

## Yao's Minimax Principle

### Complexity of algorithms

The complexity of an algorithm is usually measured with respect to the *size* of the input, where size may for example refer to

- the length of a binary word describing the input,
- the number of nodes in an input graph,
- the number of elements in an input list.

Furthermore, one may be interested in

- the worst-case complexity, i.e., the complexity on the worst input of a given size,
- the average-case complexity, i.e., the average complexity over all inputs of the same size.

Finally, the complexity may refer to

- running time, the number of comparisons made, . . . .

## Yao's Minimax Principle

### Complexity of problems: lower and upper bounds

The complexity of a problem is specified by lower and upper bounds on the complexity of algorithms that solve the problem:

- lower bounds are obtained by proving (e.g., using combinatorial arguments) that no algorithm can have a complexity smaller than the lower bound under consideration,
- upper bounds are usually obtained by constructing an algorithm for the given problem that is correct and has complexity of at most the upper bound under consideration.

### The aim is to find matching lower and upper bounds

If one can derive matching lower and upper bounds for a problem, then the algorithm that yields the upper bound has minimum complexity. Similarly, close lower and upper bounds imply that the complexity of the corresponding algorithm is close to minimum.

## Yao's Minimax Principle

### Las Vegas and Monte Carlo algorithms

A randomized algorithm is a

- Las Vegas algorithm* if the algorithm is always correct,
- Monte Carlo algorithm* if the algorithm may be incorrect.

For a Las Vegas algorithm, the outcomes of the underlying source of randomness have no influence on the correctness of the result, however they may influence the running time or other parameters.

### Examples

A typical example of a Las Vegas algorithm is randomized quicksort (i.e., the pivot elements are chosen uniformly at random).

A typical example of a Monte Carlo algorithm is integration by randomized sampling (i.e., in order to determine to a certain precision the area of a geometric figure inside a unit square, pick points in the square uniformly at random and let the area be equal to the fraction of points inside the figure).

## Yao's Minimax Principle

### Worst-case complexity of Las Vegas algorithms

In what follows, we consider Las Vegas algorithms and their **expected complexity in worst case**, that is, the expected complexity on the worst input of any given size.

On first sight, it appears to be hard to obtain good lower bounds on the expected complexity of a Las Vegas algorithm in worst case.

Yao's Minimax Principle, which is shown below, asserts that for any problem and

- for any fixed probability distribution on the inputs of some given size, that any lower bound on the average-case complexity of deterministic algorithms (where the algorithm may depend on the probability distribution)

is also a lower bound on the expected complexity in worst case of Las Vegas algorithms.

## Yao's Minimax Principle

### Representing Las Vegas algorithms

We consider problems with a notion of size for the inputs such that for each size  $n$  there are

a finite set  $\mathcal{I}_n$  of inputs,

a finite set  $\mathcal{A}_n$  of correct deterministic algorithms,

a cost function  $k_n: \mathcal{A} \times \mathcal{I} \rightarrow \mathbb{N}$ ,

where  $k_n(A, I)$  is equal to the costs of applying algorithm  $A$  for solving instance  $I$ .

Furthermore, we assume that for any input size  $n$

any Las Vegas algorithm for the problem under consideration can be represented by probability distributions  $\sigma_n$  on  $\mathcal{A}_n$

where  $\text{Prob}_{\sigma_n}[A]$  is the probability that on inputs of size  $n$  the Las Vegas algorithm behaves like algorithm  $A$  in  $\mathcal{A}_n$ .

## Yao's Minimax Principle

### Definition

An algorithm for sorting is a *black-box algorithm* if an input  $x_1, \dots, x_n$  can only be accessed by queries of the form " $x_i < x_j$ ?".

### Example Randomized Quicksort

In each recursive call of randomized Quicksort a pivot element for splitting the current list is chosen uniformly at random.

The choice of the pivot elements depends on a random source.

Once the outcomes of the random source are fixed, the algorithm works like a deterministic black-box algorithm for sorting.

The deterministic algorithms obtained this way are always correct because randomized Quicksort is a Las Vegas algorithm.

## Yao's Minimax Principle

### Example Randomized Quicksort (continued)

With a linear ordering on items understood, an input for randomized Quicksort is a list of pairwise distinct items.

The size of an input is equal to the number of items in the list.

