed in (6). This particular choice function is smooth with $\sigma = 1$ for any domain D.

Multiplicative update. The second choice function is

$$F(z) = \sum_{i=1}^{m} (p_0)_i \exp(z_i),$$

which leads to the multiplicative update rule found in (7),

$$p_{t+1} = p_0 \circ \exp(-\eta(g_1 + \cdots + g_t)) = p_t \circ \exp(-\eta g_t).$$

Here $\exp(\cdot)$ is taken elementwise, while $\circ$ denotes the elementwise (Hadamard) product. For the domain $D = \{z \mid \|z\|_\infty \leq \log(M)\}$, this choice function is smooth with $\sigma = \|p_0\|_\infty M$. (We use $\log(M)$ since a bound on the prices of order M would imply a bound on the domain size of order around $\log(M)$—this is a somewhat technical condition that can be avoided by a slightly more careful choice of F; *cf.*, appendix A.)

3.2 Constructing general regret bounds

We will combine two simple observations to give a bound on the linear term of (10). The first is that, if we define the Fenchel conjugate of F over the domain D by

$$F^*(\tilde{p}) = \sup_{z \in D} (\tilde{p}^T z - F(z)),$$

then we have the definitional inequality

$$z^T \tilde{p} - F^*(\tilde{p}) \leq F(z), \tag{12}$$

for any choice of $\tilde{p} \in \mathbf{R}^m$ and $z \in D$. The second is that, since F is smooth with constant σ, we have, by a simple exercise in integration:

$$F(z + z') \leq F(z) + \nabla F(z)^T z' + \frac{\sigma}{2} \|z'\|_2^2.$$

So, given a sequence of $g_1, \ldots, g_T$, by applying this inequality repeatedly and using the definition of the prices p_t given in (11), we have

$$F(-\eta(g_1 + \cdots + g_T)) \leq F(0) - \eta \sum_{t=1}^{T} g_t^T p_t + \frac{\sigma \eta^2}{2} \sum_{t=1}^{T} \|g_t\|_2^2.$$

Finally, setting $z = -\eta(g_1 + \cdots + g_T)$ in (12) and using the above inequality gives

$$\sum_{t=1}^{T} g_t^T (p_t - \tilde{p}) \leq \frac{F(0) + F^*(\tilde{p})}{\eta} + \frac{\sigma \eta}{2} \sum_{t=1}^{T} \|g_t\|_2^2.$$

Since the gradients g_t are bounded from above by $\|g_t\|_2 \leq \|b\|_2 = B$, this leads to the following bound on the linearized quantity:

$$\sum_{t=1}^{T} g_t^T (p_t - \tilde{p}) \leq B \sqrt{\sigma T (F(0) + F^*(\tilde{p}))}, \tag{13}$$

by choosing the step size

$$\eta = \sqrt{\frac{F(0) + F^*(\tilde{p})}{\sigma T B^2/2}},$$

for any choice of $\tilde{p}$.

Regret bound. Using (13), we can now give a bound on the regret R using the inequality of (10). More specifically, if we know that the optimal aggregate clearing price $p^\star$ lies in some set P, then, using (13), we have

$$R \leq B\sqrt{\sigma T(F(0) + F^*(p^\star))} \leq B\sqrt{\sigma T(F(0) + \sup_{p\in P} F^*(p))},$$

or, alternatively, that the average regret over that time horizon, R/T, is

$$\frac{R}{T} \leq B\sqrt{\frac{\sigma(F(0) + \sup_{p\in P} F^*(p))}{T}}. \tag{14}$$

3.3 Regret bounds for updates

Given the set up above, we are now ready to give bounds on the regret for both the gradient-descent-like update rule of (6) and the multiplicative update of (7).

