Bachelorarbeit

am

Institut für Informatik
Rheinische Friedrich-Wilhelms-Universität Bonn

Das Sekretärsproblem

Ramtin Azimi Garakani

Gutachter:
Prof. Dr. Heiko Röglin
Dr. Elmar Langetepe

29. Oktober 2015

Zusammenfassung

In dieser Bachelorarbeit befassen wir uns mit dem *Sekretärsproblem*. Hierbei gilt es, aus einer Menge von Bewerbern einen einzigen Bewerber für die Position des Sekretärs auszuwählen. Erst während des Vorstellungsgesprächs stellt sich die Eignung des Bewerbers für diese Position heraus. Man muss sich vor dem Erscheinen des nächsten Bewerbers für eine Zu- oder Absage des jetzigen Bewerbers entscheiden. Eine getroffene Entscheidung kann nicht mehr revidiert werden.

Wir beschäftigen uns mit zwei unterschiedlichen Varianten des Problems. Bei der ersten Variante, dem Best-Choice-Problem, wird das Ziel verfolgt, die Wahrscheinlichkeit, den besten Kandidaten zu erhalten, zu maximieren. Bei der zweiten Variante, dem Value-Problem, ist es das Ziel zu garantieren, dass die erwartete Eignung des Kandidaten höchstens um einen Faktor kleiner als die des besten Bewerbers ist.

Wir werden in dieser Arbeit den Zusammenhang zwischen diesen beiden Problemvarianten erläutern. Weiterhin stellen wir bekannte Algorithmen zur Lösung der zwei Problemvarianten vor und diskutieren, ob Algorithmen existieren können, die bessere Ergebnisse als die bisher bekannten Algorithmen erreichen.

Danksagung

Zunächst möchte ich mich an dieser Stelle bei all denjenigen bedanken, die mich während der Anfertigung dieser Bachelorarbeit unterstützt haben. Hierzu gehören unter anderem Lena Carta, Milad Navidizadeh und Fabian Thorand, die beim Korrekturlesen mit sehr guten Ratschlägen diese Arbeit verbessern konnten.

Ganz besonders gilt der Dank meinem Betreuer Herrn Prof. Dr. Röglin. Seine Motivation und seine Geduld waren bei unseren zahlreichen Treffen eine große Hilfe und haben den Weg zu dieser Arbeit geebnet. Ich danke auch Herrn Dr. Langetepe für seine Bereitschaft, die Zweitkorrektur meiner Bachelorarbeit zu übernehmen. An dieser Stelle möchte ich beiden Professoren für die Gutachten danken, die sie während meines Bachelorstudiums für mich verfasst haben.

Mein Dank gilt auch meiner besseren Hälfte, Sally Chau, für das Korrekturlesen der Arbeit und für ihr Fachwissen, wodurch sie sehr gute Impulse setzen konnte. Auch für ihre kontinuierliche Unterstützung, sowohl in guten als auch in schlechten Zeiten, möchte ich ihr danken.

Nicht zuletzt möchte ich mich für die tatkräftige Unterstützung meiner Familie, insbesondere der meiner Eltern, bedanken. Sie haben es mir ermöglicht einen akademischen Weg einzuschlagen.

Inhaltsverzeichnis

1 | Einleitung

1.1 Motivation

Heutzutage stehen wir in vielen alltäglichen Situationen vor Problemen, für die wir kein Wissen über die zukünftige Entwicklung haben, um diese optimal lösen zu können. Daher verfolgt man in diesen Situationen das Ziel das Optimum möglichst gut zu approximieren. Die *Online-Algorithmik* ist ein Teilbereich der Informatik, die sich mit dieser Problematik beschäftigt. Auf den ersten Blick könnte man meinen, dass ein Algorithmus gegen einem Informationsmangel wenig erreichen kann. Anhand des *Sekretärsproblem* werden wir zeigen, dass trotz des Informationsmangels Algorithmen existieren, die sich der optimalen Lösung bis zu einem konstanten Faktor annähern können.

In der Informatik können Probleme in zwei Klassen unterteilt werden: *Offline-* und *Online-Probleme*. Beim Ersteren wird einem *Offline-Algorithmus* die komplette Eingabesequenz zu Beginn präsentiert. Das Ziel besteht darin, die optimale Lösung für das Offline-Problem zu bestimmen. Einen Algorithmus, der auf allen Eingabesequenzen die optimale Lösung findet, bezeichnet man als einen *optimalen Offline-Algorithmus* (OPT). Im Gegensatz zu Offline-Problemen wird bei Online-Problemen die Eingabesequenz schrittweise aufgedeckt. Das heißt, erst nachdem der *Online-Algorithmus* eine Entscheidung auf Eingabe i getroffen hat, wird dem Algorithmus die nächste Eingabe $i+1$ präsentiert. Das Ziel eines Online-Algorithmus ist es, die Lösung des optimalen Offline-Algorithmus möglichst gut zu approximieren. Falls die Approximation von keinem weiteren Algorithmus übertroffen werden kann, nennen wir ihn den *optimalen Online-Algorithmus*.

Ein bekanntes Online-Problem ist das *One-Way-Trading* [33]. Hierbei besitzen wir einen Betrag in Euro und wollen diesen vollständig bis zum Ende des Monats in Dollar umtauschen. Die Eingabesequenz besteht aus den täglichen Wechselkursen. Weiterhin ist eine untere und obere Grenze für den Wechselkurs bekannt. Dem

optimalen Offline-Algorithmus sind alle Wechselkurse zu Beginn bekannt und er entscheidet sich für den höchsten Kurs, der ihm den größten Profit verspricht. Dem Online-Algorithmus wird dagegen sequentiell der aktuelle Wechselkurs präsentiert. Die zukünftigen Kurse sind ihm nicht bekannt. Das Ziel ist es, einen Online-Algorithmus zu entwickeln, dessen Gewinn am Ende des Monats möglichst nah an dem Gewinn des optimalen Offline-Algorithmus liegt.

1.2 Problemstellung

In dieser Arbeit liegt der Fokus auf dem Sekretärsproblem. In der Literatur ist dieses auch unter den Namen *Heiratsproblem* oder *Sultan's dowry problem* bekannt. Beim Sekretärsproblem geht es um eine Menge von Kandidaten, die sich um eine Stelle als Sekretär bewerben. Die Anzahl der Bewerber ist im Vorhinein bekannt. Das Talent eines Einzelnen erfahren wir jedoch erst beim Bewerbungsgespräch und nach Abschluss des Bewerbungsgesprächs muss sofort eine Entscheidung für die Anstellung des Bewerbers oder dessen Ablehnung getroffen werden. Insbesondere muss die Entscheidung vor dem Eintreffen des nächsten Bewerbers getroffen worden sein. Die getroffene Entscheidung kann im Nachhinein nicht mehr revidiert werden.

Das Problem kann in zwei relevante Teilprobleme bezüglich des zu untersuchenden Ziels unterteilt werden: Das *Best-Choice-Problem* und das *Value-Problem*. Bei dem Ersteren möchten wir die Wahrscheinlichkeit maximieren, den besten Kandidaten zu erhalten. Wir interessieren uns somit nur für den besten Bewerber. Alle weiteren Kandidaten werden als Misserfolg interpretiert. Intuitiv scheint dieses Ziel zu strikt. Man betrachte den Fall, dass das Talent des Zweitbesten sehr nah an dem Talent des Besten liegt. Daher sollte man nicht außer Acht lassen, dass der zweitbeste Bewerber einen höheren Gewinn für das Unternehmen erwirtschaften kann als der Schlechteste. Das Value-Problem nimmt hingegen für jeden Kandidaten entsprechend seines Talents einen Gewinn an. Hierbei gilt es zu garantieren, dass der erwirtschaftete Gewinn durch die Auswahl eines Kandidaten nur um einen konstanten Faktor kleiner ist als der vom besten Kandidaten erwirtschafteten Gewinn.

Das Best-Choice-Problem und das Value-Problem können wiederum in jeweils zwei Teilprobleme, das *Full-Information-* und *Partial-Information-Szenario*, unterteilt werden. Sie unterscheiden sich in der Art und Weise, wie die Talente der

Bewerber einem Online-Algorithmus präsentiert werden. Beim Full-Information-Szenario wird dem Algorithmus das Talent eines jeden Bewerbers als numerischer Wert präsentiert. Beim Partial-Information-Szenario wird stattdessen in jedem Schritt eine Tabelle, in der alle bisher gesehenen Bewerber entsprechend ihrer Eignungen absteigend sortiert sind, übergeben. Man kann sich vorstellen, dass ein allwissendes Orakel diese Tabelle in jedem Schritt aktualisiert. Es ist wichtig zu beachten, dass in der Tabelle die tatsächlichen Talentwerte nicht enthalten sind und diese dem Online-Algorithmus vorenthalten werden. Wir werden die hier beschriebene Tabelle für das Partial-Information-Szenario auch als *Rangliste* oder *Rangfolge* der Bewerber bezeichnen. Falls wir allgemein von dem Best-Choice-Problem sprechen, sind damit beide Informationsszenarien, Partial- und Full-Information, gemeint. Analoges gilt für das Value-Problem. Abbildung 1.1 fasst die besprochenen Problemvarianten zusammen.

Eine Anwendung des Sekretärsproblem zeigt sich an der Börse. Wir besitzen ein Aktienpapier und möchten dieses bis Ende des Monats vollständig verkaufen. Je höher der Aktienkurs liegt, desto höher ist unser Gewinn des verkauften Aktienpapiers. Der tägliche Kurs der Aktie entspricht hierbei dem Talent der Bewerber beim Sektretärsproblem. Beim Best-Choice-Problem ist für uns nur der beste Aktienkurs relevant und alle weiteren Kurse interpretieren wir als Misserfolg. Bei diesem Anwendungsbeispiel wird deutlich, dass dieses Ziel nicht praxisrelevant ist. Ein Aktienkurs, der relativ nah am höchsten Kurs liegt, verspricht uns einen höheren Gewinn als der Schlechteste und es wäre somit nicht sinnvoll diese beiden Szenarien als gleichwertig einzustufen. Mithilfe des Value-Problems können wir unser Ziel besser formulieren. Wir verfolgen das Ziel, einen Aktienkurs zu erhalten, der höchstens um einen Faktor r schlechter ist als der beste Kurs. Somit garantieren wir ebenfalls, dass der von uns erhaltene Gewinn höchstens um einen Faktor r schlechter ist als der größtmögliche Gewinn.

1.3 Beitrag der Arbeit

Diese Arbeit wird sich aus weiteren drei Kapiteln zusammensetzen. In nächsten Kapitel wird zunächst der bisherige Forschungsstand zum Sekretärsproblem erläutert und dem Leser ein Einblick in die Vielfalt dieses Problem ermöglicht. Hierbei orientieren wir uns bei der Zusammenfassung an der bekannten Arbeit „Who solved the Secretary Problem“ [16] von Fergueson und werden diese durch weitere aktuelle Forschungsergebnisse ergänzen. Insbesondere erlaubt uns dieser Einblick eine Antwort auf Ferguesons in dem Titel gestellte Frage zu finden. In Kapitel 3

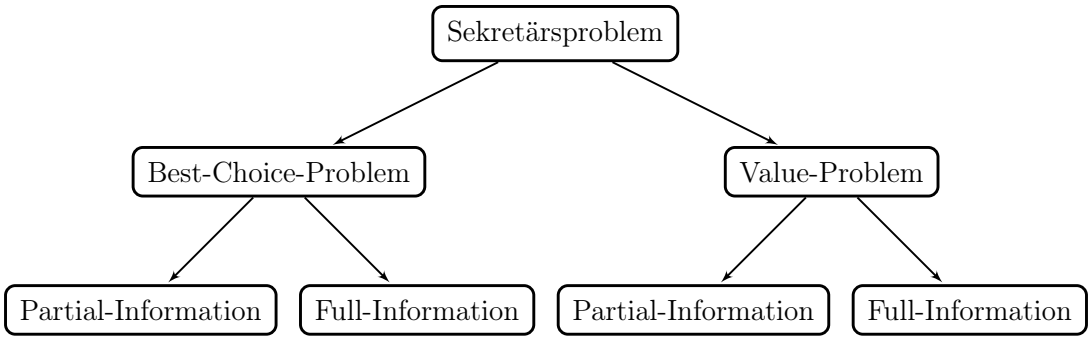

Abbildung 1.1: Variationen des Sekretärsproblems

werden wir das mathematische Modell für dieses Problem einführen und insbesondere erläutern, warum Online-Algorithmen beliebig schlecht werden können, falls die Reihenfolge der Kandidaten nicht zufällig generiert wird. Kapitel 4 stellt den Hauptteil dieser Arbeit dar. Dieses unterteilt sich in das Best-Choice und das Value-Problem. Wir werden einen bekannten Algorithmus aus der Literatur vorstellen und diesen für beide Problemvarianten bezüglich seiner Performanz analysieren. Weiterhin werden wir für beide Problemvarianten untere Schranken zeigen. Für die untere Schranke für das Best-Choice-Problem mit Partial-Information werden wir einen Beweis aus der Literatur [11] näher erläutern. Dieser Beweis nutzt ein lineares Programm. Wir werden unter anderem eine Lösung hierfür herleiten und den Wert dieses linearen Programms für eine wachsende Anzahl von Bewerbern berechnen. Für das Value-Problem mit Partial-Information werden wir zeigen, dass die untere Schranke dem vom Best-Choice-Problem gleicht. Für das Value-Problem mit Full-Information konnten wir das gleiche Resultat mit der Einschränkung der Eignungen auf natürliche Zahlen zeigen.

2 | Stand der Forschung

In diesem Kapitel werden wir uns anhand der definierten Problemvarianten aus Abbildung 1.1 einen Überblick über die Literatur des Sekretärsproblems verschaffen und Erweiterungen der Problemstellung besprechen. Das Sekretärsproblem ist ein sehr beliebtes Problem der Online-Algorithmik. Es entstand in der mathematisch stochastischen Optimierung.

2.1 Best-Choice-Problem

Die ursprüngliche Fassung des Best-Choice-Problems wurde als Full-Information-Szenario eingeführt. Sie wurde unter dem Namen *Game of Googol* [1] bekannt und zum ersten Mal in der Februar-Ausgabe von Martin Gardners Fachzeitschrift „Mathematical Games“ im Jahre 1960 veröffentlicht. Das Game of Googol ist als Zweisummenspiel definiert und lautet wie folgt:

> „*Bitte jemanden, beliebig viele Zettel zu nehmen und auf jeden eine andere positive Zahl zu schreiben. Die Zahlen können von einer Zahl kleiner als* 1 *bis hin zu einer Zahl der Größe von Googol oder sogar größer reichen. Diese Zettel werden zugedeckt und gemischt. Nacheinander werden diese aufgedeckt. Das Ziel ist es, aufzuhören, sobald man vermutet, die größte Zahl aufgedeckt zu haben. Man kann nicht zurückgehen und einen zuvor aufgedeckten Zettel wählen. Wenn man alle aufgedeckt hat, muss man den zuletzt aufgedeckten Zettel wählen.*“ [18]

[1] Googol steht für die Zahl 10^{100}. Man mag die Ähnlichkeit des Wortes mit dem Unternehmensnamen Google erkennen. In der Tat entstand der Name Google aus Googol [13].

Im Game of Googol repräsentieren die numerischen Werte auf den Zetteln die Eignungen der Bewerber im Sekretärsproblem. Wir bezeichnen die Person, die die Zettel nacheinander aufdeckt und einen Zettel auswählt, als Spieler I und die Person, die die numerischen Werte der Zettel festlegt, als Spieler II. Spieler I entspricht der Rolle eines Online-Algorithmus für das Sekretärsproblem und Spieler II versucht eine möglichst schlechte Instanz für den Online-Algorithmus zu wählen.

Bei der Lösung dieses Problems hat man sich zunächst auf das Best-Choice-Problem mit Partial-Information konzentriert. Ferguson bezeichnete diese Form des Sekretärsproblems als das *Basisproblem* [16]. Seine Arbeit erhielt sehr viel Aufmerksamkeit, sodass sich diese eingeführte Bezeichnung für das Best-Choice-Problem mit Partial-Information durchsetzte. Weiterhin ist das Basisproblem als *klassisches Sekretärsproblem* [4, 5, 10] und als *Standard-Sekretärsproblem* [6, 17] bekannt.
Der optimale Online-Algorithmus für das klassische Sekretärsproblem lehnt die ersten n/e Kandidaten ab und wählt den nächstbesten ankommenden Kandidaten. Hierbei ist n die Anzahl aller Kandidaten und e die eulersche Zahl. Der Algorithmus erhält mit einer Wahrscheinlichkeit von mindestens $1/e$ den besten Kandidaten. Wir diskutieren zunächst Arbeiten, die unterschiedliche Ansätze zur Herleitung dieses Algorithmus beschreiben und seine Optimalität beweisen. Lindley (1961, [27]) stellte als Erstes diesen Online-Algorithmus vor. Die Herleitung und Optimalität leitet Lindley mittels dynamischer Programmierung her. Beckmann (1990, [7]) konnte Lindleys Lösung mit einem einfacheren dynamischen Programm herleiten. Dynkin (1963, [14]) wiederum nutzte zur Lösung die Tatsache, dass das Best-Choice-Problem mit Partial-Information ein Spezialfall des Markov Stoppproblems darstellt. Bruss (1987, [10]) zeigte das Resultat und die Optimalität mit dem *Odds Theorem* nicht nur für das klassische Sekretärsproblem, sondern auch für eine Klasse von Problemen mit ähnlichem Charakter. Er bezeichnete den Algorithmus als das $1/e$-*Gesetz*. Der Beweis der Optimalität des Algorithmus wurde von Buchbinder et al. (2010, [11]) aufgegriffen und mittels eines linearen Programms bewiesen. Weiterhin konnten Buchbinder et al. das Resultat für Variationen des klassischen Sekretärsproblems zeigen, indem zusätzliche Bedingungen in das lineare Programm aufgenommen wurden.