Furthermore,

$\mathcal{A}_n$  is the set of all deterministic black-box algorithms that correctly sort lists of  $n$  items,

$\mathcal{I}_n$  can be assumed to be equal to the set of all permutations of the list  $1, \dots, n$  (because we consider only black-box algorithms),

$k(A, I)$  is equal to the number of queries that algorithm  $A$  asks on input  $I$ .

## Yao's Minimax Principle

### Remark

Consider the representation of a Las Vegas algorithm by probability distributions  $\sigma_n$  on the sets  $\mathcal{A}_n$  of correct algorithms. Then for inputs of size  $n$ , the expected costs in worst case are given by

$$\max_{I \in \mathcal{I}_n} \sum_{A \in \mathcal{A}_n} \text{Prob}_{\sigma_n}[A] \cdot k_n(A, I).$$

### Remark

Any sequence of probability distributions  $\sigma_0, \sigma_1, \dots$  on  $\mathcal{I}_0, \mathcal{I}_1, \dots$  can be viewed as representing a (generalized) Las Vegas algorithm.

In case the sequence of distributions is not given effectively, the corresponding Las Vegas algorithm may not be effective.

As long as we are only interested in lower bounds, considering also such not necessarily effective Las Vegas algorithms is fine.

## Yao's Minimax Principle

### Theorem (Yao's Minimax Principle)

Let  $\mathcal{A}$  and  $\mathcal{I}$  be nonempty finite sets. Let  $k: \mathcal{A} \times \mathcal{I} \rightarrow \mathbb{N}$  be a cost function, and let  $\sigma$  and  $\tau$  be probability distributions on  $\mathcal{A}$  and  $\mathcal{I}$ . Let  $A_\sigma$  be a random variable with values in  $\mathcal{A}$  and distribution  $\sigma$ , and let  $I_\tau$  be a random variable with values in  $\mathcal{I}$  and distribution  $\tau$ . Then we have

$$\min_{A \in \mathcal{A}} \mathbf{E}[k(A, I_\tau)] \leq \max_{I \in \mathcal{I}} \mathbf{E}[k(A_\sigma, I)]. \quad (1)$$

### Remark

In Yao's Minimax Principle, we have in inequality (1) on the left-hand side the average costs on inputs chosen according to  $\tau$  for the best deterministic algorithm (which "knows"  $\tau$ ), right-hand side the expected costs for the randomized algorithm determined by  $\sigma$  on the worst input in  $\mathcal{I}$ .

## Yao's Minimax Principle

### Proof of Yao's Minimax Principle.

In order to demonstrate the theorem, we show for

$$\tilde{k} = \sum_{(A, I) \in \mathcal{A} \times \mathcal{I}} \text{Prob}_\sigma[A] \cdot \text{Prob}_\tau[I] \cdot k(A, I)$$

that we have

$$\min_{A \in \mathcal{A}} \mathbf{E}[k(A, I_\tau)] \leq \tilde{k} \leq \max_{I \in \mathcal{I}} \mathbf{E}[k(A_\sigma, I)].$$

While this is not used in what follows, observe that in case the random variables  $I_\tau$  and  $A_\sigma$  are mutually independent, we have

$$\tilde{k} = \mathbf{E}[k(A_\sigma, I_\tau)].$$

## Yao's Minimax Principle

### Proof of Yao's Minimax Principle (continued).

By choice of  $\tilde{k}$ , we obtain

$$\begin{aligned} \min_{A \in \mathcal{A}} \mathbf{E}[k(A, I_\tau)] &\leq \sum_{A \in \mathcal{A}} \text{Prob}_\sigma[A] \cdot \mathbf{E}[k(A, I_\tau)] \\ &= \sum_{A \in \mathcal{A}} \text{Prob}_\sigma[A] \sum_{I \in \mathcal{I}} \text{Prob}_\tau[I] \cdot k(A, I) \\ &= \tilde{k} \\ &= \sum_{I \in \mathcal{I}} \text{Prob}_\tau[I] \sum_{A \in \mathcal{A}} \text{Prob}_\sigma[A] \cdot k(A, I) \\ &= \sum_{I \in \mathcal{I}} \text{Prob}_\tau[I] \cdot \mathbf{E}[k(A_\sigma, I)] \leq \max_{I \in \mathcal{I}} \mathbf{E}[k(A_\sigma, I)]. \end{aligned}$$

□

## Yao's Minimax Principle

### Example Finding empty columns.

Let  $\{0, 1\}^{n \times n}$  be the set of Boolean ( $n \times n$ ) matrices.