Norm-squared choice function. For the gradient-descent-like rule (6), we have the choice function $F(z) = \frac{1}{2}\|p_0 + z\|_2^2$. Applying the bound (13) gives, since $F(0) = \|p_0\|_2^2$ and $\sigma = 1$ for this choice function (from §3.1):

$$\frac{1}{T}\sum_{t=1}^{T} g_t^T(p_t - \tilde{p}) \leq B\sqrt{\frac{\|p_0\|_2^2 + F^*(\tilde{p})}{2T}}.$$

It is not hard to show that $F^*(\tilde{p}) = \frac{1}{2}\|\tilde{p}\|_2^2$, and, assuming that we have a bound $\|p^\star\|_2 \leq M$ (and similarly for p_0), then the average regret is bounded by (14),

$$\frac{R}{T} \leq \frac{BM}{\sqrt{T}}.$$

Exponential choice function. In the multiplicative update rule (7), the exponential choice function is given by $F(z) = \sum_{i=1}^m \exp(z_i)$ has $F(0) = \|p_0\|_1$ and $\sigma = M$ such that, again, using bound (13),

$$\frac{1}{T}\sum_{t=1}^{T} g_t^T(p_t - \tilde{p}) \leq B\sqrt{\frac{M(\|p_0\|_1 + F^*(\tilde{p}))}{2T}}. \tag{15}$$

It is also not hard to show that the conjugate of F is

$$F^*(\tilde{p}) = \sum_{i=1}^{m} \tilde{p}_i \log \frac{\tilde{p}_i}{(p_0)_i} - \tilde{p}_i,$$

and, if we have a bound that $0 \leq \tilde{p} \leq M\mathbf{1}$ and $\varepsilon\mathbf{1} \leq p_0 \leq M\mathbf{1}$ for $\varepsilon \leq 1$, then

$$\sup_{\|\tilde{p}\|_\infty \leq M} F^*(\tilde{p}) \leq m \max\{M \log(M/\varepsilon) - M, 0\},$$

which follows from the fact that F^* is separable and convex so is maximized at the boundary of the domain. For simplicity, we will assume that $M \geq e$, in which case the first term in the max is nonnegative. Putting it all together, we know that

$$\|p_0\|_1 \leq mM \quad \text{and} \quad F^*(p^\star) \leq m(M \log(M/\varepsilon) - M).$$

Plugging these values into (15) gives

$$\frac{R}{T} \leq BM\sqrt{\frac{m \log(M/\varepsilon)}{2T}},$$

where, from before, M is the maximum price allowed for any one resource, m is the number of resources, and ε is the minimum starting price. The latter condition cannot be dropped since, if $p_0 = 0$, no update would ever be able to increase the price, while on the order of $\sim \log(1/\varepsilon)$ updates are needed to scale the price p_0 to any constant. A sufficient condition to ensure that $p_0 \geq \varepsilon\mathbf{1}$ is that there is always some minimal demand for blockspace at a small enough price, as in, *e.g.*, [DECA23, §3.2].

3.4 A stochastic model

In this subsection, we consider a stochastic setting instead of the adversarial one above, which will give us stronger results on the convergence of the prices p_t. In particular, given a probability space with sample space Ω and distribution P, we assume that a sample $\omega_t \in \Omega$ determines the parameters of the block building problem at time t, so that,

$$q_t = \tilde{q}(\omega_t), \quad S_t = \tilde{S}(\omega_t), \quad \ell_t(\cdot) = \tilde{\ell}(\cdot, \omega_t), \quad A_t = \tilde{A}(\omega_t),$$

where the welfare $\tilde{q}$, feasible set $\tilde{S}$, losses $\ell(\cdot, \omega)$ (viewed as a function over ω), and consumption matrices $\tilde{A}$ are random variables over Ω. Given a sample $\omega \in \Omega$ and a price vector p, the utility in scenario ω for price p is given by the concave function

$$\tilde{f}(p, \omega) = \sup_y \left(p^T y - \tilde{\ell}(y, \omega)\right) + \sup_{x \in \mathbf{conv}(\tilde{S}(\omega))} (\tilde{q}(\omega) - \tilde{A}(\omega)^T p)^T x.$$