Die Variationen des klassischen Sekretärsproblems entstanden hauptsächlich seit 1972. Unter anderem haben sich über die Jahre folgende vier bekannte Problemzweige des klassischen Sekretärsproblems ergeben: Einem Online-Algorithmus wird es ermöglicht seine Entscheidung zu revidieren und einen zuvor abgelehnten Kandidaten zu akzeptieren. Bei der Auswahl eines zuvor abgelehnten Kandidaten

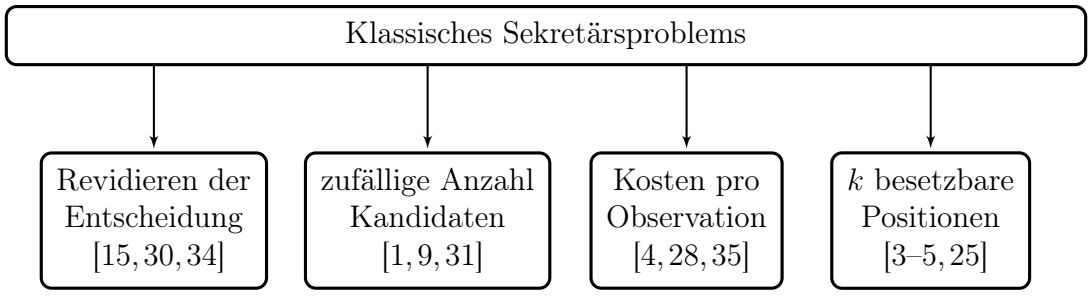

Abbildung 2.1: Variationen des klassischen Sekretärsproblems.

erhält der Online-Algorithmus einen Bruchteil des ursprünglichen Talent dieses Kandidaten. Bei einer weiteren Variante nimmt man die Anzahl der Bewerber n nicht mehr als gegeben an. Dieser Wert ist einem Online-Algorithmus entweder komplett unbekannt oder n wird gemäß einer vom Online-Algorithmus bekannten Wahrscheinlichkeitsverteilung gewählt. Der dritte Problemzweig diskutiert die Situation, dass jede Observation eines Kandidaten Kosten verursacht. Dies kann so interpretiert werden, dass für jedes Vorstellungsgespräch Zeit investiert werden muss und damit Kosten verbunden sind. Eine weitere Problemvariation besteht darin, dass mehrere Positionen zu vergeben sind und man hierfür eine Menge an Bewerbern auswählen muss. Die Variationen des klassischen Sekretärsproblems, einschließlich der relevanten Arbeiten, sind in Abbildung 2.1 dargestellt.

Wir werden zwei der in Abbildung 2.1 illustrierten Erweiterungen des klassischen Sekretärsproblems anhand der Arbeit [4] von Babaioff et al. aus dem Jahr 2009 detaillierter beschreiben. Die Variante, in der Kosten pro Observation entstehen, wird von Babaioff et al. als *Discounted-Secretary-Problem* bezeichnet. Hierzu wird angenommen, dass ein zeitabhängiger Diskontierfaktor $d(t)$ existiert. Das Talent eines akzeptierten Kandidaten zu einem Zeitpunkt t entspricht dem Produkt aus seinem ursprünglichen Talent und dem Diskontierfaktor $d(t)$. Babaioff et al. haben einen $\mathcal{O}(\log n)$-kompetitiven Online-Algorithmus und eine untere Schranke von $\Omega(\log n/(\log\log n))$ für dieses Problem beweisen können. Die Variante, in der es k besetzbare Positionen gibt, führen Babaioff et al. als das *Weighted-Secretary-Problem* ein. Jede der k Positionen besitzt einen Wert und jeder Kandidat ein Talent. Das Ziel eines Online-Algorithmus für diese Problemvariante ist es, jeder Positionen einen Kandidaten zuzuordnen. Der Profit einer Zuordnung ergibt sich als Summe über die Produkte eines Positionswertes und dem Talent seines zugeordneten Kandidaten. Diesen Term gilt es zu maximieren. Babaioff et al. stellen

einen 4-kompetitiven Online-Algorithmus hierfür vor.

Für das Best-Choice-Problem mit Partial-Information konnte, wie bereits erwähnt, ein optimaler Online-Algorithmus bereits in den Sechziger Jahren hergeleitet werden. Dieses Teilproblem wurde somit von der Forschungsgemeinde vollständig gelöst. Bis zur vollständigen Lösung des Best-Choice-Problems mit Full-Information musste sich die Forschungsgemeinde allerdings bis 1994 gedulden. Ferguson (1989, [16]) erwähnte zum ersten Mal in der Literatur, dass die untere Schranke für das ursprüngliche Problem des Game of Googol bisher ungelöst geblieben ist. Hierbei stellt er sich die Frage, ob die untere Schranke für das Best-Choice-Problem mit Partial-Information (klassisches Serkretärsproblem) und mit Full-Information (Game of Googol) identisch ist. Somit hätten beide Probleme den gleichen optimalen Online-Algorithmus. Die Frage reduziert sich auf das Problem, ob die Präsentation der numerischen Werte der Kandidateneignungen einem Online-Algorithmus einen Vorteil verschafft, den besten Kandidaten zu erhalten. Falls die Talente unabhängig und identisch gemäß einer für den Online-Algorithmus bekannten Wahrscheinlichkeitsverteilung verteilt sind, ist die Antwort auf die Frage positiv. Gilbert and Mosteller (1966, [19]) konnten zeigen, dass für diesen Fall die Wahrscheinlichkeit, den besten Kandidaten zu erhalten, für wachsendes n gegen 0.58016 strebt. Für das Best-Choice-Problem mit Full-Information (Game of Googol) konnten zum ersten Mal Silvermann und Nadas (1992, [36]) für $n = 3$ zeigen, dass die Wahrscheinlichkeit, den besten Kandidaten zu erhalten, bei $1/e$ liegt und somit der Wahrscheinlichkeit des Partial-Information-Szenario entspricht. Gnedin (1994, [20]) konnte das Resultat auf $n \geq 3$ ausweiten und somit eine scharfe untere Schranke für das Game of Googol zeigen. Die Antwort auf die Frage von Ferguson [16] „Who solved the secretary problem?“ lautet demnach: Alexander von Gnedin.

2.2 Value-Problem

Dem Value-Problem wurde in der Literatur nicht ansatzweise so viel Beachtung geschenkt, wie dem Best-Choice-Problem. Seinen Ursprung fand das Value-Problem jedoch fast ein Jahrhundert früher, im Jahre 1875, als das *Cayley-Problem*. Benannt wurde es nach seinem Verfasser Arthur von Cayley, der das Problem in dieser Form publizierte:

> „*Eine Lotterie hat folgende Spielregeln: Es gibt n Lose mit einem Wert von jeweils* $a, b, c, \ldots$ *Pounds. Eine*

> *Person zieht ein Los, schaut auf den Wert und darf bei Wunsch (aus den $n-1$ verbleibenden Losen) nochmals ziehen bis er höchstens k Mal gezogen hat. Er erhält den Wert des zuletzt gezogenen Loses. Angenommen er zieht die Lose in der vorteilhaftesten Weise. Was ergibt sich dann für seinen erwarteten Gewinn?*“ [12]

Die Beschreibung des Problems lässt darauf schließen, dass es sich um das Full-Information-Szenario handelt. Das Cayley-Problem ist lange Zeit in Vergessenheit geraten bis Moser [29] es im Jahre 1956 aufgriff und wurde seitdem ebenfalls als *Caley-Moser-Problem* bezeichnet. Hierfür nahm er allerdings an, dass die zugrunde liegende Wahrscheinlichkeitsverteilung, mit der die Talente der Kandidaten gewählt werden, einem Online-Algorithmus im Vorhinein bekannt ist. Die Werte der Talente sind unabhängig und identisch uniform auf dem Intervall (0,1) verteilt. In späteren Arbeiten wurden weitere Verteilungen, unter anderem die Normalverteilung (1960, [21]) und die Exponentialverteilung (1962, [24]), betrachtet. Bearden (2006, [6]) untersuchte hingegen das Value-Problem mit Partial-Information. Er nimmt ebenfalls an, dass dem Online-Algorithmus bekannt ist, dass die Talentwerte der Kandidaten unabhängig und identisch uniform auf dem Intervall (0,1) verteilt sind. Er kommt zu dem Schluss, dass der optimale Online-Algorithmus $\sqrt{n}$ Kandidaten ablehnt und den nächstbesten Kandidaten auswählt. Der Online-Algorithmus erhält mit dieser Strategie einen Kandidaten mit erwartetem Talent von 1 für wachsendes n.

2.3 Variation des Ziels

In der Literatur werden zusätzlich zu den zwei Zielen, dem Best-Choice- und dem Value-Problem, weitere Zielfunktionen für das Sekretärsproblem diskutiert. Anstatt die Absicht zu verfolgen den besten Kandidaten zu erhalten, untersucht Vanderbei [37] im *Postdoc*-Problem das Vorhaben den zweitbesten Kandidaten im Partial-Information-Szenario zu erhalten. Motiviert wird dieses Vorhaben durch die Tatsache, dass der beste Bewerber für eine Doktorandenstelle ein Angebot von Harvard erhalten habe und dieses auch annehmen wird. Es stellt sich heraus, dass für wachsendes n die Wahrscheinlichkeit gegen $1/4$ strebt den zweitbesten Bewerber auszuwählen und somit diese kleiner ist als $1/e$, die Wahrscheinlichkeit den Besten zu erhalten.

3 | Das mathematische Modell

3.1 Problemdefinition

Im Folgenden werden wir die Notationen und Definitionen der Online-Algorithmik in Zusammenhang mit dem Sekretärsproblem einführen. Wir orientieren uns hierbei an [33].

Beim Sekretärsproblem gibt es eine Menge von n *Bewerbern* oder *Kandidaten*, die sich um eine Stelle als Sekretär bewerben. Die Bewerber werden nacheinander zum Vorstellungsgespräch eingeladen und ein Online-Algorithmus A hat die Aufgabe, einen einzigen Kandidaten aus dieser Menge auszuwählen. Die Anzahl der Bewerber n ist dem Algorithmus A zu Beginn bekannt, aber das *Talent* beziehungsweise die *Eignung* der Bewerber für die Arbeitsstelle ist vor dem Vorstellungsgespräch unbekannt. Die Information über die Talente werden dem Online-Algorithmus nacheinander in der Eingabesequenz $\sigma = (\sigma_1, \ldots, \sigma_n)$ präsentiert. Beim Vorstellungsgespräch mit dem i-ten Bewerber tritt seine Eignung σ_i zum Vorschein. Ein Online-Algorithmus A kann den Bewerber akzeptieren oder ablehnen. Falls A ihn ablehnt, darf Bewerber i nicht mehr für die Rolle in Betracht gezogen werden und dem Algorithmus wird der nächste Bewerber mit σ_{i+1} präsentiert. Bei Akzeptanz des i-ten Bewerbers werden ihm keine der weiteren Bewerber mit $\sigma_{i+1}, \ldots, \sigma_n$ präsentiert. Falls ein Online-Algorithmus A alle $n-1$ Kandidaten ablehnt, ist er gezwungen den letzten Kandidaten auszuwählen.

In welcher Form das Talent σ_i des i-ten Bewerbers einem Online-Algorithmus A präsentiert wird, liegt von dem zugrunde liegenden Informationsszenario ab. Beim Full-Information-Szenario ist $\sigma_i \in \mathbb{R}_{\geq 0}$ der numerische Wert des Talents des i-ten Kandidaten. Beim Partial-Information-Szenario ist σ_i die aktualisierte *Rangfolge* der bisher gesehenen Bewerber. Die Bewerber sind in dieser Rangfolge absteigend gemäß ihrem Talent sortiert, aber der numerische Wert des Talents eines Bewer-

bers ist nicht enthalten.

Wir werden einige zusätzliche Notationen und Terminologien zur Vereinfachung einführen. Mit w_1 bezeichnen wir die Eignung des Besten, mit w_2 die des zweitbesten Bewerbers bis hinzu w_n, welches die Eignung des schlechtesten Bewerbers darstellt. Weiterhin nehmen wir ohne Beschränkung der Allgemeinheit an, dass $w_i \neq w_j$ für $i \neq j$. Mit $w_A(\sigma)$ bezeichnen wir das Talent des von A ausgewählten Kandidaten auf der Eingabe σ und mit $p_A(\sigma)$ den erwirtschafteten Profit dieses Kandidaten. Das Ziel eines jeden Online-Algorithmus für das Best-Choice- oder Value-Problem wird es sein den Profit $p_A(\sigma)$ zu maximieren. Der Profit ist für die zwei Problemvariationen unterschiedlich definiert.

Definition 3.1. (Profit eines Online-Algorithmus)
Sei A ein Online-Algorithmus für das Sekretärsproblem. Für das Best-Choice-Problem ist der erwirtschaftete Profit von A auf der Eingabe σ als

$$p_A^B(\sigma) = \begin{cases} 1 & \textit{falls } w_A(\sigma) = w_1, \\ 0 & \textit{falls } w_A(\sigma) \neq w_1 \end{cases}$$

definiert. Für das Value-Problem ist der Profit von A als

$$p_A^V(\sigma) = w_A(\sigma)$$

definiert.

Um die Qualität des Profits eines Online-Algorithmus A bewerten zu können, führen wir eine *kompetitive Analyse* [23] auf A durch. Hierbei wird der Profit $p_A^B(\sigma)$, den Online-Algorithmus A für das Best-Choice-Problem erzielt, mit dem Profit $\mathrm{OPT}^B(\sigma)$ des optimalen Offline-Algorithmus (OPT) verglichen. Analog gilt dies auch für das Value-Problem mit $p_A^V(\sigma)$ und $\mathrm{OPT}^V(\sigma)$. Wir werden hierzu den *kompetitiven Faktor* bestimmen, der ein Maß für die Lösung eines Online-Algorithmus A darstellt. Für das Sekretärsproblem Π bezeichnen wir die Menge aller Instanzen mit $\mathcal{I}_\Pi$.

Definition 3.2. (Kompetitiver Faktor eines Online-Algorithmus)
Ein deterministischer Online-Algorithmus A für das Sekretärsproblem Π für das Best-Choice-Problem erreicht einen kompetitiven Faktor (competitive ratio) von $r \geq 1$, wenn

$$\exists \tau \in \mathbb{R} : \forall \sigma \in \mathcal{I}_\Pi : \mathrm{OPT}^B(\sigma) \leq r \cdot p_A^B(\sigma) + \tau$$

gilt. Ein randomisierter Online-Algorithmus A für das Sekretärsproblem Π für das Best-Choice-Problem erreicht einen kompetitiven Faktor von $r \geq 1$, wenn

$$\exists \tau \in \mathbb{R} : \forall \sigma \in \mathcal{I}_\Pi : \mathrm{OPT}^B(\sigma) \leq r \cdot \mathbf{E}\left[p_A^B(\sigma)\right] + \tau$$

gilt.

Für das Value-Problem lässt sich Definition 3.2 analog formulieren, indem man $\mathrm{OPT}^B(\sigma)$ durch $\mathrm{OPT}^V(\sigma)$ und $p_A^B(\sigma)$ durch $p_A^V(\sigma)$ ersetzt. Falls $\tau = 0$ gilt, ist A sogar ein *strikt* r-kompetitiver Online-Algorithmus. Man beachte, dass dem optimalen Offline-Algorithmus zu Beginn die komplette Eingabesequenz σ bekannt ist und dieser sich, sowohl für das Best-Choice-, als auch für das Value-Problem, für den besten Bewerber entscheiden wird. Somit ergibt sich für den Profit des optimalen Offline-Algorithmus stets $\mathrm{OPT}^B(\sigma) = 1$ für das Best-Choice-Problem und stets $\mathrm{OPT}^V(\sigma) = w_1$ für das Value-Problem.

Da der Profit $p_A^B(\sigma)$ für das Best-Choice-Problem als Indikatorvariable definiert ist, erhalten wir für randomisierte Algorithmen ein wichtiges Resultat.

Lemma 3.3. *Sei A ein randomisierter Online-Algorithmus für das Best-Choice-Problem. Dann gilt*

$$\mathbf{E}[p_A^B(\sigma)] = \mathbf{Pr}[w_A(\sigma) = w_1].$$

Beweis. Dies folgt direkt aus der Definition von $p_A^B(\sigma)$ als Indikatorvariable, da

$$\mathbf{E}[p_A^B(\sigma)] = 1 \cdot \mathbf{Pr}[p_A^B(\sigma) = 1] + 0 \cdot \mathbf{Pr}[p_A^B(\sigma) = 0] = \mathbf{Pr}[w_A(\sigma) = w_1].$$

□

Lemma 3.3 sagt aus, dass sich das Problem zur Bestimmung des Profits eines randomisierten Online-Algorithmus stets auf die Bestimmung der Wahrscheinlichkeit den besten Kandidaten zu erhalten reduzieren lässt. Hieraus ergibt sich das folgende Theorem.

Theorem 3.4. *Ein Online-Algorithmus A für das Best-Choice-Problem ist genau dann strikt r-kompetitiv, wenn A den besten Kandidaten mit einer Wahrscheinlichkeit von mindestens $1/r$ erhält.*

Beweis. Sei A ein r-kompetitiver Algorithmus für das Best-Choice-Problem, dann folgt aus Lemma 3.3, dass $1 = \mathrm{OPT}^B(\sigma) \leq r \cdot \mathbf{E}[p_A^B(\sigma)] = r \cdot \mathbf{Pr}[w_A(\sigma) = w_1]$ gilt und somit erhält man

$$\frac{1}{r} \leq \mathbf{Pr}[w_A(\sigma) = w_1].$$

Die Umkehrung folgt direkt. □

Durch die Einführung des Random-Order-Modells im nächsten Abschnitt, begründen wir, warum wir uns bei der kompetitiven Analyse von Online-Algorithmen für das Sekretärsproblem auf die Definition von randomisierten Algorithmen beschränken können. Hierdurch werden die Resultate aus Lemma 3.3 und Theorem 3.4 für die Abschnitte 4.3 und 4.4 von Relevanz sein.

3.2 Das Random-Order-Modell

Als Algorithmenentwerfer für das Sekretärsproblem sind wir stets daran interessiert, einen Online-Algorithmus zu entwickeln, der auf allen Instanzen einen konstanten kompetitiven Faktor erreicht. Man kann sich vorstellen, dass die Instanzen von einem Gegner konstruiert werden. Sein Ziel ist es, eine möglichst schlechte Eingabe für den Algorithmus zu konstruieren. In dem bisher beschriebenen Modell ist der Gegner allerdings zu mächtig. Er kann für jeden Online-Algorithmus A eine Eingabe konstruieren, für die A keinen konstanten kompetitiven Faktor erreicht. Aus diesem Grund werden wir das *Random-Order-Modell* einführen. Das Ziel dieses Modells ist es, den Gegner zu schwächen.

Bevor wir näher auf das neue Modell eingehen, werden wir zunächst zeigen, dass alle Algorithmen in dem bisher betrachteten Modell nicht kompetitiv sind. Im ersten Schritt zeigen wir die Behauptung für deterministische Algorithmen und im zweiten Schritt für eine Klasse von randomisierten Algorithmen.

Theorem 3.5. *Es existiert kein deterministischer Online-Algorithmus für das Best-Choice- oder Value-Problem mit konstantem kompetitiven Faktor.*

Beweis. Sei A ein beliebiger deterministischer Algorithmus. Wir werden die Aussage

$$\nexists r \geq 1 : \forall \sigma \in \mathcal{I}_\Pi : \mathrm{OPT}(\sigma) \leq r \cdot p_A(\sigma)$$

zeigen. Dies ist äquivalent zu

$$\forall r \geq 1 : \exists \sigma \in \mathcal{I}_\Pi : \mathrm{OPT}(\sigma) > r \cdot p_A(\sigma). \tag{3.1}$$

Hierbei ist der Profit $p_A(\sigma)$, je nachdem, ob es sich um das Best-Choice oder Value-Problem handelt, entsprechend definiert.

