# Distributed Secretary Problem: A Case of Two Uneven Queues


Ke Ding
Department of Computing
Hong Kong Polytechnic University
Hong Kong, China
coco-ke.ding@connect.polyu.hk

Bo Li
Department of Computing
Hong Kong Polytechnic University
Hong Kong, China
comp-bo.li@polyu.edu.hk

Fangxiao Wang
Department of Computing
Hong Kong Polytechnic University
Hong Kong, China
fangxiao.wang@connect.polyu.hk



## Abstract

Secretary problem is one of the most widely studied online stochastic models, in which an employer wants to hire the best candidate from $n$ candidates who arrive in a random order. It is well-known that the optimal success probability is $\frac{1}{e}$. However, in reality, things are more complex because employers often have interviewers in different cities, interviewing candidates in a distributed manner. This motivates us to study the secretary problem with multiple queues. Feldman and Tennenholtz [EC 2012] studied this assuming the candidates are distributed evenly. In particular, when there are two even queues, the optimal success probability is $\frac{1}{4}$. In this work, we move to the general problem when the queues are arbitrary and design the optimal online algorithm for the case of two queues. Our results characterize the exact success probability curve, connecting the cases of a single queue and two equal queues. Our technique is grounded on the linear programming framework introduced by Buchbinder et al. [Math. Oper. Res. 2014] and a novel analysis.



## CCS Concepts

• **Theory of computation** → **Design and analysis of algorithms**.

## Keywords

Secretary Problem, Multi-Queue, Linear Programming

**ACM Reference Format:**
Ke Ding, Bo Li, and Fangxiao Wang. 2025. Distributed Secretary Problem: A Case of Two Uneven Queues. In *Companion Proceedings of the ACM Web Conference 2025 (WWW Companion '25), April 28-May 2, 2025, Sydney, NSW, Australia.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3701716.3715529


## 1 Introduction

In the classical secretary problem, an employer would like to select the best candidate among $n$ comparable candidates. The candidates are assumed to arrive in a random order. After each interview, the employer must decide whether to select the current candidate or reject her irrevocably. The objective is to hire the best candidate with the largest probability. A simple well-known two-phase algorithm interviews but rejects the first $\frac{n}{e}$ candidates and then hires the first candidate who is better than all previous candidates. This algorithm is optimal and hires the best candidate with probability $\frac{1}{e}$ [5, 8]. In practice, the problem can be more complicated. The employer may assign different interviewers to different cities, and the best candidate may appear in any city. The interviews in each city are simultaneous and independent until one candidate is hired by some queue, and all interviews halt. For example, in a job fair scenario, different recruiters are assigned to booths in various cities or sections of the venue. The best potential employee could be present at any of these booths, and recruiters conduct interviews independently until a hiring decision is made, thereby halting the process for the others. We refer the readers to Feldman and Tennenholtz [6] for more examples. This motivates us to study the distributed secretary problem with multiple queues.

In the distributed setting, the candidates in different cities are not directly comparable. There are several reasons. The interviewers may not have instant communications (for example, due to efficiency concerns) and their scores may not be comparable (for example, they may adopt different questions or have different evaluation criteria). Further, the interview speeds of different interviewers are the same. This is because each interview is usually given a fixed amount of time. When one of the interviewers decides to hire a candidate, she can notify the other interviewers to halt further interviews. The interviewers' joint objective is to hire the best candidate with the largest probability.

### 1.1 Our Model

This model has been studied by Feldman and Tennenholtz [6], where they assume the candidates are evenly and randomly partitioned into multiple queues. This may not be true in practice as different cities have different populations and different preferences. Thus, in the current work, we consider the general case when the candidates are not evenly partitioned. For the case of two queues with uneven $n_1 \le n_2$, we prove that a simple stopping strategy is surprisingly optimal:

- If the two queues differ a lot (i.e., $n_2$ is much larger than $n_1$), we simply reject all candidates in the shorter queue, and adopt the optimal two-phase algorithm for the longer queue (i.e., interviewing the first $\frac{1}{e}$ fraction of candidates without selection, and selecting the first best-so-far candidate thereafter).
- If the two queues are similar in length (i.e., $n_1$ is at least a large fraction of $n_2$), we adopt two-phase algorithms for both queues. Interestingly, the first phase in both queues has the same length. This may not be expected since the two queues still have different lengths in this case.