(In other words, we replace the problem data in the definition of the dual function (2) with the corresponding sampled versions.) Here, we will show that, in a certain sense, the time-averaged price of the algorithm minimizes the expected utility,

$$f(p) = \mathbf{E}[\tilde{f}(p, \omega)]. \tag{16}$$

For later, we will define $\tilde{p}^\star \in \operatorname{argmin} f$. Unlike in the previous set up for adversarial functions, in this problem, the observed functions are random variables $f_t(p) = \tilde{f}(p, \omega_t)$, where the ω_t are drawn i.i.d. according to P. So, in this setting, the gradient update rule of (6),

$$p_{t+1} = p_t - \eta \nabla f_t(p_t),$$

corresponds to an instance of stochastic gradient descent with stepsize given by $\eta > 0$.

Result. A standard result in stochastic gradient descent from, say [GG23, Thm. 9.5], is that,

$$\mathbf{E}[f(\bar{p})] - f(\tilde{p}^\star) \leq \frac{2M^2}{\eta T} + \frac{\eta B^4}{2}.$$

Here, $B = \|b\|_2$ and $\|\tilde{p}^\star\|_2 \leq M$ is a bound on the optimal price $\tilde{p}^\star$ that minimizes the expected utility f given in (16). (We also assume that the initial price $\|p_0\|_2 \leq M$.) Finally, the time-averaged price $\bar{p}$ is defined

$$\bar{p} = \frac{1}{T} \sum_{t=1}^{T} p_t.$$

Setting $\eta = 2M/B^2\sqrt{T}$ then gives the bound:

$$\mathbf{E}[f(\bar{p})] - f(\tilde{p}^\star) \leq \frac{B^2 M}{\sqrt{T}}.$$

Extensions. There is an immediate extension to this result. The first is to note that, by assumption, this result relies on i.i.d. draws of the sample ω_t, but this can be generalized considerably. If we define ϵ_t to be the noise in the gradient estimate at block t, *i.e.*,

$$\epsilon_t = \nabla f(p_t) - \nabla_p \tilde{f}(p_t, \omega_t),$$

then the above result requires that $\{\epsilon_t\}$ be zero mean and i.i.d. Under suitable conditions, this can be relaxed to the $\{\epsilon_t\}$ being a martingale difference sequence [HKY97, §5.2].

3.5 Discussion

With only the basic assumptions that (a) there is some bound on the maximum price $p^\star$ that an adversary may choose and (b) some hard constraint on the block size, which is enforced by the validators, we achieve average regret that scales as $1/\sqrt{T}$ for either update rule, even when the transactions and welfare, which are included in the functions f_t, are chosen adversarially. An interesting point is that this shows why a hard constraint on the resources consumed per-block is a good idea: it bounds any adversary's ability to construct arbitrarily-sized blocks which could make the regret of most reasonable rules quite large. The restriction given in (a), that the adversary has bounded prices, seems strong, but is implied

by a number of simple statements: for example, it suffices that the adversary has finite budget and must pay $p_t^T A_t x_t^\star(p_t)$ for each transaction, which is, in fact, a much stronger requirement. Two interesting questions for future research are: first, is there a more natural model for an adversary in the blockchain setting than the essentially all-powerful one provided here? And, second, given such an adversary, can we achieve better results?

Time-independent step sizes. In this model, we assumed that the step size η is fixed and can depend on the total length of observed time period, T. Indeed, the step size trades off the ability of the prices p to 'react' to observed gradients g_t with the average regret. (In particular, as T becomes large, the average regret becomes small, but η must be similarly small and must remain so for the time period in observation.) It is not hard to show that the regret bound in both the adversarial (§3.3) and stochastic case (§3.4) above for the gradient descent rule (6) (*i.e.*, the one derived from the norm-square choice function) also carries through in the case where the step sizes η are decreasing with respect to time, to get an update rule of the form