Wir weisen die Aussage zunächst für das Value-Problem nach. Sei hierzu A ein beliebiger Online-Algorithmus für das Value-Problem. Gemäß (3.1) reicht es aus eine Instanz σ zu wählen, auf der A keinen konstanten kompetitiven Faktor erreicht. Die Instanz kann auf zwei Bewerber beschränkt werden. Dem ersten Kandidaten wird eine Eignung $D \in \mathbb{R}_{\geq 0}$ mit $D > r$ zugewiesen. Je nach Entscheidung von A bezüglich des ersten Kandidaten, wird die Eignung des zweiten Bewerbers mit

$$\sigma' = \begin{cases} (D, r \cdot D + 1) & \text{falls } A \text{ den ersten Bewerber auswählt,} \\ (D, \frac{D}{r} - 1) & \text{falls } A \text{ den ersten Bewerber ablehnt,} \end{cases}$$

entsprechend gewählt. OPT wählt den Bewerber mit größerem Talent aus. Sei $r \geq 1$ beliebig, dann gilt im ersten Fall

$$\mathrm{OPT}^V(\sigma') = r \cdot D + 1 > r \cdot D = r \cdot p_A^V(\sigma')$$

und im zweiten Fall

$$\mathrm{OPT}^V(\sigma') = D = r \cdot \frac{D}{r} > r \cdot \left(\frac{D}{r} - 1\right) = r \cdot p_A^V(\sigma').$$

Somit ist die Behauptung für das Value-Problem bewiesen.

Wir zeigen die Aussage für das Best-Choice-Problem. Sei hierzu A ein beliebiger Online-Algorithmus für das Best-Choice-Problem. A wird auf der oben definierten Instanz σ' in keinem der beiden Fälle den besten Bewerber erhalten und somit gilt unabhängig von der Entscheidung von A

$$\mathrm{OPT}^B(\sigma') = 1 > r \cdot 0 = p_A^B(\sigma')$$

für jedes beliebige $r \geq 1$. Somit erreicht kein Algorithmus für das Best-Choice-Problem einen konstanten kompetitiven Faktor. □

Als Nächstes werden wir die Aussage auf randomisierte Algorithmen erweitern. Allerdings werden wir uns hierbei auf die Klasse der *skalierungsinvarianten Algorithmen* beschränken.

Definition 3.6. *Seien $\sigma \in \mathcal{I}_\Pi$ und $\alpha \in \mathbb{R}_{\geq 0}$ beliebig. Ein Algorithmus A heißt skalierungsinvariant, wenn A für die Eingaben σ und $\alpha \cdot \sigma = (\alpha \cdot \sigma_1, \ldots, \alpha \cdot \sigma_n)$ die gleiche Entscheidung trifft.*

Für den Beweis werden wir ein bekanntes Lemma der *Minimax-Theorie* nutzen. Sei hierzu Π ein Optimierungsproblem, dann definieren wir $\mathcal{A}$ als die endliche Menge aller deterministischen Algorithmen für Π, $\mathcal{I}$ als die endliche Menge aller Eingaben für Π und $C(A, I) \in \mathbb{R}_{\geq 0}$ als das Kostenmaß für einen Algorithmus $A \in \mathcal{A}$ auf einer Instanz $I \in \mathcal{I}$. Außerdem bezeichnen wir mit P und Q die Mengen aller Wahrscheinlichkeitsverteilungen auf den Mengen $\mathcal{I}$ und $\mathcal{A}$. Folgendes Resultat ist als *Yoas Ungleichung* bekannt und der Beweis hierzu kann in [33] nachgeschlagen werden.

Lemma 3.7. (Yoas Ungleichung)
Für alle Verteilungen $p \in P$ über $\mathcal{I}$ und für alle Verteilungen $q \in Q$ über $\mathcal{A}$ gilt

$$\min_{A \in \mathcal{A}} \mathbf{E}[C(I_p, A)] \leq \max_{I \in \mathcal{I}} \mathbf{E}[C(I, A_q)],$$

wobei I_p eine gemäß p zufällig gewählte Eingabe und A_q ein gemäß q zufällig gewählter Algorithmus ist.

Yoas Ungleichung sagt aus, dass für den Beweis einer unteren Schranke für randomisierte Algorithmen der beste deterministische Algorithmus auf einer randomisierten Eingabe betrachtet werden kann. Sein erwartetes Kostenmaß ist eine untere Schranke für das erwartete Kostenmaß eines randomisierten Algorithmus auf der für ihn schlechtesten Eingabe. Mit Hilfe von Yaos Ungleichung lässt sich die folgende Aussage verifizieren, dessen Beweis sich an der Arbeit [2] von Aggarwal et al. orientiert.

Theorem 3.8. *Es existiert kein randomisierter skalierungsinvarianter Algorithmus für das Best-Choice- oder Value-Problem mit konstantem kompetitiven Faktor.*

Beweis. Die Menge der Eingaben $\mathcal{I}$ ist endlich, da sich die Eingaben nur in der Reihenfolge der Bewerber unterscheiden. Bei n Bewerbern erhalten wir demnach $n!$ signifikant unterschiedliche Eingaben. Die Menge der Algorithmen $\mathcal{A}$ ist ebenfalls endlich, da sich die Algorithmen nur bezüglich ihrer Entscheidung des gewählten Bewerbers unterscheiden. Somit sind beide Voraussetzungen erfüllt, um Lemma 3.7 anzuwenden.

Wir werden eine Menge von Eingaben $\mathcal{I}' \subset \mathcal{I}$ definieren. Die Menge $\mathcal{I}'$ enthält Eingaben σ^i mit $i \in \{1, \ldots, n\}$ der Form

$$\sigma^i = (\underbrace{D^i, D^{i+1}, \ldots, D^n}_{(n-i+1)-\text{Mal}}, \underbrace{0, \ldots, 0}_{(i-1)-\text{Mal}}),$$

wobei $D \in \mathbb{R}$ eine hinreichend große Zahl ist und es sei q eine Gleichverteilung auf der Menge $\mathcal{I}'$.

Wir werden die Behauptung zunächst für das Value-Problem zeigen. Da für alle Instanzen σ^i der beste Kandidat ein Talent von D^n besitzt, ist der Profit des optimalen Offline-Algorithmus auf σ^q bestimmt durch $\text{OPT}^V(\sigma^q) = D^n$. Es gilt noch den Profit des besten deterministischen Algorithmus A auf σ^q zu bestimmen. Die Wahrscheinlichkeit, dass σ^q eine Realisierung σ^i annimmt, entspricht $q(\sigma^i) = 1/n$. Den Profit von A auf der Eingabe σ^q kann man wie folgt herleiten. Algorithmus A wählt auf der Instanz σ^1 einen Bewerber an einer beliebigen Position $k \in \{1, \ldots, n\}$ mit Eignung D^k aus. Um den Gewinn von σ^i für $i > 1$ zu bestimmen, werden wir hierzu zwei Fälle betrachten.

- **1.Fall:** Sei $i \in \{2, \ldots, n-k+1\}$.
  Für $i \in \{2, \ldots, n-k+1\}$ sind die Kandidateneignungen in σ^i beschränkt auf die ersten k Kandidaten ein Vielfaches von den ersten k Kandidaten in σ^1.

Aus der Skalierungsinvarianz des Algorithmus A gilt, dass dieser beschränkt auf den ersten k Kandidaten für σ^i die gleiche Entscheidung trifft wie zuvor auf σ^1 und den Bewerber an Postion k auswählt. Dies gilt ebenfalls für die gesamte Eingabesequenz und folgt aus der Tatsache, dass es sich bei A um einen Online-Algorithmus handelt. Da A auf σ^i stets den Kandidaten an Position k auswählt, beträgt für $i \in \{1, \ldots, n-k+1\}$ die Eignung dieses Kandidaten D^{k+i-1} und entsprechend hoch ist ebenfalls der Profit $p_A(\sigma^i)$ auf der Eingabe σ^i.

- **2.Fall:** Sei $i \in \{n-k+2, \ldots, n\}$.
  Für $i \in \{n-k+2, \ldots, n\}$ besitzen nur noch Kandidaten an Positionen kleiner als k in σ^i ein Talent, die ein Vielfaches von σ^1 ist. Für diese Kandidaten gilt, dass A die selbe Entscheidung trifft wie zuvor auf σ^1 und diese ablehnt. Welche Entscheidung A für die restlichen Kandidaten trifft, kann nicht mehr bestimmt werden. Da alle restlichen Kandidaten aber ohnehin ein Talent in Höhe von Null besitzen, entspricht der Profit von A auf σ^i unabhängig von seiner Entscheidung dem Wert Null.

Zusammenfassend ergibt sich aus diesen beiden Fällen für den erwarteten Profit von A auf σ^q

$$\begin{aligned}
\mathbf{E}[p_A^V(\sigma^q)] &= \sum_{i=1}^{n} q(\sigma^i) \cdot p_A^V(\sigma^i) = \underbrace{\sum_{i=1}^{n-k+1} \frac{1}{n} D^{k+i-1}}_{\text{1.Fall}} + \underbrace{\sum_{i=n-k+2}^{n} \frac{1}{n} \cdot 0}_{\text{2.Fall}} = \frac{1}{n} \sum_{i=k-1}^{n} D^i \\
&\leq \frac{1}{n} \sum_{i=0}^{n} D^i = \frac{1}{n} \cdot \frac{D^{n+1}-1}{D-1} \leq \frac{1}{n} \cdot \frac{D^{n+1}-1}{D/2} \leq \frac{2}{n} \cdot D^n \\
&= \frac{2}{n} \cdot \mathrm{OPT}^V(\sigma^q).
\end{aligned}$$

Für wachsendes n konvergiert der erwartete Profit gegen Null und somit existiert kein deterministischer Online-Algorithmus mit konstantem Faktor auf eine randomisierte Instanz. Laut Yoas Ungleichung gilt somit, dass kein randomisierter skalierungsinvarianter Online-Algorithmus mit konstanten kompetitiven Faktor für das Value-Problem existiert.

Als Nächstes leiten wir die Aussage für das Best-Choice-Problem her. Sei A der beste deterministische Online-Algorithmus für das Best-Choice-Problem und alles weitere sei gewählt wie zuvor. Das Verhalten von A ist ähnlich wie zuvor beim Value-Problem. Dann wird der Algorithmus A nur auf einer Instanz den besten

Kandidaten erhalten und auf allen anderen einen anderen. Dies passiert jeweils mit Wahrscheinlichkeit $1/n$. Gemäß Definition 3.1 ist der Profit von A bei Erhalt des besten Kandidaten $p_A(\sigma) = 1$ und bei Erhalt eines anderen Kandidaten gilt $p_A(\sigma) = 0$. Daraus folgt

$$\mathbf{E}[p_A^B(\sigma^q)] = \sum_{i=1}^{n} q(\sigma^i) \cdot p_A^B(\sigma^i) = \frac{1}{n} \cdot 1 = \frac{1}{n} \cdot \mathrm{OPT}^B(\sigma^q).$$

Für wachsendes n konvergiert der kompetitive Faktor gegen 0. Somit gilt ähnlich wie beim Value-Problem, dass kein randomisierter skalierungsinvarianter Online-Algorithmus für das Best-Choice-Problem mit konstanten kompetitiven Faktor existiert. □

Wir haben zur Vereinfachung der Notation in Theorem 3.5 und 3.8 die Eingaben mit den numerischen Werte für die Kandidaten belegt und somit formell den Beweis für das Full-Information-Szenario nachgewiesen. Für das Partial-Information-Szenario können die Kandidateneignungen ähnlich gewählt werden und die Rangfolgen für jedes σ_i aus den gegebenen Kandidateneignungen von einem Orakel abgeleitet werden. Der restliche Beweis ist analog.

Es konnte nachgewiesen werden, dass sich Algorithmen auf dem ursprünglichen Modell beliebig schlecht verhalten können. Damit sich Online-Algorithmen für das Problem etablieren können, führen wir das Random-Order-Modell ein. Hierfür wird der Gegner geschwächt, indem dieser nicht mehr die Reihenfolge der ankommenden Bewerber bestimmen darf. Es wird aus den $n!$ möglichen Reihenfolgen der Bewerber eine uniform zufällige ausgewählt. Die zufällig gewählte Reihenfolge in diesem Modell hat keinen Einfluss auf das Verhalten des optimalen Offline-Algorithmus, da dieser unabhängig der gewählten Reihenfolge der Kandidaten stets den besten Kandidaten auswählen wird. Allerdings ist der Profit $p_A(\sigma)$ des ausgewählten Kandidaten eines Online-Algorithmus A vom Zufall beeinflusst. Dementsprechend ist der Profit im Random-Order-Modell eine Zufallsvariable und wir werden daher vom erwarteten Profit $\mathbf{E}[p_A(\sigma)]$ eines Online-Algorithmus A auf einer Instanz σ sprechen. Daher ist die Definition aus 3.2 eines randomisierten Algorithmus für die kompetitive Analyse im Random-Order-Modell von Relevanz. Für den weiteren Verlauf der Arbeit werden wir für das Sekretärsproblem stets das Random-Order-Modell zu Grunde legen.

4 | Kompetitive Analyse

4.1 Zusammenhang der Problemvarianten

Bisher haben wir stets eine klare Trennlinie zwischen den beiden Zielfunktionen für das Sekretärsproblem, dem Best-Choice-Problem und dem Value-Problem, gezogen. Ziel dieses Unterkapitels soll es sein, den Zusammenhang zwischen diesen beiden Zielfunktionen zu verdeutlichen. Im nächsten Schritt werden ebenfalls relevante Zusammenhänge zwischen den zwei Informationsszenarien, Partial- und Full-Information-Szenario, erläutert.

Theorem 4.1. *Sei A ein strikt r-kompetitiver Online-Algorithmus für das Best-Choice-Problem. Dann ist A ein strikt r-kompetitiver Online-Algorithmus für das Value-Problem.*

Beweis. Sei A ein strikt r-kompetitiver Online-Algorithmus für das Best-Choice-Problem. Gemäß Theorem 3.4 gilt, dass A den besten Kandidaten mit einer Wahrscheinlichkeit von mindestens $1/r$ auswählt. Es folgt

$$\forall \sigma \in \mathcal{I}_\Pi : \mathbf{E}[p_A^B(\sigma)] = \mathbf{Pr}[w_A(\sigma) = w_1] \geq \frac{1}{r}. \tag{4.1}$$

Sei $\sigma \in \mathcal{I}_\Pi$ beliebig. Für den kompetitiven Faktor von A für das Value-Problem folgt

$$\begin{aligned}
\mathbf{E}\left[p_A^V(\sigma)\right] &= \sum_{i=1}^{n} \mathbf{Pr}\left[p_A^V(\sigma) = w_i\right] \cdot w_i \\
&= \mathbf{Pr}\left[p_A^V(\sigma) = w_1\right] \cdot w_1 + \sum_{i=2}^{n} \mathbf{Pr}\left[p_A^V(\sigma) = w_i\right] \cdot w_i \\
&\geq \mathbf{Pr}\left[w_A(\sigma) = w_1\right] \cdot w_1 \\
&\overset{(4.1)}{\geq} \frac{1}{r} \cdot \mathrm{OPT}^V(\sigma).
\end{aligned}$$

Da wir die kompetitive Analyse für das Value-Problem betrachten, wird in der letzten Zeile der Umformung $\mathrm{OPT}^V(\sigma) = w_1$ genutzt. Somit haben wir gezeigt, dass

$$\forall \sigma \in \mathcal{I}_\Pi : \mathrm{OPT}^V(\sigma) \leq r \cdot \mathbf{E}[p_A^V(\sigma)]$$

gilt. Es folgt, dass A ein strikt r-kompetitiver Online-Algorithmus für das Value-Problem ist. □

Die Umkehrung der Aussage aus Theorem 4.1 wird für Abschnitt 4.4 über das Value-Problem von Relevanz sein. Wir werden die Umkehrung zunächst für das Partial-Information-Szenario beweisen. Das folgende Theorem ist anschaulich folgendermaßen formuliert: Falls ein Online-Algorithmus mit besserem kompetitiven Faktor als r für das Value-Problem mit Partial-Information existiert, dann gibt es ebenfalls einen Online-Algorithmus für das Best-Choice-Problem mit Partial-Information, der besser als r-kompetitiv ist.

Theorem 4.2. *Sei A ein strikt $(r-\epsilon)$-kompetitiver Online-Algorithmus für das Value-Problem mit Partial-Information für ein beliebiges $\epsilon \in (0, r-1]$. Dann ist A ein strikt $(r-\epsilon/2)$-kompetitiver Online-Algorithmus für das Best-Choice-Problem mit Partial-Information.*

Beweis. Sei A ein strikt $(r-\epsilon)$-kompetitiver Online-Algorithmus für das Value-Problem für ein beliebiges $\epsilon \in (0, r-1]$, das heißt

$$\forall \epsilon \in (0, r-1] : \forall \sigma \in \mathcal{I}_\Pi : \mathrm{OPT}^V(\sigma) \leq (r-\epsilon)\, \mathbf{E}[p_A^V(\sigma)].$$

Es muss Folgendes gezeigt werden.

$$\forall \epsilon \in (0, r-1] : \forall \sigma \in \mathcal{I}_\Pi : \mathrm{OPT}^B(\sigma) \leq (r-\epsilon/2)\, \mathbf{E}[p_A^B(\sigma)].$$

Wir werden zunächst eine explizite Instanz wählen. Laut Voraussetzung erreicht A auf allen Instanzen einen $(r-\epsilon)$-kompetitiven Faktor für das Value-Problem mit Partial-Information. Wir werden zeigen, dass A für die gewählte Instanz einen $(r-\epsilon/2)$-kompetitiven Faktor für das Best-Choice-Problem mit Partial-Information erreicht. Zuletzt werden wir argumentieren, dass wir von dem Ergebnis für die explizite Instanz auf alle Instanzen schließen können.

Es sei $\epsilon \in (0, r-1]$ beliebig und σ' eine Instanz mit Eignungen $\{1, \epsilon', \frac{\epsilon'}{2}, \frac{\epsilon'}{3}, \ldots, \frac{\epsilon'}{n-1}\}$ mit $\epsilon' := \frac{\epsilon/2}{(r-\epsilon)(r-\epsilon/2)}$. Es gilt zu beachten, dass die Kandidateneignungen in σ' im Partial-Information-Szenario nicht präsentiert werden, sondern nur eine Rangfolge der bisher gesehenen Kandidaten. Da A ein $(r-\epsilon)$-kompetitiver Online-Algorithmus für das Value-Problem ist, gilt insbesondere auf der Instanz σ', dass

$$\mathrm{OPT}^V(\sigma') \leq (r-\epsilon)\mathbf{E}[p_A^V(\sigma')] = (r-\epsilon)\mathbf{E}[w_A(\sigma')]. \tag{4.2}$$

Abbildung 4.1: Idee des Beweises zu Theorem 4.2: Das erwartete Talent des von A gewählten Kandidaten ist mindestens $1/(r-\epsilon)$ und dieser ist auf σ' größer als das Talent ϵ' des zweitbesten Kandidaten.

