Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра математики факультету інформатики

ЗАДАЧА СЕКРЕТАРЯ

Курсова робота

Керівник курсової роботи

Доцент, кандидат ф.-м.н

Щестюк Наталія Юріївна

_____

(підпис)

«____» _____ 2023 р.

Виконала студентка

3-го року навчання спеціальності

113 «Прикладна математика»

Дацько Юлія Віталіївна

Київ 2023

ЗМІСТ

ВСТУП

Оптимальність прийняття вибору є надзвичайно важливим завданням в сучасному світі для рішень не тільки бізнесу, проте і особистих. Задача секретаря допомагає у прийнятті найкращого рішення з послідовних альтернатив, що є актуальною та цікавою темою для дослідження у галузях: теорії ймовірностей, статистиці та теорії прийняття рішень [1]. Описуючи прикладне значення задачі, неможливо оминути теми вибору найкращого кандидата на роботу в бізнесі або найбільш вигідних інвестиційних проєктів у сфері фінансів. Її застосування можливе також і у повсякденному житті при виборі товару, житла, плануванні подорожей і т.д.

Перша публікація про задачу секретаря з'явилась у лютому 1960 року у Scientific American та опублікована Мартіном Гарднером. Вирішення проблеми було оприлюднено Лео Мозером та Паудером у цьому ж журналі в березні 1960. Питання походження цієї задачі залишається спірним питанням, оскільки перші усні згадки про неї беруть початок з 1949 року [2]. Проте навіть з того часу до сьогодення продовжуються дослідження пошуку оптимальної зупинки у варіаціях класичної постановки задачі секретаря.

Мета даної курсової роботи – аналіз задачі секретаря та алгоритму оптимальної зупинки, застосування на різних наборах даних з подальшим визначенням ефективності. Отже, завдання курсової роботи складається з:

1. Розглянути постановку задачі секретаря
2. Проаналізувати стратегії вибору найкращого кандидата
3. Визначити алгоритм оптимальної зупинки
4. Розглянути алгоритм оптимальної зупинки для даних з різних випадкових розподілів
5. Переконатись або спростувати ефективність застосування алгоритму оптимальної зупинки для описаних випадкових розподілів

Основна частина курсової роботи складається з двох розділів.

У першому розділі розглянута постановка задачі секретаря та її можливі стратегії вирішення. Також подано алгоритм оптимальної зупинки для даної проблеми.

У другому розділі розглядаються поняття випадкових розподілів, їх вплив на ймовірність обрання найкращого кандидата та аналіз залежно від параметрів розподілів.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ РОЗВ'ЯЗАННЯ ЗАДАЧІ СЕКРЕТАРЯ

1.1 Постановка задачі секретаря

Однією з класичних задач теорії оптимальної зупинки є задача секретаря, яка також відома як «задача про перебірливу наречену». Її суть полягає у послідовному розгляді кандидатів без можливості повернення до попередніх. Метою задачі секретаря є максимізація ймовірності вибору найкращого можливого варіанту.

Є безліч варіацій задачі секретаря, проте розглянемо класичну версію, яка описується за такими правилами:

- Кількість n кандидатів на посаду зарання відома
- Необхідно взяти одну людину на посаду
- Претенденти надходять послідовно один за одним
- Можливо визначити відносний порядок кандидата відносно інших
- Після співбесіди робиться вибір, прийняти особу на посаду чи відмовити
- Рішення неможливо змінити [3]

Варто зауважити, що при відхиленні $n - 1$ кандидатів, необхідно прийняти останнього. Також цілком зрозуміло, що існує певна ймовірність, що найкращий кандидат буде під номером 1 та, внаслідок наявності обмеженої інформації, буде відхилений у користь «потенційно кращих» наступних кандидатів.

В даній курсовій роботі буде розглядатись класична постановка задачі секретаря, обґрунтування вибору алгоритму оптимальної зупинки та ефективність його застосування на різних наборах даних.