$$x_{t+1} = x_t - \eta_t g_t. \tag{17}$$

Here, $\eta_t = C/\sqrt{t}$ and $C > 0$ is some appropriately chosen constant that, importantly, does not depend on the observation window T, but only on the known bounds for the price $\|p\|_2 \leq M$ and the maximum resource utilizations $\|b\|_2 \leq B$. The regret for this algorithm, under the same assumptions as §3.3 for the norm-squared choice function, is no more than $\sqrt{2}BM\sqrt{T}$ in the adversarial case; see [H$^+$16, §3.1] for a proof. (The bound for the stochastic case is more refined, but see, *e.g.*, [GG23, Thm. 5.3] for the statement.) More generally, any sequence $\{\alpha_t\}$ which asymptotically satisfies

$$\sum_{t=1}^{T} \alpha_t = O(\sqrt{T})$$

and $1/\alpha_T = O(\sqrt{T})$ has a similar guarantee, up to a potentially different constant.

Strong convexity. Indeed, if the step sizes of the algorithm are allowed to vary with respect to time, we can get much stronger regret guarantees in the case that the conjugate of the loss $\ell_t^\star$ is strongly convex, using the same update rule (17). More specifically, if $\ell_t^\star$ is μ-strongly convex, then we can achieve *logarithmic* regret that, additionally, does not depend on there being any bound on the optimal price $p^\star$. (One case where ℓ_t^* is strongly convex is, *e.g.*, when $\ell_t(y) = (1/2)\|(y - b^\star)_+\|_2^2$ for some target utilizations $b^\star \in \mathbf{R}_+^m$ and y is bounded by the maximum resource utilizations $y \leq b$.) In this particular case, we have that for the update rule (17), where the step sizes are $\eta_t = \mu/t$, the regret is no larger than $B^2 \log(T)/\mu$, making the average regret quite small: no more than $B^2 \log(T)/(\mu T)$. (For a proof of this, see [H$^+$16, §3.3.1].)

4 Lower bound

In this section, we will show that, up to a scalar constant, the bounds provided in §3.3 are essentially optimal for any algorithm. In particular, we will show that any rule to choose the prices p_{t+1}, based on the previous t observed gradients $g_1, \ldots, g_t$, achieves, at best, regret on the order of $1/\sqrt{T}$, for an adversary who can control the transactions appearing in a block, subject to the constraints of the set S containing some maximum resource usage limit $b \in \mathbf{R}^m_+$, for a commonly-used loss. As in the previous section, we assume that the optimal price $p^\star \in \mathbf{R}^m$ for the aggregated program is bounded, $\|p^\star\|_\infty \leq M$ for some value $M > 0$.

4.1 Construction

The construction we provide here is very simple: the adversary is purely stochastic and picks the resource utilizations, up to the bound, from a simple distribution, independently of all of the previous gradients, such that any algorithm achieves 0 objective value, while the best possible price p achieves an objective of $-C\sqrt{T}$ for some constant $C > 0$.

Loss function and set. We construct a lower bound for any algorithm which attempts to minimize the dual problem corresponding to the commonly-used loss

$$\ell(y) = \begin{cases} 0 & 0 \leq y \leq b^\star \\ +\infty & \text{otherwise,} \end{cases} \tag{18}$$

where $b^\star \in \mathbf{R}^m_+$ is the resource utilization target. Using this definition, it is not hard to show that the Fenchel conjugate of ℓ is

$$\ell^*(p) = (b^\star)^T (p)_+,$$

where $((p)_+)_i = \max\{p_i, 0\}$ denotes the 'positive part' of p, applied elementwise. The set $S_t \subseteq \{0,1\}^{n_t}$ we assume satisfies

$$S_t \subseteq \{x_t \in \mathbf{R}^{n_t}_+ \mid A_t x_t \leq b\},$$

for some matrix $A_t \in \mathbf{R}^{m \times n_t}$ and max resource utilization $b \in \mathbf{R}^m_+$. (We note that this set up reduces exactly to the resource metering of EIP-1559 when the number of resources $m = 1$, where n_t is the number of transactions in the mempool of the block builder and b is the block limit.)