Der beste Bewerber auf σ' hat ein Talent von $w_1 = 1$, da für die Eignung des zweitbesten Kandidaten

$$\epsilon' = \frac{\epsilon/2}{(r-\epsilon)(r-\epsilon/2)} \leq \frac{(r-1)/2}{(r-\epsilon)(r-(r-1)/2)} = \frac{1}{r-\epsilon} \cdot \frac{(r-1)/2}{(r+1)/2} < \frac{1}{r-\epsilon} < 1$$

gilt.

Die Begründung, dass Algorithmus A auf σ' den besten Kandidaten mit einer höheren Wahrscheinlichkeit als $1/r$ erhält basiert auf folgender Idee, die in Abbildung 4.1 veranschaulicht wurde: Da laut Voraussetzung A ein $(r-\epsilon)$-kompetitiver Algorithmus für das Value-Problem ist, erhält er auf σ' im Erwartungswert einen Kandidaten mit Talent von mindestens $\mathbf{E}[w_A(\sigma')] \geq 1/(r-\epsilon)$. Da das Talent des zweitbesten Kandidaten kleiner als dieser Wert ist, muss A mit einer hinreichend großen Wahrscheinlichkeit den besten Kandidaten erhalten, damit die Voraussetzung weiterhin erfüllt ist. Wir leiten im nächsten Schritt formal eine untere Schranke für diese Wahrscheinlichkeit her.

Für den Profit des optimalen Offline-Algorithmus auf σ' gilt $\mathrm{OPT}^V(\sigma') = 1$. Zur Vereinfachung der Notation setzen wir $p := \mathbf{P}[w_A(\sigma') = w_1]$. Somit erhalten wir

$$\begin{aligned} 1 = \mathrm{OPT}^V(\sigma') &\overset{(4.2)}{\leq} (r-\epsilon)\mathbf{E}[w_A(\sigma')] \leq (r-\epsilon)\left(p \cdot w_1 + (1-p)\, w_2\right) \\ &= (r-\epsilon)\left(p \cdot 1 + (1-p)\,\epsilon'\right) \leq (r-\epsilon)\left(p + \epsilon'\right). \end{aligned}$$

Wir formen die erhaltene Ungleichung nach p um und leiten eine untere Schranke her.

$$p \geq \frac{1}{r-\epsilon} - \epsilon' = \frac{1}{r-\epsilon} - \frac{\epsilon/2}{(r-\epsilon)(r-\epsilon/2)} = \frac{r-\epsilon}{(r-\epsilon)(r-\epsilon/2)} = \frac{1}{r-\epsilon/2}.$$

Gemäß Lemma 3.3 gilt $\mathbf{E}[p_A^B(\sigma')] = p$, woraus

$$\mathrm{OPT}^B(\sigma') \leq \left(r - \frac{\epsilon}{2}\right) \mathbf{E}[p_A^B(\sigma')]$$

folgt. Wir haben gezeigt, dass A auf der Instanz σ' ein $(r - \epsilon/2)$-kompetitiver Online-Algorithmus ist. Wir müssen begründen, dass dieser kompetitive Faktor ebenfalls für alle Instanzen gilt. Die Eignungen der Bewerber in σ' können beliebig verändert werden, solange weiterhin die Rangfolge der Bewerber auf σ' beibehalten wird. Wir bezeichnen die hieraus entstehende Instanz als σ. Aufgrund der unveränderten Rangfolge sind die Eingaben σ und σ' im Partial-Information-Szenario identisch. Somit wird A auf σ die gleiche Entscheidung treffen wie zuvor auf σ' und insbesondere einen $(r-\epsilon/2)$-kompetitiven Faktor für das Best-Choice-Problem erreichen. □

Die Aussage aus Theorem 4.2 für das Full-Information-Szenario zu zeigen, hat sich als schwieriger erwiesen. Aus diesem Grund halten wir diese Aussage als Vermutung fest.

Vermutung 4.3. *Es sei A ein strikt r-kompetitiver Online-Algorithmus für das Value-Problem mit Full-Information. Dann existiert ein strikt r-kompetitiver Online-Algorithmus für das Best-Choice-Problem mit Full-Information.*

Wir werden stattdessen eine schwächere Aussage zeigen, indem wir die Eignungen der Kandidaten auf nicht-negative ganze Zahlen beschränken.

Theorem 4.4. *Seien $\sigma_i \in \mathbb{N}_0$ mit $1 \leq i \leq n$ und A ein strikt $(r-\epsilon)$-kompetitiver Online-Algorithmus für das Value-Problem mit Full-Information für ein beliebiges $\epsilon \in (0, r-1]$. Dann existiert ein strikt $(r - \epsilon/2)$-kompetitiver Online-Algorithmus für das Best-Choice-Problem mit Full-Information.*

Beweis. Sei A ein $(r-\epsilon)$-kompetitiver Online-Algorithmus für das Value-Problem für ein beliebiges $\epsilon \in (0, r-1]$. Dann gilt, dass

$$\forall \epsilon \in (0, r-1] : \forall \sigma \in \mathcal{I}_\Pi : \mathrm{OPT}^V(\sigma) \leq (r-\epsilon)\mathbf{E}[p_A^V(\sigma)]. \tag{4.3}$$

Es ist zu zeigen, dass ein Online-Algorithmus A' existiert, für den

$$\forall \epsilon \in (0, r-1] : \forall \sigma \in \mathcal{I}_\Pi : \mathrm{OPT}^B(\sigma) \leq \left(r - \frac{\epsilon}{2}\right) \mathbf{E}[p_{A'}^B(\sigma)]$$

gilt. Folgende Terminologie führen wir für den Beweis ein. Für eine Eingabe σ und eine Funktion $f : \mathbb{N}_0 \to \mathbb{N}_0$ definieren wir die Eingabe $f(\sigma) := (f(\sigma_1), \ldots, f(\sigma_n))$. Der Beweis kann in zwei Schritte unterteilt werden.

Schritt 1: Konstruktion von A'
Wir werden einen Algorithmus A' für das Best-Choice-Problem konstruieren, der

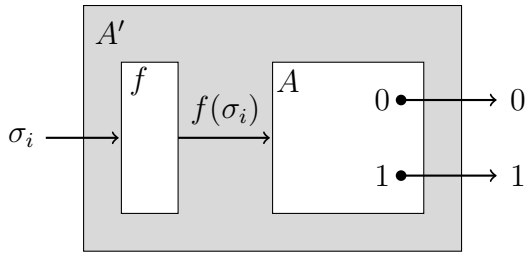

Abbildung 4.2: Funktionsweise von Algorithmus A': Die ankommenden Eignungen σ_i werden von A' zu $f(\sigma_i)$ verändert und Algorithmus A übergeben. Falls A einen Kandidaten ablehnt (0), lehnt A' diesen ebenfalls ab (0) und analog gilt dies auch bei Akzeptanz (1) eines Kandidaten.

Algorithmus A ausnutzen wird. Sei hierzu eine Funktion $f : \mathbb{N}_0 \to \mathbb{N}_0$ durch $f(x) = D^x$ mit $D := \left\lceil \frac{(r-\epsilon)(r-\epsilon/2)}{\epsilon/2} \right\rceil$ definiert. Für $D < 1$ ist f streng monoton fallend, da

$$f(x+1) = D^{x+1} = D^x \cdot D < D^x = f(x).$$

Für $D > 1$ ist f streng monoton steigend, da

$$f(x+1) = D^{x+1} = D^x \cdot D > D^x = f(x).$$

Gemäß unserer Wahl von D ergibt sich

$$D = \left\lceil \frac{(r-\epsilon)(r-\epsilon/2)}{\epsilon/2} \right\rceil \geq \left\lceil \frac{(r-(r-1))(r-(r-1)/2)}{(r-1)/2} \right\rceil = \left\lceil \frac{r+1}{r-1} \right\rceil > 1.$$

Somit folgt, dass f eine streng monoton steigende Funktion ist.

Mit Hilfe der Funktion f kann im Folgenden Algorithmus A' definiert werden.

```
A'(σ = (σ_1, ..., σ_n))
 1: for each ( Bewerber i ∈ {1, ..., n} mit Eignung σ_i)
 2:     Übergebe f(σ_i) an A;
 3:     if ( A akzeptiert Bewerber i)
 4:         Akzeptiere Bewerber i;
```

Die Funktionsweise von Algorithmus A' wird zusätzlich in Abbildung 4.2 dargestellt.

Schritt 2: Bestimmung des kompetitiven Faktors von A'

Laut Voraussetzung ist A ein Algorithmus für das Value-Problem. Im Folgenden interpretieren wir A als einen Algorithmus für das Best-Choice-Problem. Die Bestimmung des kompetitiven Faktors von A' auf der Eingabe σ für das Best-Choice-Problem reduziert sich auf die Bestimmung des kompetitiven Faktors von A auf der Eingabe $f(\sigma)$ für das Best-Choice-Problem. Dies ergibt sich aus

$$\mathbf{E}[p^B_{A'}(\sigma)] = \mathbf{Pr}[w_{A'}(\sigma) = w_1] = \mathbf{Pr}[w_A(f(\sigma)) = f(w_1)] = \mathbf{E}[p^B_A(f(\sigma))]. \quad (4.4)$$

Die erste und die letzte Gleichung ergeben sich aus Lemma 3.3. Die zweite Gleichung ergibt sich aus der folgenden Argumentation. Algorithmus A' trifft für jedes ankommende σ_i die gleiche Entscheidung wie Algorithmus A auf $f(\sigma_i)$. Da f streng monoton steigend ist, bleibt die Rangfolge der Kandidaten in σ und $f(\sigma)$ unverändert. Insbesondere ist der beste Kandidat in der Eingabe σ in $f(\sigma)$ auch weiterhin der Beste und es gilt $f(w_1) = \mathrm{OPT}^V(f(\sigma))$. Daher reduziert sich die Bestimmung der Wahrscheinlichkeit zum Erhalt des besten Kandidaten für Algorithmus A' auf σ auf das Problem die Wahrscheinlichkeit zum Erhalt des besten Kandidaten für Algorithmus A auf $f(\sigma)$ zu bestimmen, das heißt $\mathbf{Pr}[w_A(\sigma) = w_1] = \mathbf{Pr}[w_{A'}(f(\sigma)) = f(w_1)]$.

Für den weiteren Verlauf dieses Beweises konzentrieren wir uns auf die Bestimmung des kompetitiven Faktors von A auf der Eingabe $f(\sigma)$ für das Best-Choice-Problem. Dieser kann mithilfe des kompetitiven Faktors von A für das Value-Problem bestimmt werden. Zunächst leiten wir eine obere Schranke für den erwarteten Profit von A auf der Eingabe $f(\sigma)$ für das Value-Problem her. Wir setzen $p := \mathbf{Pr}[w_A(f(\sigma)) = f(w_1)]$. Es gilt

$$\mathbf{E}[p^V_A(f(\sigma))] \leq p \cdot f(w_1) + (1-p) \cdot f(w_2) \leq p \cdot f(w_1) + f(w_2). \quad (4.5)$$

Die erste Abschätzung ergibt sich aus folgender Überlegung. Wir erhalten mit einer Wahrscheinlichkeit von p den besten Bewerber und mit der Gegenwahrscheinlichkeit einen Bewerber, dessen Eignung höchstens die des Zweitbesten entspricht. Laut Voraussetzung (4.3) ist der kompetitive Faktor von A für das Value-Problem bekannt und insbesondere gilt dann $\mathrm{OPT}^V(f(\sigma)) \leq (r-\epsilon)\mathbf{E}[p^V_A(f(\sigma))]$. Zusammen mit Abschätzung (4.5) erhalten wir

$$f(w_1) = \mathrm{OPT}^V(f(\sigma)) \overset{(4.3)}{\leq} (r-\epsilon)\mathbf{E}[p^V_A(f(\sigma))] \overset{(4.5)}{\leq} (r-\epsilon)(pf(w_1) + f(w_2)).$$

Diese Ungleichung werden wir nach p umformen und die Definition von f einsetzen.

$$p \geq \frac{1}{r-\epsilon} - \frac{f(w_2)}{f(w_1)} = \frac{1}{r-\epsilon} - \frac{D^{w_2}}{D^{w_1}}.$$

Wir leiten eine untere Schranke für die Wahrscheinlichkeit p her.

$$\begin{aligned} p &\geq \frac{1}{r-\epsilon} - \frac{D^{w_2}}{D^{w_1}} \\ &\geq \frac{1}{r-\epsilon} - \frac{D^{w_2}}{D^{w_2+1}} = \frac{1}{r-\epsilon} - \frac{1}{D} \\ &= \frac{1}{r-\epsilon} - \frac{\epsilon/2}{(r-\epsilon)(r-\epsilon/2)} = \frac{r-\epsilon}{(r-\epsilon)(r-\epsilon/2)} = \frac{1}{r-\epsilon/2}. \end{aligned}$$

Bei der Abschätzung haben wir ausgenutzt, dass die Eignungen σ_i nicht-negative ganze Zahlen sind und somit $w_1 \geq w_2 + 1$ gilt. Zusammenfassend ergibt sich

$$\mathbf{E}[p^B_{A'}(\sigma)] \overset{(4.4)}{=} \mathbf{E}[p^B_A(f(\sigma))] = p \geq \frac{1}{r-\epsilon/2}.$$

Dies impliziert

$$\mathrm{OPT}^B(\sigma) \leq \left(r - \frac{\epsilon}{2}\right) \mathbf{E}[p^B_{A'}(\sigma)]$$

und somit ist A' ein $(r-\epsilon/2)$-kompetitiver Online-Algorithmus für das Best-Choice-Problem. □

Der Beweis von Theorem 4.4 beruht auf der Idee den Abstand zwischen den Eignungen w_1 des erstbesten und w_2 des zweitbesten Kandidaten mit der Funktion f zu vergrößern, sodass $w_2 < 1/r \cdot w_1$. Insbesondere heißt das, dass, abgesehen vom Besten, kein Kandidat existiert, dessen Eignung höchstens um einen Faktor r kleiner ist als der Beste. Allerdings soll laut Voraussetzung die erwartete Eignung vom Algorithmus A akzeptierten Kandidaten gerade diese Bedingung erfüllen. Damit dies weiterhin gewährleistet ist, muss A mit einer besseren Wahrscheinlichkeit als $1/r$ den besten Kandidaten erhalten. Diese Idee wird in Abbildung 4.3 illustriert.

Damit die Funktion f die beiden Talente w_1 und w_2 um einen Faktor von mehr als $1/r$ voneinander entfernen kann, war für den Beweis dieser Aussage die Einschränkung der Eignungen der Kandidaten auf die natürlichen Zahlen von Bedeutung. Ohne diese Einschränkung kann für jede von A' gewählte Funktion f, der Gegner eine Instanz konstruieren, sodass die Differenz zwischen w_1 und w_2 hinreichend klein gewählt ist. Durch diese möglichst kleine Distanz zwischen w_1 und w_2 kann der Online-Algorithmus A' mit f die Werte $f(w_1)$ und $f(w_2)$ nicht ausreichend voneinander entfernen, sodass Algorithmus A den Besten nicht mehr mit einer größeren Wahrscheinlichkeit als $1/r$ auswählen kann. Zusätzlich gilt es zu beachten, dass f nicht von den Werten w_1 und w_2 abhängen darf, da diese dem

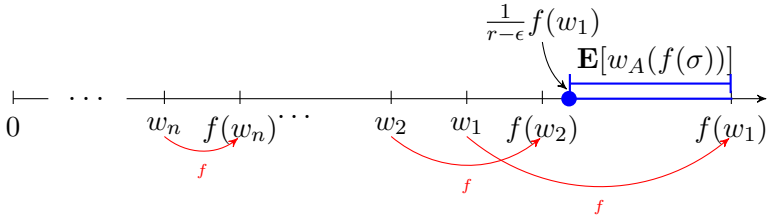

Abbildung 4.3: Idee des Beweises zu Theorem 4.4: Die Funktion f erhöht die Differenz zwischen dem Talent $f(w_2)$ des zweitbesten Kandidaten und das des besten $f(w_1)$ in einer Weise auf, dass $f(w_2)$ um einen Faktor von $r-\epsilon$ kleiner ist als $f(w_1)$. Da das erwartete Talent des von Algorithmus A akzeptierten Kandidaten nicht kleiner ist als ein Faktor von $r - \epsilon$ als $f(w_1)$, kann nur bei Akzeptanz des besten Kandidaten die Voraussetzung weiterhin erfüllt werden.

Online-Algorithmus A' im Vorhinein nicht bekannt sind.

Bevor wir Algorithmen für das Sekretärproblem vorstellen, werden wir den Bezug zwischen den von uns unterschiedenen Informationsszenarien darstellen. Im Full-Information-Szenario werden einem Online-Algorithmus die numerischen Talentwerte der Kandidaten präsentiert. Beim Partial-Information-Szenario werden nicht die numerischen Werte, sondern nur eine Rangfolge der Teilnehmer präsentiert. Daraus folgt, dass nicht jeder beliebige Algorithmus in dem Partial-Information-Szenario eine Entscheidung treffen kann, sondern nur eine bestimmte Menge von Algorithmen. Zu dieser Menge von Algorithmen gehören die *ordnungsbasierten Algorithmen*. Sie treffen Entscheidungen basierend auf der Rangfolge der Kandidaten.

Aus den präsentierten Werten der Kandidaten im Full-Information-Szenario kann ihre Rangfolge abgeleitet werden. Hieraus ergibt sich, dass Algorithmen für das Partial-Information-Szenario auch für das Full-Information-Szenario eingesetzt werden können und insbesondere den gleichen kompetitiven Faktor erreichen.

Korollar 4.5. *Sei A ein r-kompetitiver Online-Algorithmus für das Best-Choice-Problem (Value-Problem) mit Partial-Information, dann ist A ebenfalls ein r-kompetitiver Online-Algorithmus für das Best-Choice-Problem (Value-Problem) mit Full-Information.*

Die Umkehrung dieser Aussage gilt nicht. Man kann weiterhin einen Zusammenhang zwischen der unteren Schranken für das Partial-Information- und Full-Information-Szenario herleiten.

Korollar 4.6. *Für das Best-Choice-Problem (Value-Problem) mit Full-Information sei eine untere Schranke mit kompetitiven Faktor r gegeben. Dann ist r ebenfalls eine untere Schranke für den kompetitiven Faktor für das Best-Choice-Problem (Value-Problem) mit Partial-Information.*

Beweis. Angenommen die Aussage sei falsch. Dann ist r keine untere Schranke für das Partial-Information-Szenario und es existiert ein Algorithmus A, der besser als r-kompetitiv ist. Laut Korollar 4.5 ist A ebenfalls ein r-kompetitiver Online-Algorithmus für das Full-Information-Szenario. Dies ist ein Widerspruch zu unserer Annahme. □

Nach Vorstellung der zwei Zusammenhänge zwischen den Partial- und Full-Information-Szenarien, präsentieren wir im nächsten Schritt zwei Algorithmen für das Sekretärsproblem.