A column of a Boolean matrix is said to be empty if all entries in the column are equal to 0.

A black-box algorithm for deciding whether a given quadratic Boolean matrix has an empty column is a correct algorithm for this problem that accesses its input matrix only by queries of the form "Is the entry in row  $i$  and column  $j$  equal to 0?".

Let the size of an input in  $\{0, 1\}^{n \times n}$  be equal to  $n$ , and let

$\mathcal{I}_n$  be equal to  $\{0, 1\}^{n \times n}$ ,

$\mathcal{A}_n$  be the set of all deterministic black-box algorithms that decide correctly for inputs of size  $n$  whether the input has an empty column,

$k_n(A, I)$  be equal to the number of entries that algorithm  $A$  checks for being equal to 0 during processing input  $I$ .

## Yao's Minimax Principle

### Proposition (Upper bound)

For the problem of deciding whether a Boolean  $(n \times n)$  matrix has an empty column, an upper bound for the expected number of queries that are necessary in worst case is

$$g_{\text{upper}}(n) = \frac{n(n+1)}{2}.$$

### Proof.

Consider the algorithm that chooses a permutation  $\pi$  of  $\{1, \dots, n\}$  uniformly at random and then, until enough information has been obtained, successively checks the columns  $\pi(1), \pi(2), \dots$  by checking in each column successively the rows  $\pi(1), \pi(2), \dots$

On an input that has an empty column, the expected number of checked columns is at most  $(n+1)/2$ , while on any other input the expected number of checks per column is at most  $(n+1)/2$ .  $\square$

## Yao's Minimax Principle

### Proposition (Lower bound)

For the problem of deciding whether a Boolean  $(n \times n)$  matrix has an empty column, a lower bound for the expected number of queries that are necessary in worst case is

$$g_{\text{lower}}(n) = \frac{n(n+1)}{2}.$$

### Proof.

It can be shown that any black-box Las Vegas algorithm that is correct for all inputs in  $\mathcal{I}_n$  can be represented by a probability distribution  $\sigma_n$  on  $\mathcal{A}_n$ , hence the assertion of the proposition can be rephrased as

$$g_{\text{lower}}(n) \leq \max_{I \in \mathcal{I}_n} \mathbf{E}[k(A_{\sigma_n}, I)].$$

## Yao's Minimax Principle

### Proof (continued).

By Yao's Minimax Principle, we have for all probability distributions  $\sigma_n$  on  $\mathcal{A}_n$  and  $\tau_n$  on  $\mathcal{I}_n$ ,

$$\min_{A \in \mathcal{A}_n} \mathbf{E}[k(A, I_{\tau_n})] \leq \max_{I \in \mathcal{I}_n} \mathbf{E}[k(A_{\sigma_n}, I)]. \quad (2)$$

Thus in order to prove the proposition, it suffices to specify for all  $n$  a probability distribution  $\tau_n$  on  $\mathcal{I}_n$  such that the function  $g_{\text{lower}}$  is a lower bound on the left-hand side of inequality (2).

In fact, it suffices to specify for any fixed  $n$  and for any  $\varepsilon > 0$  a probability distribution  $\tau_n^\varepsilon$  on  $\mathcal{I}_n$  such that

$$(1 - \varepsilon) g_{\text{lower}}(n) \leq \min_{A \in \mathcal{A}_n} \mathbf{E}[k(A, I_{\tau_n^\varepsilon})].$$

## Yao's Minimax Principle

### Proof (continued).

In order to specify for given  $n$  and  $\varepsilon > 0$  a probability distribution  $\tau_n^\varepsilon$  on  $\mathcal{I}_n$ ,

let  $\mathcal{D}_n$  be the set of all Boolean  $(n \times n)$  matrices that have exactly one entry 1 per column.

Then define  $\tau_n^\varepsilon$  such that

the subset  $\mathcal{D}_n$  of  $\mathcal{I}_n$  has probability  $1 - \varepsilon$  and the distribution within this set is uniform,

the set  $\mathcal{I}_n \setminus \mathcal{D}_n$  has probability  $\varepsilon$  and the distribution within this set is again uniform.