Adversary. Consider a (stochastic) adversary who is able to control the amount of resources used by the network. The adversary picks the welfare to be $q_t = 0$ and picks transactions A_t such that the welfare maximization problem (3) satisfies, at price p_t,

$$A_t x_t^\star = b^\star - \varepsilon_t \circ \tilde{b},$$

where $\varepsilon_t \sim \{\pm 1\}^m$ is uniformly randomly chosen (independently, for each $t = 1, \ldots, T$) and

$$\tilde{b}_i = \min\{b_i^\star, b_i - b_i^\star\},$$

for $i = 1, \ldots, m$. (The adversary can achieve this by constructing, say, exactly one transaction that is always included with zero welfare and utilizations equal to exactly $b^\star - \varepsilon_t \circ \tilde{b}$; we assume this in what follows.) This gives

$$g_t = y_t^\star - A_t x_t^\star = b^\star - (b^\star - \varepsilon_t \circ \tilde{b}) = \varepsilon_t \circ \tilde{b}. \tag{19}$$

Proof. It suffices to show that, in expectation, the regret is $\Omega(\sqrt{T})$, which means that there is at least one choice of resource consumption trajectories (gradients $\{g_t\}$) that leads to large regret, for any possible choices of prices p_t by an algorithm. First note that the linear bound (13),

$$\sum_{t=1}^{T}(f_t(p_t) - f_t(p^\star)) \leq \sum_{t=1}^{T} g_t^T(p_t - p^\star),$$

is met at exact equality for any nonnegative p_t and $p^\star$ using our construction. To see this, note that the transactions were chosen to have zero welfare, $q_t = 0$. Using the definition of ℓ (18) means that, for any $p, p_t \geq 0$:

$$f_t(p) = \ell_t^*(p) + h_t(p) = (b^\star)^T p + (q_t - A_t^T x_t^\star(p_t))^T p = (b^\star - A_t x_t^\star(p_t))^T p = g_t^T p,$$

for every $t = 1, \ldots, T$. It will then suffice to show that the linear sum

$$\mathbf{E}\left[\sum_{t=1}^{T} g_t^T(p_t - p^\star)\right] \tag{20}$$

is 'large' in expectation, for $p^\star$ chosen with knowledge of g_t.

The first term of the loss (20) is zero since

$$\mathbf{E}[g_t^T p_t] = \mathbf{E}[g_t]^T p_t = 0,$$

as p_t can depend only on the observed gradients, $g_1, \ldots, g_{t-1}$. The expected regret then reduces to

$$\mathbf{E}\left[\sum_{t=1}^{T} g_t^T p_t - g_t^T \bar{p}\right] = \mathbf{E}\left[-\bar{p}^T \sum_{t=1}^{T} g_t\right].$$

Since $p^\star$ is chosen after observing the gradients g_t, and we are free to choose any $p^\star$ with $\|p^\star\|_\infty \leq M$ and $p^\star \geq 0$, we then this term is maximized by choosing $p_i^\star = M$ if

$$\sum_{t=1}^{T}(g_t)_i \leq 0,$$

and 0 otherwise, for $i = 1, \ldots, m$. This means that

$$\mathbf{E}\left[-\bar{p}^T \sum_{t=1}^{T} g_t\right] = M\mathbf{E}\left[\mathbf{1}^T \left(-\sum_{t=1}^{T} g_t\right)_+\right],$$

where $((z)_+)_i = \max\{z_i, 0\}$ is the 'positive part' of z_i. Using the definition of g_t above (19), we can bound the latter term

$$M\mathbf{E}\left[\mathbf{1}^T \left(-\sum_{t=1}^{T} g_t\right)_+\right] = M(\mathbf{1}^T\tilde{b})\mathbf{E}\left[\left(\sum_{t=1}^{T} \varepsilon'_t\right)_+\right] \geq M(\mathbf{1}^T\tilde{b})C\sqrt{T},$$

where $\varepsilon'_t \sim \{\pm 1\}$ for $t = 1, \ldots, T$ is uniformly and independently drawn and $C \geq 1/24$. The inequality follows from a mechanical (if involved) counting argument, or any number of lower bounds on the expected value of the positive side of a random walk. We reproduce a simple, noncounting proof of this in appendix B. We note that, compared to either of the bounds of §3.3, this is tight up to a twiddle factor of M and a square root factor of m.

