# Introduction to Probability 

Lecture 12: Online Algorithms
Mateja Jamnik, Thomas Sauerwald
University of Cambridge, Department of Computer Science and Technology email: \{mateja.jamnik,thomas.sauerwald\}@cl.cam.ac.uk


## Outline

## Stopping Problem 1: Dice Game

## Stopping Problem 2: The Secretary Problem

A Generalisation: The Odds Algorithm (non-examinable)

Appendix (non-examinable)

Introduction: Dice Game
(6) 8 8
$\square$

## Introduction: Dice Game

## 

Dice Game

- We throw a fair, six-sided dice $n$ times


## Introduction: Dice Game

 $\bullet \bullet \bullet \bullet \bullet \bullet \bullet \bullet!$Dice Game

- We throw a fair, six-sided dice $n$ times
- After each throw, you can either STOP or CONTINUE
- You win if you STOP at the last 6 within the $n$ throws


## Introduction: Dice Game

$\bullet \bullet \bullet \bullet \bullet \bullet \bullet \bullet!$

Dice Game

- We throw a fair, six-sided dice $n$ times
- After each throw, you can either STOP or CONTINUE
- You win if you STOP at the last 6 within the $n$ throws

Example ( $n=10$ )

## Introduction: Dice Game

## 

## Dice Game

- We throw a fair, six-sided dice $n$ times
- After each throw, you can either STOP or CONTINUE
- You win if you STOP at the last 6 within the $n$ throws

Example ( $n=10$ )

- $3,5,6,4,2, \underbrace{3}_{\text {Stop }}, 1,2,6,5$


## Introduction: Dice Game

## 

Dice Game

- We throw a fair, six-sided dice $n$ times
- After each throw, you can either STOP or CONTINUE
- You win if you STOP at the last 6 within the $n$ throws

Example ( $n=10$ )

- $3,5,6,4,2, \underbrace{3}_{\text {sTop }}, 1,2,6,5 \Rightarrow$ LOSE!


## Introduction: Dice Game

## 

Dice Game

- We throw a fair, six-sided dice $n$ times
- After each throw, you can either STOP or CONTINUE
- You win if you STOP at the last 6 within the $n$ throws

Example ( $n=10$ )

- 3, 5, 6, 4, 2, $\underbrace{3}_{\text {STOP }}, 1,2,6,5 \Rightarrow$ LOSE!
- $3,5, \underbrace{6}_{\text {sTop }}, 4,2,3,1,2,6,5$


## Introduction: Dice Game

## 

Dice Game

- We throw a fair, six-sided dice $n$ times
- After each throw, you can either STOP or CONTINUE
- You win if you STOP at the last 6 within the $n$ throws

Example ( $n=10$ )

- 3, 5, 6, 4, 2, $\underbrace{3}_{\text {STOP }}, 1,2,6,5 \Rightarrow$ LOSE!
- $3,5, \underbrace{6}_{\text {stop }}, 4,2,3,1,2,6,5 \Rightarrow$ LOSE!
- $3,5,6,4,2,3,1,2, \underbrace{6}_{\text {STOP }}$,


## Introduction: Dice Game

## 

Dice Game

- We throw a fair, six-sided dice $n$ times
- After each throw, you can either STOP or CONTINUE
- You win if you STOP at the last 6 within the $n$ throws

Example ( $n=10$ )

- 3, 5, 6, 4, 2, $\underbrace{3}_{\text {STOP }}, 1,2,6,5 \Rightarrow$ LOSE!
- $3,5, \underbrace{6}_{\text {stop }}, 4,2,3,1,2,6,5 \Rightarrow$ LOSE!
- $3,5,6,4,2,3,1,2, \underbrace{6}_{\text {STOP }}, 5 \Rightarrow$ WIN!


## 

## Dice Game

- We throw a fair, six-sided dice $n$ times
- After each throw, you can either STOP or CONTINUE
- You win if you STOP at the last 6 within the $n$ throws

What is the optimal strategy for maximising the probability of winning?

Example ( $n=10$ )

- 3, 5, 6, 4, 2, $\underbrace{3}_{\text {STOP }}, 1,2,6,5 \Rightarrow$ LOSE!
- $3,5, \underbrace{6}_{\text {STOP }}, 4,2,3,1,2,6,5 \Rightarrow$ LOSE!
- $3,5,6,4,2,3,1,2, \underbrace{6}_{\text {STOP }}, 5 \Rightarrow$ WIN!