1.2 Стратегії вибору найкращого кандидата

Для дослідження оптимального розв'язку задачі секретаря варто розглянути усі можливі стратегії вибору найкращого кандидата та покращити їх:

1. Вибір першого кандидата

   Дана стратегія є дещо тривіальною, саме тому і підходить для початку аналізу проблеми. З точки зору оптимальності ймовірність того, що перший кандидат виявиться найкращим рівна $\frac{1}{n}$. При збільшенні кількості претендентів виявиться майже нереальним обрати найкращого, оскільки $\lim_{n\to\infty} \frac{1}{n} \to 0$.

   Перевіримо даний результат використовуючи симуляцію Монте-Карло:

Рис. 1.1 – Симуляція для вибору першого кандидата

2. Відхилення першого кандидата

   Перевагою даної стратегії є те, що відхиляючи першого кандидата, ми запам'ятовуємо його результат і надалі кожного наступного кандидата

порівнюємо з першим. Таким чином, обираючи наступного кандидата, що є кращим за нього, ми дещо підвищуємо загальну ймовірність обрання найліпшого претендента. Ймовірність, що перший кандидат був найкращим рівна $\frac{1}{n}$, що при збільшенні n не буде підвищувати ймовірність помилкового вибору.

Перевіримо даний результат використовуючи симуляцію Монте-Карло:

Рис. 1.2 – Симуляція для відхилення першого кандидата

Спостерігаємо незначне покращення результатів, порівнюючи з попередньою стратегією.

Отже, задача секретаря полягає у знаходженні оптимального значення k – кількості кандидатів, яких варто відхилити перед прийняттям рішення.

1.3 Аналіз стратегії відхилення половини кандидатів

Дослідимо прийняття рішення про відхилення половини кандидатів та, відповідно, порівняння другої частини вибірки з найкращим кандидатом з першої.

Нехай T – вибір найкращого кандидата, а T_i – вибір i-го кандидата, який є найкращим. Ймовірності T_i є незалежними одне від одного, таким чином $P\{T\} = \sum_{i=1}^{n} T_i$. Беручи до уваги раніше зазначену умову про відхилення половини претендентів, ймовірність їх обрання буде рівна $P\{T_i\} = 0$ при $i = \overline{1, \frac{n}{2}}$. Таким чином, $P\{T\} = \sum_{i=\frac{n}{2}+1}^{n} P\{T_i\}$.

Для того, щоб обрати T_i найкращого кандидата необхідно виконання двох умов:
а) кандидат знаходиться на i-ому кроці

б) кандидат кращий за попередніх неприйнятих кандидатів з $\frac{n}{2} + 1$ кроку до i-го.

Для отримання ймовірностей $P\{T_i\}$ запишемо першу умову як A_i, а другу – B_i. Варто зауважити їх незалежність: подія A_i залежить від того, чи є кандидат на i-ому місці кращим за інших, в той час як подія B_i залежить від порядку кандидатів від 1 до $i - 1$ порядкового номеру. Отримуємо:

$$P\{T_i\} = P\{A_i \cap B_i) = P\{A_i\} \times P\{B_i\}$$

де $P\{A_i\} = \frac{1}{n}$ – ймовірність обрати кожного кандидата рівноможлива

$P\{B_i\} = \frac{\frac{n}{2}}{i-1}$ – ймовірність обрати кандидата, що є кращим за найліпшого кандидата першої половини вибірки. Таким чином, $P\{B_i\}$ буде рівноможливим для кандидатів від порядку $\frac{n}{2} + 1$ до $i - 1$.