## Yao's Minimax Principle

### Proof (continued).

A correct deterministic algorithm for the empty column problem can reject an input in  $\mathcal{D}_n$  only after reading at least one symbol 1 in every column.

For a matrix chosen uniformly at random from  $\mathcal{D}_n$ , for each column the expected number of entries that have to be read before the single 1 is found is

$$\sum_{i=1}^n \frac{i}{n} = \frac{n+1}{2}.$$

Hence the expected number of entries that have to be read in total is  $\frac{n(n+1)}{2} = g_{\text{lower}}(n)$ , where the expectation is with respect to choosing the input uniformly at random from  $\mathcal{D}_n$ , which is done with probability  $(1 - \epsilon)$ .  $\square$

## Excursus on game theory

### Example Two finger morra.

In the game of two finger morra each of two players show simultaneously either one or two fingers.

Let  $k \in \{2, 3, 4\}$  be the total count of fingers shown. In case

In case  $k$  is even, Player 1 has to pay  $k$  units to Player 2,

In case  $k$  is odd, Player 1 is paid  $k$  units by Player 2.

The payoff in this game can be represented by the matrix

$$\begin{pmatrix} -2 & 3 \\ 3 & -4 \end{pmatrix},$$

where the rows and columns correspond to the strategies in the set  $\mathcal{A} = \mathcal{I} = \{1, 2\}$  of Player 1 and 2, respectively, and the entry  $k(i, j)$  in row  $i$  and column  $j$  is equal to the gain of Player 1.

## Excursus on game theory

### Matrix games

A *matrix game*  $(\mathcal{A}, \mathcal{I}, \mathbf{k})$  is played by two players. The first player selects a strategy  $A$  from a finite set  $\mathcal{A}$ , the second player selects a strategy  $I$  from a finite set  $\mathcal{I}$ , and the first and second player try to minimize and to maximize, respectively, the payoff  $k(A, I)$ .

A matrix game can be represented by a matrix with  $|\mathcal{A}|$  rows and  $|\mathcal{I}|$  columns that has entries of the form  $k(A, I)$ .

As indicated by the notation used above, the situation of Yao's Minimax Principle can be viewed as a matrix game where the first player selects an algorithm from  $\mathcal{A}$  and the second player selects an input from  $\mathcal{I}$ , while the matrix contains entries of the form  $k(A, I)$ .

In technical terms, matrix games are finite two-player zero-sum games with incomplete information.

## Excursus on game theory

### Mixed strategies

In a matrix game  $(\mathcal{A}, \mathcal{I}, k)$ , the strategies in  $\mathcal{A}$  and  $\mathcal{I}$  are called *pure strategies* for the first and second player, respectively.

A *mixed strategy* for a player in a matrix game is a probability distribution on the set of strategies for this player, which we identify with a corresponding random variable  $A_\sigma$  or  $I_\tau$ .

When mixed strategies are allowed, the first and second player try to minimize and maximize, respectively, the expected payoff

$$\mathbf{E}[k(A_\sigma, I_\tau)] = \sum_{(A, I) \in (\mathcal{A}, \mathcal{I})} \text{Prob}_\sigma[A] \cdot \text{Prob}_\tau[I] \cdot k(A, I).$$

## Excursus on game theory

Example Two finger morra (continued).

Recall the payoff matrix  $K$  for the game of two finger morra

$$K = \begin{pmatrix} -2 & 3 \\ 3 & -4 \end{pmatrix}.$$

Any player who plays a known pure strategy will lose.

In case both players play mixed strategies with probabilities of  $1/2$  for each of their two respective pure strategies, the expected payoff of Player 1 is

$$\begin{aligned} \mathbf{E}[k(A_\sigma, l_\tau)] &= \sum_{(A,l) \in (\mathcal{A}, \mathcal{I})} \text{Prob}_\sigma[A] \cdot \text{Prob}_\tau[l] \cdot k(A, l) \\ &= \frac{1}{4}(-2 + 3 + 3 + -4) = 0. \end{aligned}$$

In fact, one of the players has a better strategy (see exercises).