The success probability curve is illustrated in Figure 1 when $n_1$ and $n_2$ change. It is worth noting that the classic problems of a single queue and two even queues are exactly the two extreme




*WWW Companion '25, Sydney, NSW, Australia*



ACM ISBN 979-8-4007-1331-6/2025/04
https://doi.org/10.1145/3701716.3715529

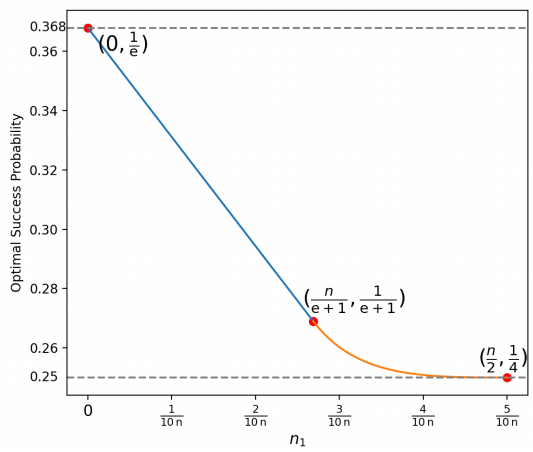


**Figure 1: Success Probability. In this curve, we set $n = n_1 + n_2$ and $n_1$ ranges from 0 to $\frac{n}{2}$. When $n_1 = 0$, our model degenerates to the classic one-queue model and has success probability of $\frac{1}{e}$; When $n_1 = \frac{n}{2}$, our model degenerates to the two-even-queue model [6] and has success probability of $\frac{1}{4}$.**

cases in our model with $n_1 = 0$ and $n_1 = n_2$. Our results connect these two models and demonstrate how the success probability shifts as the two queues become increasingly more balanced. We prove the optimality of the algorithm using the linear programming technique introduced by Buchbinder et al. [3].

## 1.2 Related Work

The classic secretary problem was introduced in a puzzle by Gardner [7]. Later, Dynkin [5], Lindley [8] gave the optimal strategy – the two-phase algorithm. Subsequently, many variants of the secretary problem have been studied in the literature. For example, Correa et al. [4] considered the fairness of the secretary problem, where the candidates are classified into different groups. There is also a line of literature that extends the single selection to multiple selections under various constraints, such as Matroids [2] and Knapsack [1].

Buchbinder et al. [3] first proposed the linear programming technique for the secretary problem, and reduce designing optimal algorithm to computing the optimal solution for LP. This approach is powerful and has been applied to more complicated settings, such as [4, 6]. Our work is most closely related to the work by Feldman and Tennenholtz [6], who proposed the secretary problem with multiple queues. They assume the candidates are evenly and randomly partitioned into $d$ queues, and designed the optimal algorithm with success probability $d^{-d/(d-1)}$ if selecting one candidate. Later, Sun et al. [9] considered a more general problem called the Q-queue J-choice K-best, but the candidates are still evenly distributed.

## 2 Preliminaries

In a distributed secretary problem, a set $\mathcal{N}$ of $n$ comparable candidates (i.e., there is a strict total order over them) are randomly partitioned into two queues, denoted by $Q_1$ and $Q_2$ with length $n_1$ and $n_2$, where $n_1 \le n_2$ and $n_1 + n_2 = n$. The candidates within each queue are ordered randomly. Two interviewers interview $Q_1$ and $Q_2$ simultaneously and independently. They do not communicate with each other (so that the candidates in different queues are not directly comparable) until one of them decides to hire one particular candidate. The interview speeds are the same and normalized to 1 so we use integers $t = 1, 2, \ldots, n_2$ to denote the time stamp when the interviewers are interviewing the $t$-th candidate $Q_i[t]$ in each queue $i = 1, 2$, or only the second queue if $t > n_1$. Throughout the paper, we assume $n$ is sufficiently large.