4.2 Algorithmen

In diesem Unterkapitel werden wir zwei Algorithmen für das Sekretärsproblem vorstellen, für die wir eine kompetitive Analyse bezüglich des Best-Choice- und Value-Problems in den Abschnitten 4.3 und 4.4 ausführen werden.

Der Online-Algorithmus UNIFORM$_k$ (U$_k$) wählt einen uniform zufälligen Bewerber k aus der Menge der n Bewerber aus. Zur Vollständigkeit stellen wir den Pseudocode hierzu vor.

UNIFORM$(\sigma = (\sigma_1, \ldots, \sigma_n), k)$
1: Wähle eine Zahl k uniform zufällig aus der Menge $\{1, \ldots, n\}$.
2: Lehne die ersten $k-1$ Bewerber ab;
3: Akzeptiere Bewerber k;

Die Information über die Eignungen der abgelehnten $k-1$ Kandidaten werden bei der Entscheidung zur Wahl des Kandidatens von U$_k$ nicht berücksichtigt.

Wie diese Information genutzt werden kann, zeigt der zweite Online-Algorithmus SECRETARY$_k$ (S$_k$). Dieser lehnt stets die ersten k Bewerber ab und speichert das

größte bisher gesehene Talent $\sigma_{max} := \max\{\sigma_1, \ldots, \sigma_k\}$. Von den verbleibenden $n-k$ Bewerbern wählt S_k den Ersten aus, dessen Eignung besser als σ_{max} ist. Falls dieser nicht existieren sollte, ist er gezwungen den letzten Bewerber zu akzeptieren. Sei $T_k = \min\{i \in \{k+1, \ldots, n\} : \sigma_i > \sigma_{max}\}$, dann lässt sich die Eignung des von S_k ausgewählten Bewerbers als

$$w_{S_k}(\sigma) = \begin{cases} \sigma_i & \text{falls } T_k = i, \\ \sigma_n & \text{falls } T_k \text{ undefiniert ist,} \end{cases}$$

beschreiben. Zur Vollständigkeit geben wir den Algorithmus S_k in Pseudocode an. Sei hierzu $k \in \{1, ..., n\}$ beliebig.

SECRETARY$(\sigma = (\sigma_1, \ldots, \sigma_n), k)$

1: Lehne die ersten k Bewerber ab und speichere das größte Talent σ_{max} ab;
2: **for each** (Bewerber $i \in \{k+1, \ldots, n\}$ mit Eignung σ_i)
3: **if** ($\sigma_i > \sigma_{max}$)
4: Akzeptiere Bewerber i;
5: **return**;
6: Akzeptiere Bewerber n;

Laut dem von uns präsentierten Pseudocode für den Algorithmus SECRETARY$_k$, basiert die Entscheidung zur Wahl eines Kandidaten auf den numerischen Werten der Eignungen und wäre somit nur für das Full-Information-Szenario geeignet. Algorithmus SECRETARY$_k$ kann allerdings auch für das Partial-Information-Szenario angewendet werden. Dies begründet sich dadurch, dass S_k in den Schritten $n-k$ jeweils nur evaluieren muss, ob der ankommende Kandidat der bisher Beste ist. Hierzu genügt die Rangfolge der Bewerber, um eine Entscheidung zu treffen. Der Online-Algorithmus UNIFORM$_k$ ist ebenfalls für das Partial-Information-Szenario geeignet. Gemäß Korollar 4.5 sind beide Algorithmen auch für das Full-Information-Szenario von Relevanz.

Im weiteren Verlauf dieser Arbeit werden wir separat auf die Zielfunktionen, das Best-Choice-Problem und das Value-Problem, eingehen. Hierzu werden wir die folgenden Unterkapiteln zu den Zielfunktionen gemäß Abbildung 1.1 strukturieren. Im ersten Schritt besprechen wir das Partial-Information-Szenario und im zweiten Schritt das Full-Information-Szenario. Für jedes Informationsszenario werden wir die Algorithmen UNIFORM$_k$ und SECRETARY$_k$ bezüglich ihres kompetitiven Faktors analysieren. Als Nächstes wenden wir uns der unteren Schranke für das jeweilige Szenario zu.

4.3 Best-Choice-Problem

4.3.1 Partial-Information

Wir beginnen mit der Diskussion des Best-Choice-Problems im Partial-Information-Szenario. Hierzu werden wir zunächst die Algorithmen aus Abschnitt 4.2 bezüglich ihres kompetitiven Faktors für das Partial-Information-Szenario analysieren. Eine kurze Analyse des Algorithmus UNIFORM$_k$ (U$_k$) zeigt, dass für wachsendes n der Algorithmus UNIFORM$_k$ nicht kompetitiv ist.

Theorem 4.7. *Der Algorithmus UNIFORM$_k$ erreicht keinen konstanten kompetitiven Faktor für das Best-Choice-Problem mit Partial-Information.*

Beweis. Wir werden untersuchen mit welcher Wahrscheinlichkeit U$_k$ den besten Bewerber erhalten wird. Da aus der Menge der n Bewerber ein uniform zufälliger Kandidat ausgewählt wird, ist die Wahrscheinlichkeit, dass dieser der Beste ist, $1/n$. Der Grund hierfür liegt darin, dass gemäß dem Random-Order-Modell jede der $n!$ Reihenfolgen mit gleicher Wahrscheinlichkeit eintreten können. Nur in $(n-1)!$ dieser Reihenfolgen befindet sich der beste Bewerber an Position k. Dies sind die für den Algorithmus U$_k$ relevanten Reihenfolgen, weil für diese der beste Kandidat ausgewählt wird. Die Wahrscheinlichkeit, dass eine dieser Reihenfolgen eintrifft, ist gerade $(n-1!)/n! = 1/n$. □

Im nächsten Schritt analysieren wir den kompetitiven Faktor des Algorithmus SECRETARY$_k$. Es stellt sich die Frage, ob die abgespeicherte Information über die bisher beste Eignung, SECRETARY$_k$ gegenüber UNIFORM$_k$ einen Vorteil verschafft. Das folgende Theorem, das erstmals von Dynkin [14] und Lindley [27] bewiesen wurde, bietet hierauf eine Antwort. Der folgende Beweis orientiert sich an der Arbeit [19] von Gilbert und Mosteller.

Theorem 4.8. *Der Online-Algorithmus SECRETARY$_k$ mit $k = \lfloor n/e \rfloor$ ist ein strikt $c(n)$-kompetitiver Algorithmus für das Best-Choice-Problem mit Partial-Information, wobei $c(n) = \frac{en}{n-e}$ für wachsendes n gegen e konvergiert.*

Beweis. Sei $k \in \{1, \ldots, n\}$ die Anzahl der ersten Bewerber, die abgelehnt werden. Wir wollen die Wahrscheinlichkeit ermitteln, mit der S_k den besten Bewerber erhält. Dazu definieren wir zunächst das Ereignis

$$\mathcal{A}_i = w_1 \text{ befindet sich an Position } i.$$

Mit Hilfe dieses Ereignisses erhält man

$$\begin{aligned}\mathbf{E}[p^B_{S_k}(\sigma)] &\overset{(1.1)}{=} \mathbf{Pr}\left[w_{S_k}(\sigma) = w_1\right] \overset{(1.2)}{=} \sum_{i=1}^{n} \mathbf{Pr}\left[w_{S_k}(\sigma) = w_1 | \mathcal{A}_i\right] \cdot \mathbf{Pr}\left[\mathcal{A}_i\right] \\ &\overset{(1.3)}{=} \sum_{i=1}^{n} \mathbf{Pr}\left[w_{S_k}(\sigma) = w_1 | \mathcal{A}_i\right] \cdot \frac{1}{n} \overset{(1.4)}{=} \frac{1}{n} \cdot \left(\sum_{i=1}^{k} 0 + \sum_{i=k+1}^{n} \frac{k}{i-1} \right) \\ &\overset{(1.5)}{\geq} \frac{k}{n} \sum_{i=k+1}^{n} \frac{1}{i} \overset{(1.6)}{\geq} \frac{k}{n} \left(\ln(n) - \ln(k)\right) = \frac{k}{n} \ln\left(\frac{n}{k}\right).\end{aligned}$$

Entsprechend der Nummerierungen werden die Umformungen im Folgenden erklärt.

(1.1) Die Gleichung folgt aus Lemma 3.3.

(1.2) Da die Ereignisse $\mathcal{A}_i$ eine Partition von Ω sind, folgt die Gleichung aus dem Gesetz der totalen Wahrscheinlichkeit.

(1.3) Da die Bewerber in einer uniform zufälligen Reihenfolge ankommen, befindet sich der beste Bewerber mit einer Wahrscheinlichkeit von $1/n$ an Position i.

(1.4) Bestimmen von $\mathbf{Pr}\left[w^B_{S_k}(\sigma) = w_1 | \mathcal{A}_i\right]$.

- **1.Fall:** Sei $i \in \{1, \ldots, k\}$.
  In diesem Fall ist der beste Kandidat unter den ersten k Kandidaten, die der Algorithmus S_k ablehnt. Es folgt $\mathbf{Pr}\left[w^B_{S_k}(\sigma) = w_1 | \mathcal{A}_i\right] = 0$, da S_k in solch einem Fall den besten Kandidaten stets ablehnen wird.
- **2.Fall:** Sei $i \in \{k+1, \ldots, n\}$.
  In diesem Fall befindet sich der beste Kandidat auf einer Position i zwischen $k+1$ und n. S_k wird einen Bewerber laut Definition an Position i auswählen, wenn in dem Intervall vom $k+1$-ten bis zum $i-1$-ten Bewerber, kein Bewerber existiert, der ein größeres Talent als σ_{max} besitzt. Dies tritt ein, wenn sich der beste Kandidat der ersten $i-1$ Kandidaten unter den ersten $k < i$ abgelehnten Kandidaten befindet. Die Wahrscheinlichkeit, dass sich der Beste der ersten $i-1$ Kandidaten unter den ersten k Kandidaten befindet, beträgt $k/(i-1)$. Somit folgt $\mathbf{Pr}\left[w^B_{S_k}(\sigma) = w_1 | \mathcal{A}_i\right] = \frac{k}{i-1}$.

(1.5) Diese Abschätzung ergibt sich aus

$$\frac{1}{n}\sum_{i=k+1}^{n}\frac{k}{i-1} = \frac{k}{n}\sum_{i=k+1}^{n}\frac{1}{i-1} = \frac{k}{n}\sum_{i=k}^{n-1}\frac{1}{i} \geq \frac{k}{n}\left(\underbrace{\left(-\frac{1}{k}+\frac{1}{n}\right)}_{\leq 0,\text{ da } k\leq n.} + \frac{1}{k} + \sum_{i=k+1}^{n-1}\frac{1}{i}\right)$$
$$\geq \frac{k}{n}\sum_{i=k+1}^{n}\frac{1}{i}.$$

(1.6) Hier wurde die Summe in zwei Summen unterteilt und jeweils die Abschätzung für die harmonische Reihe ausgenutzt.

$$\frac{k}{n}\sum_{i=k+1}^{n}\frac{1}{i} = \frac{k}{n}\left(\sum_{i=1}^{n}\frac{1}{i} - \sum_{i=1}^{k}\frac{1}{i}\right) \geq \frac{k}{n}\left(\ln(n) - \ln(k)\right).$$

Die Wahrscheinlichkeit, mit der S_k den besten Bewerber auswählt, ist zusammenfassend nach unten beschränkt durch

$$\mathbf{E}[p^B_{S_k}(\sigma)] \geq \frac{\ln(n/k)}{n/k}. \tag{4.6}$$

Diesen Term gilt es zu maximieren. Hierzu stellen wir den Term als eine Funktion f in Abhängigkeit von n/k dar. Zur Erleichterung setzen wir $x := n/k$ und erhalten $f(x) = \frac{\ln(x)}{x}$. Wir bestimmen zunächst den Punkt x, an dem die Funktion f ihr Maximum annimmt.

$$f'(x) = \frac{\frac{1}{x}\cdot x - \ln(x)}{x^2} = 0 \iff x = e \implies \frac{n}{k} = e \implies k = \left\lfloor \frac{n}{e} \right\rfloor.$$

Für $k = \lfloor n/e \rfloor$ wird die Wahrscheinlichkeit, dass S_k den besten Bewerber wählt, maximiert. Dies lässt sich in Abbildung 4.4 erkennen. Der maximale Funktionswert beträgt $1/e$. Es gilt die Wahrscheinlichkeit zu bestimmen, mit der $S_{\lfloor n/e \rfloor}$ den besten Bewerber auswählt. Wir setzen den erhaltenen Wert für k in (4.6) ein und erhalten

$$\mathbf{E}[p^B_{S_k}(\sigma)] = \mathbf{Pr}[w^B_{S_k}(\sigma) = w_1] \geq \frac{\lfloor n/e \rfloor}{n}\cdot\ln\left(\frac{n}{\lfloor n/e \rfloor}\right) \geq \frac{n/e-1}{n}\cdot\ln\left(\frac{n}{n/e}\right)$$
$$= \frac{1}{e} - \frac{1}{n} = \frac{n-e}{en} = \frac{1}{c(n)} = \frac{1}{c(n)}\cdot \mathrm{OPT}^B(\sigma).$$

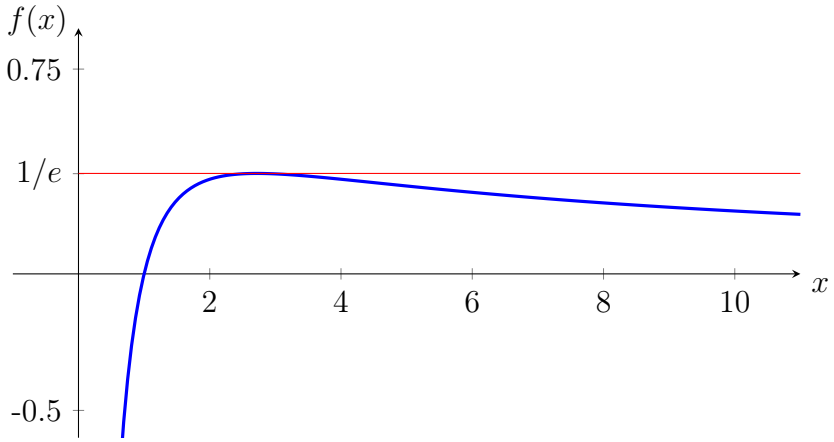

Abbildung 4.4: Der Graph der Funktion $f(x) = \frac{\ln(x)}{x}$ in blau. Der maximale Funktionswert von f ist mit einer roten Linie markiert.

Dies ist äquivalent zu

$$\mathrm{OPT}^B(\sigma) \leq c(n) \cdot \mathbf{E}[p^B_{S_k}(\sigma)].$$

Da $c(n)$ für wachsendes n gegen e konvergiert, ist S_k ein e-kompetitiver Online-Algorithmus für wachsendes n. Somit ist unsere Behauptung bewiesen. □

Es konnte gezeigt werden, dass $\mathrm{SECRETARY}_k$ ein e-kompetitiver Online-Algorithmus für das Best-Choice-Problem mit Partial-Information ist. Eine natürliche Frage ist, ob es besser geht. Wir werden uns im Folgenden daher mit der unteren Schranke für das Best-Choice-Problem mit Partial-Information beschäftigen. Es stellt sich heraus, dass kein besserer kompetitiver Algorithmus als $\mathrm{SECRETARY}_k$ existieren kann. Somit ist e eine untere Schranke für den kompetitiven Faktor für das Best-Choice-Problem mit Partial-Information. Diese Aussage wurde schon auf verschiedene Weisen in der Literatur bewiesen. Wir konzentrieren uns auf den Beweis von Buchbinder [11] und orientieren uns an [33].

Theorem 4.9. *Es gibt keinen deterministischen oder randomisierten Online-Algorithmus für das Best-Choice-Problem mit Partial-Information für $n > 2$, der besser als e-kompetitiv ist.*

Beweis. Sei A ein beliebiger deterministischer oder randomisierter Online-Algorithmus für das Best-Choice-Problem mit Partial-Information. Wir können uns auf ordnungsbasierte Algorithmen beschränken, da diese das bestmögliche Ergebnis für das Best-Choice-Problem mit Partial-Information erzielen. Nicht-ordnungsbasierte Algorithmen für das Best-Choice-Problem mit Partial-Information nutzen

nicht oder nur teilweise die Information über die Rangfolge der Kandidaten aus und können kein besseres Ergebnis als ordnungsbasierte Algorithmen erreichen. Für den weiteren Verlauf des Beweises werden wir daher A als einen ordnungsbasierten Algorithmus bezeichnen. Wir werden einen solchen Online-Algorithmus A charakterisieren und zeigen, dass die Wahrscheinlichkeit den Besten auszuwählen nach oben durch $1/e$ beschränkt ist. Gemäß Theorem 3.4 hätten wir somit bewiesen, dass ein kompetitiver Faktor von e eine untere Schranke für das Best-Choice-Problem mit Partial-Information impliziert. Der Beweis kann in folgende Schritte unterteilt werden.

1. Es werden Nebenbedingungen hergeleitet, die ein solcher Online-Algorithmus A erfüllen muss.
2. Wir definieren die Zielfunktion, den besten Bewerber auszuwählen.
3. Wir beschreiben das lineare Programm, dass sich aus den beiden vorherigen Schritten ergibt.
4. Wir bestimmen eine gültige Lösung und den Wert des linearen Programms.
5. Wir zeigen, dass der Wert des linearen Programms für wachsendes n gegen $1/e$ konvergiert.

Zur weiteren Analyse definieren wir die Ereignisse

$$\mathcal{A}_i := \text{Algorithmus } A \text{ wählt Bewerber an Position } i \text{ aus,}$$
$$\mathcal{B}_i := \text{Bewerber an Position } i \text{ ist besser als alle Vorgänger.}$$

Entsprechend setzen wir $p_i := \mathbf{Pr}[\mathcal{A}_i]$.

Schritt 1: Bestimmung der Nebenbedingungen
Es ergibt sich mit den von uns oben definierten Ereignissen

$$\begin{aligned} p_i &= \mathbf{Pr}[\mathcal{A}_i] \overset{(1.1)}{=} \mathbf{Pr}[\mathcal{A}_i \cap \mathcal{B}_i] \overset{(1.2)}{=} \mathbf{Pr}[\mathcal{A}_i|\mathcal{B}_i] \cdot \mathbf{Pr}[\mathcal{B}_i] \\ &\overset{(1.3)}{\leq} \mathbf{Pr}[\neg(\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_{i-1})|\mathcal{B}_i] \cdot \mathbf{Pr}[\mathcal{B}_i] \\ &\overset{(1.4)}{=} \mathbf{Pr}[\neg(\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_{i-1})] \cdot \frac{1}{i} \\ &\overset{(1.5)}{=} \left(1 - \sum_{j=1}^{i-1} \mathbf{Pr}[A_j]\right) \cdot \frac{1}{i} \\ &= \left(1 - \sum_{j=1}^{i-1} p_j\right) \cdot \frac{1}{i}. \end{aligned}$$

Daraus ergibt sich

$$\forall i \in \{1, \ldots, n\} : ip_i + \sum_{j=1}^{i-1} p_j \leq 1. \tag{4.7}$$

In der obigen Abschätzung wurden einige Charakteristika des Algorithmus ausgenutzt, die mit entsprechender Nummerierung hier erklärt werden.

