CSC3160: Design and Analysis of Algorithms

Week 12: Online Algorithms

Instructor: Shengyu Zhang

Offline algorithms

- Almost all algorithms we encountered in this course assume that the entire input is given all at once.
- An exception: Secretary problem.
  - The input is given gradually.
  - We need to respond to each candidate in time.
  - We care about our performance compared to the best one in hindsight.

Online algorithms

- The input is revealed in parts.
- An online algorithm needs to respond to each part (of the input) upon its arrival.
- The responding actions cannot be canceled/revoked later.
- We care about the competitive ratio, which compares the performance of an online algorithm to that of the best offline algorithm.
  - Offline: the entire input is given beforehand.

Ski rental

- A person goes to a ski resort for a long vacation.
- Two choices everyday:
  - Rent a ski: \$1 per day.
  - Buy a ski: \$$B$ once.
- An unknown factor: the number k of remaining days for ski in this season.
  - When snow melts, the ski resort closes.

Offline algorithm

- If we had known k, then it's easy.
  - If $k < B$, then we should rent everyday. The total cost is k.
  - If $k \geq B$, then we should buy on day 1. The total cost is B.
- In any case, the cost is $\min\{k, B\}$.
- Question: Without knowing k, how to make decision every day?

Deterministic algorithm

- There is a simple deterministic algorithm s.t. our cost is at most $2 \cdot \min\{k, B\}$.
  - We then say that the algorithm has a competitive ratio of 2.
- Algorithm:
  On each day $j < B$, rent.
  On day B, buy.
- If $k < B$, then our cost is k, which is optimal.
- If $k \geq B$, then our cost is
  $$B - 1 + B = 2B - 1 < 2B = 2 \cdot \min\{k, B\}$$

Randomized algorithm

- It turns out to exist a randomized algorithm with a competitive ratio of $\frac{e}{e-1} \approx 1.58$
- The algorithm uses integer programming and linear programming.

Integer programming

- There is an integer programming to solve the offline version of the ski-rental problem.
- We introduce some variables $x, z_1, z_2, \ldots, z_k \in \{0,1\}$.
  - x: indicate whether we eventually buy it.
  - z_i: indicate whether we rent on day i.
- IP:

$$
\begin{aligned}
\min \quad & B \cdot x + \sum_{j=1}^{k} z_j \\
s.t. \quad & x + z_j \geq 1, && \forall j \in [k] \\
& x, z_j \in \{0,1\} && \forall j \in [k]
\end{aligned}
$$

Solution

- It's not hard to see that the optimal solution to the IP is

$$\begin{cases} x = 0, z_j = 1, & \text{if } k < B \\ x = 1, z_j = 0, & \text{if } k \geq B \end{cases}$$

  - same as the previous optimal solution for the offline problem.
- So the IP does solve the offline problem.

Relaxation

- Relax it to LP.
- IP:

$$
\begin{aligned}
\min \quad & B \cdot x + \sum_{j=1}^{k} z_j & \\
s.t. \quad & x + z_j \ge 1, & \forall j \in [k] \\
& x, z_j \in \{0,1\} & \forall j \in [k]
\end{aligned}
$$

- LP:

$$
\begin{aligned}
\min \quad & B \cdot x + \sum_{j=1}^{k} z_j & \\
s.t. \quad & x + z_j \ge 1, & \forall j \in [k] \\
& x \ge 0, z_j \ge 0, & \forall j \in [k]
\end{aligned}
$$

The relaxation doesn't lose anything

- It is easily observed that the LP has the following optimal solution

$$\begin{cases} x = 0, z_j = 1, & \text{if } k < B \\ x = 1, z_j = 0, & \text{if } k \geq B \end{cases}$$

- This is the same as the optimal solution to the IP.
- So the LP relaxation doesn't lose anything.