An algorithm $\pi$, given its input $n_1, n_2$, describes a hiring strategy for each of the two queues. The algorithm considers the candidates sequentially. For each $t = 1, 2, \ldots, n_1$, the algorithm observes values $Q_1[t]$ and $Q_2[t]$ and their relative ranks so far in their own queues. The algorithm makes an irreversible decision whether to hire $Q_1[t]$ or $Q_2[t]$ (in which case the algorithm terminates) or reject both and move to the next time stamp. We break ties in favor of $Q_1$ so that within each time stamp, the decision is first made on $Q_1$ and then $Q_2$. If $n_1 \neq n_2$, for each $t = n_1 + 1, \ldots, n_2$, no new candidate exists in queue 1 and the algorithm only observes $Q_2[t]$ and its relative ranks so far in queue 2 in case that the algorithm still not terminates. When $t = n_2$ but the algorithm has not hired any candidate, the algorithm outputs empty. The objective is to maximize the probability of hiring the best candidate. We call an event in which the best candidate is hired a *success event*, and the probability of hiring the best candidate *success probability*.

## 3 Linear Programming Representation

In the following, we use the linear programming technique introduced by Buchbinder et al. [3] to solve the distributed secretary problem. Feldman and Tennenholtz [6] applied this technique on the case when the queues have identical lengths and we further extend it to uneven lengths.

We first assume that the algorithms we consider do not hire one candidate if she is not the best so far in its queue at every time stamp. This assumption is without loss of generality since, otherwise, the probability of hiring the globally best candidate is 0. Given an algorithm, we use $f_{i,j}$ to denote the probability that candidate $j$ in queue $i = 1, 2$ is selected, given that it is the best overall. Then we have the following linear program LP shown atop the next page.

Lemma 1. *The optimal value of* LP *is an upper bound on the success probability of any algorithm.*

Proof. Consider an arbitrary algorithm $\pi$. Let $f_{i,j}$ be the probability that $\pi$ hires the candidate at position $j$ in queue $i$, given it is the best overall. Since the best candidate appears in each position $(i, j)$, $i = 1, 2$ and $j = 1, \ldots, n_i$, with probability $\frac{1}{n}$, the probability that $\pi$ can hire the best candidate at $(i, j)$ is $\frac{1}{n} f_{i,j}$. Thus the total probability of hiring the best candidate is $\frac{1}{n} \sum_{i=1,2} \sum_{j=1}^{n_i} f_{i,j}$, which is exactly the objective of LP.

To prove the lemma, it remains to prove that $\pi$ satisfies all the conditions of LP. We first consider candidates $(1, j)$ for $j = 1, \ldots, n_1$. Given candidate $(1, j)$ is the best overall, The probability that $\pi$ hires her should not be greater than the probability that $\pi$ rejects all candidates in previous time stamps $l = 1, \ldots, j-1$ for both two queues $Q_i$, $i = 1, 2$.

$$
\begin{aligned}
\text{(LP)} \quad \max \quad & \frac{1}{n}\sum_{i=1}^{2}\sum_{j=1}^{n_i} f_{i,j} \\
\text{s.t.} \quad & f_{1,j} + \sum_{i=1}^{2}\sum_{l=1}^{j-1}\frac{f_{i,l}}{l} \le 1, && \forall\, 1 \le j \le n_1 \\
& f_{2,j} + \sum_{i=1}^{2}\sum_{l=1}^{j-1}\frac{f_{i,l}}{l} + \frac{f_{1,j}}{j} \le 1, && \forall\, 1 \le j \le n_1 \\
& f_{2,j} + \sum_{i=1}^{2}\sum_{l=1}^{n_1}\frac{f_{i,l}}{l} + \sum_{l=n_1+1}^{j-1}\frac{f_{2,j}}{j} \le 1, && \forall\, n_1 < j \le n_2 \\
& f_{1,j} \ge 0, && \forall\, 1 \le j \le n_1 \\
& f_{2,j} \ge 0, && \forall\, 2 \le j \le n_2
\end{aligned}
$$

