Randomized Algorithms

— Game Theoretic View & Minimax Principles

Joseph Chuang-Chieh Lin

Department of Computer Science & Information Engineering,
Tamkang University

Fall 2023

Outline

Outline

Payoff Matrix

	Scissors	Paper	Stone
Scissors	0	1	-1
Paper	-1	0	1
Stone	1	-1	0

- Rows: Alice's choices.
- Columns: Bob's choices.
- Entry position (i, j): state or profile.
- Entry value: the amount paid by Bob to Alice.

Payoff Matrix (the explicit form)

	Scissors	Paper	Stone
Scissors	$(0, 0)$	$(1, -1)$	$(-1, 1)$
Paper	$(-1, 1)$	$(0, 0)$	$(1, -1)$
Stone	$(1, -1)$	$(-1, 1)$	$(0, 0)$

- Rows: Alice's choices.
- Columns: Bob's choices.
- Entry position (i, j): state or profile.
- Entry value: the amount paid by Bob to Alice.

Outline

A Matrix M

	B1	B2	B3	B4	B5
A1	0	-1	2	-3	4
A2	-5	6	-7	8	-9
A3	10	-11	12	-13	14
A4	-15	16	-17	18	-19
A5	20	-21	22	-23	24

A Matrix M

	B1	B2	B3	B4	B5
A1	0	−1	2	−3	4
A2	−5	6	−7	8	−9
A3	10	−11	12	−13	14
A4	−15	16	−17	18	−19
A5	20	−21	22	−23	24

- What is $\min_j M_{1j}$? $\min_j M_{2j}$? $\min_j M_{3j}$? $\min_j M_{4j}$? $\min_j M_{5j}$?
- What is $\max_i M_{i1}$? $\max_i M_{i2}$? $\max_i M_{i3}$? $\max_i M_{i4}$? $\max_i M_{i5}$?

A Matrix M

	B1	B2	B3	B4	B5
A1	0	-1	2	-3	4
A2	-5	6	-7	8	-9
A3	10	-11	12	-13	14
A4	-15	16	-17	18	-19
A5	20	-21	22	-23	24

- What is $\min_j M_{1j}$? $\min_j M_{2j}$? $\min_j M_{3j}$? $\min_j M_{4j}$? $\min_j M_{5j}$?
- What is $\max_i M_{i1}$? $\max_i M_{i2}$? $\max_i M_{i3}$? $\max_i M_{i4}$? $\max_i M_{i5}$?

A Matrix M

	B1	B2	B3	B4	B5
A1	0	-1	2	-3	4
A2	-5	6	-7	8	-9
A3	10	-11	12	-13	14
A4	-15	16	-17	18	-19
A5	20	-21	22	-23	24

- $\max_i \min_j M_{ij} =$

A Matrix M

	B1	B2	B3	B4	B5
A1	0	-1	2	-3	4
A2	-5	6	-7	8	-9
A3	10	-11	12	-13	14
A4	-15	16	-17	18	-19
A5	20	-21	22	-23	24

- $\max_i \min_j M_{ij} = -3$.

A Matrix M

	B1	B2	B3	B4	B5
A1	0	-1	2	-3	4
A2	-5	6	-7	8	-9
A3	10	-11	12	-13	14
A4	-15	16	-17	18	-19
A5	20	-21	22	-23	24

- $\max_i \min_j M_{ij} = -3.$
- $\min_j \max_i M_{ij} =$

A Matrix M

	B1	B2	B3	B4	B5
A1	0	-1	2	-3	4
A2	-5	6	-7	8	-9
A3	10	-11	12	-13	14
A4	-15	16	-17	18	-19
A5	20	-21	22	-23	24

- $\max_i \min_j M_{ij} = -3$.
- $\min_j \max_i M_{ij} = 16$.

Exercise

Observation

For all payoff matrices M,

$$\max_i \min_j M_{ij} \leq \min_j \max_i M_{ij}$$

Exercise

Observation

For all payoff matrices M,

$$V_R = \max_i \min_j M_{ij} \leq \min_j \max_i M_{ij} = V_C$$

- When the equality holds, the game is said to have a solution (saddle point) and the value is $V = V_R = V_C$.

Hint

$$\min_j M_{ij} \leq \max_i Mij?$$

Let

- $f(i) = \min_j M_{ij}, \quad j^* = \arg\min_j M_{ij}$.
- $g(j) = \max_i M_{ij}, \quad i^* = \arg\max_i M_{ij}$.

