CS880: Approximation and Online Algorithms	**Scribe:** Dongqiangzi Ye
Lecture 16: Online Algorithms and Online Primal-Dual	**Date:** 10/23/2019

16.1 Online Algorithms

Definition 16.1.1 *Online Algorithms: Online algorithms pertain to the problems where the input is not revealed at once in the beginning, but is given to the user one at a time. Because it does not know the whole input, an online algorithm is forced to make decisions that may later turn out not to be optimal, and the study of online algorithms has focused on the quality of decision-making that is possible in this setting.*

16.2 Ski Rental Problems

Ski Rental Problem is a really simple and basic online problem. Following is the definition.

Definition 16.2.1 *Ski Rental Problem: At each step, decide to rent at cost 1 or buy at cost B. If bought once, no need to rent or buy in the future. Number of steps is unknown (Assume it's k).*

Before we analyze Ski Rental Problem, let us make denotations as follow:

- Sequence of input - σ
- The performance of an Online Algorithm - $ALG(\sigma)$
- Hindsight optimum (offline optimum) - $OPT(\sigma)$

For a minimization problem, we define the competitive ratio of an algorithm ALG to be:

Definition 16.2.2 *Competitive Ratio of ALG* $= \max_{\sigma} \frac{ALG(\sigma)}{OPT(\sigma)}$

It's different from approximation ratio in principle. In approximation ratio, we don't have the computational power to achieve $OPT(\sigma)$, while competitive ratio arises in any situation where we cannot achieve $OPT(\sigma)$ due to the lack of information. In other word, in the competitive radio, there is no way we can achieve $OPT(\sigma)$ even we have unlimited computational power.

Definition 16.2.3 *Analysis that bound competitive radio call are called competitive analysis.*

Worst case σ is generated by adversary.

- Oblivious adversary: Knows the code of the algorithm but not any random bits flipped during the course of the algorithm.
- Adaptive adversary: Can observe all actions. Then Randomness is not useful.

Let us analyze deterministic algorithms at First.

Algorithm 1 Deterministic Algorithm for Ski-Rental

Rent up to the first B times
Then buy

Let us denote k as the number of times and we can make the following analysis:

- If $k \leq B$, $ALG = OPT = k$
- If $k > B$, $ALG = 2B$ and $OPT = B$

So competitive radio of this algorithm is 2. If you think about deterministic algorithms for Ski-Rental, you can always construct a k that costs your algorithm to spend at least twice what the optimum spends.

Before we come up with a random algorithm for Ski-Rental, we try to come up with a fractional algorithm. Here is the LP relaxation for Ski-Rental:

- x_i = total amount bought up to day i
- z_i = amount rented on day i
- $x = x_k \longleftarrow$ total amount bought overall

So writing down a LP formulation for online problems is a bit early, because we don't actually know what entire sequence is. One way is to consider the offline instance. Then we have:

$$
\begin{array}{lll}
 & \textit{Primal} & \\
min & Bx + \sum_{i=1}^{k} z_i & \\
s.t. & x + z_i \geq 1, \forall i & \\
 & x, z_i \geq 0, \forall i &
\end{array}
\qquad\qquad
\begin{array}{lll}
 & \textit{Dual} & \\
max & \sum_{i=1}^{k} y_i & \\
s.t. & \sum_{i=1}^{k} y_i \leq B & \\
 & 0 \leq y_i \leq 1, \forall i &
\end{array}
$$

Then we can design the online Primal-Dual algorithm:

Algorithm 2 Primal-Dual Algorithm for Ski-Rental

Start: $x_1 = 0$
for each step i **do**
  Set $z_i = 1 - x_i$
  if $i \leq B$ **then**
    Set $y_i = 1$
  end if
  Set $x_{i+1} = x_i + \frac{x_i}{B} + \frac{\alpha}{B} = x_i(1 + \frac{1}{B}) + \frac{\alpha}{B}$
end for
return $Bx + \sum_{i=1}^{k} z_i$

From algorithm 2, we can conclude that:

- If $i \geq 1$,

$$x_{i+1} = \frac{\alpha}{B} \sum_{t=1}^{i} (1 + \frac{1}{B})^t = \alpha((1 + \frac{1}{B})^{i+1} - 1) \tag{16.2.1}$$