5 Conclusion

In this paper, we have shown that two simple update rules: a gradient update rule and a multiplicative update rule, lead to low average regret for the discovery and assignment of prices for blockchain resources, even under adversarial conditions. This implies that, on average, over a long enough timescale, the difference between correctly setting resource prices with oracular knowledge of the future (to roughly match supply and demand over the period of time in question) and these two simple update rules, is very small. We have also shown that, under the assumptions above, this is essentially the best we can hope to do by also showing a matching lower bound. Since Ethereum's EIP-1559 is the single-dimensional special case of the multiplicative rule presented above, the analysis also applies directly, along with the proposed pricing rule for EIP-4844, which is the two-dimensional special case. Future potential avenues for research include weaker, but potentially more realistic adversarial models (as the adversary has a lot of power in this particular formulation) which may yield tighter bounds to the expected performance.

References

[Adl19a] John Adler. Eip-2242: Transaction postdata, 2019.

[Adl19b] John Adler. Multi-threaded data availability on eth 1. Ethresearch, 2019.

[Adl22] John Adler. Always has been (or, wait, it's all resource pricing? part 2). EthCC, 2022.

[ALQS22] Itai Ashlagi, Jacob Leshno, Pengyu Qian, and Amin Saberi. Price discovery in waiting lists: A connection to stochastic gradient descent. *Available at SSRN 4192003*, 2022.

[Ang23] Guillermo Angeris. A non-counting lower bound for the expected distance of a simple random walk. `https://angeris.github.io/blog/content/simple-proof-rademacher/`, 2023.

[BCD+19] Vitalik Buterin, Eric Conner, Rick Dudley, Matthew Slipper, Ian Norden, and Abdelhamid Bakhta. Eip-1559: Fee market change for eth 1.0 chain, 2019.

[BD21] Vitalik Buterin and Ansgar Dietrichs. Eip-4488: Transaction calldata gas cost reduction with total calldata limit, 2021.

[BDHN19] Sébastien Bubeck, Nikhil R Devanur, Zhiyi Huang, and Rad Niazadeh. Multi-scale online learning: Theory and applications to online auctions and pricing. *Journal of Machine Learning Research*, 20(62):1–37, 2019.

[BEOS23] Soumya Basu, David Easley, Maureen O'Hara, and Emin Gün Sirer. Stablefees: A predictable fee market for cryptocurrencies. *Management Science*, 2023.

[BGR23] Maryam Bahrani, Pranav Garimidi, and Tim Roughgarden. Transaction fee mechanism design with active block producers. *arXiv preprint arXiv:2307.01686*, 2023.

[But18] Vitalik Buterin. Blockchain resource pricing. 2018.

[But22] Vitalik Buterin. Multidimensional eip 1559. *ethresear.ch*, 2022.

[BV04] Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK; New York, 2004.

[CF08] Richard Cole and Lisa Fleischer. Fast-converging tatonnement algorithms for one-time and ongoing market problems. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 315–324, 2008.

[CMW23] Davide Crapis, Ciamac C Moallemi, and Shouqiao Wang. Optimal dynamic fees for blockchain resources. *arXiv preprint arXiv:2309.12735*, 2023.

[CS23] Hao Chung and Elaine Shi. *Foundations of Transaction Fee Mechanism Design*, pages 3856–3899. 2023.

