Probability and Computing – Lower bounds using Yao’s Principle

Stefan Walzer, Maximilian Katzmann | WS 2023/2024
Some of this lecture’s content is covered in Thomas Worsch’s notes from 2019.
Content

1. Nash Equilibria in 2-Player Zero-Sum Games
   - Games and Nash Equilibria
   - Two Player Zero Sum Games
   - Loomis’ Theorem for Two-Player Zero Sum Games

2. Yao’s Minimax Principle

3. Applications of Yao’s Principle
   - Evaluation of \( \bar{\Lambda} \)-Trees
     - Proof Sketch of Tarsi's Theorem (nicht prüfungsrelevant)
   - The Ski-Rental Problem
Content

1. Nash Equilibria in 2-Player Zero-Sum Games
   - Games and Nash Equilibria
   - Two Player Zero Sum Games
   - Loomis’ Theorem for Two-Player Zero Sum Games

2. Yao’s Minimax Principle

3. Applications of Yao’s Principle
   - Evaluation of $\tilde{\Lambda}$-Trees
   - Proof Sketch of Tarsi’s Theorem (nicht prüfungsrelevant)
   - The Ski-Rental Problem
1. Nash Equilibria in 2-Player Zero-Sum Games
   - Games and Nash Equilibria
   - Two Player Zero Sum Games
   - Loomis’ Theorem for Two-Player Zero Sum Games

2. Yao’s Minimax Principle

3. Applications of Yao’s Principle
   - Evaluation of $\bar{\Lambda}$-Trees
     - Proof Sketch of Tarsi’s Theorem (nicht prüfungsrelevant)
   - The Ski-Rental Problem
Prisoner’s Dilemma

Setting

- strategies 😈 and 😂 available to both players
- table shows payoffs for players depending on chosen strategies
- here: always better to choose 😂
  → pair (😂, 😈) is unique equilibrium

Definition: Equilibrium

Combination of strategies such that no one can profit by unilaterally switching his or her own strategy.
A cat and mouse game

Someone always regrets their decision

<table>
<thead>
<tr>
<th></th>
<th>reaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>🍀 -42</td>
<td>🍀 should have played 🍀</td>
</tr>
<tr>
<td>🍀 2\ 1</td>
<td>🍀 should have played 🍀</td>
</tr>
<tr>
<td>0\0</td>
<td>🍀 should have played 🍀</td>
</tr>
<tr>
<td>0\1</td>
<td>🍀 should have played 🍀</td>
</tr>
</tbody>
</table>

→ No combination of *pure* strategies is an *equilibrium*.

Equilibrium

Combination of strategies such that no one can profit by unilaterally switching his or her own strategy.
Nash Equilibria

What a Game is

- Finite sets $S_1$, $S_2$ of pure strategies.
- Utility functions $u_1, u_2 : S_1 \times S_2 \rightarrow \mathbb{R}$.

How a Game is played

- Players pick a strategy simultaneously $\rightarrow$ gives pair $(s_1, s_2) \in S_1 \times S_2$.
- Player 1 gets payoff $u_1(s_1, s_2)$ and player 2 gets $u_2(s_1, s_2)$.

Existence of Mixed-Strategy Nash Equilibria

There exist distributions $S_1$ on $S_1$ and $S_2$ on $S_2$, called mixed strategies such that $(S_1, S_2)$ is an equilibrium:

player 1 cannot increase expected payoff: $\mathbb{E}_{s_1 \sim S_1, s_2 \sim S_2}[u_1(s_1, s_2)] = \max_{s_1 \in S_1} \mathbb{E}_{s_2 \sim S_2}[u_1(s_1, s_2)]$.

player 2 cannot increase expected payoff: $\mathbb{E}_{s_1 \sim S_1, s_2 \sim S_2}[u_2(s_1, s_2)] = \max_{s_2 \in S_2} \mathbb{E}_{s_1 \sim S_1}[u_2(s_1, s_2)]$.

Remark: Theorem holds for $n \geq 3$ players as well.
Nash Equilibrium in Cat & Mouse Game

<table>
<thead>
<tr>
<th></th>
<th>🦷</th>
<th>🧶</th>
</tr>
</thead>
<tbody>
<tr>
<td>🦷</td>
<td>-4\2</td>
<td>2\1</td>
</tr>
<tr>
<td>🧶</td>
<td>0\0</td>
<td>0\1</td>
</tr>
</tbody>
</table>

Equilibrium

\[ S_мышь = \{ \begin{array}{c} \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \end{pmatrix} \end{array} \} \]

\[ S_мышь = \{ \begin{array}{c} \begin{pmatrix} 1 \\ 3 \end{pmatrix}, \begin{pmatrix} 2 \\ 3 \end{pmatrix} \end{array} \} \]

Verification of Equilibrium Property: Calculating Expected Payoffs

**for 🦷:**
- playing 🤡 gives expected payoff
  \[
  \frac{1}{3} \cdot (-4) + \frac{2}{3} \cdot 2 = 0
  \]
- playing 🧶 gives expected payoff
  \[
  \frac{1}{3} \cdot 0 + \frac{2}{3} \cdot 0 = 0
  \]
- playing \( S_мышь \) is a mix of both
  \( \rightarrow \) also expected payoff 0.