One crucial observation is that, the algorithm's behavior depends only on the relative ranks of previously interviewed candidates. Thus, at position $(i, j)$ the algorithm acts the same if candidate $(i, j)$ is the best overall, or is the best among the first $j$ candidates in $Q_i$. As we fixed candidate $(1, j)$ being the best overall, the probability that $\pi$ hires $(i, l)$ at some previous time stamp equals

$$
\begin{aligned}
& \Pr[\pi \text{ hires } (i,l) \mid (1,j) \text{ is the best overall}] \\
=& \Pr[\pi \text{ hires } (i,l)] \\
=& \Pr[\pi \text{ hires } (i,l) \mid (i,l) \text{ is the best so far in } Q_i] \\
& \cdot \Pr[(i,l) \text{ is the best so far in } Q_i] \\
=& \Pr[\pi \text{ hires } (i,l) \mid (i,l) \text{ is the best overall}] \\
& \cdot \Pr[(i,l) \text{ is the best so far in } Q_i] \\
=& f_{i,l} \cdot \frac{1}{l},
\end{aligned}
$$

which gives the first constraint $f_{1,j} \le 1 - \sum_{i=1}^{2}\sum_{l=1}^{j-1}\frac{f_{i,l}}{l}$ for $Q_1$.

Since $Q_1$ makes a decision before $Q_2$ within each time stamp, for the candidate at $(2, j)$ we also need to consider the probability that $\pi$ does not hire at time stamps prior to $(2, j)$. Thus, for $(2, j)$, $j = 1, \ldots, n_1$, we have $f_{2,j} \le 1 - \sum_{i=1}^{2}\sum_{l=1}^{j-1}\frac{f_{i,l}}{l} - \frac{f_{1,j}}{j}$, which gives the second constraint. Similarly, when $j > n_1$, only $Q_2$ remains in the algorithm, and we have the third constraint, which completes the proof of the lemma. □

To bound the optimal value of LP, denoted by $OPT_{\mathsf{LP}}$, we consider its dual DP shown atop this page, and denote the optimal value of DP by $OPT_{\mathsf{DP}}$. Note that $OPT_{\mathsf{LP}} = OPT_{\mathsf{DP}}$.

We use the indices $(i, j)$, $i = 1, 2$ and $j = 1, \ldots, n_i$, of the first variable to refer to each constraint in DP. Further, for simplicity, let $H_k = 1 + \frac{1}{2} + \cdots + \frac{1}{k}$ and $G_k = 1 + \frac{1}{2^2} + \cdots + \frac{1}{k^2}$ for any integer $k \ge 1$. Let $H_0 = G_0 = 0$.

Lemma 2. *If $n_1 - 1 < \frac{n_2 - 1}{e}$, then $OPT_{\mathsf{DP}} \le \frac{n_2}{n \cdot e}$.*

Proof. To prove the lemma, it suffices to construct a feasible solution whose objective value is $\frac{n_2}{n \cdot e}$.

Denote by $T$ the integer that satisfies

$$
\sum_{i=T+1}^{n_2-1}\frac{1}{i} < 1 \le \sum_{i=T}^{n_2-1}\frac{1}{i}.
$$

$$
\begin{aligned}
\text{(DP)} \quad \min \quad & \sum_{j=1}^{n_1} x_{1,j} + \sum_{j=1}^{n_2} x_{2,j} \\
\text{s.t.} \quad & j \cdot x_{1,j} + \sum_{l=j+1}^{n_1} x_{1,l} + \sum_{l=j}^{n_2} x_{2,l} \ge \frac{j}{n}, && \forall\, 1 \le j \le n_1 \\
& j \cdot x_{2,j} + \sum_{l=j+1}^{n_1} x_{1,l} + \sum_{l=j+1}^{n_2} x_{2,l} \ge \frac{j}{n}, && \forall\, 1 \le j \le n_1 \\
& j \cdot x_{2,j} + \sum_{l=j+1}^{n_2} x_{2,l} \ge \frac{j}{n}, && \forall\, n_1 < j \le n_2 \\
& x_{1,j} \ge 0, && \forall\, 1 \le j \le n_1 \\
& x_{2,j} \ge 0, && \forall\, 1 \le j \le n_2
\end{aligned}
$$