Розпишемо ймовірність вибору найкращого претендента:

$$P\{T\} = \sum_{i=\frac{n}{2}+1}^{n} P\{T_i\} = \sum_{i=\frac{n}{2}+1}^{n} P\{A_i\} \times P\{B_i\} =$$

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

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

Перевіримо, яких значень набуває $P\{T\}$ при $n \to \infty$:

$$\lim_{n\to\infty} \frac{1}{2} \sum_{i=\frac{n}{2}}^{n-1} \frac{1}{i-1} = \frac{\log(2)}{2} \approx 0{,}35$$

Перевіримо даний результат використовуючи симуляцію Монте-Карло:

Рис. 1.3 – Симуляція для відхилення половини кандидатів

Надалі ми розглянемо алгоритм оптимальної зупинки та порівняємо його результати з розв'язком для половини відхилених кандидатів, на основі чого зможемо зробити висновок про оптимальний поріг для початку прийняття рішення щодо кандидатів.

1.4 Алгоритм оптимальної зупинки в задачі секретаря

Вище розглянуті стратегії мають певний поріг в k претендентів, після яких ми приймаємо першого, що є кращим за переглянутих. Проте за малого порогового значення ймовірність обрати найкращого кандидата низька:

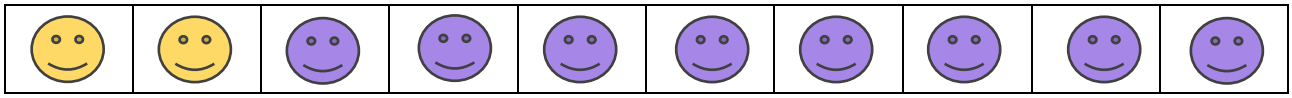

Рис. 1.4 – Приклад низького порогового значення для задачі секретаря

За великого k – ймовірність пропустити найкращого кандидата збільшується:

Рис. 1.5 – Приклад високого порогового значення для задачі секретаря

Таким чином приходимо до алгоритму оптимальної зупинки – його проблема полягає у знаходженні такого значення k, за якого ймовірність знайти найкращого кандидата максимально можлива.

Користуючись описаною у попередньому підрозділі логікою, варто замінити значення $\frac{n}{2}$ на k:

$$P\{T\} = \sum_{i=k+1}^{n} P\{T_i\} = \sum_{i=k+1}^{n} P\{A_i\} \times P\{B_i\} =$$

$$= \sum_{i=k+1}^{n} \frac{1}{n} \times \frac{k}{i-1} = \frac{k}{n} \sum_{i=k+1}^{n} \frac{1}{i-1} = \frac{k}{n} \sum_{i=k}^{n-1} \frac{1}{i-1} \quad (*)$$

Апроксимувавши суму, отримуємо:

$$\frac{k}{n}\sum_{i=k}^{n-1}\frac{1}{i-1}=\frac{k}{n}(\log(n)-\log(k))$$

Для знаходження максимального значення виразу диференціюємо по k та прирівнюємо до нуля:

$$\frac{d}{dk}\left(\frac{k}{n}(\log(n)-\log(k))\right)=\frac{1}{n}\frac{d}{dk}(\mathrm{k}\ \log(n)-\mathrm{k}\ \log(k))$$

$$=\frac{1}{n}(\log(n)-\log(k)-1)=0$$

$$\log(k)=\log(n)-1\Rightarrow k=\frac{n}{e}$$

Підставимо отримане значення в (*) та перевіримо, яких значень набуває $P\{T\}$ при $n\to\infty$:

$$\lim_{n\to\infty}\frac{\frac{n}{e}}{n}\sum_{i=\frac{n}{e}}^{n-1}\frac{1}{i-1}=\lim_{n\to\infty}\frac{1}{e}\sum_{i=\frac{n}{e}}^{n-1}\frac{1}{i-1}=\frac{1}{e}\approx 0{,}37$$

Перевіримо даний результат використовуючи симуляцію Монте-Карло:

Рис. 1.5 – Симуляція для відхилення $\frac{n}{e}$ кандидатів

Порівнюючи графіки алгоритму оптимальної зупинки та попередніх стратегій, бачимо значне покращення – при $n \to \infty$ ймовірність обрати найкращого кандидата приблизно рівна $0{,}37$.