**for 🧶:**
- playing 🤡 gives expected payoff
  \[
  \frac{1}{2} \cdot 2 + \frac{1}{2} \cdot 0 = 1
  \]
- playing 🧶 gives expected payoff
  \[
  \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 1 = 1
  \]
- playing \( S_мышь \) is a mix of both
  \( \rightarrow \) also expected payoff 1.

Nash Equilibria in 2-Player Zero-Sum Games

Yao’s Minimax Principle

Applications of Yao’s Principle

ITI, Algorithm Engineering & Scalable Algorithms
1. Nash Equilibria in 2-Player Zero-Sum Games
   - Games and Nash Equilibria
   - Two Player Zero Sum Games
   - Loomis’ Theorem for Two-Player Zero Sum Games

2. Yao’s Minimax Principle

3. Applications of Yao’s Principle
   - Evaluation of \( \bar{\mathcal{T}} \)-Trees
     - Proof Sketch of Tarsi's Theorem (nicht prüfungsrelevant)
   - The Ski-Rental Problem
Two Player Zero Sum Games and their Matrix Formulation

- Finite sets of pure strategies
  - $S_1$ for player 1
  - $S_2$ for player 2
- utility function $u : S_1 \times S_2 \rightarrow \mathbb{R}$
  - player 1 gets $u(s_1, s_2)$
  - player 2 gets $-u(s_1, s_2)$

- Implicit sets of pure strategies
  - $S_1 = [n]$ for the row player
  - $S_2 = [m]$ for the column players
- matrix $M \in \mathbb{R}^{n \times m}$
  - row player gets $M_{s_1, s_2}$
  - column player gets $-M_{s_1, s_2}$

<table>
<thead>
<tr>
<th></th>
<th>🧥</th>
<th>📄</th>
<th>🐿</th>
</tr>
</thead>
<tbody>
<tr>
<td>🧥</td>
<td>0</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>📄</td>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>🐿</td>
<td>-1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Unique equilibrium of 🧥 📄 🐿

$S_1 = S_2 = \{ \text{🧥} : \frac{1}{3}, \text{프로그램} : \frac{1}{3}, \text{🪨} : \frac{1}{3} \}$

Nash Equilibria in 2-Player Zero-Sum Games

Yao's Minimax Principle

Applications of Yao's Principle
Nash Equilibria for Two-Player Zero-Sum Games

Nash’s Theorem (1950), Special Case for Two-Player Zero-Sum Games

For any $M \in \mathbb{R}^{n \times m}$ there exist distributions $S^*_1$ on $[n]$ and $S^*_2$ on $[m]$ such that

$$E_{s_1 \sim S^*_1, s_2 \sim S^*_2}[M_{s_1, s_2}] = \max_{s_1 \in [n]} E_{s_2 \sim S^*_2}[M_{s_1, s_2}] = \min_{s_2 \in [m]} E_{s_1 \sim S^*_1}[M_{s_1, s_2}].$$

Intuition: When the players play according to $S^*_1$ and $S^*_2$, then no player can benefit by deviating from his strategy.

Corollary: Loomis (1946) Von Neumann (1928)

For any $M \in \mathbb{R}^{n \times m}$ we have

$$\max_{S_1} \min_{s_2 \in [m]} E_{s_1 \sim S_1}[M_{s_1, s_2}] = \min_{S_2} \max_{s_1 \in [n]} E_{s_2 \sim S_2}[M_{s_1, s_2}]$$

where $S_1$ and $S_2$ are distributions on $[n]$ and $[m]$, respectively

Intuition

No first-mover disadvantage if

- first player plays mixed strategy
- second player (wlog) pure strategy

Next: Proof of Loomis’ Theorem assuming Nash’s Theorem.
Lemma: First Mover’s Disadvantage

Lemma Φ: Exchanging min and max

Let $X$ and $Y$ be sets and $u : X \times Y \rightarrow \mathbb{R}$ a function. In our setting\(^1\)

$$\max_{y \in Y} \min_{x \in X} u(x, y) \leq \min_{x \in X} \max_{y \in Y} u(x, y)$$

\(^1\) In general min and max may not be well-defined...

Proof.

$$\max_{y \in Y} \min_{x \in X} u(x, y) = \min_{x \in X} u(x, y^*) \leq \min_{x \in X} \max_{y \in Y} u(x, y).$$

Relevance

Being the second player to choose is never a disadvantage.
Lemma: When pure strategies are sufficient

Lemma $\Delta$: Minima over sets of distributions

Let $X$ be a set and $u : X \to \mathbb{R}$ a function. Let $D$ be the set of all distributions on $X$. In our setting:

$$\min_{x \in X} u(x) = \min_{\mathcal{X} \in D} \mathbb{E}_{x \sim \mathcal{X}}[u(x)].$$

Relevance for us

The last player to choose a strategy may always choose a pure strategy.