## Excursus on game theory

Definition

For a matrix game  $(\mathcal{A}, \mathcal{I}, k)$ , a mixed strategy  $\sigma^*$  for Player 1 is *optimal*, if it holds that

$$\max_{\tau} \mathbf{E}[k(A_{\sigma^*}, l_\tau)] = \min_{\sigma} \max_{\tau} \mathbf{E}[k(A_\sigma, l_\tau)] \quad (3)$$

Similarly, a mixed strategy  $\tau^*$  for Player 2 is *optimal*, if it holds that

$$\min_{\sigma} \mathbf{E}[k(A_\sigma, l_{\tau^*})] = \max_{\tau} \min_{\sigma} \mathbf{E}[k(A_\sigma, l_\tau)] \quad (4)$$

Remark

In a matrix game, each player has an optimal strategy, which, however, is not necessarily unique. Here it suffices to observe that payoffs, hence also expected payoffs are bounded, and that the set of mixed strategies of each player is compact (in the sense of calculus), hence all minima and maxima in the definition of optimal strategy exist and are attained for appropriate mixed strategies.

## Excursus on game theory

Remark

By playing an optimal mixed strategy  $\sigma^*$ , Player 1 can enforce that the expected payoff is at most

$$k_1^* = \max_{\tau} \mathbf{E}[k(A_{\sigma^*}, l_\tau)] = \min_{\sigma} \max_{\tau} \mathbf{E}[k(A_\sigma, l_\tau)],$$

no matter what mixed strategy Player 2 plays.

Similarly, by playing an optimal mixed strategy  $\tau^*$ , Player 2 can enforce that the expected payoff is at least

$$k_2^* = \min_{\sigma} \mathbf{E}[k(A_\sigma, l_{\tau^*})] = \max_{\tau} \min_{\sigma} \mathbf{E}[k(A_\sigma, l_\tau)].$$

Remark

Observe that the values  $k_1^*$  and  $k_2^*$  do not depend on the choice of the optimal strategies  $\sigma^*$  and  $\tau^*$  that occur in their definitions.

## Excursus on game theory

Remark

For any pair  $\sigma^*$  and  $\tau^*$  of optimal strategies for the two players in a matrix game, we have

$$k_2^* \leq \mathbf{E}[k(A_{\sigma^*}, l_{\tau^*})] \leq k_1^*,$$

hence, in particular,

$$k_2^* \leq k_1^*.$$

By optimality of  $\sigma^*$  and by definition of  $k_1^*$ , the expected payoff when playing  $\sigma^*$  and  $\tau^*$  must be at most  $k_1^*$ , and must be at least  $k_2^*$  by a similar argument for  $\tau^*$ . In particular, since optimal strategies always exist,  $k_2^* \leq k_1^*$

Von Neumann's celebrated Minimax Theorem asserts that for any matrix game the values  $k_2^*$  and  $k_1^*$  are in fact the same. Before we review this theorem (without proving it), we discuss an equivalent formulation that asserts the existence of equilibrium points.

## Excursus on game theory

### Definition

For any matrix game  $(\mathcal{A}, \mathcal{I}, k)$ , a pair of mixed strategies  $\sigma_0$  and  $\tau_0$  for the first and second player, respectively, is an *equilibrium point* in case

for all mixed strategies  $\sigma$  of Player 1 it holds that

$$\mathbf{E}[k(A_{\sigma_0}, I_{\tau_0})] \leq \mathbf{E}[k(A_{\sigma}, I_{\tau_0})],$$

and for all mixed strategies  $\tau$  of Player 2 it holds that

$$\mathbf{E}[k(A_{\sigma_0}, I_{\tau_0})] \geq \mathbf{E}[k(A_{\sigma_0}, I_{\tau})].$$

## Excursus on game theory

### Von Neumann's Minimax Theorem

For any matrix game  $(\mathcal{A}, \mathcal{I}, k)$ , it holds that

$$\max_{\tau} \min_{\sigma} \mathbf{E}[k(A_{\sigma}, I_{\tau})] = \min_{\sigma} \max_{\tau} \mathbf{E}[k(A_{\sigma}, I_{\tau})] \quad (5)$$

The following reformulation of von Neumann's Minimax Theorem relies on the fact that the optimum strategy against a fixed mixed strategy can always be chosen to be a pure strategy.