- At every step, dual spends 1, while primal spends $1 - x_i + B(\frac{x_i}{B} + \frac{\alpha}{B}) = 1 + \alpha$

It's obvious to probe dual feasibility. In order to prove primal feasibility, we must show that $x_B = 1$. From (16.2.1), we can infer:

$$x_B = \alpha((1 + \frac{1}{B})^B - 1) = 1$$

$$\alpha = \frac{1}{(1 + \frac{1}{B})^B - 1}$$

So Competitive Ratio $= 1 + \alpha = \frac{(1+\frac{1}{B})^B}{(1+\frac{1}{B})^B - 1} \approx \frac{e}{e-1}$

As we can see from Figure 16.2.1, we can make a rounding procedure:

- Pick $\beta \sim U[0, 1]$
- Buy on step $t = argmin\{t : x_t \geq \beta\}$

So we can conclude that:

- Expected buying cost $= Pr[\beta \leq x_k]B = x_k B$
- Expected rent cost $= \sum_{i=1}^{k} 1 \times Pr[\beta > x_i] = \sum_{i=1}^{k} z_i$
- Expected cost of algorithm up to day t $= \frac{e}{e-1} min(B, t), \forall t$

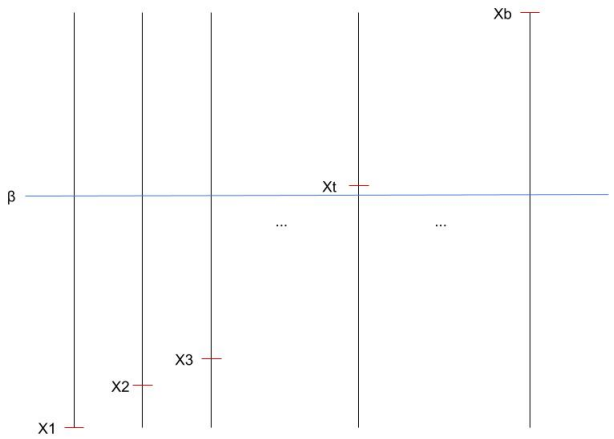

Figure 16.2.1: Random Algorithm for Ski-Rental

16.3 Online Set Cover

Definition 16.3.1 *Offline Set Cover: Given a set of elements $E = \{e_1, e_2, ..., e_n\}$ and a collection of subsets $S = \{s_1, s_2, ..., s_m\}$ with cost c_i, $s_j \subseteq E$ for $1 \leq j \leq m$. The offline set cover problem is to identify the sub-collection of S whose union equals E with minimum overall cost.*

Definition 16.3.2 *Online Set Cover: Given m sets with cost c_i, Elements arrive in sequence. We must ensure they are covered.*

Again, we can consider a fractional offline instance, and write down the LP formulation:

$$
\begin{array}{lll}
 & \textit{Primal} & \\
min & \sum_i x_i c_i & \\
s.t. & \sum_{i:j \in S_i} x_i \geq 1, \forall j & \\
 & x_i \geq 0, \forall i &
\end{array}
\qquad\qquad
\begin{array}{lll}
 & \textit{Dual} & \\
max & \sum_j y_j & \\
s.t. & \sum_{j \in S_i} y_j \leq c_i, \forall i & \\
 & y_j \geq 0, \forall j &
\end{array}
$$

In additional, this Dual LP is also called Packing Dual LP.

Now, we can come up with a high level Primal-Dual Algorithm for Online Set Cover, which we will discuss in the next lecture.

Algorithm 3 Primal-Dual Algorithm for Set Cover

Start: $x_i = 0, \forall i$ and $y_j = 0, \forall j$
for each step j (new j shows up) **do**
  if j is unsatisfied (i.e. $\sum_{i:j \in S_i} x_i < 1$) **then**
    Increase y_j
    Increase x_i for all i s.t. $j \in S_i$ as some function of y_j
  end if
  [Ensure that $\sum_{i:j \in S_i} x_i$ reaches 1 before some dual constraints become tight]
end for
return $\sum_i x_i c_i$

From high level algorithm 3, we can conclude that:

- $\sum_i c_i \frac{\mathrm{d}x_i}{\mathrm{d}t} \propto \sum_j \frac{\mathrm{d}y_j}{\mathrm{d}t}$
- Ensure that $\sum_{i:j \in S_i} x_i$ reaches 1 before some dual constraints become tight