CS 49/149: 21st Century Algorithms (Fall 2018)

Paper: Optimal Auctions via the Multiplicative Weight Method

1 Auctions and Multiplicative Weight Update

New research in the field of mechanism design, traditionally a part of microeconomics and game theory, has recently began to draw inspiration from algorithms in computer science research. Many papers in this emerging sector of "computational mechanism design" are concerned with designing computationally feasible and economically feasible mechanisms to distribute resources or penalties among different actors (Parkes, 2001). In general, the mechanism design problem arises from a situation where a variety of actors compete for the same set of resources, but with incomplete information about each other. In this auction setting, bidders and the auctioneer have certain information only about their own preferences (expressed by type, utility, and valuation functions), and have information about the strategy of other agents in the game only insofar as other agents are willing to reveal their strategies. The goal of a good auction mechanism is to generate the maximum amount of revenue for the auctioneer while ensuring fairness for the bidders (for example, not somehow forcing the winning bidder double the winning bid). Other examples of constraints will be provided in section 2 of this summary. We will be examining this paper, which leverages a general form of the multiplicative weight update algorithm to arrive at an mechanism that exceeds individual bidder budgets by no more than an additive-ϵ error and is generates no less than an additive-ϵ away from the optimal revenue for the auctioneer.

The algorithm proposed in "Optimal Auctions via the Multiplicative Weight Method" (Bhalgat et. al., 2013) is one such example of applying optimization algorithms to mechanism design. The presented algorithm can be used to re-derive recent results results in the field (e.g. auctions with multi-dimensional type spaces and allocation constraints), but also solves many different auction problems where the only known previous solutions were approximation algorithms. Another advantage of this mechanism is that it is relatively simple and doesn't depend on relating different ex-interim variables (allocations, payments, etc.) but instead explicitly encodes all of this information in the dual variables. In general, the proposed algorithm has runtime polynomial in n, the number of buyers, m, the number of items being auctioned, $\frac{1}{\epsilon}$, the reciprocal of the error tolerance, and the size of the types available to each buyer.

2 Constraints

In arriving at an optimal mechanism, we must ensure that several groups of constraints are satisfied. Some are related to the amount of revenue that can be generated by the mechanism, while others are related to individual buyer behavior and mechanism resource allocation. We outline each in subsections.

2.1 Fair Mechanism constraints

First, we present the set of constraints we refer to as "fair mechanism constraints", which constrains the possible allocation of the auctioned-off items. Here, x_{iq} is an indicator function taking 1 if person i receives q items, v_i is the valuation function, taking the number of items and a type space and outputs a real number, m refers to the total number of items, and B_i refers to the budget of bidder i. Then,

$$x_{iq}(\mathbf{t}) \in \{0,1\} \quad \forall i, q \tag{1}$$

$$\sum_q x_{iq} = 1 \qquad \forall i \tag{2}$$

$$\sum_i q x_{iq}(\mathbf{t}) \leq m \tag{3}$$

$$\sum_q v_i(q, t_i) x_{iq}(\mathbf{t}) \geq p_i(\mathbf{t}) \quad \forall i, q \tag{4}$$

$$p_i(\mathbf{t}) \in [0, B_i] \quad \forall i \tag{5}$$

These constraints are collectively referred to as $\mathcal{F}(t)$. All of these together ensure that the solution mechanism is 'fair' (i.e. doesn't sell more items than the auctioneer owns, somehow charge a bidder more than they bid, etc.).

2.2 Bayesian Incentive constraints and optimal revenue constraint

For constraints, we must also keep in mind the Bayesian Incentive Compatibility constraint, which enforces the constraints over bidder behavior, where U_i is the utility function for bidder i, taking in a true type and revealed type and outputting a real number (utility):

$$\begin{aligned} U_i(t_i, t_i) &\geq U_i(t_i', t_i) \qquad \forall t_i, t_i' \in T_i \\ U_i(t_i, t_i) &\geq 0 \qquad\qquad\quad \forall t_i \in T_i \end{aligned}$$

These two above constraints can be combined into the following constraint, referred to as the BIC constraint, where $X_{iq}(t_i)$ is the probability that buyer i gets q items playing type t_i, and $P_i(t_i)$ be the expected payment of the same buyer playing type t_i:

$$\sum_q v_i(q, t_i) X_{iq}(t_i) - P_i(t_i) \geq \sum_q v_i(q, t_i) X_{iq}(t_i') - P_i(t_i') \qquad \forall i, t_i, t_i' \in T_i \tag{6}$$

We must also check to make sure our approximation generates the same amount of revenue as the optimal mechanism for the auctioneer. The authors denote the optimal amount of revenue for any given set of constraints as OPT. Thus, the last constraint can be denoted

$$\sum_{i, t_i \in T_i} f_i(t_i) P_i(t_i) \geq OPT \tag{7}$$

where $f_i(t_i)$ = the probability that the buyer i is of strategy type t_i.