Досягнути незначного покращення отриманого результату можливо при $k = \frac{\left(n-\frac{1}{2}\right)}{e} + \frac{1}{2}$ як зазначено у праці [4].

На прикладі таблиці [5] змоделюємо ситуацію з відхиленими $\frac{n}{e}$ кандидатами та подаймо результати у вигляді таблиці для порівняння результатів:

Кількість кандидатів	Кількість пропущених кандидатів	Ймовірність обрання найкращого кандидата
3	1 (33.33%)	49.9%
4	1 (25.0%)	45.9%
5	2 (40.0%)	43.2%
6	2 (33.33%)	42.8%
7	3 (42.86%)	40.9%
8	3 (37.5%)	40.9%
9	3 (33.33%)	40.8%
10	4 (40.0%)	39.9%
20	7 (35.0%)	38.4%
30	11 (36.67%)	37.8%
50	18 (36.0%)	37.3%
100	37 (37.0%)	37.1%
1000	368 (36.8%)	36.8%

Рис. 1.6 – Таблиця ймовірностей обрання найкращого кандидата при n

Як бачимо, при збільшенні кількості кандидатів n ймовірність прямує до $\frac{1}{e}$.

Розглянутий випадок задачі секретаря описаний для рівномірного дискретного розподілу, проте цікаво також розглянути його ефективність для інших розподілів.

У наступному розділі буде проведений аналіз алгоритму оптимальної зупинки для неперервного рівномірного, нормального, експоненційного розподілу та розподілу Парето.

РОЗДІЛ 2. СИМУЛЯЦІЯ ЗАДАЧІ СЕКРЕТАРЯ ПРИ РІЗНИХ РОЗПОДІЛАХ КАНДИДАТІВ

2.1 Математичні моделі досліджуваних випадкових розподілів

У реальному житті дані досить часто розподілені згідно різних розподілів, тому досліджуючи задачу секретаря варто дослідити ймовірні розподіли даних. У цьому розділі ми проаналізуємо ефективність алгоритму для деяких розподілів та як вона змінюється залежно від параметрів розподілу випадкової величини.

1. Рівномірний випадковий розподіл

   Даний розподіл зазвичай розглядається при моделюванні задачі секретаря через свою простоту. Параметрами є значення a та b $(a < b)$, в діапазоні яких з однаковою ймовірністю можна обрати кожне число:

$$f(x) = \frac{1}{b-a}, \text{де } a \leq x \leq b$$

Рис. 2.1 – Частота обрання кандидатів для рівномірного розподілу

2. Нормальний випадковий розподіл

Описаний розподіл достатньо поширений в реальному житті, навіть при перегляді кандидатів неможливо заперечувати їх можливу симетричність відносно середнього значення. Параметрами є значення μ – математичне сподівання та σ^2 – дисперсія випадкової величини:

$$f(x, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{(x-\mu)^2}{2\sigma^2}\}$$

Рис. 2.2 – Частота обрання кандидатів для нормального розподілу

3. Експоненційний випадковий розподіл

Даний розподіл теж може мати застосування у моделюванні проблеми секретаря через власну безперервність та незалежність подій одна від одної. Параметром є значення λ – інтенсивність подій за одиницю часу ($\lambda > 0$):

$$f(x) = \lambda \exp\{-\lambda x\}$$

Рис. 2.3 – Частота обрання кандидатів для експоненціального розподілу

4. Розподіл випадковий Парето

   Зазначений розподіл використовується для явищ з важким хвостом та може мати своє місце у задачі секретаря. Також принцип генерації розподілу описується як «правило 80-20», що у нашому випадку ми можемо описати як 80% результатів дають 20% кандидатів. Параметрами є значення α – форма ($\alpha > 0$) та x_m– параметр мінімального значення ($x_m > 0$):

$$f(x) = \frac{\alpha {x_m}^{\alpha}}{x^{\alpha+1}}$$

Рис. 2.4 – Частота обрання кандидатів для розподілу Парето

2.2 Порівняння ефективності алгоритму для різних випадкових розподілів за допомогою Python