Proof by Example

Here is a list of numbers $X = \{12, 42, 73, 101\}$.

- Task 1: Find the minimum!
  \[\text{Duh, it’s 12.}\]
- Task 2: Design a wheel of fortune involving only numbers from $X$ with minimum expectation!
  \[\text{Duh, take 12 everywhere.}\]
Proof of Loomis’s Theorem

Let $S_1^*$ and $S_2^*$ be the mixed strategies from Nash’s Theorem.

$$
\min_{S_2} \max_{s_1 \in [n]} \mathbb{E}_{s_2 \sim S_2}[M_{s_1, s_2}]
\leq \max_{s_1 \in [n]} \mathbb{E}_{s_2 \sim S_2^*}[M_{s_1, s_2}]
= \min_{s_2 \in [m]} \mathbb{E}_{s_1 \sim S_1^*}[M_{s_1, s_2}]
= \min_{S_2} \mathbb{E}_{s_1 \sim S_1^*, s_2 \sim S_2}[M_{s_1, s_2}]
\leq \max_{S_1} \min_{S_2} \mathbb{E}_{s_1 \sim S_1, s_2 \sim S_2}[M_{s_1, s_2}]
= \max_{S_1} \mathbb{E}_{s_1 \sim S_1}[M_{s_1, s_2}]
\leq \min_{S_2} \max_{S_1} \mathbb{E}_{s_1 \sim S_1, s_2 \sim S_2}[M_{s_1, s_2}]
= \min_{S_2} \mathbb{E}_{s_2 \sim S_2}[M_{s_1, s_2}]
\leq \max_{S_1} \min_{S_2} \mathbb{E}_{s_1 \sim S_1, s_2 \sim S_2}[M_{s_1, s_2}]
= \max_{S_1} \mathbb{E}_{s_1 \sim S_1}[M_{s_1, s_2}]
\leq \min_{S_2} \max_{S_1} \mathbb{E}_{s_1 \sim S_1, s_2 \sim S_2}[M_{s_1, s_2}]
= \min_{S_2} \mathbb{E}_{s_2 \sim S_2}[M_{s_1, s_2}]
$$

Start and end with same term. Hence all “≤” are “=”. Hence terms of interest are “=”.

Corollary: Loomis (1946) Von Neumann (1928)

For any $M \in \mathbb{R}^{n \times m}$ we have

$$
\max_{S_1} \min_{S_2} \mathbb{E}_{s_1 \sim S_1}[M_{s_1, s_2}] = \min_{S_2} \max_{S_1} \mathbb{E}_{s_2 \sim S_2}[M_{s_1, s_2}]
$$

Nash Equilibria in 2-Player Zero-Sum Games

Yao’s Minimax Principle

Applications of Yao’s Principle

15/39

WS 2023/2024

Stefan Walzer, Maximilian Katzmann: Yao’s Principle

ITI, Algorithm Engineering & Scalable Algorithms
Content

1. Nash Equilibria in 2-Player Zero-Sum Games
   - Games and Nash Equilibria
   - Two Player Zero Sum Games
   - Loomis’ Theorem for Two-Player Zero Sum Games

2. Yao’s Minimax Principle

3. Applications of Yao’s Principle
   - Evaluation of Ā-Trees
     - Proof Sketch of Tarsi's Theorem (nicht prüfungsrelevant)
   - The Ski-Rental Problem
Algorithm Design as a 2-Player Zero-Sum Game

Setting
For a given computational problem $P$ let

- **Algos**: finite set of deterministic algorithms
- **Inputs**: finite set of inputs
- $C(A, I) \in \mathbb{R}$ cost of $A \in \text{Algos}$ on $I \in \text{Inputs}$.

Example: Sorting
For given $n \in \mathbb{N}$ (finite, though possibly $n \to \infty$ later)

- $P = \text{“sort } n \text{ numbers comparison-based”}$
- $C(A, I) = \# \text{ of comparisons of } A \text{ for input } I$
- **Inputs** = $S_n$ //permutations of $[n]$
- **Algos** = e.g. suitable set of decision trees

A Two-Player Zero-Sum Game

- Designer chooses (randomised) algorithm, i.e. a distribution on **Algos**.
  $\leftrightarrow$ Goal: Minimise (expected) cost.
- Adversary chooses (randomised) input, i.e. a distribution on **Inputs**.
  $\leftrightarrow$ Goal: Maximise (expected) cost.