Dual LP

Primal

$$\begin{aligned} \min \quad & Bx + \textstyle\sum_{j=1}^{k} z_j & \\ s.t. \quad & x + z_j \geq 1, & \forall j \\ & x \geq 0, z_j \geq 0, & \forall j \end{aligned}$$

Dual

$$\begin{aligned} \max \quad & \textstyle\sum_{j=1}^{k} y_j & \\ s.t. \quad & \textstyle\sum_{j=1}^{k} y_j \leq B & \forall j \\ & y_j \in [0,1] & \forall j \end{aligned}$$

- Consider the following algorithm, which defines variables x, y_j, z_j.
- $x = 0, y_j = 0$

  for each new $j = 1,2, \dots, k$

      if $x < 1$

          $x \leftarrow x + \frac{x}{B} + \frac{1}{cB}$, where $c = \left(1 + \frac{1}{B}\right)^B - 1$

          $z_j = 1 - x$

          $y_j = 1$
- Output $x, y_1, \dots, y_k, z_1, \dots, z_k$.

Property 1

- Theorem. The above algorithm produces a feasible solution (x, z_j) to Primal LP and a feasible solution y_j to Dual LP.
- Feasible to Primal LP:
  - $x \geq 0$ always holds.
  - $z_j = 1 - x > 0$ always holds since we assign $z_j = 1 - x$ only if $x < 1$.
  - $x + z_j = 1$ when $x < 1$, and $x + z_j \geq x \geq 1$ when $x \geq 1$. So $x + z_j \geq 1$ always holds.

Property 1

- Theorem. The above algorithm produces a feasible solution (x, z_j) to Primal LP and a feasible solution y_j to Dual LP.
- Feasible to Dual LP:
  - $y_j \in \{0,1\} \subseteq [0,1]$.
  - To show $\sum_j y_j \leq B$, we need to show that the algorithm stops after $\leq B$ iterations.

- Consider $x_j \stackrel{\text{def}}{=}$ the increment of x in iteration j.
- $x_1 = \frac{1}{cB}$, $x_2 = \frac{x_1}{B} + \frac{1}{cB} = \frac{1}{cB}\left(1 + \frac{1}{B}\right)$.
- In general, it's not hard to prove that

$$x_j = \frac{1}{cB}\left(1 + \frac{1}{B}\right)^{j-1}$$

- So after B iterations, x increases to

$$\sum_{j=1}^{B} \frac{1}{cB}\left(1 + \frac{1}{B}\right)^{j-1} = \frac{\left(1+\frac{1}{B}\right)^{B} - 1}{c} = 1.$$

- So only the first B dual variables $y_j = 1$, resulting in $\sum_j y_j = B$. Thus y is dual feasible.

Property 2

- The outputted variables x, y_j, z_j satisfy

$$\overbrace{Bx + \sum_j z_j}^{\text{primal obj value}} \leq \left(1 + \frac{1}{c}\right) \overbrace{\sum_j y_j}^{\text{dual obj value}}$$

- Actually, we will show something stronger: In every iteration, the increment of primal obj value is at most $(1 + 1/c)$ · that of dual.
- The increment of dual is always $y_j = 1$ before x reaches 1.

