A Survey of Yao's Minimax Principle

Brian Shimanuki *
Mayuri Sridhar †
Chelsea Voss ‡

May 10, 2015

Abstract

1 Introduction

TODO: describe what Yao's minimax principle is, and what it is good for

We will begin with a reminder of the intuition behind Yao's minimax principle.

1.1 Statement of the principle

Yao's principle states that *the best deterministic algorithm, when run on any distribution over random inputs, is at least as good as any randomized algorithm when run on the worst-case input.* In other words,

$$\max_{x \in \mathcal{X}} \mathop{\mathbb{E}}_{r \in R} texttttcost(r, x) \geq \min_{r \in \mathcal{R}} \mathop{\mathbb{E}}_{x \in X} texttttcost(r, x)$$

where

$\mathcal{X}$ = the set of all possible inputs to the algorithm.
$\mathcal{R}$ = the set of all possible strings defining deterministic algorithms.
X = any probability distribution over inputs to the algorithm.
R = any probability distribution over strings defining deterministic algorithms.
$texttttcost(r, x)$ = the cost (usually running time) of the algorithm running on input x, if the specific algorithm to use is defined by string r.

*bshimanuki@mit.edu
†mayuri@mit.edu
‡csvoss@mit.edu

1.2 Proof of the principle

First, consider game theory. Each player i has a set of moves M_i that they may choose from. A two-player game is defined by a *payoff matrix* of outcomes for each player, a function mapping $M_1 \times M_2 \to \mathbb{R} \times \mathbb{R}$.

In a *pure strategy*, player 1 chooses exactly one move x from M_i to play. In a *mixed strategy*, player 1 instead chooses a probability distribution $\pi_i(x)$ of moves from M_i.

Let $\texttt{payoff}_1(x, y)$ denote the payoff to player 1 when the players choose moves x and y, respectively.

Lemma: If you "announce" your mixed strategy – thus committing to not change it – then there exists a pure strategy for me that does at least as good as any mixed strategy for me. That is:

$$\forall \pi_1, \pi_2, \exists x_{opt}, \mathop{\mathbb{E}}_{y \in \pi_2} \texttt{payoff}_1(x_{opt}, y) \geq \mathop{\mathbb{E}}_{y \in \pi_2, x \in \pi_1} \texttt{payoff}_1(x, y)$$

Proof. Each component move x_i of π_1 contributes a component $\pi_1(x_i)\texttt{payoff}_1(x_i, y)$ to the sum $\mathbb{E}_{x \in \pi_1} \texttt{payoff}_1(x, y)$. There exists some move x_{opt} for which $\texttt{payoff}_1(x_i, y)$ is maximized. If we choose that x_{opt} as a pure strategy instead, then $\mathbb{E}_{y \in \pi_2} \texttt{payoff}_1(x_{opt}, y) \geq \mathbb{E}_{y \in \pi_2, x \in \pi_1} \texttt{payoff}_1(x, y)$ as desired.

□

TODO: something about how this leads to the below

$$\max_{b \in \lfloor} \mathop{\mathbb{E}}_{a \in A} payoff(a, b) \leq \min_{a \in \mathcal{A}} \mathop{\mathbb{E}}_{b \in B} payoff(a, b)$$

where

$\mathcal{A}$ = set of moves available to player 1.

$\mathcal{B}$ = set of moves available to player 2.

A = any probability distribution over moves to player 1 – that is, any mixed strategy for player

B = any probability distribution over moves to player 2 – that is, any mixed strategy for player

$\texttt{payoff}(a, b)$ = the payoff to player 1

Given this inequality, we need only observe the following correspondence between algorithms and strategies and between inputs and strategies in order to yield Yao's principle.

$$\max_{x \in \mathcal{X}} \mathop{\mathbb{E}}_{r \in R} texttttcost(r, x) \geq \min_{r \in \mathcal{R}} \mathop{\mathbb{E}}_{x \in X} texttttcost(r, x)$$

- The set of moves for player 1 ($\mathcal{A}$) is the same as the set of deterministic algorithms ($\mathcal{R}$).
- The set of moves for player 2 ($\mathcal{B}$) is the same as the set of possible inputs ($\mathcal{X}$).
- A mixed strategy for player 1 (A) is the same as a randomized algorithm – that is, a probability distribution over deterministic algorithms (R).

- A mixed strategy for player 2 (B) is the same as a probability distribution over inputs to the algorithm.
- The payoff to player 1 (`payoff`) is the negated cost to player 1 ($-$`cost`).

The following table summarizes each of the parts of Yao's principle, and their roles. As we demonstrate examples which utilize Yao's principle, this table will be used in order to clarify how the principle applies to each new situation.