Example: Sorting ($x, y, z$)

<table>
<thead>
<tr>
<th>Adversary</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 2, 3)</td>
</tr>
<tr>
<td>$x &lt; y$ then $y &lt; z$ then $z &lt; x$</td>
</tr>
<tr>
<td>$y &lt; z$ then $z &lt; x$ then $x &lt; y$</td>
</tr>
<tr>
<td>$\ldots$</td>
</tr>
<tr>
<td>$x &lt; y$ then $y &lt; z$ then $z &lt; x$</td>
</tr>
<tr>
<td>$y &lt; z$ then $z &lt; x$ then $x &lt; y$</td>
</tr>
<tr>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

Recall: Exercise Sheet 0, Exercise 1.
Definition: Randomised Complexity

\[ C := \min_{A \text{ dist. on Algos}} \max_{I \in \text{Inputs}} \mathbb{E}_{A \sim A}[C(A, I)] \]

\[ \text{designer moves first} \]

\[ \text{Loomis} \]

\[ = \max_{I \text{ dist. on Inputs}} \min_{A \in \text{Algos}} \mathbb{E}_{I \sim I}[C(A, I)] \]

\[ \text{adversary moves first} \]

Yao’s Principle: (Upper and) Lower Bounds on \( C \)

Let \( A_0 \) be a distribution on \( \text{Algos} \) and \( I_0 \) a distribution on \( \text{Inputs} \). Then

\[ \max_{I \in \text{Inputs}} \mathbb{E}_{A \sim A_0}[C(A, I)] \geq C \geq \min_{A \in \text{Algos}} \mathbb{E}_{I \sim I_0}[C(A, I)]. \]

\text{Tightness:} Loomis implies that “=” is possible.

\[ \leftrightarrow \text{Can attain lower bounds on } C \text{ by thinking about deterministic algorithm only!} \]
Content

1. Nash Equilibria in 2-Player Zero-Sum Games
   - Games and Nash Equilibria
   - Two Player Zero Sum Games
   - Loomis’ Theorem for Two-Player Zero Sum Games

2. Yao’s Minimax Principle

3. Applications of Yao’s Principle
   - Evaluation of $\overline{A}$-Trees
     - Proof Sketch of Tarsi’s Theorem (nicht prüfungsrelevant)
   - The Ski-Rental Problem
Computational Problem: $\bar{\wedge}$-Tree-Evaluation

Problem: Evaluate $\bar{\wedge}$-Tree of depth $d$
- \textbf{Inputs} = $\{0, 1\}^n$ for $n = 2^d$. Specify bits at leaves.
- \textbf{Algos} = Algorithms computing value at root.
- $C(A, I) = \#$ bits of $I$ that $A$ examines
  $\leftrightarrow$ query complexity of $A$ on $I$

Goal
Bound randomised query complexity

$$C = \min_{A \text{ dist. on Algos}} \max_{I \in \text{Inputs}} \mathbb{E}_{A \sim A}[C(A, I)].$$

Example and possible formalisation of \textbf{Algos} (that we won’t use)
Each $A \in \text{Algos}$ corresponds to a \textit{decision tree}. In the example:
- $C(A, (1, 0, 1, 0)) = 4$
- $C(A, (0, 1, 0, 1)) = 2$

Each leaf queried at most once per path
$\Rightarrow$ depth $\leq n \Rightarrow |\text{Algos}| < \infty$
∧-∨-trees are  ∨-trees are  ∧-trees

See exercise sheet 1 ("Die Wälder von NORwegen")

Deterministic Query Complexity is $n$ (Lecture 1, Slide 8)

For all $A \in \text{Algos}$ there exists $I \in \text{Inputs}$ such that $C(A, I) = n$.

Randomised Query Complexity is $O(n \log_4(3)) \approx O(n^{0.792})$ (Lecture 1, Slide 10)

Let $\mathcal{A}$ be the randomised algorithm that evaluates one of the two depth $d - 1$ subtrees at random (recursively) and, if that yields 1, also evaluates the other subtree (recursively).

$$\max_{l \in \text{Inputs}} \mathbb{E}_{A \sim \mathcal{A}}[C(A, I)] = O(3^{d/2}) = O(n \log_4(3)).$$

**Goal:** Show lower bound of $\Omega(\varphi^d) \approx \Omega(n^{0.694})$ using Yao's Principle ($\varphi$ is the golden ratio).