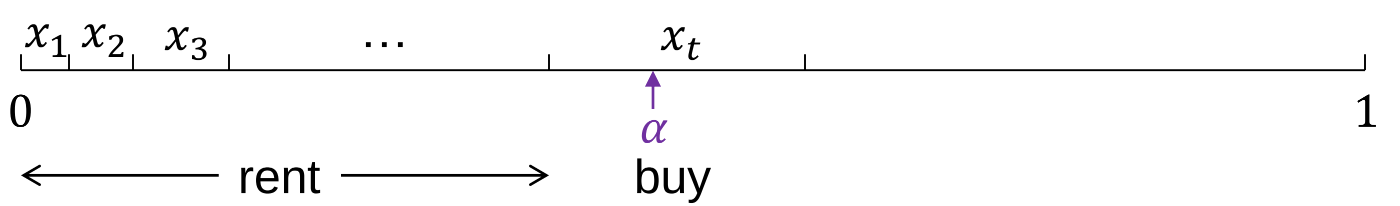
- The increment of primal is

$$Bx_j + z_j = x_{<j} + \frac{1}{c} + 1 - x_{\le j} \le 1 + 1/c.$$

  - $x_{<j} = \sum_{i=1}^{j-1} x_i$ and $x_{\le j} = \sum_{i=1}^{j} x_i$ are the x before and after iteration j, respectively.
  - Recall update: $x \leftarrow x + \frac{x}{B} + \frac{1}{cB}$. So $Bx_j = x_{<j} + \frac{1}{c}$.
  - Recall update: $z_j = 1 - x$. So $z_j = 1 - x_{\le j}$.
- So the increment of primal obj value is at most $(1 + 1/c) \times$ that of dual.

Turning into an online algorithm

- The above algorithm just gives (x, z_j, y_j).
- Now we give an online algorithm based on it.
- Pick $\alpha \in [0,1]$ uniformly at random.
- Suppose t is the first day that $\sum_{j=1}^{t} x_j \geq \alpha$, then rent in all days before t and buy on day t.



Expected cost

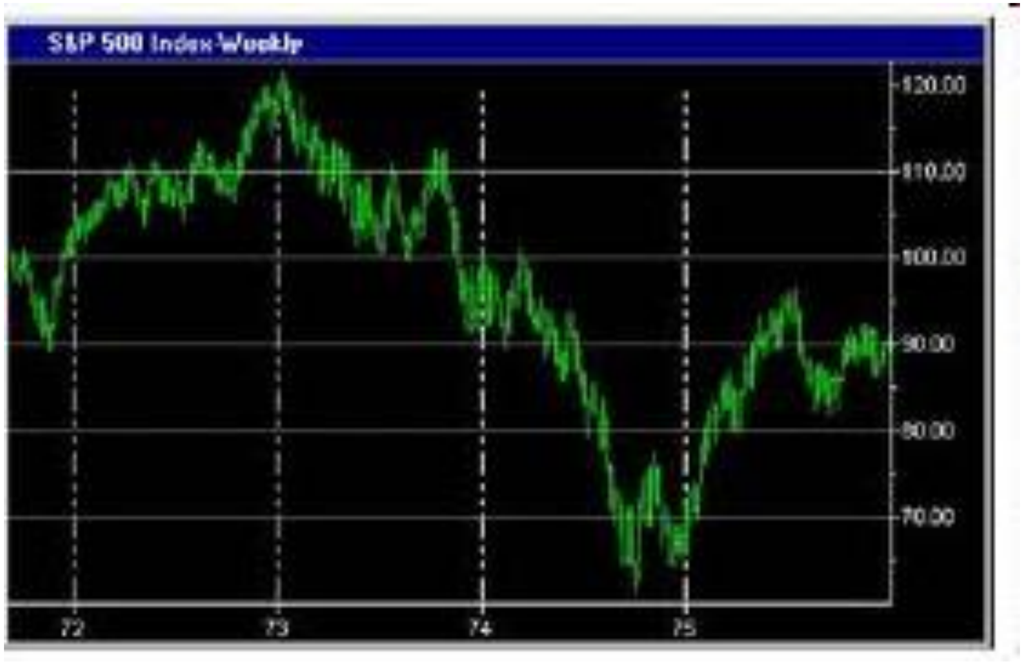
- Theorem. $\mathbf{E}[cost] \le \left(1 + \frac{1}{c}\right) \mathrm{OPT}$.
- There are two costs. One is buying cost, and the other is renting cost.
- Obs. $\mathbf{Pr}[\text{buy in day } \boldsymbol{i}] = x_i$.
- So $\mathbf{E}[buying\ cost] = B \sum_{j=1}^{k} x_i = Bx$, the first term of the obj function of Primal.
- $\mathbf{Pr}[\text{rent in day } j] = \mathbf{Pr}[\text{no buy in days } 1, \dots, j]$ $= 1 - \sum_{i=1}^{j} x_i \le 1 - \sum_{i=1}^{j-1} x_i = z_j$.