Hint

$$\min_j M_{ij} \leq \max_i Mij?$$

Let

- $f(i) = \min_j M_{ij}, \quad j^* = \arg\min_j M_{ij}.$
- $g(j) = \max_i M_{ij}, \quad i^* = \arg\max_i M_{ij}.$

We have

- $\forall j, \;\; M_{i,j^*} \leq M_{ij}.$

Hint

$$\min_j M_{ij} \leq \max_i Mij?$$

Let

- $f(i) = \min_j M_{ij}, \quad j^* = \arg\min_j M_{ij}.$
- $g(j) = \max_i M_{ij}, \quad i^* = \arg\max_i M_{ij}.$

We have

- $\forall j,\ M_{i,j^*} \leq M_{ij}.$
- $\forall i,\ M_{i,j} \leq M_{i^*j}.$

Hint

$$\min_j M_{ij} \leq \max_i Mij?$$

Let

- $f(i) = \min_j M_{ij}, \quad j^* = \arg\min_j M_{ij}.$
- $g(j) = \max_i M_{ij}, \quad i^* = \arg\max_i M_{ij}.$

We have

- $\forall j,\ M_{i,j^*} \leq M_{ij}.$
- $\forall i,\ M_{i,j} \leq M_{i^*j}.$
- $\forall \boldsymbol{i}\, \forall \boldsymbol{j}, \quad M_{i,j^*} \leq M_{i^*,j}.$

Hint

$$\min_j M_{ij} \le \max_i Mij?$$

Let

- $f(i) = \min_j M_{ij}, \quad j^* = \arg\min_j M_{ij}.$
- $g(j) = \max_i M_{ij}, \quad i^* = \arg\max_i M_{ij}.$

We have

- $\forall j, \ \ M_{i,j^*} \le M_{ij}.$
- $\forall i, \ \ M_{i,j} \le M_{i^* j}.$
- $\forall \boldsymbol{i} \, \forall \boldsymbol{j}, \quad M_{i,j^*} \le M_{i^*,j}.$ (since $M_{i,j^*} \le M_{ij} \le M_{i^*,j}$)

Example

	Scissors	Paper	Stone
Scissors	0	1	2
Paper	-1	0	1
Stone	-2	-1	0

- Now, we have $V_R = V_C = 0$, so $V = 0$.

Example

	Scissors	Paper	Stone
Scissors	0	1	2
Paper	-1	0	1
Stone	-2	-1	0

- Now, we have $V_R = V_C = 0$, so $V = 0$.
- What if a game has no solution (i.e., no saddle point)?

Example

	Scissors	Paper	Stone
Scissors	0	1	2
Paper	-1	0	1
Stone	-2	-1	0

- Now, we have $V_R = V_C = 0$, so $V = 0$.
- What if a game has no solution (i.e., no saddle point)?

⋆ Introduce randomization in the choice of strategies.

Example

	Scissors (33%)	Paper (33%)	Stone (33%)
Scissors (33%)	0	1	2
Paper (33%)	-1	0	1
Stone (33%)	-2	-1	0

- Now, we have $V_R = V_C = 0$, so $V = 0$.
- What if a game has no solution (i.e., no saddle point)?

⋆ Introduce randomization in the choice of strategies.

Mixed Strategies

Mixed Strategies

A mixed strategy is a probability distribution on the set of possible strategies.

- $\mathbf{p} = (p_1, \ldots, p_n)$: probability distribution on the rows of $\boldsymbol{M}$.
- $\mathbf{q} = (q_1, \ldots, q_m)$: probability distribution on the columns of $\boldsymbol{M}$.
- The payoff (of Alice) now becomes a random variable.

Mixed Strategies

Mixed Strategies

A mixed strategy is a probability distribution on the set of possible strategies.

- $\mathbf{p} = (p_1, \ldots, p_n)$: probability distribution on the rows of $\boldsymbol{M}$.
- $\mathbf{q} = (q_1, \ldots, q_m)$: probability distribution on the columns of $\boldsymbol{M}$.
- The payoff (of Alice) now becomes a random variable.

$$\mathbb{E}[\text{payoff}] = \mathbf{p}^\top \boldsymbol{M}\mathbf{q} = \sum_{i=1}^{n} \sum_{j=1}^{m} p_i M_{ij} q_j.$$

Best over distributions

$$
\begin{aligned}
V_R &= \max_{\mathbf{p}} \min_{\mathbf{q}} \mathbf{p}^\top \boldsymbol{M} \mathbf{q} \\
V_C &= \min_{\mathbf{q}} \max_{\mathbf{p}} \mathbf{p}^\top \boldsymbol{M} \mathbf{q}
\end{aligned}
$$