Randomized algorithm chooses:	some distribution R over strings r as its source of randomness
Deterministic algorithm chooses:	some fixed string r
Mixed-strategy adversary chooses:	some distribution X over inputs x for the algorithm
Pure-strategy adversary chooses:	some fixed input x
Adversary wants to maximize:	the running time of the algorithm, $textttcost(r, x)$

2 Removable Online Knapsack Problem

In this section, we present the 2-competitive solution of Han et al. [**?**] to the removable online knapsack problem as well as use Yao's Principle show $1 + 1/e$ is a lower bound on the competitiveness of any algorithm against an oblivious adversary.

The knapsack problem asks: Given a set of items e_i with weights w_i and values v_i, select a subset with the maximum sum of values such that the sum of the weights is at most the capacity of the knapsack. We call this the value of the knapsack. The offline version is a classic NP-hard optimization problem, though it has a simple dynamic programming pseudo-polynomial solution. Here, we consider the *online* version, meaning each item is given to the algorithm after the algorithm makes a decision on the previous item. *Removable* means the knapsack is a working set, from which we can remove items and add new items, but not add previously removed items. At all times, the knapsack is limited by its capacity. The objective is to maximize the value of the final knapsack. We say an algorithm is c-competitive if the expected value of the final knapsack is at least $\frac{1}{c}$ times the expected value of the optimal knapsack.

We assume the knapsack has capacity 1. This is valid because only weights relative to the knapsack's capacity matters for the capacity constraint, so we can always scale down to a unit capacity.

2.1 Upper Bound

To prove an upper bound on the competitiveness of algorithms which yield a solution to the problem, we present a 2-competitive randomized algorithm.

Since the online nature of the problem is what makes it impossible to always achieve the optimal solution, it is insightful to first consider algorithms which behave similarly in both online and offline situations. We look at MAX and $GREEDY$.

MAX simply selects the item with the largest value whose weight does not exceed 1. Since in the offline case, we can process each item in an arbitrary order and keep track of the best item so far, in the online case, we can do exactly the same with the order given to us, storing the best item so far in our knapsack.

$GREEDY$ selects the k items with the highest ratios of value to weight $\frac{v_i}{w_i}$, maximizing k while keeping the sum of the weights of the selected items at most 1. In the offline case, this can be done trivially by sorting the items by ratio and then selecting the highest k items. An online algorithm can perform at least as well by keeping a working set of the items with the highest ratio while processing each input.

$S \leftarrow \emptyset$ item e_i, in order of arrival $w_i \leq 1$ $S \leftarrow S \cup \{e_i\}$ $\sum_{e_j \in S} w_j > 1$ Remove the item from S with the smallest ratio

Since when the sum of the weights in S is greater than 1, an item with ratio less than that of the k items with the largest ratios must be in S, the k items will never be removed. Thus S is a superset of the items the offline $GREEDY$ algorithm selects.

Neither of these algorithms alone are competitive. Consider an input sequence $\{(w_i, v_i)\}$ of $\{(\epsilon, 2\epsilon),\ (\epsilon, \epsilon),\ (\epsilon, \epsilon),\ \ldots\}$ on the MAX algorithm and $\{(\epsilon, 2\epsilon),\ (1, 1)\}$ on the $GREEDY$ algorithm. In fact, no deterministic online algorithm is competitive with a constant factor [**?**]. However, using both with randomness can yield a competitive algorithm.

Theorem 1. [**?**] *Running MAX and $GREEDY$ uniformly at random is a 2-competitive algorithm.*

Proof. For a given input sequence T, let $GREEDY(T)$, $MAX(T)$, and $OPT(T)$ be the values of the sets returned by $GREEDY$, MAX, and an optimal solution, respectively. Consider the fractional knapsack solution to the offline version, that is, when we are allowed to take fractional parts of items. Then a greedy solution is optimal: Sort by ratio and take the highest ratio items until the knapsack is full, potentially taking only part of the last item in the knapsack. In the fractional knapsack problem, we have relaxed the integrality constraint, so the fractional solution is not worse than the integral solution.

Now the offline $GREEDY$ algorithm by definition selects all of these except the fractional item (if one exists). Thus the online $GREEDY$ algorithm also selects all except the fractional item. Finally the MAX algorithm selects an item with value at least that of the fractional item, so $GREEDY(T) + MAX(T) \geq OPT(T)$. Thus the competitive ratio is $\frac{2 \cdot OPT(T)}{GREEDY(T)+MAX(T)} \leq 2$. □

2.2 Lower Bound

We use Yao's Principle to prove a lower bound of $1 + 1/e$ for the competitive ratio of any removable online knapsack algorithm.