(1.1) Man kann ohne Beschränkung der Allgemeinheit davon ausgehen, dass A nur einen Bewerber auswählt, wenn dieser der bis dato Beste ist. Wäre dies nicht der Fall, so wäre der Bewerber unter den bisher gesehenen schon nicht der Beste und die Wahrscheinlichkeit den Besten insgesamt zu erhalten würde sinken.

(1.2) Dies ergibt sich aus der Definition der bedingten Wahrscheinlichkeit.

(1.3) Hier wird $\mathcal{A}_i$ nach oben abgeschätzt. Wenn A einen Bewerber an Position i akzeptiert, muss dieser alle vorherigen Bewerber $1, \ldots, i-1$ abgelehnt haben. Die Rückrichtung gilt jedoch nicht. Wenn A Bewerber $1, \ldots, i-1$ ablehnt, wird A nicht zwangsweise Bewerber i wählen, da die Bewerber an Positionen $i+1, \ldots, n$ weiterhin in Frage kommen. Mit dieser Argumentation gilt

$$\neg\mathcal{A}_1 \cap \cdots \cap \neg\mathcal{A}_{i-1} = \mathcal{A}_i \cup \cdots \cup \mathcal{A}_n$$

und insbesondere gilt

$$\mathcal{A}_i \subseteq \neg\mathcal{A}_1 \cap \cdots \cap \neg\mathcal{A}_{i-1} = \neg\left(\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_{i-1}\right).$$

Gleichheit gilt für $i = n$.

(1.4) Hier wurden zwei Aspekte ausgenutzt. Da ein A ein Online-Algorithmus ist, ist ihm zum Zeitpunkt der Ablehnung der Kandidaten 1 bis $i-1$ nicht bekannt, ob der Kandidat an Position i besser als seine Vorgänger ist. Somit kann $\mathcal{B}_i$ vernachlässigt werden. Zweitens beträgt die Wahrscheinlichkeit, dass Bewerber i der Beste unter den ersten i Bewerber ist, genau $1/i$.

(1.5) Die Ereignisse $\mathcal{A}_1, \ldots, \mathcal{A}_n$ sind paarweise disjunkt. Angenommen dies wäre nicht der Fall, so folgt, dass Algorithmus A zwei Kandidaten ausgewählt haben muss. Das wäre ein Widerspruch zur Definition des Sekretärsproblem, dass nur ein einziger Kandidaten ausgewählt werden darf. Aufgrund der Disjunktheit lassen sich die Wahrscheinlichkeiten der $\mathcal{A}_i$ aufsummieren.

Gemäß der Definition von Wahrscheinlichkeiten gilt $p_i \geq 0$. Mit der Bedingung aus (4.7) erhält man die folgenden Bedingungen für die Werte $p_1, \ldots, p_n$:

$$\begin{aligned} &\forall i \in \{1, \ldots, n\} : ip_i + \sum_{j=1}^{i-1} p_j \leq 1, \\ &\forall i \in \{1, \ldots, n\} : p_i \geq 0. \end{aligned} \tag{4.8}$$

Schritt 2: Bestimmung der Zielfunktion
Unser Ziel ist es, die Wahrscheinlichkeit, den besten Bewerber auszuwählen, zu maximieren. Wir werden diese Ziel mithilfe der Werte p_i ausdrücken. Dazu definieren wir das Ereignis

$$\mathcal{C}_i := \text{Bester Bewerber befindet sich an Position } i.$$

Sei $\sigma \in \mathcal{I}_\Pi$ beliebig, dann folgt

$$\begin{aligned} \mathbf{E}[p_A^B(\sigma)] &= \mathbf{Pr}\left[w_A(\sigma) = w_1\right] \\ &\overset{(2.1)}{=} \sum_{i=1}^{n} \mathbf{Pr}[\mathcal{A}_i \cap \mathcal{C}_i] \overset{(1.2)}{=} \sum_{i=1}^{n} \mathbf{Pr}[\mathcal{A}_i|\mathcal{C}_i] \cdot \mathbf{Pr}[\mathcal{C}_i] \overset{(2.2)}{=} \sum_{i=1}^{n} \mathbf{Pr}[\mathcal{A}_i|\mathcal{B}_i] \cdot \frac{1}{n} \\ &\overset{(1.2)}{=} \sum_{i=1}^{n} \frac{\mathbf{Pr}[\mathcal{A}_i \cap \mathcal{B}_i]}{\mathbf{Pr}[B_i]} \cdot \frac{1}{n} \overset{(1.1)}{=} \sum_{i=1}^{n} \frac{\mathbf{Pr}[\mathcal{A}_i]}{\mathbf{Pr}[B_i]} \cdot \frac{1}{n} = \sum_{i=1}^{n} \frac{p_i}{\frac{1}{i}} \cdot \frac{1}{n} = \sum_{i=1}^{n} \frac{i}{n} \cdot p_i. \end{aligned}$$

Auch hier werden die Gleichungen entsprechend ihrer Nummerierung erläutert:

(2.1) A erhält den besten Bewerber, wenn der beste Bewerber sich an Position i befindet und A den Bewerber an Position i auswählt.

(2.2) Da A ein Online-Algorithmus ist, kann A nicht den Wert des besten Bewerbers kennen. Somit ist der bisher beste Bewerber gleichzeitig auch der insgesamt beste Kandidat für Algorithmus A. Somit kann $\mathcal{C}_i$ durch $\mathcal{B}_i$ ersetzt werden. Die Wahrscheinlichkeit, dass sich der Beste aus der Menge aller Bewerber an Position i befindet, beträgt $1/n$.

Für das Best-Choice-Problem gilt es somit $\mathbf{E}[p_A^B(\sigma)] = \sum_{i=1}^{n} \frac{i}{n} p_i$ zu maximieren.

Schritt 3: Bestimmung des linearen Programms
Die Nebenbedingungen aus (4.8) zusammen mit der hergeleiteten Zielfunktion in Schritt 2 charakterisieren vollständig einen Online-Algorithmus A, mit dem Ziel,

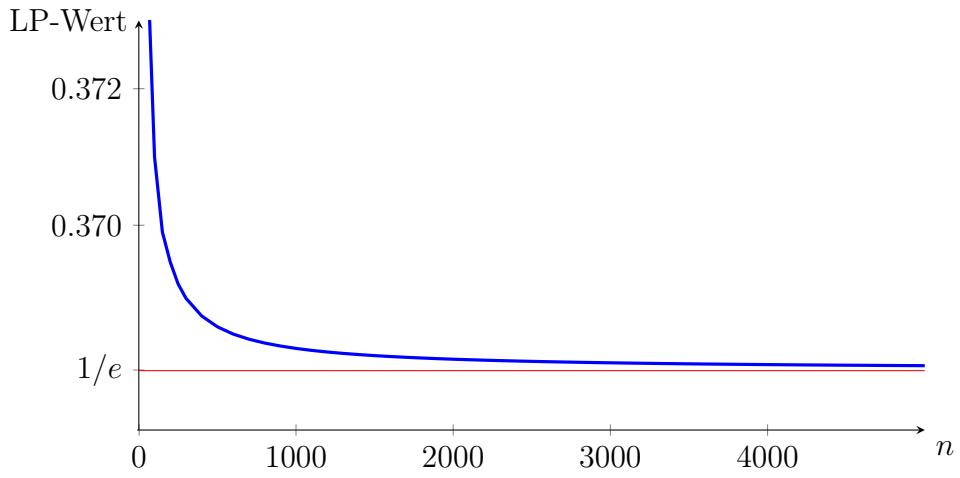

Abbildung 4.5: Der Graph des LP aus (4.9) wurde für verschiedene n mithilfe eines LP-Solvers ausgewertet. Die Werte können in Tabelle A.1 nachgelesen werden. Der daraus resultierende Graph ist blau dargestellt und es ist erkennbar, dass er sich dem Wert $1/e$ nähert. Eine detailliertere Beschreibung zur Verwendung eines LP-Solvers wird im Anhang A erklärt.

die Wahrscheinlichkeit, den besten Bewerber zu erhalten, zu maximieren. Man erhält somit folgendes lineares Programm (LP).

$$\begin{aligned} &\max \sum_{i=1}^{n} \frac{i}{n} \cdot p_i \text{ mit} \\ &\forall i \in \{1, \ldots, n\} : \sum_{j=1}^{i-1} p_j + i p_i \leq 1, \\ &\forall i \in \{1, \ldots, n\} : p_i \geq 0. \end{aligned} \tag{4.9}$$

In Abbildung 4.5 wurde der Wert des linearen Programmes 4.9 für explizite n graphisch dargestellt. Anhand dieser Abbildung ist zu erkennen, dass der Wert des LPs gegen $1/e$ konvergiert.

Um dies formal zu zeigen, werden wir eine Technik aus der *linearen Optimierung* nutzen, das *Dualitätsprinzip* [22]: Sei ein *lineares Programm* gegeben mit

$$\max\{c^T x : Ax \leq b, x \geq 0\}, \tag{P}$$

dann definieren wir das *duale LP* als lineares Programm

$$\min\{y^T b : y^T A \geq c^T, y \geq 0\}. \tag{D}$$

Hierbei bezeichnet (P) das *primale LP*. Nach dem *schwachen Dualitätssatz* gilt

$$\max\{c^T x : Ax \leq b, x \geq 0\} \leq \min\{y^T b : y^T A \geq c^T, y \geq 0\}.$$

Somit ist (D) eine obere Schranke für den Wert von (P). Der Beweis hierzu kann in [22] nachgelesen werden.
Diese Erkenntnis werden wir im Folgenden für unser lineares Programm anwenden. Da wir an einer oberen Schranke, für die Wahrscheinlichkeit den besten Kandidaten zu erhalten, interessiert sind, reicht es aus (D) zu lösen. Man kann das in (4.9) formulierte lineare Programm als

$$\max\{c^T p : Ap \leq b, p \geq 0\}$$

mit

$$c^T := \frac{1}{n}\begin{pmatrix}1 & 2 & \dots & n\end{pmatrix},\ p := \begin{pmatrix}p_1 & p_2 & \dots & p_n\end{pmatrix}^T,\ b := \begin{pmatrix}1 & 1 & \dots & 1\end{pmatrix}^T \text{ und}$$

$$A := \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 2 & 0 & \dots & 0 \\ 1 & 1 & 3 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 1 & \dots & \dots & 1 & n \end{pmatrix} \in \mathbb{N}_0^{n \times n}$$

interpretieren. Gemäß dem Dualitätsprinzip erhalten wir als obere Schranke

$$\min\{x^T b : x^T A \geq c^T, x \geq 0\} = \min\{x^T b : A^T x \geq c, x \geq 0\}. \tag{4.10}$$

Hierbei bezeichnen wir $A^T \in \mathbb{N}_0^{n \times n}$ als die transponierte Matrix von A. Da A eine untere Dreiecksmatrix ist, ergibt sich für die Transponierte eine obere Dreiecksmatrix.

Schritt 4: Bestimmung einer gültigen Lösung des LPs

Wir bestimmen eine gültige Lösung für das duale LP (4.10). A^T ist gegeben durch a_{ij}^T mit

$$a_{ij}^T = \begin{cases} 1 & \text{falls } i < j, \\ i & \text{falls } i = j, \\ 0 & \text{falls } i > j, \end{cases}$$

wobei a_{ij}^T den Eintrag in der i-ten Zeile und j-ten Spalte der Matrix A^T bezeichnet. Zur Bestimmung einer Lösung des LPs werden wir uns zunächst auf die erste

Bedingung $A^T x \geq c$ konzentrieren und eine gültige Lösung für diese Ungleichung bestimmen. Die Zielfunktion $x^T b$ wird genau dann minimal, wenn $A^T x$ minimal wird. Dies ergibt sich aus den nicht-negativen Einträgen von x und A^T. Da $A^T x$ allerdings durch den Vektor c nach unten beschränkt ist, können wir ihn nicht beliebig klein wählen, sondern setzen $A^T x = c$. Somit erhalten wir ein lineares Gleichungssystem und es genügt zur Bestimmung der Lösung die Inverse von A^T zu berechnen. Aus der Linearen Algebra ist bekannt, dass jede obere Dreiecksmatrix mit von 0 verschiedenen Diagonaleinträgen, insbesondere die Matrix A^T, eine Inverse besitzt. Dann ist die Lösung gegeben durch $x = (A^T)^{-1} \cdot c$. Somit gilt es nur noch die Inverse von A^T zu bestimmen. Zur Vereinfachung der Notation setzen wir $A' := (A^T)^{-1}$.

Durch ein paar Rechnungen erhält man für A' die Einträge

$$a'_{ij} = \begin{cases} \frac{-1}{j(j-1)} & \text{falls } i < j \\ \frac{1}{j} & \text{falls } i = j \\ 0 & \text{falls } i > j. \end{cases}$$

Man kann verifizieren, dass dies tatsächlich die Inverse von A^T ist. Hierzu betrachten wir die Fälle $i < j$, $i \neq j$ und $i > j$ separat.

- **1.Fall:** Sei $i < j$.

$$\begin{aligned}
(A^T \cdot A')_{ij} &= \sum_{k=1}^{n} a_{ik} \cdot a'_{kj} \\
&= \underbrace{\sum_{k=1}^{i-1} a^T_{ik} \cdot a'_{kj}}_{0} + a^T_{ii} \cdot a'_{ij} + \sum_{k=i+1}^{j-1} a^T_{ik} \cdot a'_{kj} + a^T_{ij} \cdot a'_{jj} + \underbrace{\sum_{k=j+1}^{n} a^T_{ik} \cdot a'_{kj}}_{0} \\
&= i \cdot \frac{-1}{j(j-1)} + \sum_{k=i+1}^{j-1} \frac{-1}{j(j-1)} + \frac{1}{j} \\
&= \frac{-i - ((j-1) - (i+1) + 1) + j - 1}{j(j-1)} \\
&= \frac{0}{j(j-1)} = 0.
\end{aligned}$$

- **2.Fall:** Sei $i = j$.

$$(A^T \cdot A')_{ii} = \underbrace{\sum_{k=1}^{i-1} a_{ik}^T \cdot a'_{ki}}_{0} + a_{ii}^T \cdot a'_{ii} + \underbrace{\sum_{k=i+1}^{n} a_{ik}^T \cdot a'_{ki}}_{0} = i \cdot \frac{1}{i} = 1.$$

- **3.Fall:** Sei $i > j$.

$$(A^T \cdot A')_{ij} = \sum_{k=1}^{n} a_{ik}^T \cdot a'_{kj} = \underbrace{\sum_{k=1}^{i-1} a_{ik}^T \cdot a'_{kj}}_{0} + \underbrace{\sum_{k=i}^{n} a_{ik}^T \cdot a'_{kj}}_{0} = 0.$$

Für $A' \cdot A^T$ erhält man das gleiche Resultat, welches aus der Kommutativität der nicht-negativen Einträge der Matrizen folgt. Zusammenfassend ergibt sich

$$\left(A^T \cdot A'\right)_{ij} = \left(A' \cdot A^T\right)_{ij} = \left\{ \begin{matrix} 0 & \text{falls } i \neq j \\ 1 & \text{falls } i = j \end{matrix} \right\} = \delta_{i,j}.$$

Damit wurde gezeigt, dass A' die Inverse von A^T ist. Die Lösung des linearen Gleichungssystems kann als Nächstes mittels der Inversen A' berechnet werden. Sei $i \in \{1, \ldots, n\}$, dann erhalten wir für den i-ten Eintrag des Vektors x

$$\begin{aligned} x_i = \sum_{j=1}^{n} a'_{ij} \cdot c_j &= \underbrace{\sum_{j=1}^{i-1} a'_{ij} \cdot c_j}_{0} + a'_{ii} \cdot c_i + \sum_{j=i+1}^{n} a'_{ij} \cdot c_j \\ &= \frac{1}{i} \cdot \frac{i}{n} + \sum_{j=i+1}^{n} \frac{-1}{j(j-1)} \cdot \frac{j}{n} \\ &= \frac{1}{n} \left(1 - \sum_{j=i+1}^{n} \frac{1}{j-1}\right) = \frac{1}{n} \left(1 - \sum_{j=i}^{n-1} \frac{1}{j}\right). \end{aligned}$$

Die daraus resultierende Lösung für x_i ergibt sich durch die erste Bedingung. Dabei haben wir die zweite Bedingung außer Acht gelassen. Damit diese erfüllt ist, muss zusätzlich

$$x_i = \frac{1}{n} \left(1 - \sum_{j=i}^{n-1} \frac{1}{j}\right) \geq 0 \tag{4.11}$$

gelten. Damit dies erfüllt ist, muss $\sum_{j=i}^{n-1} \frac{1}{j} \leq 1$ gelten. Wir definieren ein $\tau \in \mathbb{N}$ als natürliche Zahl, ab welcher die Bedingung erfüllt ist, mit

$$\sum_{j=\tau}^{n-1} \frac{1}{j} < 1 \leq \sum_{j=\tau-1}^{n-1} \frac{1}{j}. \tag{4.12}$$

Somit ergibt sich, dass für alle $i \leq \tau - 1$ der Term $\sum_{j=i}^{n-1} 1/j$ mindestens 1 ist und x_i nicht positiv ist. Dies würde die Bedingung aus (4.11) verletzten und wir setzen daher $x_i = 0$. Für alle $i > \tau$ hat der Term einen Wert kleiner als 1 und somit ist dies eine gültige Lösung für x_i. Zusammenfassend erhalten wir

$$x_i = \begin{cases} 0 & \text{falls } 1 \leq i < \tau, \\ \frac{1}{n}\left(1 - \sum\limits_{j=i}^{n-1} \frac{1}{j}\right) & \text{falls } \tau \leq i \leq n. \end{cases} \tag{4.13}$$

Das duale Programm nimmt durch das Einsetzen der Lösung für x_i aus (4.13) folgenden Wert an.

$$\sum_{i=0}^{n} x_i = \sum_{i=0}^{\tau-1} x_i + \sum_{i=\tau}^{n} x_i \overset{(4.13)}{=} \sum_{i=0}^{\tau-1} 0 + \sum_{i=\tau}^{n} \frac{1}{n}\left(1 - \sum_{j=i}^{n-1} \frac{1}{j}\right) = \frac{\tau - 1}{n} \sum_{j=\tau-1}^{n-1} \frac{1}{j}.$$

Die letzte Gleichung ergibt sich aus den folgenden Umformungen.

$$\begin{aligned} \sum_{i=\tau}^{n} \frac{1}{n}\left(1 - \sum_{j=i}^{n-1} \frac{1}{j}\right) &= \frac{1}{n}\left(\sum_{i=\tau}^{n} 1 - \sum_{i=\tau}^{n} \sum_{j=i}^{n-1} \frac{1}{j}\right) \\ &= \frac{1}{n}\left((n - \tau + 1) - \sum_{j=0}^{n-\tau-1} \frac{j+1}{\tau + j}\right) \\ &= \frac{1}{n}\left(n - \tau + 1 - \sum_{j=\tau}^{n-1} \frac{j + 1 - \tau}{j}\right) \\ &= \frac{1}{n}\left(n - \tau + 1 - \sum_{j=\tau}^{n-1} 1 + (1 - \tau) \sum_{j=\tau}^{n-1} \frac{1}{j}\right) \\ &= \frac{1}{n}\left(1 + (\tau - 1) \sum_{j=\tau}^{n-1} \frac{1}{j}\right) \\ &= \frac{\tau - 1}{n}\left(\frac{1}{\tau - 1} + \sum_{j=\tau}^{n-1} \frac{1}{j}\right) = \frac{\tau - 1}{n} \sum_{j=\tau-1}^{n-1} \frac{1}{j}. \end{aligned}$$

Schritt 5: Berechnung der Konvergenz des Wertes des LP für wachsendes n

Es muss gezeigt werden, dass der Wert $\frac{\tau-1}{n}\sum_{j=\tau-1}^{n-1}\frac{1}{j}$ für wachsendes n gegen $1/e$ konvergiert. Wir werden zunächst sowohl eine untere als auch eine obere Schranke für τ herleiten. Dazu wenden wir folgendes Lemma aus der Analysis [8] an.