- So $\mathbf{E}[renting\ cost] = \sum_{j=1}^{k} z_j$, the second term of the obj function of Primal.
- $\mathbf{E}[cost] = \mathbf{E}[buying\ cost] + \mathbf{E}[renting\ cost]$
  $= Bx + \sum_{j=1}^{k} z_j$, the objective function value.
- So $\mathbf{E}[cost]$
  $= Primal\ obj$ // above
  $\leq \left(1 + \frac{1}{c}\right) dual\ obj$ // Property 2
  $\leq \left(1 + \frac{1}{c}\right) OPT$. // dual feasible $\leq$ OPT.

- So the online algorithm achieves a competitive ratio of $\left(1+\frac{1}{c}\right)$.
- Recall that $c = (1 + 1/B)^B - 1$, which is close to $e - 1$ for large B.
- Thus the competitive ratio is $1 + \frac{1}{c} = \frac{e}{e-1} \approx 1.58$, as claimed.

- Optimality: Both deterministic and randomized algorithms are optimal.
  - No better competitive ratio is possible.
- Reference: **The design of competitive online algorithms via a primal dual approach**, Niv Buchbinder and Joseph Naor, *Foundations and Trends in Theoretical Computer Science*, Vol. 3, pp. 93-263, 2007.
- Next: Another learning algorithm

Stock market



- Simplification: Only consider up or down.

Which expert to follow?

- Each day, stock market goes up or down.

- Each morning, n "experts" predict the market.
- How should we do? Whom to listen to? Or combine their advice in some way?

Which expert to follow?

- Each day, stock market goes up or down.

- At the end of the day, we'll see whether the market actually goes up or down.
- We lose 1 if our prediction was wrong.

- After a year, we'll see with hindsight that one expert is the best.
  - But, of course, we don't know who in advance.
- We'll think "If we had followed his advice…"
- Theorem: We have a method to perform close to the best expert!
  - We don't assume anything about the experts.
    - They may not know what they are talking about.
    - They may even collaborate in any bad manner.

Method and intuition

- Algorithm: *Randomized Weighted Majority*

- Use random choice: following expert i with probability p_i
- If an expert predicts wrongly: punish him by decreasing the probability of choosing him/her in next round.
  - If someone gives you wrong info, then you tend to trust him less in future.

Randomized Weighted Majority

$w_i^{(t)}$: weight of expert i at time t

$p_i^{(t)}$: probability of choosing expert i at time t

- **for** each $i \in [n]$

  $$w_i^{(1)} = 1, \; p_i^{(1)} = 1/n$$

- **for** each $t > 1$, $\forall i \in [n]$:
  - **if** expert i was wrong at step $t-1$

    $$w_i^{(t)} = w_i^{(t-1)}(1-\varepsilon)$$

    Decrease your weight!

    else

    $$w_i^{(t)} = w_i^{(t-1)}$$

  - $p_i^{(t)} = w_i^{(t)} / \sum_i w_i^{(t)}$

    Probability is proportional to weight

  - Choose i with prob. $p_i^{(t)}$, and follow expert i's advice.

Example (n=5, T=6, ε = 1/4)

	1	2	3	4	5	our	real
1	1, ↑	1, ↑	1, ↓	1, ↑	1, ↓	↑	↑
2	1, ↑	1, ↓	0.75, ↑	1, ↑	0.75, ↑	↑	↑
3	1, ↑	0.75, ↑	0.75, ↓	1, ↓	0.75, ↑	↓	↓
4	0.75, ↑	0.5625, ↑	0.75, ↓	0.75, ↓	0.5625, ↑	↑	↓
5	0.5625, ↓	0.4219, ↑	0.75, ↑	0.75, ↓	0.4219, ↓	↓	↑
6	0.4219, ↑	0.4219, ↑	0.75, ↓	0.5625, ↑	0.3164, ↑	↓	↓
loss	4	4	1	2	5	2	

- Numbers: weight
- Arrows: predications. Red: wrong.

