Optimality in Blockchain Fee Markets

Theo Diamandis and Guillermo Angeris

Columbia Cryptoeconomics Workshop, 2023

Question: how do we set fees in an unknown, possibly adversarial environment?

Question: how do we set fees in an unknown, possibly adversarial environment?

Answer: online convex optimization!

(a no-regret algorithm)

Outline

Multidimensional fee markets

The resource allocation problem

Why gradient descent?

Formalizing multidimensional fee markets

- A **transaction** j consumes a vector of resources $a_j \in \mathbb{R}_+^m$
  - Entry $(a_j)_i$ denotes the amount of resource i consumed by tx j

Formalizing multidimensional fee markets

- A **transaction** j consumes a vector of resources $a_j \in \mathbb{R}_+^m$
  - Entry $(a_j)_i$ denotes the amount of resource i consumed by tx j
- The vector $x \in \{0,1\}^n$ records which of n possible txns are included in a block
  - Entry $x_j = 1$ if tx j is included and 0 otherwise

Formalizing multidimensional fee markets

- A **transaction** j consumes a vector of resources $a_j \in \mathbb{R}_+^m$
  - Entry $(a_j)_i$ denotes the amount of resource i consumed by tx j
- The vector $x \in \{0,1\}^n$ records which of n possible txns are included in a block
  - Entry $x_j = 1$ if tx j is included and 0 otherwise
- The quantity of resources consumed by this block is then

$$y = \sum_{j=1}^{n} x_j a_j = Ax$$

Formalizing multidimensional fee markets

- A **transaction** j consumes a vector of resources $a_j \in \mathbb{R}^m_+$
  - Entry $(a_j)_i$ denotes the amount of resource i consumed by tx j
- The vector $x \in \{0, 1\}^n$ records which of n possible txns are included in a block
  - Entry $x_j = 1$ if tx j is included and 0 otherwise
- The quantity of resources consumed by this block is then

$$y = \sum_{j=1}^{n} x_j a_j = Ax$$

- Each txn j has a utility q_j if included $\implies$ block utility is $q^T x$

We constrain & charge for each resource used

- Define a set of allowable transactions $S \subseteq \{0, 1\}^n$
  - Network constraints, *e.g.*, $Ax \leq b$
  - Interactions among txns, *e.g.*, bidders for MEV opportunity

We constrain & charge for each resource used

- Define a set of allowable transactions $S \subseteq \{0, 1\}^n$
  - Network constraints, *e.g.*, $Ax \le b$
  - Interactions among txns, *e.g.*, bidders for MEV opportunity
- Charge for usage of each resource (burned)
  - Prices p, mean that transaction j costs (this is burned, *i.e.*, this is the base fee)

$$p^T a_j = \sum_{i=1}^{m} p_i (a_j)_i$$

We constrain & charge for each resource used

- Define a set of allowable transactions $S \subseteq \{0,1\}^n$
  - Network constraints, *e.g.*, $Ax \leq b$
  - Interactions among txns, *e.g.*, bidders for MEV opportunity
- Charge for usage of each resource (burned)
  - Prices p, mean that transaction j costs (this is burned, *i.e.*, this is the base fee)

$$p^T a_j = \sum_{i=1}^{m} p_i (a_j)_i$$

- How do we determine the price update rule? (*e.g.*, EIP-1559)

Update rules are implicitly solving an optimization problem

Update rules are implicitly solving an optimization problem

Specific choice of objective by network designer $\implies$ specific update rule

Outline

Multidimensional fee markets

The resource allocation problem

Why gradient descent?

The resource allocation problem

$$
\begin{array}{ll}
\text{maximize} & q^T x - \ell(y) \\
\text{subject to} & y = Ax \\
& x \in \mathrm{conv}(S).
\end{array}
$$

The resource allocation problem

$$\begin{array}{ll} \text{maximize} & q^T x - \ell(y) \\ \text{subject to} & y = Ax \\ & x \in \mathrm{conv}(S). \end{array}$$

- **Objective:** Maximize utility of included txns minus the loss incurred by the network

The resource allocation problem

$$\begin{array}{ll} \text{maximize} & q^T x - \ell(y) \\ \text{subject to} & y = Ax \\ & x \in \mathbf{conv}(S). \end{array}$$

- **Objective:** Maximize utility of included txns minus the loss incurred by the network
- **Constraints:** Utilization y is resource usage of included txns, and x is in the set of allowable txns $S \subseteq \{0, 1\}^n$ (can be very complex/hard to solve!)

Dual problem decouples tx producers and network

- Dual problem is to find the resource prices p that minimize dual function $g(p)$
- Duality theory: dual problem has same optimal value as original problem

Dual problem decouples tx producers and network

- Dual problem is to find the resource prices p that minimize dual function $g(p)$
- Duality theory: dual problem has same optimal value as original problem
- Problem is separable, so $g(p)$ decomposes into two easily interpretable terms:

$$g(p) = \underbrace{\sup_y \left(p^T y - \ell(y)\right)}_{\text{network}} + \underbrace{\sup_{x \in \mathrm{conv}(S)} (q - A^T p)^T x}_{\text{tx producers}}$$

- Evaluating the 1st term is easy (conjugate function): can be done on chain!
- Second term is exactly block building problem; network can observe soln

Cool. So how do we minimize $g(p)$?