[DECA23] Theo Diamandis, Alex Evans, Tarun Chitra, and Guillermo Angeris. Designing Multidimensional Blockchain Fee Markets. In Joseph Bonneau and S. Matthew Weinberg, editors, *5th Conference on Advances in Financial Technologies (AFT 2023)*, volume 282 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 4:1–4:23, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[FMPS21] Matheus V. X. Ferreira, Daniel J. Moroz, David C. Parkes, and Mitchell Stern. Dynamic posted-price mechanisms for the blockchain transaction-fee market. In *Proceedings of the 3rd ACM Conference on Advances in Financial Technologies*, pages 86–99, 2021.

[GG23] Guillaume Garrigos and Robert M Gower. Handbook of convergence theorems for (stochastic) gradient methods. *arXiv preprint arXiv:2301.11235*, 2023.

[H+16] Elad Hazan et al. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

[HKY97] J Harold, G Kushner, and George Yin. Stochastic approximation and recursive algorithm and applications. *Application of Mathematics*, 35(10), 1997.

[KSST09] Sham M. Kakade, Shai Shalev-Shwartz, and Ambuj Tewari. Regularization techniques for learning with matrices. *Journal of Machine Learning Research*, 13:1865–1890, 2009.

[LLN+22] Yulin Liu, Yuxuan Lu, Kartik Nayak, Fan Zhang, Luyao Zhang, and Yinhong Zhao. Empirical analysis of eip-1559: Transaction fees, waiting times, and consensus security. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, page 2099–2113, New York, NY, USA, 2022. Association for Computing Machinery.

[LRMP23] Stefanos Leonardos, Daniël Reijsbergen, Barnabé Monnot, and Georgios Piliouras. Optimality despite chaos in fee markets. In *Financial Cryptography and Data Security: 27th International Conference, FC 2023, Bol, Brač, Croatia, May 1–5, 2023, Revised Selected Papers, Part II*, page 346–362, Berlin, Heidelberg, 2023. Springer-Verlag.

[O'G23] Patrick O'Grady. Multidimensional fees + fifo mempool. `https://github.com/ava-labs/hypersdk/pull/352`, 2023.

[Pen23] Penumbra Team. Groundwork for multidimensional gas fees. `https://github.com/penumbra-zone/penumbra/issues/2970`, 2023.

[Rou21] Tim Roughgarden. Transaction fee mechanism design. *SIGecom Exch.*, 19(1):52–55, jul 2021.

[RSUW20] Aaron Roth, Aleksandrs Slivkins, Jonathan Ullman, and Zhiwei Steven Wu. Multidimensional dynamic pricing for welfare maximization. *ACM Transactions on Economics and Computation (TEAC)*, 8(1):1–35, 2020.

[SS+12] Shai Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

[Uza60] Hirofumi Uzawa. Walras' tatonnement in the theory of exchange. *The Review of Economic Studies*, 27(3):182–194, 1960.

[YZ24] Anatoly Yakovenko and Tao Zhu. Simd-110: Exponential fee for write lock accounts, 2024.

A Alternative multiplicative choice function

Instead of setting a bound on the gradients such that the multiplicative update choice function of §3.1 has arguments $z \in \mathbf{R}^m$ with $\|z\|_\infty \leq \log(M)$, we can instead choose the function F in the following way: if we know that $\|p\|_\infty \leq M$ (or if we wish to have a rule that satisfies this condition) then we can set the function F to be, instead,

$$F(z) = \sum_{i=1}^{m} \min\{(p_0)_i \exp(z_i), M(z_i - \log(M/(p_0)_i) + 1)\}.$$

We then have that

$$(\nabla F(z))_i = \begin{cases} (p_0)_i \exp(z_i) & z_i \leq \log(M/(p_0)_i) \\ M & \text{otherwise.} \end{cases}$$

It is not hard to show that F is indeed convex by noting that it is both separable and the derivative of any one term is nondecreasing by construction. Additionally, F is smooth with constant mM, which is easy to see by noting that, again, F is separable and the second derivative of each term is bounded from above by M. By construction, we will have that any

$$\tilde{p} \in \nabla F(z),$$

satisfies $\|\tilde{p}\|_\infty \leq M$, as required, while

$$F^*(p) = \sum_{i=1}^{m} \left(p_i \log\left(\frac{p_i}{(p_0)_i}\right) - p_i \right),$$

whenever $0 \leq p \leq M\mathbf{1}$ (where we define $0 \log 0 = 0$ for convenience).