## Introduction: Dice Game

$\bullet \bullet \bullet \bullet \bullet \bullet \bullet \bullet$

## Dice Game

- We throw a fair, six-sided dice $n$ times
- After each throw, you can either STOP or CONTINUE
- You win if you STOP at the last 6 within the $n$ throws

What is the optimal strategy for maximising the probability of winning?

Example ( $n=10$ )

- 3, 5, 6, 4, 2, $\underbrace{3}_{\text {STOP }}, 1,2,6,5 \Rightarrow$ LOSE!

This boils down to finding a threshold from which we STOP as soon as a 6 is thrown.

- $3,5, \underbrace{6}_{\text {STop }}, 4,2,3,1,2,6,5 \Rightarrow$ LOSE!
$\Rightarrow$ WIN!
- $3,5,6,4,2,3,1,2, \underbrace{6}_{\text {STOP }}$,


## Dice Game (Solution)

$\mathbf{P}$ [obtain exactly one 6 in last $k$ throws ] =

## Dice Game (Solution)

$\mathbf{P}$ [obtain exactly one 6 in last $k$ throws ] $=\binom{k}{1} \cdot \frac{1}{6} \cdot\left(\frac{5}{6}\right)^{k-1}$

## Dice Game (Solution)

$$
\mathbf{P}[\text { obtain exactly one } 6 \text { in last } k \text { throws }]=\binom{k}{1} \cdot \frac{1}{6} \cdot\left(\frac{5}{6}\right)^{k-1}=\frac{k}{6} \cdot\left(\frac{5}{6}\right)^{k-1}
$$

## Dice Game (Solution)

$\mathbf{P}$ [obtain exactly one 6 in last $k$ throws ] $=\binom{k}{1} \cdot \frac{1}{6} \cdot\left(\frac{5}{6}\right)^{k-1}=\frac{k}{6} \cdot\left(\frac{5}{6}\right)^{k-1}$


## Dice Game (Solution)

$\mathbf{P}$ [obtain exactly one 6 in last $k$ throws ] $=\binom{k}{1} \cdot \frac{1}{6} \cdot\left(\frac{5}{6}\right)^{k-1}=\frac{k}{6} \cdot\left(\frac{5}{6}\right)^{k-1}$


- This is maximised for $k=6$ (or $k=5$ )


## Dice Game (Solution)

$$
\mathbf{P}[\text { obtain exactly one } 6 \text { in last } k \text { throws }]=\binom{k}{1} \cdot \frac{1}{6} \cdot\left(\frac{5}{6}\right)^{k-1}=\frac{k}{6} \cdot\left(\frac{5}{6}\right)^{k-1}
$$



- This is maximised for $k=6$ (or $k=5) \Rightarrow$ best strategy: wait until we have 6 (5) throws left, and then stop at the first 6


## Dice Game (Solution)

$$
\mathbf{P}[\text { obtain exactly one } 6 \text { in last } k \text { throws }]=\binom{k}{1} \cdot \frac{1}{6} \cdot\left(\frac{5}{6}\right)^{k-1}=\frac{k}{6} \cdot\left(\frac{5}{6}\right)^{k-1}
$$



- This is maximised for $k=6$ (or $k=5$ ) $\Rightarrow$ best strategy: wait until we have 6 (5) throws left, and then stop at the first 6
- Probability of success is:


## Dice Game (Solution)

$$
\mathbf{P} \text { [ obtain exactly one } 6 \text { in last } k \text { throws ] }=\binom{k}{1} \cdot \frac{1}{6} \cdot\left(\frac{5}{6}\right)^{k-1}=\frac{k}{6} \cdot\left(\frac{5}{6}\right)^{k-1}
$$



- This is maximised for $k=6($ or $k=5) \Rightarrow$ best strategy: wait until we have 6 (5) throws left, and then stop at the first 6
- Probability of success is:

$$
\left(\frac{5}{6}\right)^{5}
$$

## Dice Game (Solution)

$$
\mathbf{P} \text { [ obtain exactly one } 6 \text { in last } k \text { throws ] }=\binom{k}{1} \cdot \frac{1}{6} \cdot\left(\frac{5}{6}\right)^{k-1}=\frac{k}{6} \cdot\left(\frac{5}{6}\right)^{k-1}
$$