### Variant of von Neumann's Minimax Theorem

For any matrix game  $(\mathcal{A}, \mathcal{I}, k)$ , it holds that

$$\max_{\tau} \min_{A \in \mathcal{A}} \mathbf{E}[k(A, I_{\tau})] = \min_{\sigma} \max_{I \in \mathcal{I}} \mathbf{E}[k(A_{\sigma}, I)].$$

## Excursus on game theory

### Proposition

A matrix game  $(\mathcal{A}, \mathcal{I}, k)$  has an equilibrium point  $(\sigma_0, \tau_0)$  if and only if the values of  $k_1^*$  and  $k_2^*$  are the same.

### Proof.

If there is an equilibrium point  $(\sigma_0, \tau_0)$ , the corresponding expected payoff  $k$  must be at least  $k_1^*$ , because Player 1 can enforce an expected payoff of at most  $k$  by playing  $\sigma_0$ , while by definition  $k_1^*$  is the minimum expected payoff that Player 1 can enforce. By a similar argument,  $k \leq k_2^*$ , hence by  $k_2^* \leq k_1^*$  as shown above,

$$k_1^* \leq k \leq k_2^* \leq k_1^*.$$

In case  $k_1^* = k_2^*$ , any pair of optimal strategies  $\sigma^*$  and  $\tau^*$  is an equilibrium point with expected gain of  $k_1^* = k_2^*$ . As long as Player 1 sticks to  $\sigma^*$ , Player 2 cannot force the expected payoff to be larger than  $k_1^*$ , and similarly for the symmetric case.

## Excursus on game theory

### Proof that the variant is equivalent to the Minimax Theorem.

For arbitrary mixed strategies  $\sigma$  and  $\tau$  it holds that

$$\begin{aligned} \mathbf{E}[k(A_{\sigma}, I_{\tau})] &= \sum_{(A, I) \in (\mathcal{A}, \mathcal{I})} \text{Prob}_{\sigma}[A] \cdot \text{Prob}_{\tau}[I] \cdot k(A, I) \\ &= \sum_{I \in \mathcal{I}} \text{Prob}_{\tau}[I] \cdot \mathbf{E}[k(A_{\sigma}, I)] \leq \max_{I \in \mathcal{I}} \mathbf{E}[k(A_{\sigma}, I)], \end{aligned}$$

hence for any given  $\sigma$ ,  $\max_{\tau} \mathbf{E}[k(A_{\sigma}, I_{\tau})] = \max_{I \in \mathcal{I}} \mathbf{E}[k(A_{\sigma}, I)]$ , where  $\geq$  is trivial and  $\leq$  holds by the preceding discussion. Then

$$\min_{\sigma} \max_{\tau} \mathbf{E}[k(A_{\sigma}, I_{\tau})] = \min_{\sigma} \max_{I \in \mathcal{I}} \mathbf{E}[k(A_{\sigma}, I)],$$

i.e., the right-hand sides of the equations that occur in the Minimax Theorem and its variant have the same value. A similar argument for the left-hand sides then concludes the proof.  $\square$

## Excursus on game theory

From von Neumann's Minimax Theorem we obtain as a corollary the following reformulation of Yao's Minimax Principle.

### Theorem (Equivalent form of Yao's Minimax Principle)

Let  $(\mathcal{A}, \mathcal{I}, k)$  be a matrix game and let  $A_{\sigma_0}$  and  $I_{\tau_0}$  be mixed strategies for the first and second player. Then it holds that

$$\min_{A \in \mathcal{A}} \mathbf{E}[k(A, I_{\tau_0})] \leq \max_{I \in \mathcal{I}} \mathbf{E}[k(A_{\sigma_0}, I)].$$

### Proof.

The variant of von Neumann's Minimax Theorem yields

$$\begin{aligned} \min_{A \in \mathcal{A}} \mathbf{E}[k(A, I_{\tau_0})] &\leq \max_{\tau} \min_{A \in \mathcal{A}} \mathbf{E}[k(A, I_{\tau})] \\ &= \min_{\sigma} \max_{I \in \mathcal{I}} \mathbf{E}[k(A_{\sigma}, I)] \leq \max_{I \in \mathcal{I}} \mathbf{E}[k(A_{\sigma_0}, I)]. \end{aligned}$$

□