Since $\ln \frac{n_2 - 1}{T+1} < 1$, we have $T > \frac{n_2}{e} - 1 > (n_1 - 1) - 1$ so that $T \ge n_1 - 1$. Consider the following solution:

$$
x_{1,j} = 0, \text{ for all } j = 1, \ldots, n_1,
$$

and

$$
x_{2,j} = \begin{cases} 0, & \text{for } 1 \le j \le T \\ \frac{1}{n}(1 - H_{n_2-1} + H_{j-1}), & \text{for } T < j \le n_2. \end{cases}
$$

The non-negativity of $x_{i,j}$ directly follows from the definition of $T$. We prove that the solution is feasible via backward induction. First observe that for all $T + 1 \le j \le n_2 - 1$, $x_{2,j}$ decreases as $j$ increases, and in particular,

$$
x_{2,j} = x_{2,j+1} - \frac{1}{nj}.
$$

The constraint $(2, n_2)$ holds since $x_{2,n_2} = \frac{1}{n}$. In fact, this constraint is tight where equality holds. Given that constraint $(2, j)$ holds with equalities for $T + 1 < j \le n_2$, for constraint $(2, j - 1)$, we have

$$
\begin{aligned}
(j-1) \cdot x_{2,j-1} + \sum_{l=j}^{n_2} x_{2,l} &= (j-1)\left(x_{2,j} - \frac{1}{n(j-1)}\right) + \sum_{l=j}^{n_2} x_{2,l} \\
&= \left(j \cdot x_{2,j} + \sum_{l=j+1}^{n_2} x_{2,l}\right) - \frac{1}{n} \\
&= \frac{j}{n} - \frac{1}{n} = \frac{j-1}{n},
\end{aligned}
$$

where the second last equality holds because of the induction step for constraint $(2, j)$. For the remaining constraints $j \le T$, we note that the value of the left-hand side of the inequality increases but the value of the right-hand side decreases as $j$ decreases, we directly have the correctness of these inequalities.

Next, we verify the objective value of this solution.

$$\begin{aligned}obj &= \sum_{j=1}^{n_1} x_{1,j} + \sum_{j=1}^{n_2} x_{2,j} = \sum_{j=T+1}^{n_2} x_{2,j} \\ &= -T \cdot x_{2,T+1} + (T+1) \cdot x_{2,T+1} + \sum_{j=T+2}^{n_2} x_{2,j} \\ &= -\frac{T}{n}(1 - H_{n_2-1} + H_T) + \frac{T+1}{n} \\ &= \frac{T}{n}(H_{n_2-1} - H_{T-1}) \\ &\approx \frac{T}{n} \ln \frac{n_2 - 1}{T-1} \approx \frac{n_2}{n \cdot e},\end{aligned}$$

where the fourth equality holds because constraint $(2, T+1)$ holds with equality, i.e.,

$$(T+1) \cdot x_{2,T+1} + \sum_{j=T+2}^{n_2} x_{2,j} = \frac{T+1}{n}.$$

The last two approximations hold when $n$ goes large. □

Lemma 3. *If* $n_1 - 1 \geq \frac{n_2-1}{e}$,

$$OPT_{\text{DP}} \leq \frac{n_1}{n \cdot \left(2 - H_{n_2-1} + H_{n_1-1}\right)} + o(\frac{1}{n}).$$

Proof. Similar to Lemma 2, it suffices for us to construct a feasible solution with the desired objective value. Denote by $T$ the integer that satisfies the following condition,

$$\sum_{i=T+1}^{n_1-1} \frac{1}{i^2} < \frac{1 - H_{n_2-1} + H_{n_1-1}}{n_1} \leq \sum_{i=T}^{n_1-1} \frac{1}{i^2}.$$