- This is maximised for $k=6($ or $k=5) \Rightarrow$ best strategy: wait until we have 6 (5) throws left, and then stop at the first 6
- Probability of success is:

$$
\left(\frac{5}{6}\right)^{5} \approx 0.40 .
$$

## Illustration of the Three Possibilities



## Illustration of the Three Possibilities



## Outline

## Stopping Problem 1: Dice Game

Stopping Problem 2: The Secretary Problem

## A Generalisation: The Odds Algorithm (non-examinable)

Appendix (non-examinable)

## The Secretary Problem

The Problem

- We are interviewing $n$ candidates for one job in a sequential, random order


## The Secretary Problem

The Problem

- We are interviewing $n$ candidates for one job in a sequential, random order
- A candidate must be accepted (STOP) or rejected immediately after the interview and cannot be recalled


## The Secretary Problem

The Problem

- We are interviewing $n$ candidates for one job in a sequential, random order
- A candidate must be accepted (STOP) or rejected immediately after the interview and cannot be recalled
- Goal: maximise the probability of hiring the best candidate


## The Secretary Problem

The Problem

- We are interviewing $n$ candidates for one job in a sequential, random order
- A candidate must be accepted (STOP) or rejected immediately after the interview and cannot be recalled
- Goal: maximise the probability of hiring the best candidate
also known as marriage problem (Kepler 1613), hiring problem or best choice problem.


## The Secretary Problem

The Problem

- We are interviewing $n$ candidates for one job in a sequential, random order
- A candidate must be accepted (STOP) or rejected immediately after the interview and cannot be recalled
- Goal: maximise the probability of hiring the best candidate
also known as marriage problem (Kepler 1613), hiring problem or best choice problem.

Further Remarks

## The Secretary Problem

The Problem

- We are interviewing $n$ candidates for one job in a sequential, random order
- A candidate must be accepted (STOP) or rejected immediately after the interview and cannot be recalled
- Goal: maximise the probability of hiring the best candidate
also known as marriage problem (Kepler 1613), hiring problem or best choice problem.


## Further Remarks

- After seeing candidate $i$, we only know the relative order among the first $i$ candidates.


## The Secretary Problem

## The Problem

- We are interviewing $n$ candidates for one job in a sequential, random order
- A candidate must be accepted (STOP) or rejected immediately after the interview and cannot be recalled
- Goal: maximise the probability of hiring the best candidate
also known as marriage problem (Kepler 1613), hiring problem or best choice problem.


## Further Remarks

- After seeing candidate $i$, we only know the relative order among the first $i$ candidates.
$\Rightarrow$ For our problem we may as well assume that the only information we have when interviewing candidate $i$ is whether that candidate is best among $\{1, \ldots, i\}$ or not.

Illustration
unknown permutation:


Illustration
unknown permutation:
4,


Illustration
unknown permutation:
4,


Illustration
unknown permutation:
4, 7,


Illustration
unknown permutation:
4, 7,


Illustration
unknown permutation:
4, 7, 8,


Illustration
unknown permutation:
4, 7, 8,


Illustration
unknown permutation:
4, 7, 8, 6,


Illustration
unknown permutation:
4, 7, 8, 6,


Illustration
unknown permutation:
4, 7, 8, 6,


Illustration
unknown permutation:
4, 7, 8, 6,


Illustration
unknown permutation:
4, 7, 8, 6, 18,


Illustration
unknown permutation:
4, 7, 8, 6, 18,


Illustration
unknown permutation:
$4,7,8,6,18,11$,


Illustration
unknown permutation:
4, 7, 8, 6, 18, 11,


Illustration
unknown permutation:
4, 7, 8, 6, 18, 11,


Illustration
unknown permutation:
$4,7,8,6,18,11$,


Illustration
unknown permutation:
$4,7,8,6,18,11,3$,


Illustration
unknown permutation:
$4,7,8,6,18,11,3$,


Illustration
unknown permutation:
$4,7,8,6,18,11,3$,


Illustration
unknown permutation:
4, 7, 8, 6, 18, 11, 3,


Illustration
unknown permutation:
$4,7,8,6,18,11,3,5$,


Illustration
unknown permutation:
$4,7,8,6,18,11,3,5$,

unknown permutation:
$4,7,8,6,18,11,3,5$,

unknown permutation:
$4,7,8,6,18,11,3,5$,

unknown permutation:
$4,7,8,6,18,11,3,5,9$,

unknown permutation:
$4,7,8,6,18,11,3,5,9$,

unknown permutation:
$4,7,8,6,18,11,3,5,9$,

unknown permutation:

$$
4,7,8,6,18,11,3,5,9
$$


unknown permutation:
4, 7, 8, 6, 18, 11, 3, 5, 9, 13,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13$,
value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,
$$

value

unknown permutation:
4, 7, 8, 6, 18, 11, 3, 5, 9, 13,
value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,17
$$

value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17$,

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20$,
value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,17,2,20,14,
$$

value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,17,2,20,14,
$$

value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12$,
value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15
$$

value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15
$$

value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15
$$

value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15
$$

value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10
$$

value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10
$$

value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16,19$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16,19$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16,19$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16,19$,
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16,19,1$.
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16,19,1$.
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16,19,1$.
value

unknown permutation:
$4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16,19,1$.
value

unknown permutation:

$$
4,7,8,6,18,11,3,5,9,13,17,2,20,14,12,15,10,16,19,1 .
$$

value


## Two Basic Strategies

$\square$

## Two Basic Strategies

Naive Approach

- Always pick the first (or any other) candidate


## Two Basic Strategies

Naive Approach

- Always pick the first (or any other) candidate
- Probability for success is:

$$
\mathbf{P} \text { [ hire best candidate ] }
$$

## Two Basic Strategies

Naive Approach

- Always pick the first (or any other) candidate
- Probability for success is:

$$
\mathbf{P}[\text { hire best candidate }]=\frac{1}{n} .
$$

## Two Basic Strategies

Naive Approach

- Always pick the first (or any other) candidate
- Probability for success is:

$$
\mathbf{P}[\text { hire best candidate }]=\frac{1}{n}
$$

Smarter Approach

## Two Basic Strategies

Naive Approach

- Always pick the first (or any other) candidate
- Probability for success is:

$$
\mathbf{P}[\text { hire best candidate }]=\frac{1}{n}
$$

Smarter Approach

- Reject the first $n / 2$ candidates, then take the first candidate that is better than the first $n / 2$


## Two Basic Strategies

Naive Approach

- Always pick the first (or any other) candidate
- Probability for success is:

$$
\mathbf{P}[\text { hire best candidate }]=\frac{1}{n}
$$

Smarter Approach

- Reject the first $n / 2$ candidates, then take the first candidate that is better than the first $n / 2$ (if none is taken before, take last candidate)


## Two Basic Strategies

Naive Approach

- Always pick the first (or any other) candidate
- Probability for success is:

$$
\mathbf{P}[\text { hire best candidate }]=\frac{1}{n}
$$

## A typical exploration-exploitation based strategy.

Smarter Approach


- Reject the first $n / 2$ candidates, then take the first candidate that is better than the first $n / 2$ (if none is taken before, take last candidate)


## Two Basic Strategies

Naive Approach

- Always pick the first (or any other) candidate
- Probability for success is:

$$
\mathbf{P}[\text { hire best candidate }]=\frac{1}{n}
$$

A typical exploration-exploitation based strategy.
Smarter Approach


- Reject the first $n / 2$ candidates, then take the first candidate that is better than the first $n / 2$ (if none is taken before, take last candidate)

How good is this approach?

## Analysis of the Refined Approach

## Example 1

Find a lower bound on the success probability of the refined approach (picking the first candidate better than the first $n / 2$ ).

## Finding the Optimal Strategy (1/2)

- Observation 1: At interview $i$, it only matters if current candidate is best so far (i.e., no benefit in counting how many "best-so-far" candidates we had).

- Observation 2: If at interview $i$, the best strategy is to accept the candidate (if it is "best-so-far"), then the same holds for interview $i+1$

Optimal Strategy

- Explore but reject the first $x-1$ candidates
- Accept first candidate $i \geq x$ which is better than all candidates before


## Example 2

Find $x$ which maximises the probability of hiring the best candidate.

## Probability for Success (Illustration)

Suppose $n=50$ :
$\mathbf{P}$ [success]


## Another Variant of the Secretary Problem

## Postdoc Problem (Vanderbei'80)

- same setup as in the classical secretary problem
- difference: we want to pick the second-best (the best postdoc will go somewhere else)
- Success probability of the optimal strategy is:

$$
\frac{0.25 n^{2}}{n(n-1)} \quad \xrightarrow{n \rightarrow \infty} \frac{1}{4}
$$

- Thus it is easier to pick the best than the second-best(!)