У даному підрозділі проведемо аналіз ефективності алгоритму оптимальної зупинки для різних випадкових розподілів: рівномірного, нормального, експоненційного та розподілу Парето. Для цього скористаємось мовою програмування Python, його бібліотеками та початковим кодом [6]:

1. Рівномірний розподіл

   Для подальшого порівняння згенеруємо числа на проміжку від 1 до 101 використовуючи функцію numpy.random.uniform():

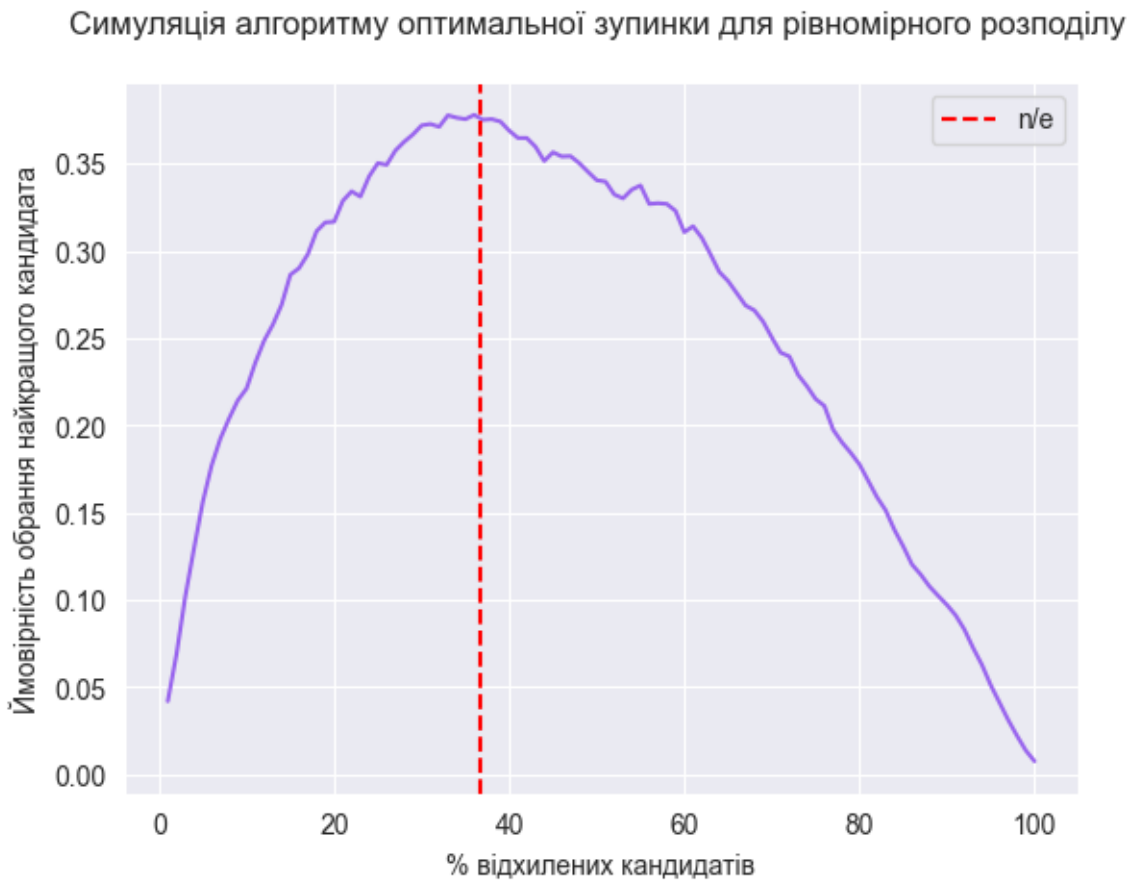

Рис. 2.5 – Оптимальний % відхилених кандидатів для рівномірного розподілу

При редагуванні значень параметрів рівномірного розподілу змін у оптимальному відсотку відхилених кандидатів не спостерігаємо.