Best over distributions

$$\begin{aligned} V_R &= \max_{\mathbf{p}} \min_{\mathbf{q}} \mathbf{p}^\top \boldsymbol{M} \mathbf{q} \\ V_C &= \min_{\mathbf{q}} \max_{\mathbf{p}} \mathbf{p}^\top \boldsymbol{M} \mathbf{q} \end{aligned}$$

von Neumann's Minimax Theorem

For any two-player zero-sum game specified by a matrix $\boldsymbol{M}$,

$$\max_{\mathbf{p}} \min_{\mathbf{q}} \mathbf{p}^\top \boldsymbol{M} \mathbf{q} = \min_{\mathbf{q}} \max_{\mathbf{p}} \mathbf{p}^\top \boldsymbol{M} \mathbf{q}.$$

- The saddle-point exists here and the two distributions $\mathbf{p}$ and $\mathbf{q}$ are called optimal mixed-strategies.

von Neumann's Minimax Theorem

For any two-player zero-sum game specified by a matrix $\boldsymbol{M}$,

$$\max_{\mathbf{p}} \min_{\mathbf{q}} \mathbf{p}^\top \boldsymbol{M} \mathbf{q} = \min_{\mathbf{q}} \max_{\mathbf{p}} \mathbf{p}^\top \boldsymbol{M} \mathbf{q}.$$

- Once $\mathbf{p}$ is fixed, $\mathbf{p}^\top \boldsymbol{M} \mathbf{q}$ is a linear function of $\mathbf{q}$ and can be minimized by setting 1 to the q_j with the smallest coefficient in the function.
- If C knows the distribution $\mathbf{p}$ being used by R, then its optimal strategy is a pure strategy.

Loomis' Theorem

Loomis' Theorem

For any two-player zero-sum game specified by a matrix $\boldsymbol{M}$,

$$\max_{\mathbf{p}} \min_{j} \mathbf{p}^{\top} \boldsymbol{M} \mathbf{e}_j = \min_{\mathbf{q}} \max_{i} \mathbf{e}_i^{\top} \boldsymbol{M} \mathbf{q}.$$

- $\mathbf{e}_k$: a unit vector with value 1 in the kth position and 0's elsewhere.

Example (when $\mathbf{q}$ is fixed)

	$q_1 = \frac{1}{8}$ Scissors	$q_2 = \frac{1}{2}$ Paper	$q_3 = \frac{3}{8}$ Stone
p_1 Scissors	0	1	-1
p_2 Paper	-1	0	1
p_3 Stone	1	-1	0

Example (when $\mathbf{q}$ is fixed)

	$q_1 = \frac{1}{8}$ Scissors	$q_2 = \frac{1}{2}$ Paper	$q_3 = \frac{3}{8}$ Stone
p_1 Scissors	0	1	-1
p_2 Paper	-1	0	1
p_3 Stone	1	-1	0

- $\mathbf{p}^\top \boldsymbol{M}\mathbf{q} =$

Example (when $\mathbf{q}$ is fixed)

	$q_1 = \frac{1}{8}$ Scissors	$q_2 = \frac{1}{2}$ Paper	$q_3 = \frac{3}{8}$ Stone
p_1 Scissors	0	1	-1
p_2 Paper	-1	0	1
p_3 Stone	1	-1	0

- $\mathbf{p}^\top \boldsymbol{M}\mathbf{q} = \frac{1}{8}p_1 + \frac{1}{4}p_2 + (-\frac{3}{8})p_3$.
  So we should choose $\mathbf{p} = [0 \;\; 1 \;\; 0]^\top$ for utility maximization.

Example (when **q** is fixed; Nash equilibrium)

	$q_1 = \frac{1}{3}$	$q_2 = \frac{1}{3}$	$q_3 = \frac{1}{3}$
	Scissors	Paper	Stone
p_1 Scissors	0	1	-1
p_2 Paper	-1	0	1
p_3 Stone	1	-1	0

Example (when $\mathbf{q}$ is fixed; Nash equilibrium)

	$q_1 = \frac{1}{3}$ Scissors	$q_2 = \frac{1}{3}$ Paper	$q_3 = \frac{1}{3}$ Stone
p_1 Scissors	0	1	-1
p_2 Paper	-1	0	1
p_3 Stone	1	-1	0