To do so, we must find a distribution of inputs for which no deterministic algorithm performs "well", which means the expected value of the output from any deterministic algorithm is at most $\frac{1}{1+1/e}$ times the expected optimum. As the power of an offline algorithm is in the additional information it has over an online algorithm, it is reasonable that in constructing our "bad" input, we would want to limit the information the algorithm has about future items. Thus it is not surprising that we will construct a family of inputs where each input is a prefix of the same base sequence.

Let the probability of $k + 1$ items be p_k. For simplicity in analysis, we will let the first item $e_0 = (w_0, v_0) = (1, 1)$ with future items having much smaller weights and values but a higher ratio so that the problem is essentially reduced to determining when to remove the first item from the knapsack.

This is reminiscent of the *ski rental problem*, one of the most well-understood online problems [**?**], where a client must choose whether and when to buy skis for a fixed cost versus paying a smaller cost each day. Our knapsack problem is the reverse of this: instead of deciding when to switch from renting skis to buying skis to minimize total cost, we are deciding when to switch from having a single large item in our knapsack to getting a stream of smaller items with a better ratio to maximize total value.

Finally, we want the expected value of removing e_0 and selecting all following items to be close to 1 at any given point. A simple way to achieve this is to let the remaining items have a constant value and have the p_k form an exponential distribution.

This leads us towards the input sequence given by Han et al. [**?**]:

$$(1,1),\ (1/n^2,1/n),\ (1/n^2,1/n),\ldots\ (1/n^2,1/n)$$

for a given value of n and where there are $k+1$ items with probability $p_k = \frac{1-e^{-1/n}}{1-e^{-n}} \cdot e^{-(k-1)/n}$ for $k = 1, 2, \ldots, n^2$.

Theorem 2. [**?**] *No online algorithm has competitive ratio less than* $1+1/e$ *for the removable online knapsack problem.*

Proof. We apply Yao's Principle on the input distribution above. The optimal strategy with full information is to keep e_0 for $k \leq n$ and remove e_0 immediately otherwise. This leads us to the expected optimal value

$$\sum_{i=1}^{n} 1 \cdot p_i + \sum_{i=n+1}^{n^2} \frac{i}{n} \cdot p_i = \frac{1-e^{-1/n}}{1-e^{-n}} \left(\sum_{i=1}^{n} e^{-(i-1)/n} + \frac{1}{n} \sum_{i=n+1}^{n^2} i \cdot e^{-(i-1)/n} \right)$$

which has value $1 + 1/e$ as $n \to \infty$.

Now an online deterministic algorithm can only decide on which item e_l to remove e_0. Then the expected value of this algorithm is

$$\sum_{i=1}^{l} 1 \cdot p_i + \sum_{i=l+1}^{n^2} \frac{i-l}{n} \cdot p_i = \frac{1-e^{-1/n}}{1-e^{-n}} \left(\sum_{i=1}^{l} e^{-(i-1)/n} + \frac{1}{n} \sum_{i=l+1}^{n^2} (i-1) \cdot e^{-(i-1)/n} \right)$$

which has value 1 as $n \to \infty$.

Thus any online algorithm has competitive ratio at least $1 + 1/e$. □

2.3 Remarks

Yao's Principle is used here to give a lower bound on the competitiveness of algorithms for the removable online knapsack problem, which is an upper bound on their effectiveness. It can be used to show the tightness of a particular solution.

Also, as we saw here, there is a dual between using an algorithm to demonstrate a lower bound on the solvability of a problem (alternatively, an upper bound on the hardness of a problem), and using a series of inputs to demonstrate an upper bound on the solvability of a problem (alternatively, an lower bound on the hardness). As more work is done on a particular problem, the gap between the two shrink until a provably optimal algorithm is found.

3 Yao's Principle in Quantum Information

Yao's principle isn't just used for proving worst-case bounds on the running time of algorithms. Another area where Yao's principle has found an application is in the study of quantum information, permitting proof of worst-case bounds for strategies in certain quantum information "games". In this section, we will examine how Yao's minimax principle may be used for this purpose, as demonstrated in *Worst Case Analysis of Non-local Games* by Ambainis et al.

3.1 Quantum Preliminaries

Consider setting up a "game" between two players and an adversary, where the two players receive random input from the adversary and must each return specific outputs based on those inputs without communicating with each other. Analyzing the best strategies for these games, under the assumptions of both classical physics and quantum physics, can be used to demonstrate that better strategies exist if the players are permitted access to quantum information.

An example of such a game is the CHSH (Clauser-Horne-Shimonyi-Holt) game:

- The two players may collude to set up a strategy beforehand, but may not communicate afterwards.
- The adversary chooses a random $\vec{x} = (x_1, x_2) \in X = \{(0,0), (0,1), (1,0), (1,1)\}$. The random choice is uniform, with $\frac{1}{4}$ probability each.
- The players 1 and 2 receive the values x_1 and x_2, respectively.
- In the quantum setting, the players 1 and 2 may also possess an entangled 2-part quantum state $|\psi\rangle$, which they may measure.
- Using their strategy, the players 1 and 2 output the values a_1 and a_2, respectively.
- The players win if $a_1 \oplus a_2 = x_1 \wedge x_2$.