2.3 Equality Constraints

Finally, we must also enforce equality constraints, which make the mechanism consistent with itself. The two constraints in this domain are as follows:

$$f_i(t_i)X_{iq}(t_i) = \sum_{\mathbf{t}|t_i \in \mathbf{t}} \mu(\mathbf{t}) x_{iq}(\mathbf{t})$$
$$f_i(t_i)P_i(t_i) = \sum_{\mathbf{t}|t_i \in \mathbf{t}} \mu(\mathbf{t}) p_i(\mathbf{t})$$

where t_i is a particular strategy within $\mathbf{t}$, a vector of reported strategies. The first constraint says that the probability that buyer i is allocated q items under strategy t_i times the probability of buyer i playing strategy t_i is equal to sum of the densities where t_i is a strategy in $\mathbf{t}$ (since x_{iq} is an indicator function). Likewise, the second condition says that the expected payment from buyer i after playing type i times the probability of choosing type i is equal to the sum of payments under every type vector containing strategy type t_i times the payments under those scenarios.

3 Oracle Subproblem

The efficiency of the overall algorithm requires us to solve the Oracle subproblem in an efficient manner. $Oracle(\alpha, \beta)$ is the same as solving

$$\begin{aligned} \max \quad & -\sum_{i,t_i \in T_i} f_i(t_i) \left(\sum_q \alpha_{iqt_i} X_{iq}(t_i) + \beta_{it_i} P_i(t_i) \right) \\ & + \sum_t \mu(t) \left(\sum_i \left(\sum_q (a_{iqt_i} x_{iq}(\mathbf{t})) + \beta_{it_i} p_i(\mathbf{t}) \right) \right) \end{aligned}$$

subject to the same constraints as outlined above in section 2 (the mechanism and bayesian incentive/optimal revenue constraints). We notice that this problem breaks down into two separate optimization problems, namely minimizing the quantity in the first line and maximizing the quantity in the second line. The first optimization problem (i.e. minimizing the negative of the first line) can be solved with Lagrangian multipliers. The authors have omitted the workings to the procedure.

Let's now examine this second optimization problem (i.e. maximizing the second line), constrained by $\mathcal{F}(\mathbf{t})$, which the authors denote $\mathcal{A}(\alpha, \beta)$. A smart observation can be made about $\mathcal{A}(\alpha, \beta)$: $\beta_{it_i} \leq 0 \implies p_i(\mathbf{t}) = 0$, and $\beta_{it_i} > 0 \implies p_i(\mathbf{t}) = \min\{B_i, v_i(q, t_i)\}$. Then, we can solve this problem in polynomial time via dynamic programming. We fill in a matrix $A[i, k]$ where $A[i, k] := \mathcal{A}(\alpha, \beta)$ for buyers i through n and for 0 through k items. We fill in the table from the nth person backwards, using the folowing rule:

$$A[i, k] = \min_{0 \leq j \leq k} (A[i + 1, k - j] + j \times \alpha_{ijt_i} F + \max\{\beta_{it_i} \times \min\{B_i, v_i(j, t_i)\}, 0\})$$

4 Algorithm

Let L be a quantity bounding the absolute value of all payments (buyer budgets), valuations, and utilities for any possible mechanism. We know that $\max_i |T_i| \leq L$, and thus the number of equality constraints are bounded by $2nmL$. The support for any type for a buyer is also bounded by $1/L$. Define α^l, β^l to be the dual variables in iterations $1...K$ when running MWU. Given an ϵ, our procedure is as follows:

MECHANISM(**T**)

- run sampled multiplicative weight update algorithm (see below)
- set $\delta = \frac{\epsilon}{nmL}$ and $K = O(\frac{L^2 \log(nmL)}{\delta^2})$
- choose an integer l between 1 and K uniformly at random
- allocate items and payments according to the solution of $\mathcal{A}(\alpha^l, \beta^l)$

Claim: This mechanism yields an ϵ-BIC, ex-post individual rational mechanism such that a buyer never pays more than their budget and expected revenue for the auctioneer $OPT - \epsilon$

Proof. By the properties of the multiplicative update algorithm, each constraint in the set of equality constraints will have an error of at most $\epsilon/(nL)$. Since we have bounded $v_i(q, t_i)$ by L for all possible combinations of people and items, the error in each BIC constraint is bounded by ϵ/n, and since payments are bounded by L as well, the revenue is at most ϵ away from optimal. □

Note that each constraint in the set of equality constraints is exponential in the number of buyers. We can relax each constraint in this set of constraints by $\delta = \frac{\epsilon}{mnL}$ to produce the following contraints:

$$X_{iq}(t_i) \in \frac{\sum_{\mathbf{t}|t\in\mathbf{t}} \mu(\mathbf{t}) x_{iq}(\mathbf{t})}{f_i(t_i)} \pm \delta$$

$$P_i(t_i) \in \frac{\sum_{\mathbf{t}|t\in\mathbf{t}} \mu(\mathbf{t}) p_i(\mathbf{t})}{f_i(t_i)} \pm \delta$$

To cut down on runtime, we use a sampled version of multiplicative weight update method which runs in polynomial time:

SAMPLED MWU

- $S_l := O(nmL^3 log(nmLK)/\epsilon)$ samples from data. Let the size of this set be C.
- solve Oracle(α^l, β^l) using sampled version of equality constraints
- rest of weight updates are run the same as normal MWU (see Rui's notes)

Before we prove correctness of this procedure, let us observe a lemma that will be helpful.