## Outline

## Stopping Problem 1: Dice Game

## Stopping Problem 2: The Secretary Problem

A Generalisation: The Odds Algorithm (non-examinable)

Appendix (non-examinable)

## Details of the Odds Algorithm

- Let $I_{1}, I_{2}, \ldots, I_{n}$ be a sequence of independent indicators and let $p_{j}=\mathbf{E}\left[I_{j}\right]$
- Let $r_{j}:=\frac{p_{j}}{1-p_{j}}$ (the odds) and $p_{j} \in(0,1)$ for all $j=1,2, \ldots, n$


## Example 3

What is the probability that after trial $k$, there is exactly one success?

$$
\mathbf{P}\left[\sum_{j=k}^{n} l_{j}=1\right]=\sum_{j=k}^{n} p_{j} \cdot \prod_{k \leq j \leq n, j \neq i}^{n}\left(1-p_{i}\right)=\sum_{j=k}^{n} r_{j} \cdot\left(\prod_{i=k}^{n}\left(1-p_{i}\right)\right)
$$

- it turns out that $\mathbf{P}\left[\sum_{j=k}^{n} l_{j}=1\right]$ is unimodal in $k \Rightarrow$ there is an ideal point from which on we should stop at the first success!

Odds Algorithm ("Sum the Odds to One and Stop", F. Thomas Bruss, 2000)

1. Let $k^{*}$ be the largest $k$ such that $\sum_{j=k}^{n} r_{j} \geq 1$
2. Ignore everything before the $k^{*}$-th trial, then stop at the first success.

- The success probability is $\sum_{j=k^{*}}^{n} r_{j} \cdot\left(\prod_{i=k^{*}}^{n}\left(1-p_{i}\right)\right)$.
- This algorithm always executes the optimal strategy!

Illustration of the probability of having the last success $(n=100)$


Source: Group Fibonado

## Example 4

## Use the Odds Algorithm to analyse the Secretary Problem.

- Let $l_{j}=1$ if and only if secretary $j$ is the best secretary so far.
- The $l_{j}$ 's are independent (this is an question is on the exercise sheet)
- Then:

$$
\begin{aligned}
p_{j} & =\mathbf{P}\left[l_{j}=1\right]=\frac{1}{j} \\
r_{j} & =\frac{p_{j}}{1-p_{j}}=\frac{1 / j}{(j-1) / j}=\frac{1}{j-1}
\end{aligned}
$$

- Largest $k$ for which $\sum_{j=k}^{n} \frac{1}{j-1} \geq 1$ is $k=1 / e \cdot n$
- Probability for success:

$$
\mathbf{P}\left[\sum_{j=k}^{n} l_{j}=1\right]=\sum_{j=k}^{n} r_{j} \cdot\left(\prod_{i=k}^{n}\left(1-p_{i}\right)\right)
$$

We re-derived the solution of the secretary problem as a special case!

$$
\begin{aligned}
& =\sum_{j=k}^{n} \frac{1}{j-1} \cdot\left(\prod_{i=k}^{n} \frac{i-1}{i}\right) \\
& =\sum_{j=k}^{n} \frac{1}{j-1} \cdot \frac{k-1}{n} \approx \frac{1}{e}
\end{aligned}
$$

## Outline

## Stopping Problem 1: Dice Game

## Stopping Problem 2: The Secretary Problem

## A Generalisation: The Odds Algorithm (non-examinable)

Appendix (non-examinable)

## Even More Variants of the Secretary Problem...

In 1990, Robbins introduced four versions of the Secretary Problem:

1. Classical Secretary Problem: Observe the relative ranks. Goal: Maximise probability for hiring best candidate (We studied this!)
2. Informed Secretary Problem: Each candidate has a random value in $[0,1]$ (determining their ranks), which we see after interviewing. Goal: Maximise probability for hiring best candidate
3. No-Information Expected-Rank Problem: Observe the relative ranks. Goal: Minimise the expected rank (lower rank = better)
4. Full-Information Expected-Rank Problem (a.k.a. Robbins problem): Each candidate has a random value in $[0,1]$, which we see after interviewing. Goal: Minimise the expected rank. only this is still unsolved!

Many more versions could be considered, e.g.:

- Classical Secretary Problem: Maximise probability for hiring worst candidate (this could be solved by "inversion")
- Secretary Problem with Payoffs: Each candidate has a value drawn randomly from $[0,1]$. Goal: Maximise the expected value.


## Thank you and Best Wishes for the Exam!