2. Нормальний розподіл

   Для подальшого порівняння згенеруємо числа з параметрами $\mu = 0, \delta^2 = 1$ використовуючи функцію numpy.random.normal():

Рис. 2.6 – Оптимальний відсоток відхилених кандидатів для нормального розподілу

При редагуванні значень параметрів рівномірного розподілу змін у оптимальному відсотку відхилених кандидатів не спостерігаємо – алгоритм дає результати схожі до рівномірного розподілу та межа у $\frac{n}{e}$ підходить для вибору найкращого кандидата.

3. Експоненційний розподіл

   Для подальшого порівняння згенеруємо числа з параметром $\lambda = 1$ використовуючи функцію numpy.random.exponential():

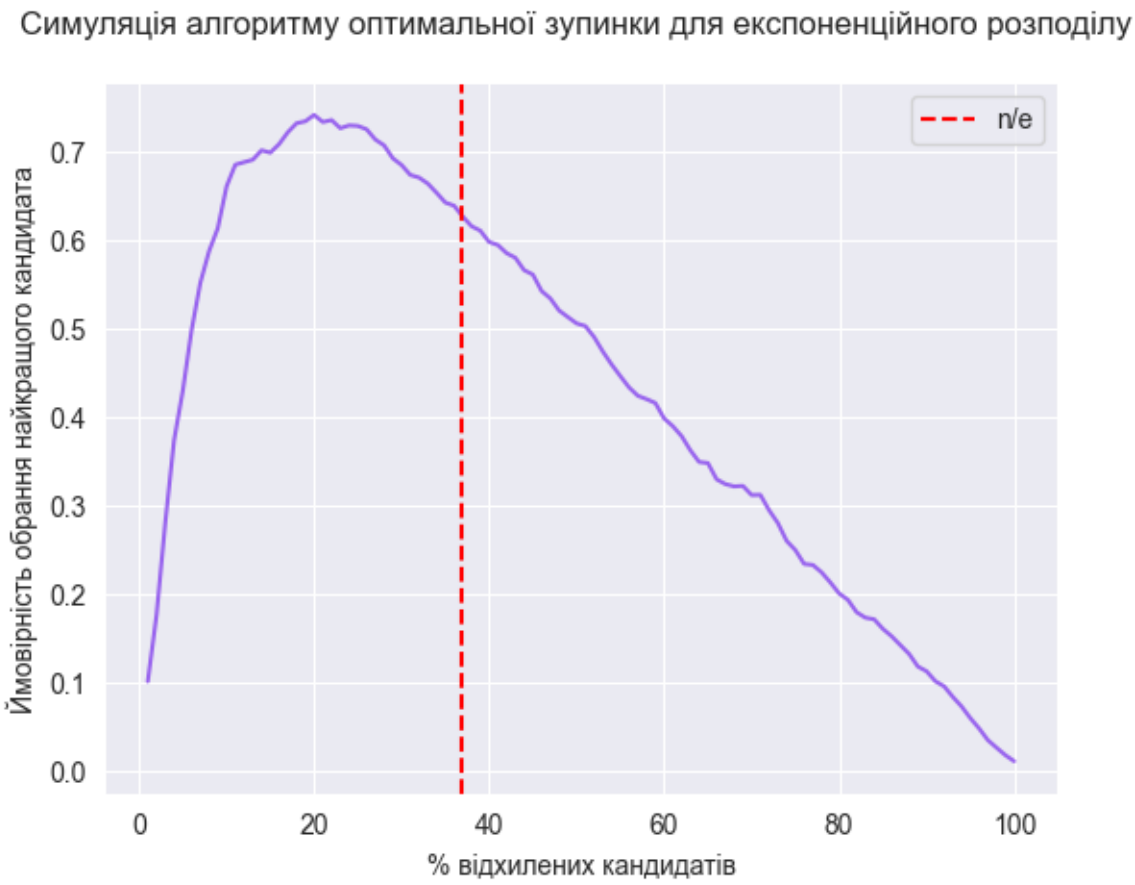

Рис. 2.7 – Оптимальний відсоток відхилених кандидатів для експоненційного розподілу