Lemma: With high probability (i.e. with probability $\geq 1 - 1/poly(nmLK/\epsilon)$), for all $1 \leq l \leq K$, we have:

$$\left| \frac{\sum_{t \in S_{it_i l}} x^l_{iq}(\mathbf{t})}{|S_{it_i l}|} - \frac{\sum^l_{t \in D | t_i \in T} \mu(\mathbf{t}) x^l_{iq}(\mathbf{t})}{f_i(t_i)} \right| \leq \delta$$
$$\left| \frac{\sum_{t \in S_{it_i l}} x^*_{iq}(\mathbf{t})}{|S_{it_i l}|} - \frac{\sum^l_{t \in D | t_i \in T} \mu(\mathbf{t}) x^*_{iq}(\mathbf{t})}{f_i(t_i)} \right| \leq \delta$$

where x^* denotes the feasible solution for the linear program with no slack

Proof. Let A_{it_i} be the event that $|S_{it_i l}| \notin (1 \pm \epsilon)C \times f_i(t_i)$, where $C = O(nmL^3 log(nmLK)/\epsilon)$ and $N_{it_i l}$ be the event that one sample isn't of type A_{it_i}. Fix i, t_i, l. We apply Hoeffding's bounds here: $\mathbb{P}(\frac{1}{l}\sum^l_{i=1}(N_{it_i l} - \mathbb{E}[N_{it_i l}]) \geq t) \leq 2exp(-2lt^2)$. Using linearity of expectation, we notice that the left hand side is the the deviation from $C \times f_i(t_i)$, which is the quantity we want to bound. Plugging in $\epsilon C \times f_i(t_i)$ for t, we note that the right hand turns into a polynomial function of n, m, K, L since C is a polynomial function of the same variables and $f_i(t_i) \geq 1/L$. Using this and the fact $e^{-x} \leq x + 1, x > 0$. Thus, $\mathbb{P}(A_{it_i}) = poly(nmLK/\epsilon)$ Thus, $\forall i, t_i, l, |S_{it_i l}| = \Omega(nmL^2K/\epsilon)$ w.h.p. (omitting lower order terms here).

$x^l_{iq}(\mathbf{t})$ is generated by solving $\mathcal{A}(\alpha^l, \beta^l)$. Since S_l is chosen independently of previous rounds and of α^l, β^l, the values for $x_i q(\mathbf{t})$ can be viewed as drawing values from a distribution parameterized by α^l, β^l. As each $x_{iq}(\mathbf{t}) \in \{0, 1\}$, we can once again apply Hoeffding's inequality and union bound to prove the first equation.

The proof for the second equation is similar, noting that $x^*_{iq}(\mathbf{t}) \in \{0, 1\}$ and sampling valuation vectors is the same as sampling from distribution $\{x^*_{iq}\}$. We can again apply Hoeffding's inequality and union bound. (The authors omit the full proof for all three of these). □

Using this, the authors establish correctness of the algorithm by examining an arbitrarily chosen constraint:

$$\frac{\sum_{\mathbf{t} \in S_{it_i l}} x_{iq}(\mathbf{t})}{|S_{it_i l}|} - X_{iq}(t_i) \geq -\delta \quad \forall i, t_i \in T_i$$
$$\implies \frac{1}{K} \times \sum_{1 \leq l \leq K} \left(\frac{\sum_{\mathbf{t} \in S_{it_i l}} x_{iq}(\mathbf{t})}{|S_{it_i l}|} - X_{iq}(t_i) \right) \geq -\delta - \delta \qquad \text{(General property of MWU)}$$
$$\implies \sum_{i \leq l \leq K} \frac{\sum_{\mathbf{t} \in D | t_i \in \mathbf{t}} \mu(\mathbf{t}) x^l_{iq}(\mathbf{t})/f_i(t_i) - X^l_{iq}(t_i)}{K} \geq -3\delta \qquad \text{(above lemma)}$$
$$\implies \sum_{\mathbf{t} \in D | t_i \in \mathbf{t}} \frac{\mu(\mathbf{t})}{f_i(t_i)} \times \sum_{1 \leq l \leq K} \frac{x^l_{iq}(\mathbf{t})}{K} - \sum_{1 \leq l \leq K} \frac{X^l_{iq}(t_i)}{K} \geq -3\delta$$

Which implies that the algorithm is ϵ-BIC with revenue $OPT - \epsilon$.

References

Arora, S., Hazan, E., & Kale, S. (2012). The Multiplicative Weights Update Method: a Meta-Algorithm and Applications. Theory of Computing, 8(1), 121-164.

Bhalgat, A., Gollapudi, S., & Munagala, K. (2013, June). Optimal auctions via the multiplicative weight method. In Proceedings of the fourteenth ACM conference on Electronic commerce (pp. 73-90). ACM.

Parkes, D. C. (2001). Classic mechanism design. Iterative Combinatorial Auctions: Achieving Economic and Computational Efficiency. Ph. D. dissertation, University of Pennsylvania.