- We can compute the gradient:

$$\nabla g(p) = y^\star(p) - Ax^\star(p)$$

Cool. So how do we minimize $g(p)$?

- We can compute the gradient:

$$\nabla g(p) = y^\star(p) - Ax^\star(p)$$

- Network determines $y^\star(p)$ (computationally easy)

Cool. So how do we minimize $g(p)$?

- We can compute the gradient:

$$\nabla g(p) = y^\star(p) - Ax^\star(p)$$

- Network determines $y^\star(p)$ (computationally easy)
- Network observes $x^\star(p)$ from previous block (block building problem soln)

Cool. So how do we minimize $g(p)$?

- We can compute the gradient:

$$\nabla g(p) = y^\star(p) - Ax^\star(p)$$

- Network determines $y^\star(p)$ (computationally easy)
- Network observes $x^\star(p)$ from previous block (block building problem soln)
- Then network applies gradient descent:

$$p^{k+1} = p^k - \eta \nabla g(p^k)$$

Outline

Let's play a game

- Two players: network and block producers

Let's play a game

- Two players: network and block producers
  1. Network chooses prices p^k with gradient descent

Let's play a game

- Two players: network and block producers
  1. Network chooses prices p^k with gradient descent
  2. Users submit txns (with utilities q^k, resources A^k), possibly adversarially!

Let's play a game

- Two players: network and block producers
  1. Network chooses prices p^k with gradient descent
  2. Users submit txns (with utilities q^k, resources A^k), possibly adversarially!
  3. Network receives payoff $g_k(p^k)$ (from duality)

Let's play a game

- Two players: network and block producers
  1. Network chooses prices p^k with gradient descent
  2. Users submit txns (with utilities q^k, resources A^k), possibly adversarially!
  3. Network receives payoff $g_k(p^k)$ (from duality)
- Metric: *regret* of the network ('welfare gap')

$$\frac{1}{T}\left(\sum_{k=1}^{T} g_k(p^k) - \min_{p^\star} \sum_{k=1}^{T} g_k(p^\star)\right)$$

- Interpretation: difference between our update rule and the best fixed prices $p^\star$
  - Knowing $p^\star$ requires omniscience: assumes you know all future txns!

Main result:

- Gradient descent price update with fixed step size $\eta = M/B\sqrt{T}$ gives

$$\frac{1}{T}\left(\sum_{k=1}^{T} g_k(p^k) - \min_{p^\star} \sum_{k=1}^{T} g_k(p^\star)\right) \leq \frac{4MB}{\sqrt{T}}$$

where B and M are constants.

Main result:

- Gradient descent price update with fixed step size $\eta = M/B\sqrt{T}$ gives

$$\frac{1}{T}\left(\sum_{k=1}^{T} g_k(p^k) - \min_{p^\star}\sum_{k=1}^{T} g_k(p^\star)\right) \leq \frac{4MB}{\sqrt{T}}$$

  where B and M are constants.

- Regret is $O(1/\sqrt{T})$ and goes to 0 as T gets large!

Main result:

- Gradient descent price update with fixed step size $\eta = M/B\sqrt{T}$ gives

$$\frac{1}{T}\left(\sum_{k=1}^{T} g_k(p^k) - \min_{p^\star} \sum_{k=1}^{T} g_k(p^\star)\right) \leq \frac{4MB}{\sqrt{T}}$$

  where B and M are constants.

- Regret is $O(1/\sqrt{T})$ and goes to 0 as T gets large!

- This result does not assume any model or notion of stochasticity
  - No assumption that there exists a particular distribution for txns
  - Agents mess with your protocol! Need adversarial bounds.

Main result:

- Gradient descent price update with fixed step size $\eta = M/B\sqrt{T}$ gives

$$\frac{1}{T}\left(\sum_{k=1}^{T} g_k(p^k) - \min_{p^\star} \sum_{k=1}^{T} g_k(p^\star)\right) \leq \frac{4MB}{\sqrt{T}}$$

  where B and M are constants.

- Regret is $O(1/\sqrt{T})$ and goes to 0 as T gets large!

- This result does not assume any model or notion of stochasticity
  - No assumption that there exists a particular distribution for txns
  - Agents mess with your protocol! Need adversarial bounds.

- Online convex optimization shines in this setting (common in blockchains!)
  - Note: does not require that we ever converge to $p^\star$!!!

Conclusion: online convex opt is powerful tool for pricing problems

No difference between ‘correctly’ fixing prices with oracle knowledge of future and using online gradient descent algorithm.

Conclusion: online convex opt is powerful tool for pricing problems

No difference between ‘correctly’ fixing prices with oracle knowledge of future and using online gradient descent algorithm.

These results hold without assumptions of demand distributions or of market-clearing prices!

For more info on multidimensional fees, check out our paper!

Multidimensional Fees Paper

Thank you!

Theo Diamandis

theodiamandis.com

tdiamand@mit.edu

@theo_diamandis