**Remark:** actual complexity is $\Theta(n \log_4(3))$, but that's more difficult.
Warm Up: A simple lower bound

**Observation**
For any even $d \in \mathbb{N}$ and $A \in \text{Algos}$ we have $C(A, (0, \ldots, 0)) \geq 2^{d/2}$.

**Corollary:** Randomised Complexity is $\Omega(\sqrt{n})$

$$
C = \min_{\mathcal{A} \text{ dist. on Algos}} \max_{I \in \text{Inputs}} \mathbb{E}_{A \sim \mathcal{A}}[C(A, I)] \\
\geq \min_{\mathcal{A} \text{ dist. on Algos}} \mathbb{E}_{A \sim \mathcal{A}}[C(A, (0, \ldots, 0))] \\
\geq \min_{\mathcal{A} \text{ dist. on Algos}} \mathbb{E}_{A \sim \mathcal{A}}[2^{d/2}] \\
\geq 2^{d/2} = 2^{\log_2(n)/2} = n^{1/2}.
$$

Note Yao’s spirit: Lower bound on randomised complexity from result on deterministic algorithms.
A stronger lower bound

**Theorem (Tarsi 1984)**

For any \( p \in [0, 1] \) simpleEval is optimal for input distribution \( I_p \), i.e.

\[
\min_{A \in \text{Algos}} \mathbb{E}_{I \sim I_p} [C(A, l)] = \mathbb{E}_{I \sim I_p} [C(\text{simpleEval}, l)].
\]

**Lemma**

If \( p_0 = \frac{\sqrt{5} - 1}{2} \) and \( \varphi \) is the golden ratio then

\[
\mathbb{E}_{I \sim I_{p_0}} [C(\text{simpleEval}, l)] = (1 + p_0)^d = \varphi^d.
\]

**Corollary:** \( C = \Omega(\varphi^d) \approx \Omega(n^{0.694}) \)

\[
C \geq \min_{A \in \text{Algos}} \mathbb{E}_{I \sim I_{p_0}} [C(A, l)] = \mathbb{E}_{I \sim I_{p_0}} [C(\text{simpleEval}, l)]
\]

\[
= \varphi^d = \varphi^{\log_2 n} = n^{\log_2 \varphi} \approx n^{0.694}.
\]

---

**Independent Bernoulli Inputs**

Let \( I_p = \text{Ber}(p)^n \) be the distribution where leafs are assigned independently values with distribution \( \text{Ber}(p) \).

**Deterministic Algorithm**

**Algorithm** simpleEval\((T)\):

- if \( T = \text{leaf}(b) \) then
  - return \( b \)
- else
  - \((T_\ell, T_r) \leftarrow T\)
    - if simpleEval\((T_\ell) = 0\) then
      - return \( 1 \)
    - else
      - return \(-\text{simpleEval}(T_r)\)
Proof of Lemma: Cost of simpleEval on $I_{p_0}$

**Lemma**

If $p_0 = \frac{\sqrt{5} - 1}{2}$ and $\phi$ is the golden ratio then

$$\mathbb{E}_{l \sim I_{p_0}} [C(\text{simpleEval}, l)] = (1 + p_0)^d = \phi^d.$$  

**Proof (cf. Exercise “Die Wälder von NORwegen”)**

- $p_0 = \frac{\sqrt{5} - 1}{2}$ is the solution to $p = 1 - p^2$.
- If $a, b \sim \text{Ber}(p_0)$ then $a \wedge b \sim \text{Ber}(1 - p_0^2) = \text{Ber}(p_0)$.
- For $l \sim I_{p_0}$ the probability that an internal tree node evaluates to 1 is $p_0$.
- Let $c_d := \mathbb{E}_{l \sim I_{p_0}} [C(\text{simpleEval}, l)]$ for trees of depth $d$. Then
  - $c_0 = 1$ // tree of depth 0 is just the leaf
  - $c_d = c_{d-1} + p_0 \cdot c_{d-1} = (1 + p_0) c_{d-1} \equiv (1 + p_0)(1 + p_0)^{d-1} = (1 + p_0)^d$
    // Always one recursive call, with probability $p$ a second one.

**Deterministic Algorithm**

**Algorithm** `simpleEval(T)`:

```
if $T = \text{leaf}(b)$ then
    return $b$
else
    $(T_\ell, T_r) \leftarrow T$
    if `simpleEval($T_\ell$)` = 0 then
        return 1
    else
        return $\neg$`simpleEval($T_r$)`
```

Nash Equilibria in 2-Player Zero-Sum Games

Yao's Minimax Principle

Applications of Yao's Principle
Content

1. Nash Equilibria in 2-Player Zero-Sum Games
   - Games and Nash Equilibria
   - Two Player Zero Sum Games
   - Loomis’ Theorem for Two-Player Zero Sum Games

2. Yao’s Minimax Principle

3. Applications of Yao’s Principle
   - Evaluation of $\tilde{\Lambda}$-Trees
     - Proof Sketch of Tarsi's Theorem (nicht prüfungsrelevant)
   - The Ski-Rental Problem
Tarsi's Theorem

Theorem (Tarsi 1984)

For any $p \in [0, 1]$ simpleEval is optimal for input distribution $\mathcal{I}_p$, i.e.

$$\min_{A \in \text{Algos}} \mathbb{E}_{I \sim \mathcal{I}_p} [C(A, I)] = \mathbb{E}_{I \sim \mathcal{I}_p} [C(\text{simpleEval}, I)].$$

Proof idea:

- Take optimal Algorithm $A$.
- Transform $A$ into simpleEval step by step.
- Show: Expected query complexity never increases.
Definition: Superleafs

A superleaf consists of two sibling leafs and their parent.