We try to bound the value of $T$. Since $\frac{1}{i(i+1)} < \frac{1}{i^2} < \frac{1}{(i-1)i}$, we have

$$\frac{1}{T+1} - \frac{1}{n_1} < \frac{1 - H_{n_2-1} + H_{n_1-1}}{n_1} < \frac{1}{T-1} - \frac{1}{n_1 - 1} < \frac{1}{T-1} - \frac{1}{n_1}.$$

$$\frac{n_1}{2 - H_{n_2-1} + H_{n_1-1}} - 1 < T < \frac{n_1}{2 - H_{n_2-1} + H_{n_1-1}} + 1.$$

Note that when $n_1 - 1 \geq \frac{n_2-1}{e}$, $0 \leq H_{n_2-1} - H_{n_1-1} \leq 1$, and thus $T \leq n_1 \leq n_2$. Consider the following solution:

- For $i = 1, 2$ and $1 \leq j \leq T$, $x_{i,j} = 0$.
- For $i = 1$ and $T+1 \leq j \leq n_1$,

$$x_{i,j} = \frac{1}{n}\frac{j-1}{n_1}(1 - H_{n_2-1} + H_{n_1-1}) - \frac{j-1}{n}(G_{n_1-1} - G_{j-1}).$$

- For $i = 2$ and $T+1 \leq j \leq n_1$,

$$x_{i,j} = \frac{1}{n}\frac{j}{n_1}(1 - H_{n_2-1} + H_{n_1-1}) - \frac{j}{n}(G_{n_1-1} - G_{j-1})$$

- For $i = 2$ and $n_1 < j \leq n_2$,

$$x_{i,j} = \frac{1}{n}(1 - H_{n_2-1} + H_{j-1}).$$

We first prove the feasibility of the solution. For $i = 2$ and $n_1 \leq j \leq n_2$, constraints $(i, j)$ hold with equality since the values of $x_{i,j}$ (note that this includes $j = n_1$) are the same as the solution constructed in the proof of Lemma 2.

For $T+1 \leq j \leq n_1$, by the design of the solution, we have

$$x_{1,j} = x_{2,j} \cdot \frac{j-1}{j} \text{ and } x_{2,j-1} = x_{1,j} - \frac{1}{n(j-1)}.$$

We continue to prove by backward induction and recall that the $(2, n_1)$ constraint holds with equality. Given constraint $(2, j)$ holds with equality for $t+1 \leq j \leq n_1$, for constraint $(1, j)$,

$$\begin{aligned}&j \cdot x_{1,j} + \sum_{l=j+1}^{n_1} x_{1,l} + \sum_{l=j}^{n_2} x_{2,l} \\ =&j \cdot x_{2,j} \cdot \frac{j-1}{j} + \sum_{l=j+1}^{n_1} x_{1,l} + \left(x_{2,j} + \sum_{l=j+1}^{n_2} x_{2,l}\right) \\ =&j \cdot x_{2,j} + \sum_{l=j+1}^{n_1} x_{1,l} + \sum_{l=j+1}^{n_2} x_{2,l} = \frac{j}{n},\end{aligned}$$

where the last equality holds because of the induction on constraint $(2, j)$. Next, given constraint $(1, j)$ holds for $t+1 < j \leq n_1$, for constraint $(2, j-1)$,

$$\begin{aligned}&(j-1) \cdot x_{2,j-1} + \sum_{l=j}^{n_1} x_{1,l} + \sum_{l=j}^{n_2} x_{2,l} \\ =&(j-1)\left(x_{1,j} - \frac{1}{n(j-1)}\right) + \left(x_{1,j} + \sum_{l=j+1}^{n_1} x_{1,l}\right) + \sum_{l=j}^{n_2} x_{2,l} \\ =&j \cdot x_{1,j} + \sum_{l=j+1}^{n_1} x_{1,l} + \sum_{l=j}^{n_2} x_{2,l} - \frac{1}{n} = \frac{j}{n} - \frac{1}{n} = \frac{j-1}{n},\end{aligned}$$

where the last equality is by the induction on constraint $(1, j)$.

The remaining constraints for $j \leq T$ are satisfied via a similar argument in the proof of Lemma 2.