При редагуванні значень параметрів експоненційного розподілу змін у оптимальному відсотку відхилених кандидатів не спостерігаємо, проте наявна помітна різниця в порівнянні з попередніми розподілами. Оптимальний відсоток відхилених кандидатів близький до $\frac{n}{5}$, що потребує подальшого дослідження та розробки алгоритму для роботи з даними експоненційного розподілу.

4. Розподіл Парето

   Для подальшого порівняння згенеруємо числа з параметром $\alpha = 1$ використовуючи функцію numpy.random.pareto():

Рис. 2.8 – Оптимальний відсоток відхилених кандидатів для розподілу Парето

При редагуванні значень параметра розподілу Парето спостерігаємо зміни у оптимальному відсотку відхилених кандидатів. При подальшому дослідженні виявляємо, що при $\alpha < 1$ ймовірність обрання найкращого кандидата наближається до 1, а при $\alpha > 1$ – оптимальний відсоток відхилених кандидатів близький до $\frac{n}{5}$ (як у і експоненційного розподілу).

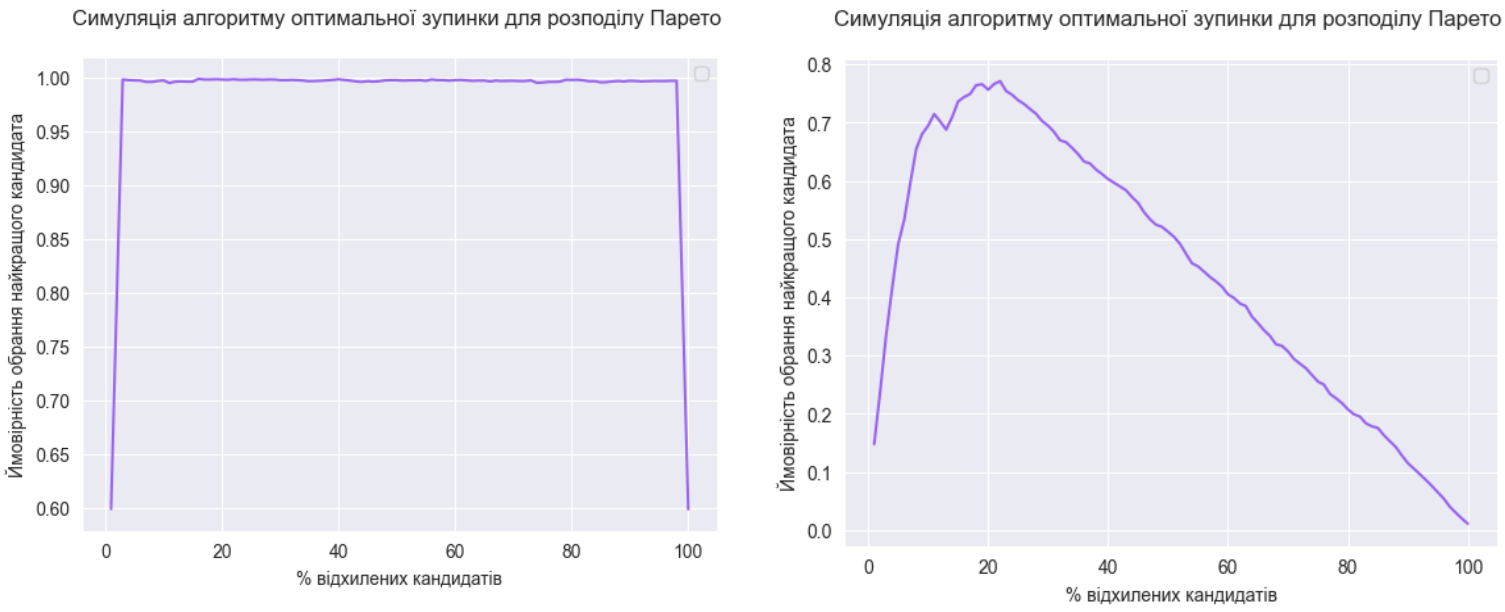

Рис. 2.9 – Оптимальний відсоток відхилених кандидатів для розподілу Парето при $\alpha = 0.1$ (зліва) та $\alpha = 100$ (справа)

Отримані результати потребують подальшого дослідження та розробки алгоритму для роботи з даними описаного розподілу.

У даному розділі ми перевірили результати виконання алгоритму оптимальної зупинки для даних розподілів: рівномірного, нормального, експоненціального та розподілу Парето. Також за нашими спостереженнями час виконання алгоритму для перших двох розподілів був приблизно однаковим, що пов’язано з їх симетричністю та рівномірним розподілом значень. Оскільки для інших розподілів ці умови не виконуються, то цей час дещо різниться, особливо для розподілу Парето через його важкий хвіст.

ВИСНОВКИ

У даній курсовій роботі було проведено аналіз алгоритму оптимальної зупинки для класичної постановки задачі секретаря для різних випадкових розподілів. Дослідження впливу виду розподілу на ефективність алгоритму може дати підстави для проведення додаткових експериментів.