Lemma

For any \( p \in [0, 1] \) and any \( A \in \text{Algos} \) there exists \( A' \in \text{Algos} \) such that

\[
E_{I \sim I_p}[C(A', I)] \leq E_{I \sim I_p}[C(A, I)]
\]

\( A' \) behaves on any superleaf \( T = (\ell, r) \) like simpleEval:

- Property i: never visits \( r \) before \( \ell \)
- Property ii: never visits \( r \) if \( \ell = 0 \)
- Property iii: immediately visits \( r \) after visiting \( \ell \) if \( \ell = 1 \)

Proof Idea

- We fix every superleaf one by one. Let \( T \) be the superleaf that needs fixing.
- Property i: Switch roles of \( \ell \) and \( r \) if needed. Does not change the expected cost.
- Property ii: \( r \) does not contribute to result. Not visiting \( r \) reduces expected cost.
- Property iii: More difficult. See next slide.
\[ C_A := \mathbb{E}\{C(A, I)\} = \mathbb{E}\{C_0 + \bar{\alpha} \cdot (1 + \bar{p}C_1 + p \cdot (C_2 + \bar{\beta}(1 + \bar{p}C_3 + pC_4)))\} \]
\[ C_B := \mathbb{E}\{C(B, I)\} = \mathbb{E}\{C_0 + \bar{\alpha} \cdot (1 + \bar{p}C_1 + p \cdot (1 + \bar{p}C_1 + p(C_2 + \bar{\beta}C_4)))\} \]
\[ C_D := \mathbb{E}\{C(D, I)\} = \mathbb{E}\{C_0 + \bar{\alpha} \cdot (C_2 + \bar{\beta}(1 + \bar{p}C_3 + p(1 + \bar{p}C_3 + pC_4)))\} \]

\[ (C_B - C_A) + p \cdot (C_D - C_A) = \ldots = 0 \]
\[ \Rightarrow C_B - C_A \leq 0 \lor C_D - C_A \leq 0 \]
\[ \Rightarrow B \text{ or } D \text{ (or both) are at least as good as } A \text{ and both visit superleaf } (\ell, r) \text{ as desired.} \]
Theorem (Tarsi 1984)

For any \( p \in [0, 1] \) simpleEval is optimal for input distribution \( \mathcal{I}_p \), i.e.

\[
\min_{A \in \text{Algos}} \mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [C(A, I)] = \mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [C(\text{simpleEval}, I)].
\]

We use induction on \( d \). For \( d = 0 \) simpleEval is clearly optimal. Let now \( d \geq 1 \).

Let \( A \in \text{Algos} \) be an algorithm minimising \( \mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [C(A, I)] \).

By Lemma: There exists \( A' \in \text{Algos} \) that behaves like simpleEval on superleafs such that

\[
\mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [C(A', I)] \leq \mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [C(A, I)].
\]

Let \( L' \) be the number of superleafs visited by \( A' \) and \( L \) the number of superleafs visited by simpleEval.

Superleafs evaluate to 1 with probability \( 1 - p^2 \) independently and are in a complete binary tree of depth \( d - 1 \).

Apply induction for \( d' = d - 1 \) and \( p' = 1 - p^2 \).

\[
\mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [L] \leq \mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [L'].
\]

The expected cost for evaluating a superleaf is \( 1 + p \).

Hence

\[
\mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [C(A', I)] = (1 + p)\mathbb{E}[L']
\]
\[
\mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [C(A, I)] = (1 + p)\mathbb{E}[L]
\]

Finally we obtain:

\[
\mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [C(\text{simpleEval}, I)] = (1 + p)\mathbb{E}[L] \leq (1 + p)\mathbb{E}[L']
\]
\[
= \mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [C(A', I)] \leq \mathbb{E}_{\mathcal{I} \sim \mathcal{I}_p} [C(A, I)].
\]

Hence, simpleEval is optimal for \( \mathcal{I}_p \). \( \square \)
1. Nash Equilibria in 2-Player Zero-Sum Games
   - Games and Nash Equilibria
   - Two Player Zero Sum Games
   - Loomis' Theorem for Two-Player Zero Sum Games

2. Yao's Minimax Principle

3. Applications of Yao's Principle
   - Evaluation of $\bar{\Delta}$-Trees
     - Proof Sketch of Tarsi's Theorem (nicht prüfungsrelevant)
   - The Ski-Rental Problem
Ski Rental – A Prototypical Online Problem

Setting: You are on a ski trip
Trip lasts for unknown number of days $l \in \mathbb{N}$
("as long as there is snow").
Every day, if no skis bought yet:
- RENT skis for one day for cost 1 or
- BUY skis for cost $B \in \mathbb{N}$.