Secondly, to see the non-negativity of the solution, we only need to show the non-negativity of $x_{1,T+1}$ due to the monotonicity of $x$,

$$\begin{aligned}x_{1,T+1} &= \frac{1}{n}\frac{T}{n_1}(1 - H_{n_2-1} + H_{n_1-1}) - \frac{T}{n}(G_{n_1-1} - G_T) \\ &= \frac{T}{n}\left(\frac{1 - H_{n_2-1} + H_{n_1-1}}{n_1} - \sum_{i=T+1}^{n_1-1} \frac{1}{i^2}\right) \\ &> 0,\end{aligned}$$

where the inequality follows from the definition of $T$.

We finally upper-bound the objective value of the above solution,

$$\begin{aligned}obj &= \sum_{j=1}^{n_1} x_{1,j} + \sum_{j=1}^{n_2} x_{2,j} = \sum_{j=T+1}^{n_1} x_{1,j} + \sum_{j=T+1}^{n_2} x_{2,j} \\ &= \left((T+1) \cdot x_{1,T+1} + \sum_{j=T+2}^{n_1} x_{1,j} + \sum_{j=T+1}^{n_2} x_{2,j}\right) - T \cdot x_{1,T+1} \\ &= \frac{T+1}{n} - T \cdot x_{1,T+1} \leq \frac{T+1}{n} \\ &= \frac{n_1}{n \cdot \left(2 - H_{n_2-1} + H_{n_1-1}\right)} + o(\frac{1}{n}),\end{aligned}$$

where the second last inequality holds because the $(1, T+1)$ constraint holds with equality and $x_{1,T+1} > 0$, and the last equality follows from the bound of $T$. □

## 4 The Optimal Algorithm for Two Queues

We now design the optimal algorithm for two uneven queues.

**Algorithm 1:** The optimal algorithm for two queues

**Input:** $n_1$ and $n_2$, where $n_1 \le n_2$ and $n_1 + n_2 = n$.

**if** $n_1 - 1 < \frac{n_2-1}{e}$ **then**
- Reject all candidates in $Q_1$.
- Reject the first $\lfloor \frac{n_2}{e} \rfloor$ candidates in $Q_2$. For the remaining candidates in $Q_2$, hire the first one who is better than all previous candidates in $Q_2$.

**else**
- Reject the first $\lfloor \frac{n_1}{2-H_{n_2}+H_{n_1}} \rfloor$ candidates in $Q_1$ and $Q_2$.
- For the remaining candidates in $Q_1$ and $Q_2$, hire the first one who is better than all previous candidates in its queue. If two candidates are hired at the same time, hire the one in $Q_1$.

Theorem 4. *Algorithm 1 is optimal for any two-queue instance.*

The proof of Theorem 4 relies on the following Lemmas 5 and 6.

Lemma 5. *If $n_1 - 1 < \frac{n_2-1}{e}$, Algorithm 1 has success probability*

$$\frac{n_2}{n \cdot e} > \frac{1}{e+1}.$$

Proof. If we always reject all candidates in $Q_1$, the success probability is

$$\Pr[\text{the best in } Q_2] \cdot \Pr[\text{select the best in } Q_2 \mid \text{the best in } Q_2].$$

It is clear that $\Pr[\text{the best in } Q_2] = \frac{n_2}{n}$, and $\Pr[\text{select the best in } Q_2 \mid \text{the best in } Q_2] = \frac{1}{e}$ as this degenerates to the classic single-queue problem, which proves the lemma. □

Lemma 6. *If $n_1 - 1 \ge \frac{n_2-1}{e}$, Algorithm 1 has success probability*

$$\frac{n_1}{n(2 - H_{n_2-1} + H_{n_1-1})} \ge \frac{1}{4}.$$

Proof. When $n_1 - 1 \ge \frac{n_2-1}{e}$, Algorithm 1 sets threshold $T \le n_1$. We use the probability method to calculate the success probability. We claim that our algorithm hires the best candidate at position $i > T$ in queue $Q_j$, $j = 1, 2$, if the following three conditions hold simultaneously:

- The best candidate appears at position $i$ in $Q_j$, which happens with probability $\frac{1}{n}$;
- The best candidate in $Q_j[1, \ldots, i-1]$ appears in the first $T$ positions, which happens with probability $\frac{T}{i-1}$;
- The best candidate in $Q_{3-j}[1, \ldots, i]$ appears in the first $T$ positions, which happens with probability $\frac{T}{i}$.