Both under classical physics and under any "local hidden variable theory" – explanation of quantum phenomena as being due to extra local hidden variables – the best that the players can achieve is a win probability of 0.75. If the players are allowed to share entangled quantum bits, however, then they can win more often: $\frac{1}{2} + \frac{1}{2\sqrt{2}} = 0.8535...$ becomes the best win probability that the players can achieve. [[cite Clauser]] [[cite Ambainis 2013]]

The CHSH game was designed so as to be an actual realizable experiment, and experiments have since confirmed that the way the universe works is consistent with quantum entanglement, and not consistent with local hidden variable theories. [[cite Aspect]]

In *Worst Case Analysis of Non-local Games* by Ambainis et al., the authors examine generalized games beyond the simple CHSH game, and analyze the worst-case performance of strategies in both classical and quantum physics. [[cite Ambainis 2013]]

3.2 Generalized CHSH

The CHSH game is played between two players and an adversary; one simple generalization that can be made is to extend the game from two players to n players. Ambainis et al. call this the *n-party AND game*.

- The n players may collude to set up a strategy beforehand, but may not communicate afterwards.
- The adversary chooses a random $\vec{x} = (x_1, ..., x_n) \in X_1 \times ... \times X_n$, for some $X_1, ..., X_n$. The random choice is governed by a probability distribution π; each $\vec{x}$ is chosen with probability $\pi(\vec{x})$.
- The players $1, ..., n$ receive the values $x_1, ..., x_n$, respectively.
- In the quantum setting, the players may also possess an entangled n-part quantum state $|\psi\rangle$, which they may measure.
- Using their strategy, the players $1, ..., n$ output the values $\vec{a} = a_1, ..., a_n$, respectively.
- The players win if $\bigoplus \vec{a} = \bigwedge \vec{x}$.

The authors note that this game has not been studied before because of its triviality in the classical setting: by outputting $a_i = 0$, the players are guaranteed to win unless all $x_i = 1$. Against an adversary that chooses $\vec{x}$ uniformly, the players can win with probability $1 - 2^n$.

However, this game becomes more interesting when we consider worst-case bounds: if the adversary is allowed to choose π, the optimal strategy for the players becomes less trivial.

The authors show that $\lim_{n\to\infty} Pr(win) = 2/3$ in both the classical case and the quantum case.

[[cite Ambainis 2010]]

3.3 Fully Generalized Games

- The n players may collude to set up a strategy beforehand, but may not communicate afterwards.
- The adversary chooses a random $\vec{x} = (x_1, ..., x_n) \in X_1 \times ... \times X_n$, for some $X_1, ..., X_n$. The random choice is governed by a probability distribution π; each $\vec{x}$ is chosen with probability $\pi(\vec{x})$.
- The players $1, ..., n$ receive the values $x_1, ..., x_n$, respectively.
- In the quantum setting, the players may also possess an entangled n-part quantum state $|\psi\rangle$, which they may measure.
- Using their strategy, the players $1, ..., n$ output the values $\vec{a} = a_1, ..., a_n$, respectively.
- The players win if some proposition $Win(\vec{a}|\vec{x})$ is true.

To-do: the following citations

Ambainis, A., A. Bačkurs, K. Balodis, A. Škuškovniks, J. Smotrovs, M. Virza: Worst Case Analysis of Non-local Games. P. van Emde Boas et al. (Eds.): SOFSEM 2013, LNCS 7741, pp. 121-132, 2013. http://arxiv.org/abs/1112.2856

Ambainis, A., Kravchenko, D., Nahimovs, N., Rivosh, A.: Nonlocal Quantum XOR Games for Large Number of Players. In: Kratochvíl, J., Li, A., Fiala, J., Kolman, P. (eds.) TAMC 2010. LNCS, vol. 6108, pp. 72-83. Springer, Heidelberg (2010).

Clauser, J., M. Horne, A. Shimony, R. Holt: Proposed experiment to test local hidden-variable theories. Physical Review Letters 23, 880 (1969). http://journals.aps.org/prl/abstract/10.1103/Ph

Aspect, A., P. Grangier, G. Roger: Experimental Tests of Realistic Local Theories via Bell's Theorem. Physical Review Letters 47, 460 (1981). http://journals.aps.org/prl/abstract/10.1103/Phy

Yao, A.: Probabilistic computations: Toward a unified measure of complexity. Proceedings of the 18th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 222-227 (1977).

4 Conclusion