Game Tree Evaluation

2k levels
4^k leaves

Deterministic LB of 4^k
Randomized Alg: Top down evaluation

Analysis: If OR returns 0,
\[ T(OR, k, 0) = 2 \cdot 3^{k-1} \]
If OR returns 1,
\[ T(OR, k, 1) = \frac{3}{2} \cdot 3^{k-1} \]
If AND returns 1,
\[ T(AND, k, 1) = 2 \cdot T(OR, k, 1) = 3 \cdot 3^{k-1} = 3^k \]
If AND returns 0,
\[ T(AND, k, 0) = T(OR, k, 0) + \frac{1}{2} \cdot T(OR, k, 1) \]
\[ \leq 3 \cdot 3^{k-1} = 3^k = 0.743 \]

Game Theory

Zero-sum game: sum of payoffs is 0

<table>
<thead>
<tr>
<th></th>
<th>s1</th>
<th>s2</th>
<th>s3</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>s2</td>
<td>-1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>s3</td>
<td>-2</td>
<td>-1</td>
<td>0</td>
</tr>
</tbody>
</table>

R (strategy i)  \( \begin{align*} \text{Max} \min (M_{ij}) \end{align*} \)
C (strategy i)  \( \begin{align*} \text{Min} \max (M_{ij}) \end{align*} \)

R will play
Det., C will play
Ex: Max Min (M_{ij})

\( \begin{align*} i, j \end{align*} \)
\( C(\text{strategy } j) \) \leq \min_j \max_i (M_{ij})

Randomized Strategies: \( \bar{p} \) for \( P \) and \( \bar{q} \) for \( C \)

\[ E[\text{payoff}] = p^T M q \]

Thm. (Von Neumann): \( \max_p \min_q p^T M q = \min_q \max_p p^T M q \)

\[ \max_p \min_q p^T M q = \min_q \max_i e_i^T M q \]

Yao's Minimax Principle

Alg designer: \( C \) (chooses algorithm)
Adv adversary: \( P \) (chooses input)
Payoff: Running time / Approximation / 

Mixed strategy for \( C \): Las Vegas algo
Mixed strategy for \( P \): Input distribution

Yao's Minimax Principle:

\[ \max_p \min_q E[p(C(p, A)) = \min_q \max_{\pi \in \Pi} E[C(\pi, A)] \]

In other words, if there exists a input distribution such that every deterministic algorithm has expected running time (or approx, etc.) at least \( \alpha \) then the running time (or approx, etc.) of the best Las Vegas algorithm is also at least \( \alpha \).
For each leaf to 1 w/p \( p = \frac{3 - \sqrt{5}}{2} \)

\[ \Pr(\text{output is 1}) = \left(1 - \frac{3 - \sqrt{5}}{2}\right)^2 = \frac{3 - \sqrt{5}}{2} = p \]

Any deterministic algo uses depth-first pruning

\[ W(h) = W(h-1) + (1 - p) \cdot W(h-1) \]

\[ \Rightarrow W(h) \geq n^{0.649} \]