Goal: Minimise Competitive Ratio
The competitive ratio of distribution $\mathcal{A}$ on $\text{Algos}$ is

$$C_{\mathcal{A}} = \sup_{l \in \text{Inputs}} \frac{\mathbb{E}_{A \sim \mathcal{A}}[C(A, l)]}{\text{OPT}(l)}.$$  

Framing using Online Algorithms
- **Inputs** = $\mathbb{N}$: number of days (not known in advance)
- **Algos** = $\mathbb{N}$: specify day for choosing BUY
- cost for $A \in \text{Algos}$ on $l \in \text{Inputs}$:

$$C(A, l) = \begin{cases} l & \text{if } l < A \\ A - 1 + B & \text{otherwise.} \end{cases}$$

- cost of optimum offline solution

$$\text{OPT}(l) = \begin{cases} l & \text{if } l < B \\ B & \text{otherwise.} \end{cases}$$
Break-Even is the best deterministic algorithm

**Observation**

The algorithm breakEven := B has competitive ratio \( \frac{2B-1}{B} \approx 2 \).

All other \( A \in \text{Algos} \) have competitive ratio \( \geq 2 \).

**Proof**

The worst ratio for \( A \in \text{Algos} \) with \( A > B \) is attained for input \( I = A \).

\[
C_A = \sup_{I \in \mathbb{N}} \frac{C(A, I)}{\text{OPT}(I)} = \frac{C(A, A)}{\text{OPT}(A)} = \frac{A - 1 + B}{B} = 1 + \frac{A - 1}{B} \geq 1 + 1 = 2.
\]

Recall

\( B \) is the cost to \text{BUY}. 

\[ \text{cost} \]

\[ C(A, I) \]

\[ \geq 2 \]

\[ \text{OPT}(I) \]

\[ B \]

\[ A \]

\[ I \]
A randomised algorithm can beat break-even

**Observation (assuming wlog that $B$ is a multiple of 3)**

The randomised algorithm $\mathcal{A} = \mathcal{U}(\{\frac{2}{3}B, B\})$ has competitive ratio $\approx 1 + \frac{5}{6}$.

**Proof**

The competitive ratio of $\mathcal{A}$ “spikes” for inputs $\frac{2}{3}B$ and $B$. It is decreasing in between and constant after $B$.

$$
\mathbb{E}_{A \sim A}[C(A, I)] = \frac{2}{3}B - 1 + \frac{1}{3}(1 + B) < \frac{7}{6}B, \quad \text{OPT}(\frac{2}{3}B) = \frac{2}{3}B,
$$

$$
\mathbb{E}_{A \sim A}[C(A, B)] = B + \frac{2}{3}B - 1 + \frac{1}{2}(\frac{1}{3}B) < \frac{11}{6}B, \quad \text{OPT}(B) = B.
$$

Hence $C_A = \sup_{I \in \mathbb{N}} \frac{\mathbb{E}_{A \sim A}[C(A, I)]}{\text{OPT}(I)} \leq \max \left\{ \frac{7}{6}, \frac{11}{6} \right\} = \frac{11}{6}$. 
Goal: Lower bound

No randomised algorithm has competitive ratio better than $\approx 1.582$. 

What’s next?
Yao’s Principle for Online Algorithms

Theorem (see Online Optimization Lecture, Corollary 3.8, Prof. Yann Disser, Darmstadt, 2023)

For any distribution $\mathcal{A}_0$ on $\text{Algos}$ and any distribution $\mathcal{I}_0$ on $\text{Inputs}$ we have

$$C_{\mathcal{A}_0} \overset{\text{def}}{=} \sup_{I \in \text{Inputs}} \mathbb{E}_{A \sim \mathcal{A}_0}[C(A, I)] \geq \inf_{A \in \text{Algos}} \mathbb{E}_{I \sim \mathcal{I}_0}[C(A, I)] \mathbb{E}_{I \sim \mathcal{I}_0}[\text{OPT}(I)].$$

Remark

- Yao’s principle exists for other settings as well.
- Proof of “$\geq$” relatively easy to prove.
- Tightness typically follows from duality of optimisation problems or fixed point theorems.
  (though I’m not sure how it works here)
A hard distribution for Ski-Rental: Intuition

\[ I_0 := \text{Geo}(\frac{1}{B}). \]

Why \( I_0 \)?

- distribution is memoryless.
  
  Assume no skis bought on day \( i \): Minimising expected future cost is the same problem as on day 1.
  
  \( \leftrightarrow \) wlog: either buy right away or not at all.

- expectation tuned such that

\[
\mathbb{E}_{l \sim I_0}[C(\text{never buy, } l)] = \mathbb{E}_{l \sim I_0}[C(\text{immediately buy, } l)] = B.
\]

\( \leftrightarrow \) all strategies equally good
A hard distribution for Ski-Rental: Analysis

Lemma

Let $I_0 := \text{Geo}(\frac{1}{B})$ and $q := 1 - \frac{1}{B} = \Pr[\bigstar]$. Then

i. $\mathbb{E}_{I \sim I_0}[C(A, I)] = B$ for all $A \in \mathbb{N}$.

ii. $\mathbb{E}_{I \sim I_0}[\text{OPT}(I)] = B(1 - (1 - \frac{1}{B})^B)$.