Lemma 4.10. *Seien g eine stetig fallende Funktion und $a, b \in \mathbb{N}$. Dann gilt*

$$\int_a^{b+1} g(x)dx \overset{(i)}{\leq} \sum_{i=a}^{b} g(i) \overset{(ii)}{\leq} \int_{a-1}^{b} g(x)dx.$$

Wir wählen $g(x) = 1/x$, dann ist g sowohl stetig, als auch streng monoton fallend für $x > 0$. Damit sind die Voraussetzungen erfüllt und wir können Lemma 4.10 anwenden. Wir beginnen mit der Herleitung einer unteren Schranke für τ.

$$\begin{aligned}
&\int_\tau^n \frac{1}{x}dx = \int_\tau^n g(x)dx \overset{(i)}{\leq} \sum_{i=\tau}^{n-1} g(i) = \sum_{i=\tau}^{n-1}\frac{1}{i} \overset{(4.12)}{<} 1 = \ln(e)\\
\iff &\int_\tau^n \frac{1}{x}dx = [\ln(x)]_\tau^n = \ln(n) - \ln(\tau) < \ln(e)\\
\iff &\ln(\tau) > \ln(n) - \ln(e)\\
\iff &\ln(\tau) > \ln\left(\frac{n}{e}\right)\\
\iff &\tau \geq \left\lceil \frac{n}{e} \right\rceil.
\end{aligned}$$

Die letzte Äquivalenz folgt daraus, dass die Logarithmusfunktion streng monoton wachsend ist. Analog leiten wir eine obere Schranke für τ her.

$$\begin{aligned}
&\int_{\tau-2}^{n-1} \frac{1}{x}dx = \int_{\tau-2}^{n-1} g(x)dx \overset{(ii)}{\geq} \sum_{i=\tau-1}^{n-1} g(i) = \sum_{i=\tau-1}^{n-1}\frac{1}{i} \overset{(4.12)}{\geq} 1 = \ln(e)\\
\iff &\int_{\tau-2}^{n-1} g(x)dx = [\ln(x)]_{\tau-2}^{n-1} = \ln(n-1) - \ln(\tau-2) \geq \ln(e)\\
\iff &\ln(\tau-2) \leq \ln(n-1) - \ln(e)\\
\iff &\ln(\tau-2) \leq \ln\left(\frac{n-1}{e}\right)\\
\iff &\tau \leq \frac{n-1}{e} + 2.
\end{aligned}$$

Wir fassen unser bisheriges Ergebnis mit

$$\left\lceil \frac{n}{e} \right\rceil \leq \tau \leq \frac{n-1}{e} + 2 \tag{4.14}$$

zusammen. Als Nächstes lässt sich die Behauptung zeigen, dass der Wert des LPs gegen $1/e$ konvergiert, das heißt

$$\lim_{n\to\infty} \frac{\tau-1}{n} \sum_{j=\tau-1}^{n-1} \frac{1}{j} = \frac{1}{e}.$$

Hierfür ersetzen wir τ mit der aus (4.14) hergeleiteten unteren und oberen Schranke und erhalten entsprechend eine untere und obere Schranke für den Wert des dualen linearen Programms mit

$$\frac{\lceil \frac{n}{e} \rceil - 1}{n} \sum_{j=\lceil \frac{n}{e} \rceil - 1}^{n-1} \frac{1}{j} \leq \frac{\tau-1}{n} \sum_{j=\tau-1}^{n-1} \frac{1}{j} \leq \frac{\frac{n-1}{e}+1}{n} \sum_{j=\frac{n-1}{e}+1}^{n-1} \frac{1}{j}. \tag{4.15}$$

Wir werden zeigen, dass sowohl die untere, als auch die obere Schranke gegen $1/e$ konvergiert. Mit dem Einschnürungssatz aus der Analysis, folgt dann dass der Wert des dualen LPs gegen $1/e$ konvergiert. Zunächst zeigen wir die Konvergenz der unteren Schranke.

$$\begin{aligned}
\lim_{n\to\infty} \frac{\lceil \frac{n}{e} \rceil - 1}{n} \sum_{j=\lceil \frac{n}{e} \rceil - 1}^{n-1} \frac{1}{j} &\geq \lim_{n\to\infty} \frac{\frac{n}{e} - 1}{n} \sum_{j=\lceil \frac{n}{e} \rceil - 1}^{n-1} \frac{1}{j} = \lim_{n\to\infty} \left(\frac{1}{e} - \frac{1}{n} \right) \lim_{n\to\infty} \sum_{j=\lceil \frac{n}{e} \rceil - 1}^{n-1} \frac{1}{j} \\
&= \frac{1}{e} \lim_{n\to\infty} \sum_{j=\lceil \frac{n}{e} \rceil - 1}^{n-1} \frac{1}{j} \overset{(i)}{\geq} \frac{1}{e} \lim_{n\to\infty} \int_{\lceil \frac{n}{e} \rceil - 1}^{n} \frac{1}{x} dx \\
&= \frac{1}{e} \lim_{n\to\infty} \left(\ln(n) - \ln\left(\left\lceil \frac{n}{e} \right\rceil - 1 \right) \right) \\
&\geq \frac{1}{e} \lim_{n\to\infty} \left(\ln(n) - \ln\left(\frac{n}{e} + 1 - 1 \right) \right) = \frac{1}{e} \ln(e) = \frac{1}{e}.
\end{aligned}$$

Als Nächstes zeigen wir die Konvergenz der oberen Schranke.

$$\begin{aligned}
\lim_{n\to\infty} \frac{\frac{n-1}{e}+1}{n} \sum_{j=\frac{n-1}{e}+1}^{n-1} \frac{1}{j} &= \lim_{n\to\infty} \left(\frac{n}{ne} - \frac{1}{ne} + \frac{1}{n} \right) \lim_{n\to\infty} \sum_{j=\frac{n-1}{e}+1}^{n-1} \frac{1}{j} \\
&= \frac{1}{e} \lim_{n\to\infty} \sum_{j=\frac{n-1}{e}+1}^{n-1} \frac{1}{j} \overset{(ii)}{\leq} \frac{1}{e} \lim_{n\to\infty} \int_{\frac{n-1}{e}}^{n-1} \frac{1}{x} dx \\
&= \frac{1}{e} \lim_{n\to\infty} \left(\ln(n-1) - \ln\left(\frac{n-1}{e} \right) \right) \\
&= \frac{1}{e} \lim_{n\to\infty} \left(\ln\left(\frac{n-1}{n-1} \cdot e \right) \right) = \frac{1}{e}.
\end{aligned}$$

Zusammenfassend erhält man

$$\frac{1}{e} \leq \lim_{n\to\infty} \frac{\left\lceil \frac{n}{e} \right\rceil - 1}{n} \sum_{j=\left\lceil \frac{n}{e} \right\rceil - 1}^{n-1} \frac{1}{j} \leq \lim_{n\to\infty} \frac{\tau - 1}{n} \sum_{j=\tau-1}^{n-1} \frac{1}{j} \leq \lim_{n\to\infty} \frac{\frac{n-1}{e} + 1}{n} \sum_{j=\frac{n-1}{e}+1}^{n-1} \frac{1}{j} \leq \frac{1}{e}$$

und das impliziert

$$\frac{1}{e} \leq \lim_{n\to\infty} \frac{\tau - 1}{n} \sum_{j=\tau-1}^{n-1} \frac{1}{j} \leq \frac{1}{e}.$$

Nach dem Einschnürungssatz gilt somit

$$\lim_{n\to\infty} \frac{\tau - 1}{n} \sum_{j=\tau-1}^{n-1} \frac{1}{j} = \frac{1}{e}.$$

□

In diesem Abschnitt konnte gezeigt werden, dass $\text{SECRETARY}_{\lfloor n/e \rfloor}$ ein e-kompetitiver Online-Algorithmus für das Best-Choice-Problem mit Partial-Information ist. Insbesondere ist e eine untere Schranke für den kompetitiven Faktor dieses Szenarios. Daraus folgt, dass $\text{SECRETARY}_{\lfloor n/e \rfloor}$ der optimale Online-Algorithmus für das Best-Choice-Problem mit Partial-Information ist.

4.3.2 Full-Information

Wir werden uns für diesen Abschnitt auf das Best-Choice-Problem mit Full-Information konzentrieren. Wie zuvor werden wir zunächst den kompetitiven Faktor der Algorithmen UNIFORM_k und SECRETARY_k für das Best-Choice-Problem mit Full-Information analysieren.

Wir beginnen mit der Analyse des Online-Algorithmus UNIFORM_k.

Korollar 4.11. *Der Algorithmus $UNIFORM_k$ erreicht keinen konstanten kompetitiven Faktor für das Best-Choice-Problem mit Full-Information.*

Beweis. Gemäß Theorem 4.7 erreicht der Online-Algorithmus U_k keinen konstanten kompetitiven Faktor für das Best-Choice-Problem mit Partial-Information. Aus Korollar 4.5 folgt, dass U_k für das Best-Choice-Problem mit Full-Information ebenfalls keinen konstanten kompetitiven Faktor erreicht. □

Mithilfe von Theorem 4.8 kann man analog zeigen, dass $\text{SECRETARY}_{\lfloor n/e \rfloor}$ für das Best-Choice-Problem mit Full-Information ein strikt $c(n)$-kompetitiver Online-Algorithmus ist.

Korollar 4.12. *Der Online-Algorithmus SECRETARY$_k$ mit $k = \lfloor n/e \rfloor$ ist ein strikt $c(n)$-kompetitiver Algorithmus für das Best-Choice-Problem mit Full-Information, wobei $c(n) = \frac{en}{n-e}$ für wachsendes n gegen e konvergiert.*

Die untere Schranke für das Best-Choice-Problem mit Partial Information kann auf das Best-Choice-Problem mit Full-Information erweitert werden. Dies hat Gnedin im Jahre 1994 gezeigt [20]. Gnedin argumentiert, dass eine Wahrscheinlichkeitsverteilung existiert, für die der beste ordnungsbasierte Online-Algorithmus für das Best-Choice-Problem mit Full-Information unter allen Online-Algorithmen stets die höchste Wahrscheinlichkeit, den besten Kandidaten zu akzeptieren, besitzt. Gemäß Theorem 4.9 ist e die untere Schranke für den kompetitiven Faktor für das Best-Choice-Problem mit Partial-Information. Diese Schranke gilt insbesondere für ordnungsbasierte Algorithmen. Daher folgt, dass kein Algorithmus für das Best-Choice-Problem mit Full-Information existieren kann, der besser als e-kompetitiv ist. Zusammenfassend erhält man das folgende bewiesene Theorem von Gnedin.

Theorem 4.13. *Es gibt keinen deterministischen oder randomisierten Algorithmus für das Best-Choice-Problem mit Full-Information für $n > 2$, der besser als e-kompetitiv ist.*

Mit diesem Theorem ist der Abschnitt zum Best-Choice-Problem abgeschlossen. Die untere Schranke von e gilt sowohl für das Best-Choice-Problem mit Partial-Information, als auch mit Full-Information. Hieraus folgt, dass für das Best-Choice-Problem allgemein kein besserer kompetitiver Faktor als e erreicht werden kann. Da SECRETARY$_{\lfloor n/e \rfloor}$ einen kompetitiven Faktor von e für das Best-Choice-Problem erreicht, stellt dieser insbesondere den optimalen Online-Algorithmus für das Best-Choice-Problem dar.

4.4 Value-Problem

In dem letzten Abschnitt haben wir uns mit dem Best-Choice-Problem beschäftigt. Wir wenden uns in diesem Abschnitt dem Value-Problem zu. Hierbei werden die Ergebnisse für das Best-Choice-Problem auch für das Value-Problem von Relevanz sein. Wir werden, wie zuvor beim Best-Choice-Problem, diesen Abschnitt in zwei Blöcke untergliedern: Das Value-Problem mit Partial-Information und mit Full-Information. Es werden ebenfalls für jede der Informationsszenarien die Algorithmen UNIFORM$_k$ und SECRETARY$_k$ analysiert und im nächsten Schritt eine untere Schranke besprochen.

4.4.1 Partial-Information

Wir werden für das Value-Problem mit Partial-Information den kompetitiven Faktor für die Online-Algorithmen UNIFORM_k und SECRETARY_k bestimmen. Die Erkenntnisse aus Abschnitt 4.3 werden mithilfe von Theorem 4.1 auf das Value-Problem übertragen.

Korollar 4.14. *Der Online-Algorithmus $UNIFORM_k$ erreicht keinen konstanten kompetitiven Faktor für das Value-Problem mit Partial-Information.*

Beweis. Wir haben in Theorem 4.7 bereits bewiesen, dass U_k für das Best-Choice-Problem n-kompetitiv ist. Aus Theorem 4.1 folgt daher, dass U_k ebenfalls ein n-kompetitiver Online-Algorithmus für das Value-Problem ist. Dieser Faktor konvergiert für wachsendes n gegen 0 und somit erreicht UNIFORM_k keinen konstanten kompetitiven Faktor für das Value-Problem mit Partial-Information. □

Ähnlich zum Beweis des kompetitiven Faktors von UNIFORM_k in Korollar 4.14, werden wir den kompetitiven Faktor von SECRETARY_k für das Value-Problem mit Partial-Information zeigen.

Korollar 4.15. *Der Online-Algorithmus $SECRETARY_k$ mit $k = \lfloor n/e \rfloor$ ist ein strikt $c(n)$-kompetitiver Algorithmus für das Value-Problem mit Partial-Information, wobei $c(n) = \frac{en}{n-e}$ für wachsendes n gegen e konvergiert.*

Beweis. Wir haben in Theorem 4.8 bereits bewiesen, dass $S_{\lfloor n/e \rfloor}$ für das Best-Choice-Problem $c(n)$-kompetitiv ist. Aus Theorem 4.1 folgt daher, dass $S_{\lfloor n/e \rfloor}$ ebenfalls ein $c(n)$-kompetitiver Online-Algorithmus für das Value-Problem ist. □

Es stellt sich die Frage, ob für das Value-Problem mit Partial-Information ein besserer kompetitiver Faktor als e erreicht werden kann. Die Antwort auf diese Frage ist negativ.

Theorem 4.16. *Es gibt keinen deterministischen oder randomisierten Online-Algorithmus für das Value-Problem mit Partial-Information für $n > 2$, der besser als e-kompetitiv ist.*

Beweis. Angenommen A sei ein Online-Algorithmus, der einen besseren kompetitiven Faktor als e für das Value-Problem mit Partial-Information erreicht. Dann gilt gemäß Theorem 4.2, dass ein solcher Algorithmus auch für das Best-Choice-Problem mit Partial-Information existiert. Dies ist ein Widerspruch zu Theorem 4.9 und somit war unsere Annahme falsch. □

Theorem 4.16 zeigt, dass die untere Schranke für das Value-Problem mit Partial-Information identisch zu der unteren Schranke für das Best-Choice-Problems mit Partial-Information ist.

4.4.2 Full-Information

In diesem Abschnitt konzentrieren wir uns auf das Value-Problem mit Full-Information. Es stellt sich die Frage, ob ähnlich zum Best-Choice-Problem die Offenbarung der numerischen Werte der Kandidateneignungen uns keinen Vorteil beim Entwurf eines Online-Algorithmus verschafft. Wir werden zunächst den kompetitiven Faktor der Algorithmen UNIFORM_k und SECRETARY_k für dieses Problem bestimmen. Im nächsten Schritt werden wir uns mit der unteren Schranke dieses Problems befassen.

Der kompetitive Faktor für die zwei Algorithmen für das Full-Information-Szenario können wir aus den Ergebnissen für das Partial-Information-Szenario ableiten.

Korollar 4.17. *Der Online-Algorithmus $UNIFORM_k$ erreicht keinen konstanten kompetitiven Faktor für das Value-Problem mit Full-Information.*

Beweis. Wir haben in Theorem 4.14 bereits bewiesen, dass U_k für das Value-Problem mit Partial-Information n-kompetitiv ist. Aus Theorem 4.5 folgt daher, dass U_k ebenfalls ein n-kompetitiver Online-Algorithmus für das Value-Problem mit Full-Information ist und somit keinen konstanten kompetitiven Faktor erreicht. □

Der Beweis des kompetitiven Faktors für den Online-Algorithmus SECRETARY_k ist ähnlich zu Korollar 4.17. Dieser nutzt die Ergebnisse aus Theorem 4.15 und Korollar 4.5 aus.

Korollar 4.18. *Der Online-Algorithmus $SECRETARY_k$ mit $k = \lfloor n/e \rfloor$ ist ein strikt $c(n)$-kompetitiver Algorithmus für das Value-Problem mit Full-Information, wobei $c(n) = \frac{en}{n-e}$ für wachsendes n gegen e konvergiert.*

Somit haben die beiden Algorithmen den gleichen kompetitiven Faktor für das Value-Problem mit Full-Information wie zuvor beim Value-Problem mit Partial-Information. Aus den bisher bewiesenen Aussagen ergibt sich folgendes Resultat für den Algorithmus $\text{SECRETARY}_{\lfloor n/e \rfloor}$.