В результаті виконання задач курсової роботи було отримано рішення задачі секретаря та підтверджено ефективність алгоритму оптимальної зупинки для рівномірного та нормального випадкових розподілів. Також було отримано результати для експоненційного та розподілу Парето, що підштовхують до додаткових досліджень у галузі пошуку алгоритму оптимальної зупинки для різного виду вхідних даних.

Було написано код мовою програмування Python для симуляції результатів різних видів випадкових розподілів та візуалізації отриманих підсумків.

Ця курсова робота підштовхує до додаткових досліджень у галузі оптимальної зупинки, спираючись на наявність умов невизначеності даних у реальному житті. Покращення аналізу для окремих випадкових видів розподілу призведе до отримання надійних методів прийняття ефективних рішень.

СПИСОК ДЖЕРЕЛ ІНФОРМАЦІЇ

1. Wikipedia The Free Encyclopedia – Secretary problem [Електронний ресурс] https://en.wikipedia.org/wiki/Secretary_problem
2. Ferguson T. S. Who solved The Secretary Problem. *Statistical Science*. 1989. Vol. 4, no. 3. https://projecteuclid.org/journals/statistical-science/volume-4/issue-3/Who-Solved-the-Secretary-Problem/10.1214/ss/1177012493.full
3. Plaasch S., Rubio M., Verbeelen Y. The Secretary Problem – University of Luxembourg, 2020 https://math.uni.lu/eml/assets/reports/PVR.pdf
4. Ajta M., Megido N., Waarts O. Improved algorithms and analysis for secretary problems and generalizations. 23–25 October 1995. http://theory.stanford.edu/~megiddo/pdf/secrfin.pdf
5. Chamberlain H. Book Review: ALGORITHMS TO LIVE BY: The Computer Science of Human Decisions [Електронний ресурс] https://comicsgrinder.com/2016/05/07/book-review-algorithms-to-live-by-the-computer-science-of-human-decisions/
6. Khan I. Solving the secretary problem with Python [Електронний ресурс] https://www.imrankhan.dev/pages/Solving%20the%20secretary%20problem%20with%20Python.html#:~:text=Solving%20the%20secretary%20problem%20with%20Python%20The%20secretary,and%20mathematical%20solution%20is%20outlined%20quite%20clearly%20here

Додаток А. Код мовою програмування Python

До курсової роботи додано окремий файл з розширенням .pdf, у якому наведено код та його результати виконання.