Our analysis here ignores the ties, which happens with probability $o(1)$. Summing over all $i = T+1, \ldots, n_2$, we have the success probability of Algorithm 1 being

$$\begin{aligned}
&2\sum_{i=T+1}^{n_1} \frac{1}{n} \cdot \frac{T}{i-1} \cdot \frac{T}{i} + \sum_{i=n_1+1}^{n_2} \frac{1}{n} \cdot \frac{T}{n_1} \cdot \frac{T}{i-1} \\
=&\frac{2T^2}{n}\left(\frac{1}{T} - \frac{1}{n_1}\right) + \frac{T^2}{n \cdot n_1}\left(\sum_{i=n_1+1}^{n_2} \frac{1}{i-1}\right) \\
=&\frac{2T}{n} - \frac{T^2}{n \cdot n_1}(2 - H_{n_2-1} + H_{n_1-1}) \\
>&\frac{T}{n} - o\left(\frac{1}{n}\right) = \frac{n_1}{n \cdot (2 - H_{n_2} + H_{n_1})} - o\left(\frac{1}{n}\right).
\end{aligned}$$

When $n$ goes to infinity, the lower term goes to 0, completing the proof of the lemma. □

Note that the worst-case probability when $n_1 - 1 \ge \frac{n_2-1}{e}$ happens when $n_1 = n_2$, and $\frac{n_1}{n(2-H_{n_2}+H_{n_1})} = \frac{1}{4}$. This result matches the results from [6] and indicates that when the queues are more balanced, the chance of success gets lower.

Combining the above lemmas and Lemmas 1, 2 and 3, our algorithm has success probability as large as $OPT_{\mathsf{LP}}$ and thus is optimal when $n$ gets sufficiently large.

## 5 Conclusion

In this article, we initiate the study of the distributed secretary problem with multiple uneven queues. Our model is well justified by real-world problems. For the case of two queues, we show how to use linear programming techniques to design the optimal algorithm. Our paper uncovers many interesting research problems. For example, it is intriguing to extend our model to more than two queues and the selection of multiple candidates.

## Acknowledgments

This work is funded by the Hong Kong SAR Research Grants Council (No. PolyU 15224823) and the Guangdong Basic and Applied Basic Research Foundation (No. 2024A1515011524).

## References

[1] Moshe Babaioff, Nicole Immorlica, David Kempe, and Robert Kleinberg. 2007. A Knapsack Secretary Problem with Applications. In *APPROX-RANDOM (Lecture Notes in Computer Science, Vol. 4627)*. Springer, 16–28.
[2] Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. 2007. Matroids, secretary problems, and online mechanisms. In *SODA*. SIAM, 434–443.
[3] Niv Buchbinder, Kamal Jain, and Mohit Singh. 2014. Secretary Problems via Linear Programming. *Math. Oper. Res.* 39, 1 (2014), 190–206.
[4] José Correa, Andrés Cristi, Paul Duetting, and Ashkan Norouzi-Fard. 2021. Fairness and Bias in Online Selection. In *ICML (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, 2112–2121.
[5] Evgenii Borisovich Dynkin. 1963. The Optimum Choice of the Instant for Stopping a Markov Process. *Scientific American* 4 (1963), 627–629.
[6] Moran Feldman and Moshe Tennenholtz. 2012. Interviewing secretaries in parallel. In *EC*. ACM, 550–567.
[7] Martin Gardner. 1960. Mathematical games. *Scientific American* 202, 5 (1960), 174–188.
[8] Denis V Lindley. 1961. Dynamic programming and decision theory. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 10, 1 (1961), 39–51.
[9] Xiaoming Sun, Jia Zhang, and Jialin Zhang. 2014. Solving Multi-choice Secretary Problem in Parallel: An Optimal Observation-Selection Protocol. In *ISAAC (Lecture Notes in Computer Science, Vol. 8889)*. Springer, 661–673.