Korollar 4.19. *Der Algorithmus $SECRETARY_{\lfloor n/e \rfloor}$ erhält mit einer Wahrscheinlichkeit von mindestens $1/e$ den besten Kandidaten und sonst einen Kandidaten, dessen erwartete Eignung höchstens um einen Faktor von e kleiner ist als die des besten Kandidaten.*

Es stellt sich die Frage, ob ein besserer kompetitiver Faktor als e für das Full-Information-Szenario erreicht werden kann. Hierzu konzentrieren wir uns im Folgenden auf die untere Schranke für das Value-Problem mit Full-Information.

Die Vermutung liegt nahe, dass wie bereits beim Value-Problem mit Partial-Information in Theorem 4.16, der kompetitive Faktor von e eine untere Schranke für das Value-Problem mit Full-Information darstellt. Wir werden unsere Vermutung zunächst festhalten.

Vermutung 4.20. *Es gibt keinen deterministischen oder randomisierten Online-Algorithmus für das Value-Problem mit Full-Information für $n > 2$, der besser als e-kompetitiv ist.*

Für die Aussage in Vermutung 4.20 werden wir zunächst einen Beweis aus der Literatur besprechen und argumentieren, warum dieser einen logischen Fehler beinhaltet. Im nächsten Schritt werden wir eigene Ansätze zu diesem Problem besprechen.

Zu der Vermutung 4.20 wurde bereits in Korollar 2.4 aus [32] ein Beweis verfasst. Der Ansatz ist allerdings nicht korrekt und wir werden dies kurz erläutern. Es soll gezeigt werden, dass kein Algorithmus für das Value-Problem mit Full-Information existiert, der einen besseren kompetitiven Faktor als e erreicht. Diese Aussage soll durch einen Widerspruchsbeweis gezeigt werden. Hierfür wird angenommen, dass ein Algorithmus A existiert, der einen besseren kompetitiven Faktor als e erreicht. Algorithmus A ist insbesondere auf der Instanz $\sigma' = \{D, \epsilon, \epsilon, \ldots, \epsilon\}$ e-kompetitiv. Die Zahl D kann hinreichend groß und ϵ hinreichend klein gewählt werden, sodass Algorithmus A mit einer besseren Wahrscheinlichkeit als $1/e$ den besten Kandidaten auf σ' erhalten muss. In der Arbeit [32] wird argumentiert, dass dies somit ein Widerspruch zu dem Beweis von Theorem 4.13 darstellt. Theorem 4.13 sagt aus, dass kein Algorithmus für *alle* Instanzen mit einer besseren Wahrscheinlichkeit als $1/e$ den besten Kandidaten erhalten kann. Der Algorithmus A erfüllt diese Eigenschaft lediglich für die Instanz σ', aber es wurde nicht gezeigt, dass dies ebenfalls für alle Instanzen gilt.

Die untere Schranke für das Value-Problem mit Full-Information stellt sich als etwas schwieriger heraus, als zu Beginn vermutet. Wir zeigen daher eine untere Schranke mit der Einschränkung, dass die Eignungen der Kandidaten nichtnegative ganze Zahlen sind. Wir greifen für diesen Beweis auf Theorem 4.4 zurück.

Korollar 4.21. *Seien $\sigma_i \in \mathbb{N}_0$. Dann existiert kein deterministischer oder randomisierter Algorithmus für das Value-Problem mit Full-Information für $n > 2$, der besser als e-kompetitiv ist.*

Beweis. Angenommen es existiert ein Online-Algorithmus A für das Value-Problem mit Full-Information, der besser als e-kompetitiv ist. Laut Theorem 4.4 exis-

tiert ebenfalls ein Algorithmus für das Best-Choice-Problem im Full-Information-Szenario mit einem besserem kompetitiven Faktor als e. Dies ist jedoch ein Widerspruch zu Theorem 4.13. □

Die Vermutung 4.3 weitet die Aussage aus Theorem 4.4 auf nicht-negative reelle Eignungen aus. Allerdings konnten wir diese Aussage in Vermutung 4.3 in dieser Arbeit nicht beweisen. Falls wir annehmen, dass Vermutung 4.3 gilt, dann kann man analog zu Korollar 4.21 zeigen, dass e eine untere Schranke für den kompetitiven Faktor für das Value-Problem mit Full-Information ist.

Korollar 4.22. *Falls Vermutung 4.3 gilt, dann existiert kein deterministischer oder randomisierter Online-Algorithmus für das Value-Problem mit Full-Information für $n > 2$, der besser als e-kompetitiv ist.*

Somit wurde das Problem der unteren Schranke in Korollar 4.22 auf die Vermutung 4.3 reduziert. Falls die Vermutung bewiesen wird, gilt die untere Schranke von e für das Value-Problem mit Full-Information.

5 | Fazit und Ausblick

In dieser Arbeit haben wir uns mit dem Sekretärsproblem beschäftigt. Wir haben uns zunächst mit den beiden Problemvarianten, dem Best-Choice- und dem Value-Problem, vertraut gemacht. Zwischen den Problemvarianten konnten nützliche Zusammenhänge dargestellt werden, die bei der kompetitiven Analyse und bei dem Beweis von den unteren Schranken an Bedeutung fanden. Weiterhin wurde der Algorithmus SECRETARY$_{\lfloor n/e \rfloor}$ vorgestellt, der sowohl für das Best-Choice, als auch für das Value-Problem e-kompetitiv ist. Somit ergab sich, dass dieser mit einer Wahrscheinlichkeit von $1/e$ den besten Kandidaten erhält und sonst einen Kandidaten, dessen Talent höchstens um einen Faktor von e kleiner ist, als das Talent des besten Kandidaten.

Bei der Analyse der unteren Schranke für das Best-Choice-Problem zeigte sich, dass sowohl für das Partial-Information, als auch für das Full-Information-Szenario, kein kompetitiver Faktor besser als e erreicht werden kann. Somit wurde dieses Problem vollständig gelöst. Für das Value-Problem mit Partial-Information konnte die untere Schranke von e ebenfalls gezeigt werden. Für das Value-Problem mit Full-Information ist uns der Beweis der unteren Schranke allerdings nicht gelungen. Stattdessen haben wir zwei schwächere Aussagen zeigen können. Die erste Aussage reduziert das Problem der unteren Schranke auf eine Vermutung und die zweite Aussage zeigt eine untere Schranke von e bei Einschränkung der Eignungen auf natürliche Zahlen. Somit bleibt das Problem der unteren Schranke für das Value-Problem mit Full-Information für reelle Eignungen ein offenes Problem.

Mit dieser Arbeit erhoffen wir uns die Relevanz des Value-Problems im Zusammenhang mit dem Sekretärsproblems hervorgehoben zu haben und dass diese Arbeit der Forschungsgemeinde zum Lösen des Problems der unteren Schranke für das Value-Problem mit Full-Information zu helfen vermag.

A | Anhang

An dieser Stelle wollen wir eine Anleitung zum Lösen des primalen LPs in 4.9 aus Theorem 4.9 mithilfe eines LP-Solvers für explizite n präsentieren.

Wir werden die Schritte einzeln auflisten.

1. Besuchen Sie die Website `http://scip.zib.de`.

2. Wählen Sie unter Downloads die Binaries-Packages für Windows aus. Die *Visual C++ Redistributable Packages* müssen mit installiert werden.

3. Nachdem Sie sowohl die Binaries-Packages als auch die Visual C++ Packages heruntergeladen haben, gilt es, das Binary-Package zu öffnen. Es befinden sich drei Dateien in diesem Ordner, unter anderem auch eine *.exe*-File. Letzteres wählen Sie aus.

4. Es öffnet sich ein neues Fenster. Dies stellt die Konsole dar. Von hier aus lässt sich das LP-Format einlesen und lösen. Wir werden die Benutzung der Konsole genauer unter Punkt 9 besprechen. Vorher muss allerdings erst einmal unser mathematisch definiertes LP in ein für das SCIP angemessenes Format übersetzt werden.

5. Sie schreiben hierzu das gewünschte LP in einer .txt-Datei. Zur Übersetzung des mathematischen Modells in eine für den Computer verständliche Sprache kann die ZIMPL-Sprache [26] verwendet werden. Es gibt auch andere Sprachen für die Übersetzung des LPs. ZIMPL hat den Vorteil, dass für komplexere LPs der Code weiterhin sehr übersichtlich ist. Zum besseren Überblick der Funktionsweisen ist die Einleitung zur ZIMPL-Sprache unter `http://zimpl.zib.de/download/zimpl.pdf` empfehlenswert.

6. Unser lineares Programm aus 4.9 hat in ZIMPL folgende Form.

```
#Specify the number of n for the linear program.
param n := 100;

#We declare a set of numbers from 1 to n.
set N := {1 .. n};
#We declare the variables for p_1,...p_n.
var p[<i> in N] real >= 0;

#These are the values of the vector c.
param c[<i> in N] := i/n ;

#We define the entries a_{ij} of the Matrix A.
defnumb a(i,j) :=
      if i < j then 0
      else if i == j then i
      else 1 end end;

#This is our objective function.
maximize prob: sum <i> in N : c[i] * p[i];

#The condition which needs to be fulfilled.
subto condition:
      forall <i> in N:
        sum <j> in N: a(i,j) * p[j] <= 1;
```

Listing A.1: LP 4.9 aus Theorem 4.9 in ZIMPL.

Die Zeilen, die zu Beginn mit # versehen sind, enthalten Kommentare zum besseren Verständnis des Codes. Das explizite n, wofür das LP gelöst werden soll, wird bei *param n* in der zweiten Zeile spezifiziert. Dieser ist auf 100 gesetzt, allerdings können Sie diese beliebig verändern. Es sei jedoch Vorsicht geboten, da für große n eine lange Laufzeit beansprucht werden kann.

7. Den vorgestellten Code können Sie in Ihre .txt-Datei kopieren. Diesen speichern Sie mit einem bestimmten Namen zum Beispiel „LP“ und der Endung .zpl. In unserem Fall würden wir die Datei mit folgendem Namen abspeichern: *LP.zpl*

8. Um das in ZIMPL definierte lineare Programm zu lösen werden wir die in Punkt 5 erwähnte Konsole öffnen. Zu Beginn wird in der letzten Zeile

   SCIP>

   stehen.

9. Wir werden unser Programm einlesen. Führen Sie hierzu den Befehl

   read LP.zpl

   aus, wobei *LP.zpl* die Datei ist, wo der Code abgespeichert wurde.

10. Falls es keine Fehlermeldungen geben sollte, kann mit dem Befehl

    optimize

    das LP gelöst werden. Der Wert des LPs kann direkt auf der Konsole unter dem Begriff „Primal Bound“ abgelesen werden.

11. Um sich das Ergebnis der Variablen auf der Konsole ausgeben zu lassen, führen Sie

    display solution

    auf der Konsole aus. Falls es bevorzugt wird, das Ergebnis in einer separaten Datei namens „LPSolution“ zu speichern, gibt man hierzu den Befehl

    write solution LPSolution.sol

    ein. Sie sollten in der Lage sein, entsprechend Ihrer Auswahl, die Lösung des LPs in der Konsole oder in der Datei abzulesen.

Das oben beschrieben lineare Programm wurde für verschiedene Werte von n gelöst. Die hierbei erhaltenen Werte werden wir in der folgenden Tabelle auflisten. Auf Grundlage dieser Werte ist der Graph in 4.5 entstanden.

n	LP-Wert
100	0,371000000
200	0,369460000
300	0,368930000
400	0,368671000
500	0,368512200
600	0,368406000
700	0,368331000
800	0,368274800
900	0,368230900
1000	0,368195610
1100	0,368166706
1200	0,368142923
1300	0,368122705
1400	0,368105306
1500	0,368090175
1600	0,368076978
1700	0,368065422
1800	0,368055105
1900	0,368045839
2000	0,368037400
2100	0,368029960
2200	0,368023149
2300	0,368016903
2400	0,368011157
2500	0,368005853

n	LP-Wert
2600	0,368001020
2700	0,367996532
2800	0,367992348
2900	0,367988438
3000	0,367984000
3100	0,367981413
3200	0,367978234
3300	0,367975235
3400	0,367972403
3500	0,367969747
3600	0,367967251
3700	0,367964880
3800	0,367962626
3900	0,367960480
4000	0,367958460
4100	0,367956542
4200	0,367954707
4300	0,367952950
4400	0,367951269
4500	0,367949684
4600	0,367948161
4700	0,367946698
4800	0,367945290
4900	0,367943943
5000	0,367942660

Tabelle A.1: Werte für verschiedene n für das lineare Programm aus 4.9. Der dazugehörige Plot wird in Abbildung 4.5 dargestellt.

Literaturverzeichnis

[1] AR Abdel-Hamid, JA Bather, and GB Trustrum. **The secretary problem with an unknown number of candidates**. *Journal of Applied Probability*, pages 619–630, 1982.

[2] Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. **Online Vertex-Weighted Bipartite Matching and Single-bid Budgeted Allocations**. *CoRR*, abs/1007.1271, 2010.

[3] Miklós Ajtai, Nimrod Megiddo, and Orli Waarts. **Improved algorithms and analysis for secretary problems and generalizations**. *SIAM Journal on Discrete Mathematics*, 14(1):1–27, 2001.

[4] Moshe Babaioff, Michael Dinitz, Anupam Gupta, Nicole Immorlica, and Kunal Talwar. **Secretary problems: weights and discounts**. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1245–1254. Society for Industrial and Applied Mathematics, 2009.

[5] Mohammad Hossein Bateni, Mohammad Taghi Hajiaghayi, and Morteza Zadimoghaddam. **Submodular secretary problem and extensions**. *ACM Transactions on Algorithms (TALG)*, 9(4):32, 2013.

[6] J Neil Bearden. **A new secretary problem with rank-based selection and cardinal payoffs**. *Journal of Mathematical Psychology*, 50(1):58–59, 2006.

[7] MJ Beckmann. **Dynamic programming and the secretary problem**. *Computers & Mathematics with Applications*, 19(11):25–28, 1990.

[8] F. Bornemann. **Konkrete Analysis: für Studierende der Informatik**. eXamen.press. Springer Berlin Heidelberg, 2008.

[9] F Thomas Bruss. **On an optimal selection problem of Cowan and Zabczyk**. *Journal of Applied Probability*, pages 918–928, 1987.

[10] F Thomas Bruss and Stephen M Samuels. **A unified approach to a class of optimal selection problems with an unknown number of options**. *The Annals of Probability*, pages 824–830, 1987.

[11] Niv Buchbinder, Kamal Jain, and Mohit Singh. **Secretary problems via linear programming**. In *Integer Programming and Combinatorial Optimization*, pages 163–176. Springer, 2010.

[12] A Cayley. **Problem 3564, Mathematical Questions with their Solutions from the Educational Times**. *XVII, CF Hodgson & Sons, London*, page 72, 1872.

[13] Google Company. **History**. `http://www.google.com/about/company/history`, 2015.

[14] Eugene B Dynkin. **The optimum choice of the instant for stopping a Markov process**. In *Soviet Math. Dokl*, volume 4, 1963.

[15] EZ Ferenstein and EG Enns. **Optimal sequential selection from a known distribution with holding costs**. *Journal of the American Statistical Association*, 83(402):382–386, 1988.

[16] Thomas S. Ferguson. **Who Solved the Secretary Problem**. *Statist. Sci.*, 4(3):294–296, 08 1989.

[17] PR Freeman. **The secretary problem and its extensions: A review**. *International Statistical Review/Revue Internationale de Statistique*, pages 189–206, 1983.

[18] Martin Gardner. **Scientific American**. Feurary, 1960.

[19] John P Gilbert and Frederick Mosteller. **Recognizing the maximum of a sequence**. In *Selected Papers of Frederick Mosteller*, pages 355–398. Springer, 2006.

[20] Alexander V. Gnedin. **A Solution to the Game of Googol**. *Ann. Probab.*, 22(3):1588–1595, 07 1994.

[21] Irwin Guttman. **On a problem of L. Moser**. *Canadian Mathematical Bulletin*, 3:35–39, 1960.

[22] Stephan Held. **Lineare und Ganzzahlige Optimierung**. `http://www.or.uni-bonn.de/~held/lpip/1314/skript/Lingo1314.pdf`, Vorlesungsskript, Universität Bonn, Wintersemester 2013/14.

[23] Anna R Karlin, Mark S Manasse, Larry Rudolph, and Daniel D Sleator. **Competitive snoopy caching**. *Algorithmica*, 3(1-4):79–119, 1988.

[24] Samuel Karlin. **Stochastic models and optimal policy for selling an asset**. *Studies in applied probability and management science*, pages 148–158, 1962.

[25] Robert Kleinberg. **A multiple-choice secretary algorithm with applications to online auctions**. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 630–631. Society for Industrial and Applied Mathematics, 2005.

[26] Thorsten Koch. **Rapid Mathematical Programming**. PhD thesis, Technische Universität Berlin, 2004. ZIB-Report 04-58.

[27] Denis V Lindley. **Dynamic programming and decision theory**. *Applied Statistics*, pages 39–51, 1961.

[28] Thomas J Lorenzen. **Optimal stopping with sampling cost: the secretary problem**. *The Annals of Probability*, pages 167–172, 1981.

[29] Leo Moser. **On a problem of Cayley**. 1956.

[30] Joseph D Petruccelli. **Best-choice problems involving uncertainty of selection and recall of observations**. *Journal of Applied Probability*, pages 415–425, 1981.

[31] Ernst L'vovich Presman and Isaac Mikhailovich Sonin. **The best choice problem for a random number of objects**. *Teoriya Veroyatnostei i ee Primeneniya*, 17(4):695–706, 1972.

[32] Klaus Radke. **Online Optimization in the Random-Order Model**. PhD thesis, Hochschulbibliothek der Rheinisch-Westfälischen Technischen Hochschule Aachen, 2014.

[33] Heiko Röglin. **Online-Algorithmen**. `http://www.roeglin.org/teaching/WS2013/AlgoI/AlgoI.pdf`, Vorlesungsskript, Universität Bonn, Sommersemester 2015.

[34] John S Rose. **Optimal sequential selection based on relative ranks with renewable call options**. *Journal of the American Statistical Association*, 79(386):430–435, 1984.

[35] Stephen M Samuels. **A best-choice problem with linear travel cost**. *Journal of the American Statistical Association*, 80(390):461–464, 1985.

[36] Stephen Silverman and Arthur Nadas. **On the Game of Googol as the Secretary**. In *Strategies for sequential search and selection in real time*, volume 125, page 77. American Mathematical Soc., 1992.

[37] Robert J Vanderbei. **The postdoc variant of the secretary problem**.

Eidesstattliche Erklärung

Hiermit versichere ich an Eides statt und durch meine Unterschrift, dass die vorliegende Arbeit von mir selbständig und ohne fremde Hilfe angefertigt worden ist. Inhalte und Passagen, die aus fremden Quellen stammen und direkt oder indirekt übernommen worden sind, wurden als solche kenntlich gemacht. Ferner versichere ich, dass ich keine andere, außer der im Literaturverzeichnis angegebenen Literatur, verwendet habe. Die Arbeit wurde bisher keiner Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Bonn, den 29.Oktober 2015, Ramtin Azimi