# $5^{\text{TH}}$ TUTORIAL ON RANDOMIZED ALGORITHMS

Yao's principle for proving lower bounds on randomized algorithms

**1**. *Deterministic game tree lower bound.* Show that every deterministic algorithm for evaluating a game tree needs to read the values of all leaves on some input. For simplicity, consider only complete balanced binary trees of height $2k$ with NOR nodes and binary values $x_i$ in the leaves (the number of leaves is thus $n = 4^k$).

Additionally, argue that a lower bound for NOR trees implies the same lower bound for AND/OR trees.

**2**. *Sorting.* Show a lower bound on the expected running time of any Las Vegas comparison-based algorithm for sorting $n$ numbers.

**3**. *Searching.* Show a lower bound on the expected running time of any Las Vegas comparison-based algorithm for searching in a sorted array.

**4**. *Perfect matching.* Let $G$ be an $n$-vertex graph for an even $n$ such that we only have a query access to the edges, namely, we can only ask whether two nodes are connected or not. Show that $\Omega(n^2)$ queries are needed in expectation for any algorithm that correctly determines whether $G$ has a perfect matching or not.

**5**. *Bonus: Majority element in a query model.* Given a list of values $v_1, \ldots, v_n$, the goal is to find an index $i$, if one exists, such that the value $v_i$ occurs more than $n/2$ times in the list. Determine a lower bound on the expected running time of any Las Vegas algorithm that solves the problem, but is restricted to only ask equality queries; that is, in each step the algorithm specifies indexes $i, j$ and is told whether $v_i = v_j$ or not. The algorithm cannot access $v_i$'s in any other way. (Can you match the lower bound up to an $O(1)$ factor by a deterministic algorithm?)

**6**. *Eigenvalues warm-up.* Let $A \in \mathbb{R}^{n \times n}$ be a matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$. Show that the matrix $A + dI_n$ has eigenvalues $d + \lambda_1, \ldots, d + \lambda_n$. What other basic properties of eigenvalues do you know?