- L_{RWM}: expected loss of our algorithm
- L_{min}: loss of the best expert

- Theorem. For $\epsilon < 1/2$, the loss on any sequence of $\{0,1\}$ in time T satisfies

$$L_{RWM} \leq (1+\epsilon)L_{min} + \ln(n)/\epsilon.$$

Proof

- Key: Consider the total weight $W^{(t)}$ at time t.
- Fact: Any time our algorithm has significant expected loss, the total weight drops substantially.
- $l_i^{(t)}$: 1 if expert i is wrong at step t (and 0 otherwise)
- Let $F^{(t)} = (\sum_{i:l_i^{(t)}=1} w_i^{(t)})/W^{(t)}$. Two meanings:
  - The fraction of the weight on wrong experts
  - The expected loss of our algorithm at step t
- Note: $W^{(t+1)} = F^{(t)}W^{(t)}(1-\epsilon) + (1-F^{(t)})W^{(t)}$

  $= W^{(t)}(1-\epsilon F^{(t)})$

- Last slide: $W^{(t+1)} = W^{(t)}\left(1 - \epsilon F^{(t)}\right)$
- So $W^{(T+1)} = W^{(T)}(1 - \epsilon F^{(T)})$

$$= W^{(T-1)}\left(1 - \epsilon F^{(T-1)}\right)\left(1 - \epsilon F^{(T)}\right)$$

$$= \ \ldots$$

$$= W^{(1)}(1 - \epsilon F^{(1)}) \ \ldots (1 - \epsilon F^{(T)})$$

- On the other hand,

$$W^{(T+1)} \geq \max_i w_i^{(T+1)} = (1-\epsilon)^{L_{min}^{(T)}}$$

- So $(1-\epsilon)^{L_{min}^{(T)}} \leq W^{(1)}(1-\epsilon F^{(1)}) \ldots (1-\epsilon F^{(T)})$
- Note: $L_{min}^{(T)}$ is the loss of the best expert.

$$(1-\epsilon)^{L_{min}^{(T)}} \le W^{(1)}(1-\epsilon F^{(1)}) \ldots (1-\epsilon F^{(T)})$$

- Note that $W^{(1)} = n$ since $w_i^{(1)} = 1, \forall i$
- Take log:

$$L_{min}^{(T)} \ln(1-\epsilon) \le \ln(n) + \sum_{t=1,\ldots,T} \ln(1-\epsilon F^{(t)})$$

$$\le \ln(n) - \sum_{t=1,\ldots,T} \epsilon F^{(t)} \quad \because \ln(1-z) \le -z$$

$$= \ln(n) - \epsilon L_{RWM}^{(T)} \qquad \because L_{RWM}^{(T)} = \sum_{t=1,\ldots,T} F^{(t)}$$

  - $L_{RWM}^{(T)}$ is the loss of our algorithm.

- Rearranging the inequality and using

$$-\ln(1-z) \le z + z^2, \qquad 0 \le z \le 1/2$$

we get the inequality in the theorem.

$$L_{RWM} \le (1+\epsilon)L_{min} + \ln(n)/\epsilon.$$

Extensions

- The case that T is unknown.
- The case that loss is in $[0,1]$ instead of $\{0,1\}$

- References:
  - **The Multiplicative Weights Update Method: a Meta-Algorithm and Applications**, Sanjeev Arora, Elad Hazan, and Satyen Kale, Theory of Computing, Volume 8, Article 6 pp. 121-164, 2012.
  - Chapter 4 of *Algorithmic Game Theory*, available at http://www.cs.cmu.edu/~avrim/Papers/regret-chapter.pdf

Summary

- Online algorithms:
  - The input is revealed in parts.
  - We need to respond to each part upon its arrival.
  - The responding actions cannot be revoked later.
- competitive ratio: performance of an online algorithm vs. performance of the best offline algorithm.
- Primal-dual method.
- Multiplicative weight update method.