Competitive Analysis

Rent-or-Buy a Tuxedo

Rent: $50
Buy: $300

Def: Opt = optimal cost with hindsight (foresight)
Our Cost = cost of our alg.

Competitive-Ratio \( (CR) = \max \left( \frac{\text{Our Cost}}{\text{Opt}} \right) \)

\( \max \) is over worst case seq of events.

eg Alg: "Buy Right Away"
Worst Case: Use only once

\[ CR = \frac{300}{50} = 6 \]
Alg: "Rent Forever"

Worst Case: \( c \) events

\[ CR = \frac{c \cdot 50}{300} \text{ (unbounded)} \]

Claim: \( \exists \text{ Alg (strategy)} \text{ s.t. } CR = (2 - \frac{r}{p}) \)

- \( r = \) cost to rent
- \( p = \) cost to buy
- \( c = \) event #

Strategy: while \((c+1)r \leq p\) rent
then buy

ie. Rent while \( CR \leq p - r \)

Worst Case Seq = stop after "buy".

Our Cost = "Rent Cost" + "Buy Cost"

\[ \leq (p - r) + p = 2p - r \]

\( Opt = p \)

\( CR \leq \frac{2p - r}{p} = 2 - \frac{r}{p} \)
Claim: Any Deterministic Alg $\text{CR} \geq 2 - \frac{r}{\rho}$

$\text{Alg: rent } k \text{ times then buys}$

$\text{Worst Case: } k+1 \text{ events}$

$\text{Opt: min}\{(k+1)r, \rho\}$

$\text{Case 1: } \text{Opt} = \rho \text{ i.e. } \rho \leq (k+1)r \text{ or } \rho - k \leq kr \ (\times)$

$\text{CR} = \frac{kr + \rho}{\rho} \geq \frac{(\rho - r) + \rho}{\rho} = 2 - \frac{r}{\rho}$

$\text{Case 2: } \text{Opt} = (k+1)r \text{ i.e. } (k+1)r \leq \rho \ (\times)$

$\text{CR} = \frac{kr + \rho}{(k+1)r} = \frac{(k+1)r + (\rho - r)}{(k+1)r} = 1 + \frac{\rho - r}{(k+1)r} \geq 1 + \frac{\rho - r}{\rho} = 2 - \frac{r}{\rho}$
Def: Alg A is c-competitive if \( \exists A \forall (A, B) \forall (input_1, input_2) \)
\[
\text{Cost}(A, T) \leq c \cdot \text{Cost}(B, T) + a
\]

For randomized A use \( C(A, T) = \text{expected cost} \).
The List Update Prob

Assume fixed length list

<table>
<thead>
<tr>
<th>Ops</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access(x)</td>
<td>index of location containing x</td>
</tr>
<tr>
<td>Swap(i,i+1)</td>
<td>1</td>
</tr>
</tbody>
</table>

A)gorithms

Single Exchange: After access(x) swap x one to front

Freg Count: Keep inorder of fregs.

Move-to-Front(MTF): After access "MTF" using swaps
Single Exchange (SE)

Claim SE is \( \Omega(n) \)-competitive

Consider \( \text{list}_n = x \) & \( \text{list}_{n-1} = y \)

requests \( \text{access}(x), \text{access}(y), \text{access}(x) \), ...

Claim Freq Count (FC)

1. Init
   \[ \text{freq} (\text{list}_1) = n, \ldots \text{freq} (\text{list}_n) = 1 \]

2. a) access(\( \text{list}_n \)) \( n \) times
   b) access(\( \text{list}_{n-1} \)) \( n \) times

   access(\( \text{list}_1 \)) \( n \) times

Our Cost = \( \Omega(n^3) \)
Opt\( ^* = O(n^3) \)

FC \( \in \Omega(n) \)-competitive
Thm: $MTF$ is 4-competitive for list-update.

Proof:
Let $B$ be any alg.
Use potential fn $avg$.

$$
\Phi \left( \text{list}_{MTF}, \text{list}_B \right) = 2 \left( \# \text{ of inversions} \right)
= 2 \left( \# \text{ swaps to go from list}_B \text{ to list}_{MTF} \right)
$$

Example:

```
\[\begin{array}{c}
\text{a} & \text{c} & \text{d} & \text{e} \\
\text{c} & \text{e} & \text{a} & \text{d}
\end{array}\]
```

inversions = 3

We charge $MTF$ the amortized-cost (AC)

$B$ unit-cost

\[\begin{array}{c}
\text{c} & \text{e} & \text{a} & \text{d} \\
\text{e} & \text{a} & \text{d} & \text{c}
\end{array}\]
\[ S = \{ y \mid y \text{ before } x \text{ in } \text{MTF} \land y \text{ before } x \text{ in } B \} \]

\[ T = \{ y \mid \neg y \text{ after } x \} \]

\[ C_A = \text{cost of access + swaps for alg A.} \]

\[ C_{\text{MTF}} = (|S| + |T|) + \left( |S| + |T| \right) = 1 + 2(|S| + |T|) \]

\[ C_B \geq |S| + 1 \quad (\text{all of } S \text{ before } x \text{ in } B) \]

\[ (B \text{ not doing any swaps}) \]
\[ \Delta \Xi = 2|s| - 2|t| = 2(|s| - |t|) \]

\[ AC_{MTF} = 1 + 2(|s| + |t|) + 2(|s| - |t|) = 1 + 4|s| \leq 4(1 + |s|) = 4C_B \]

---

**Analysis of B doing a swap**.

\[ C_{MTF} = 0 \quad C_B = 1 \]

\[ \Delta \Xi \leq 2 \]

\[ AC_{MTF} \leq 0 + 2 \leq 2C_B \leq 4C_B \]

---

\[ \text{Total Cost}_{MTF} \leq 4(\text{Total Cost}_B) + \overline{\Pi}_0 - \overline{\Pi}_{final} \]

\[ \overline{\Pi}_0 - \overline{\Pi}_{final} \leq 2 \left( \begin{array}{c} n \\ 2 \end{array} \right) = n(n-1) \] (a constant)

**WEO**
Migration Prob

Input: Weighted graph $G$
Stores: Pages $\rightarrow$ Nodes

Cost: $\text{access}(P,a) \equiv \text{dist from } a \text{ to location storing } P$
$\text{move}(P,a,b) = |P| \cdot \text{dist}(a,b)$ $P=|P|$ 

Prob: On-line migration

Simple case $G = \begin{array}{c}
1 \\
2
\end{array}$ (two node graph)
One page

Counting Alg

Node count init $C_1 = 0 \& C_2 = 0$

Access $(P,1)$ if $P \neq 1$ do nothing
else $C_1 \leftarrow C_1 + 1$
if $C_1 = 2P$ then move $P$ to $1$
else $C_2 \leftarrow C_2$

b) $C_1 = 0$
Thm: Count Alg is 3-competitive which is optimal
(no proof)