- $\mathbf{p}^\top \boldsymbol{M}\mathbf{q} =$

Example (when $\mathbf{q}$ is fixed; Nash equilibrium)

	$q_1 = \frac{1}{3}$	$q_2 = \frac{1}{3}$	$q_3 = \frac{1}{3}$
	Scissors	Paper	Stone
p_1 Scissors	0	1	-1
p_2 Paper	-1	0	1
p_3 Stone	1	-1	0

- $\mathbf{p}^\top \boldsymbol{M}\mathbf{q} = \frac{1}{3}p_1 + \frac{1}{3}p_2 + \frac{1}{3}p_3$.
  So we should choose $\mathbf{p} = [?\ \ ?\ \ ?]^\top$ for utility maximization.

Example (when $\mathbf{q}$ is fixed; Nash equilibrium)

	$q_1 = \frac{1}{3}$ Scissors	$q_2 = \frac{1}{3}$ Paper	$q_3 = \frac{1}{3}$ Stone
p_1 Scissors	0	1	-1
p_2 Paper	-1	0	1
p_3 Stone	1	-1	0

- $\mathbf{p}^\top \boldsymbol{M}\mathbf{q} = \frac{1}{3}p_1 + \frac{1}{3}p_2 + \frac{1}{3}p_3$.

  So we should choose $\mathbf{p} = [? \;\; ? \;\; ?]^\top$ for utility maximization.

  Can you find any $\mathbf{p} \neq \left[\frac{1}{3} \;\; \frac{1}{3} \;\; \frac{1}{3}\right]^\top$ which leads to better expected payoff?

Outline

The Intuitive Idea

- View the algorithm designer as the column player C.
  - The columns: the set of all possible algorithms.
  - Each column: a pure strategy of C; a deterministic algorithm which is always correct.
  - ⋆ V_C: the worst-case running time of any deterministic algorithm.
- View the adversary choosing the input as the row player R.
  - The rows: the set of all possible inputs (of fixed size).
  - Each row: a pure strategy of R; a specific input.
  - ⋆ V_R: the non-deterministic complexity of the problem.
- The payoff from C to R: some real-valued measure of the performance of an algorithm.
  - E.g., running time, solution quality, space, etc.

When considering mixed-strategies

- A mixed-strategy for C: a probability distribution over the space of always correct deterministic algorithms (Las Vegas).
- A mixed-strategy for R: a probability distribution over the space of all inputs.

Distributional Complexity

The expected running time of the best deterministic algorithm for the worst distribution on the inputs.

When considering mixed-strategies

- A mixed-strategy for C: a probability distribution over the space of always correct deterministic algorithms (Las Vegas).
- A mixed-strategy for R: a probability distribution over the space of all inputs.

Distributional Complexity

The expected running time of the best deterministic algorithm for the worst distribution on the inputs.

- Smaller than the deterministic complexity since the algorithms knows the input distribution.

When considering mixed-strategies

- A mixed-strategy for C: a probability distribution over the space of always correct deterministic algorithms (Las Vegas).
- A mixed-strategy for R: a probability distribution over the space of all inputs.

Distributional Complexity

The expected running time of the best deterministic algorithm for the worst distribution on the inputs.

- Smaller than the deterministic complexity since the algorithms knows the input distribution.
- Loomis' Theorem implies that
  the distributional complexity = the least possible expected running time achievable by any randomized algorithm.

Corollary

- Let Π be a problem with a finite set $\mathcal{I}$ of input instances of fixed size.
- Let $\mathcal{A}$ be a finite set of deterministic algorithms.
- Let $C(I, A)$ denote the running time of algorithm $A \in \mathcal{A}$ on input $I \in \mathcal{I}$.
- Let $\mathbf{p}$ be a probability distribution over $\mathcal{I}$.
- Let $\mathbf{q}$ be a probability distribution over $\mathcal{A}$.