Seen before:

Any random variable $X$ with values in $\mathbb{N}$ satisfies

$$\mathbb{E}[X] = \sum_{j \geq 1} \Pr[X \geq j].$$

Lower bound for Ski-Rental

By Yao’s theorem any randomised algorithm $A$ for ski-rental has competitive ratio at least

$$c_A \geq \inf_{A \in \text{Algos}} \frac{\mathbb{E}_{I \sim I_0}[C(A, I)]}{\mathbb{E}_{I \sim I_0}[\text{OPT}(I)]} = \frac{B}{B(1 - (1 - \frac{1}{B})^B)} = \frac{1}{1 - (1 - \frac{1}{B})^B}.$$  

For large $B$ the lower bound converges to

$$\lim_{B \to \infty} \frac{1}{1 - (1 - \frac{1}{B})^B} = \frac{1}{1 - 1/e} = \frac{e}{e - 1} \approx 1.582.$$
Upper bound for Ski-Rental

Remark: The lower bound is tight (Karlin et al. 1994)
There exists a distribution $\mathcal{A}$ on $[B]$ such that $c_\mathcal{A} \leq \frac{e}{e-1}$.

Applications

Very basic online question:

Should I pay a small possibly recurring cost or a large one time cost?

Occurs in:
- Cache management.
- Networking.
- Scheduling.
- …
Algorithm Design as a Two-Player Game

- “we” choose algorithm to minimise cost
- “adversary” chooses input to maximise cost
- Nash/Loomis: It does not matter who moves first if mixed strategy is allowed for first player.

Yao’s Principle

Lower bound on worst-case expected cost of any randomised algorithm $A_0$ by analysing any deterministic algorithm on specific input distribution $I_0$.

$$\max_{i \in \text{Inputs}} \mathbb{E}_{A \sim A_0} [C(A, i)] \geq C \geq \min_{A \in \text{Algos}} \mathbb{E}_{I \sim I_0} [C(A, I)].$$

Can narrow down randomised complexity $C$ of underlying problem from both sides.
Anhang: Mögliche Prüfungsfragen I

Spieltheorie:
- Was ist ein Zwei-Spieler-Spiel im Sinne der Spieltheorie?
- Was ist ein Nash-Equilibrium?
- Gibt es immer ein Nash-Equilibrium?
- Was ist ein Nullsummenspiel?
- Was besagt der Satz von Nash (für Zwei-Spieler Nullsummenspiele)?
- Was besagt der Satz von Loomis?
- Beweise den Satz von Loomis! (anspruchsvolle Aufgabe)

Yaos Prinzip:
- Worin besteht die Verbindung zwischen Spieltheorie und dem Entwurf von Algorithmen?
- Wie ist die randomisierte Komplexität (bzgl. einer Kostenfunktion $C$) normalerweise definiert? Welche andere Sichtweise ergibt sich darauf durch den Satz von Loomis?
- Formuliere Yaos Prinzip! Wofür ist es nützlich?
Anhang: Mögliche Prüfungsfragen II

Anwendung auf $\wedge$-Bäume:

- Welches Ziel haben wir uns bei der Auswertung von $\wedge$-Bäumen gesetzt? (Anfragekomplexität minimieren)
- Welche Worst-Case Kosten lassen sich mit einem deterministischen Algorithmus erreichen?
- Können randomisierte Algorithmen das besser? Wie?
- Man kann sich recht leicht überlegen, dass die randomisierte Komplexität $\Omega(\sqrt{n})$ beträgt. Wie ging das?
- Wir haben auch eine schärfere Analyse gesehen. Welche Komponenten hatte diese? Insbesondere: Wie kommt dabei Yao Prinzip zur Anwendung?
- Was besagt der Satz von Tarsi?

Ski-Rental-Problem:

- Formuliere das Ski-Rental Problem.
- Wie nennt man diese Art von Problem? (Online Problem)
- Spielt das nur im Wintersport eine Rolle? (nur Stichworte)
- Wie ist der kompetitive Faktor definiert?
Anhang: Mögliche Prüfungsfragen III

- Was ist der beste deterministische Algorithmus? Wie kann man das einsehen?
- Gibt es einen randomisierten Algorithmus der Break-Even schlagen kann? (nur die Idee)
- Formuliere Yaos Prinzip für Online Algorithmen.
- Welche Eingabeverteilung haben wir für die untere Schranke für Ski-Rental zugrunde gelegt? Was ist die Intuition?
- Welche Kosten ergeben sich für Online und Offline Algorithmen für diese Eingabeverteilung? Was lässt sich entsprechend über den kompetitiven Faktor sagen?