B Noncounting lower bound

In this appendix, we give a simple lower bound for the expected value of a random walk. More specifically, given $\varepsilon'_t \in \{\pm 1\}$ chosen uniformly and independently for $t = 1, \ldots, T$, we will show that there exists a constant $C' \geq 1/12$ such that

$$\mathbf{E}\left[\left|\sum_{t=1}^{T} \varepsilon'_t\right|\right] \geq C'\sqrt{T}.$$

Since ε'_t is symmetric about 0 (*i.e.*, ε'_t and $-\varepsilon'_t$ have the same distribution) then the positive part would satisfy

$$\mathbf{E}\left[\left(\sum_{t=1}^{T}\varepsilon'_t\right)_+\right] \geq \frac{C'}{2}\sqrt{T} \geq \frac{1}{24}\sqrt{T},$$

given the above. This material is a shorter version of [Ang23].

Proof. For succinctness, we will write the sum as

$$X = \sum_{t=1}^{T}\varepsilon'_t.$$

The odd moments of X vanish, while the second and fourth moments are

$$\mathbf{E}[X^2] = T, \quad \mathbf{E}[X^4] = 3T^2 - 2T \leq 3T^2.$$

which is easy to see from expanding the sum and noting that any term containing an odd power of ε'_t has expectation equal to zero. The bound will come from the simple observation that, for any $a \geq 0$,

$$\mathbf{E}[|X|] \geq a\,\Pr(|X| \geq a). \tag{21}$$

We bound the second term in the inequality from below by expanding the second moment,

$$\mathbf{E}[X^2] \leq a^2 + \mathbf{E}[X^2\mathbf{1}_{|X|\geq a}] \leq a^2 + \sqrt{\mathbf{E}[X^4]}\sqrt{\mathbf{E}[\mathbf{1}_{|X|\geq a}]} = a^2 + \sqrt{\mathbf{E}[X^4]}\sqrt{\Pr(|X| \geq a)}, \tag{22}$$

for any fixed constant $a \geq 0$, where $\mathbf{1}_{|X|\geq a}$ is the zero-one indicator function for the event $|X| \geq a$. Here, the first inequality follows from the pointwise inequality

$$X^2 \leq a^2\mathbf{1}_{|X|<a} + X^2\mathbf{1}_{|X|\geq a},$$

while the second inequality follows from an application of Cauchy–Schwarz to the latter term and the fact that $\mathbf{1}^2_{|X|\geq a} = \mathbf{1}_{|X|\geq a}$. Rearranging the second moment bound (22) above, we find

$$\Pr(|X| \geq a) \geq \frac{(\mathbf{E}[X^2] - a^2)^2}{\mathbf{E}[X^4]} \geq \frac{(T - a^2)^2}{3T^2},$$

where we have implicitly assumed that $a^2 \leq T$. Plugging this in to the basic bound (21) on the expectation gives, for any $0 \leq a \leq \sqrt{T}$,

$$\mathbf{E}[|X|] \geq a\frac{(T - a^2)^2}{3T^2}.$$

The right hand quantity is maximized at $a = \sqrt{T/5}$ (which is easy to see by using the first-order optimality conditions applied to a) so we get

$$\mathbf{E}[|X|] \geq \frac{16}{75\sqrt{5}}\sqrt{T} \geq \frac{1}{12}\sqrt{T},$$

which means that $C' \geq 1/12$, as required. We note that this constant is loose, as a simple counting argument shows that, in fact

$$C' \sim \sqrt{2/\pi},$$

for T large enough, but this bound is enough to show the lower bound holds up to a constant.