Let $I_{\mathbf{p}}$ be a random input chosen according to $\mathbf{p}$ and $A_{\mathbf{q}}$ be a randomized algorithm chosen according to $\mathbf{q}$. Then

$$\max_{\mathbf{p}} \min_{\mathbf{q}} \mathbb{E}[C(I_{\mathbf{p}}, A_{\mathbf{q}})] = \min_{\mathbf{q}} \min_{\mathbf{p}} \mathbb{E}[C(I_{\mathbf{p}}, A_{\mathbf{q}})]$$

and

$$\max_{\mathbf{p}} \min_{A \in \mathcal{A}} \mathbb{E}[C(I_{\mathbf{p}}, A)] = \min_{\mathbf{q}} \min_{I \in \mathcal{I}} \mathbb{E}[C(I, A_{\mathbf{q}})].$$

Result by Andrew C.-C. Yao

Yao's Minimax Principle

For all distributions **p** over $\mathcal{I}$ and **q** over $\mathcal{A}$,

$$\min_{A \in \mathcal{A}} \mathbb{E}[C(I_{\mathbf{p}}, A)] \leq \max_{I \in \mathcal{I}} \mathbb{E}[C(I, A_{\mathbf{q}})]$$

Result by Andrew C.-C. Yao

Yao's Minimax Principle

For all distributions **p** over $\mathcal{I}$ and **q** over $\mathcal{A}$,

$$\min_{A \in \mathcal{A}} \mathbb{E}[C(I_{\mathbf{p}}, A)] \leq \max_{I \in \mathcal{I}} \mathbb{E}[C(I, A_{\mathbf{q}})]$$

The expected running time of the optimal deterministic algorithm for an arbitrarily chosen input distribution **p** is a lower bound on the expected running time of the optimal Las Vegas randomized algorithm for problem Π.

Result by Andrew C.-C. Yao

Yao's Minimax Principle

For all distributions **p** over $\mathcal{I}$ and **q** over $\mathcal{A}$,

$$\min_{A \in \mathcal{A}} \mathbb{E}[C(I_{\mathbf{p}}, A)] \leq \max_{I \in \mathcal{I}} \mathbb{E}[C(I, A_{\mathbf{q}})]$$

The expected running time of the optimal deterministic algorithm for an arbitrarily chosen input distribution **p** is a lower bound on the expected running time of the optimal Las Vegas randomized algorithm for problem Π.

- Trick: choose a suitable **p** and be aware of that the deterministic algorithm knows **p**.

Extension to Monte Carlo Type Randomized Algorithms

Proposition [Yao FOCS 1977]

For

- all distributions $\mathbf{p}$ over $\mathcal{I}$,
- all distributions $\mathbf{q}$ over $\mathcal{A}$,
- any $\epsilon \in [0, 1/2]$,

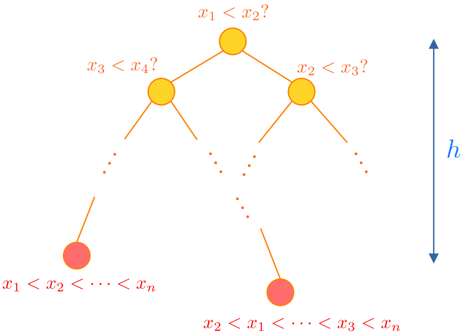
we have

$$\frac{1}{2}\left(\min_{A \in \mathcal{A}} \mathbb{E}[C_{2\epsilon}(I_{\mathbf{p}}, A)]\right) \leq \max_{I \in \mathcal{I}} \mathbb{E}[C_{\epsilon}(I, A_{\mathbf{q}})]$$

- $\mathbb{E}[C_\epsilon(I_{\mathbf{p}}, A)]$: the expected running time of a deterministic algorithm A that errs with probability $\leq \epsilon$.

Outline

Comparison-Based Sorting Algorithms



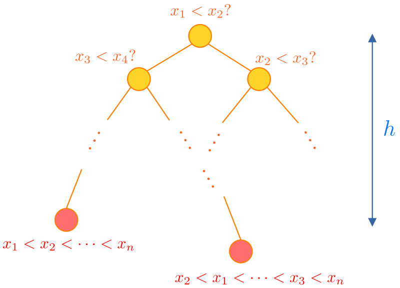
- Examples: MergeSort, QuickSort, BubbleSort, SelectionSort, HeapSort, etc.
- Non-examples: RadixSort, BucketSort, etc.

Our Goal

Theorem

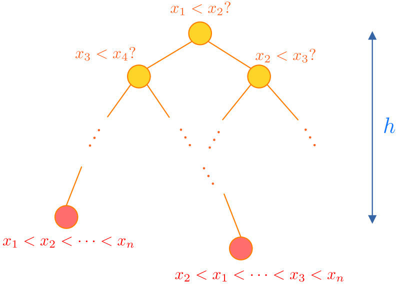
Any comparison-based Las Vegas sorting algorithm requires expected $\Omega(n \log n)$ time steps.

Analysis (1/3)



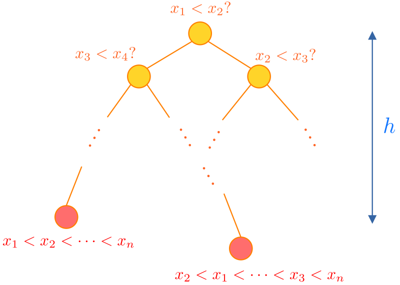
- A decision tree which models any comparison-based sorting algorithm.

Analysis (1/3)



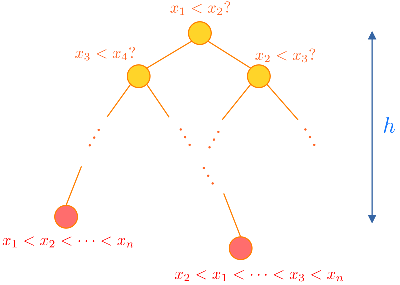
- A decision tree which models any comparison-based sorting algorithm.
- Each tree leaf corresponds to a permutation (i.e., sorted result).

Analysis (1/3)



- A decision tree which models any comparison-based sorting algorithm.
- Each tree leaf corresponds to a permutation (i.e., sorted result).
  - Assume that the set of all permutations is uniformly distributed.

Analysis (1/3)



- A decision tree which models any comparison-based sorting algorithm.
- Each tree leaf corresponds to a permutation (i.e., sorted result).
  - Assume that the set of all permutations is uniformly distributed.
- Tree depth h: number of comparisons made by the algorithm.

Analysis (2/3)

- By the pigeonhole principle, we must have $2^h \geq n!$.

[1] Note that $\lg_2(\cdot) = \log_2(\cdot)$.

Analysis (2/3)

- By the pigeonhole principle, we must have $2^h \geq n!$.
- Thus[1],

$$\begin{aligned} h \geq \lg n! &= \lg n(n-1)\cdots 2\cdot 1 = \sum_{i=2}^{n} \lg i \\ &\geq \sum_{i=n/2+1}^{n} \lg i \geq \sum_{i=n/2+1}^{n} \lg\left(\frac{n}{2}\right) \\ &= \frac{n}{2}\lg\left(\frac{n}{2}\right) = \Omega(n\log n). \end{aligned}$$

[1] Note that $\lg_2(\cdot) = \log_2(\cdot)$.

Analysis (2/3)

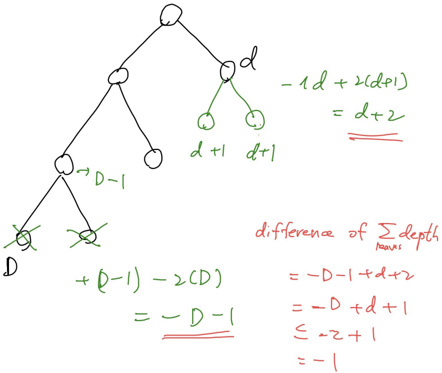
- By the pigeonhole principle, we must have $2^h \geq n!$.
- Thus[1],

$$\begin{aligned} h \geq \lg n! &= \lg n(n-1)\cdots 2 \cdot 1 = \sum_{i=2}^{n} \lg i \\ &\geq \sum_{i=n/2+1}^{n} \lg i \geq \sum_{i=n/2+1}^{n} \lg \left(\frac{n}{2}\right) \\ &= \frac{n}{2} \lg \left(\frac{n}{2}\right) = \Omega(n \log n). \end{aligned}$$

- **Note:** This only bounds the maximum depth of a leaf in the tree.

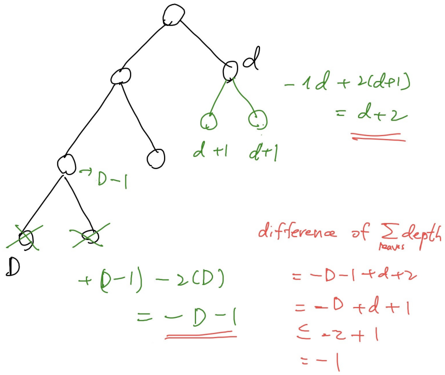
[1] Note that $\lg_2(\cdot) = \log_2(\cdot)$.

Analysis (3/3)



The average (i.e., expected) depth of the decision tree minimized when the tree is a completely balanced.

Analysis (3/3)



The average (i.e., expected) depth of the decision tree minimized when the tree is a completely balanced. $\Longrightarrow$ $\Omega(\lg n!) = \Omega(n \log n)$ expected depth.

Discussions