Geometric On-line Ray Searching Under Probability of Placement Scenarios

by

Ying Liu

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2010

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Online computation is a model for formulating decision making under uncertainty. In an online problem, the algorithm does not know the entire input from the beginning; the input is revealed in a sequence of steps. At each step, the algorithm should make its decisions based on the past and without any knowledge about the future. Many important real-life problems such as robot navigation are intrinsically online and thus the design and analysis of online algorithms is one of the main research areas in theoretical computer science.

Competitive analysis is the standard measure for analysis of online algorithms [19, 22, 30, 38]. It has been applied to many online problems in diverse areas ranging from robot navigation, to network routing, to scheduling, to online graph coloring. In this thesis, we first survey three classic online problems, namely the cow-path problem, the Processor-Allocation problem and the Robots-Search-Rays problem and highlight connections between them.

Second, the main result is for the One-Robot-Searches-Two-Rays problem for which we consider the weighted scenario, in which the robot is located on a ray with a preferential probability p. We term the One-Robot-Searches-Two-Rays-And-Weighted problem as 1-STRAW (and in general k-STRAW for k searchers).

In the 1-STRAW problem, we propose a search strategy which is optimal among weighted geometric states. In addition, we prove a tight lower bound of the worst case competitive ratio and conjecture a lower bound of the average case competitive ratio for the 1-STRAW problem. Additionally, we compare our search strategy and its performance with the doubling strategy [41] and the SmartCow algorithm[36].

Acknowledgments

I am very thankful to my supervisor, Professor Alejandro López-Ortiz, for the many discussions and suggestions on my research topic, as well as improving the clarity and readability of this thesis. He is always patient and offering his exceptional insights when I asked questions and I am amazed by his knowledge and experience in various aspects. I feel really lucky to be the student of Professor Alejandro López-Ortiz.

I want to thank my parents. Life is full of ups and downs but my parents are always there for me with their unconditional love. Because of their encouragements and supports, I can exert my full energy to my research and make steady progress. I am also grateful to my dear friends—Xi Han and her husband Lin He, Yan Mao, Ting Liu and my officemates Raju and Kameran as well as many friends who enrich my life.

Last but not least, I want to thank Professor Ian Munro and Professor Anna Lubiw for taking the time to read my thesis and provide helpful feedback and suggestions.

Dedication

This work is dedicated to my parents.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Online computation is a model for formulating decision making under uncertainty [5, 15, 39, 47]. In an online problem, the algorithm does not know the entire input of the problem. The input is rather revealed in a sequence of steps. An online algorithm makes its decisions only based on the observed past and without any given knowledge about the input sequence in the future. The cost of a decision taken by the online algorithm cannot be undone.

1.1 Online Target Searching

Online target searching is the class of robot searching problems in which the robot knows only partially about some of the configuration of the search domain, the shape of the terrain, the target's location and its own position. The problem of target searching involves an agent or a robot exploring a given domain, with the purpose of guarding it or finding a given target under some conditions of uncertainty [41].

The quality of a search depends on the searching abilities of the robot and its knowledge about the terrain being searched. Therefore, it is important to consider various cases given the robot's different capabilities.

Targets for searches are classified into two main types: **mobile** and **immobile**. Domains are divided into **bounded** and **unbounded**. Among the first, we consider searches on line segments. For unbounded domains, we consider the real lines.

Robots are also classified according to their abilities in several classes: **tactile, visual, navigational tools,** and **computing resources**. A tactile robot identifies the target when the robot is located at an ϵ distance or less of the object. A robot with vision comes with a system that provides a visibility map of its local environment. However, a robot moving under restricted memory may not be able to use previous visibility maps. Additionally, a robot with no memory or knowledge is termed **oblivious** whereas the robot is **non-oblivious**.

These conditions introduce a degree of uncertainty in the search that is revealed as the robot searches the terrain. Thus, searches can be considered as an online problem because the robot can only attain more information while active, as opposed to the offline version, in which all the information is given at once before the start of the computation.

In this thesis, we will focus on the One-Robot-Searches-Two-Rays-And-Weighted problem. By *weighted*, we mean that the target hides on one of the rays with a random probability p, where $p \in [0, 1]$. We will discuss the Robots-Search-Rays problem in details in Chapter 2 Section 2.3 and the One-Robot-Searches-Two-Rays-And-Weighted problem in Chapter 3.

1.2 Online Measures

Competitive analysis is the standard measure of online algorithms. In this model, we compare the performance of an online algorithm with an offline optimal algorithm OPT which knows the entire input in advance. Intuitively, we want to measure how close is the performance of the online algorithm to the optimal case. Competitive analysis became popular after two papers by Sleator and Tarjan [19, 30] in 1985. The term competitive analysis was introduced by Karlin et al [38]. Additionally, the competitive framework was thoroughly studied by Dorrigiv and López-Ortiz [22, 23].

Competitive analysis is a relatively simple measure to apply but efficient to quantify the performance of the online algorithms, which was a major breakthrough in the study of online algorithms. It computes a partial solution to a problem with incomplete information of a problem. It indicates the performance drop of the online algorithm given the absence of the input information of the problem. Therefore, we compare the cost of the online algorithm with the one computed with the full information, namely the cost of the optimal offline algorithm OPT. Without loss of generality, we denote the cost of an algorithm $\mathcal{A}$ on a sequence σ by $\mathcal{A}(\sigma)$. An online algorithm $\mathcal{A}$ is said to have competitive ratio c if $\mathcal{A}(\sigma) \leq c \cdot OPT(\sigma)$ for all sequences σ.

Alternatively, a $C(n)$-competitive algorithm is defined as that, for all sequences σ, an online algorithm $\mathcal{A}$ is said to have the competitive ratio $C(n)$ if $\mathcal{A}(\sigma) \leq C(|\sigma|) \cdot OPT(\sigma)$. More generally, we have that an algorithm is $C(n)$-competitive if and only if

$$C(n) = \max_{|\sigma|=n} \left\{ \frac{\mathcal{A}(\sigma)}{OPT(\sigma)} \right\}$$

We will use the notation of the competitive ratio throughout the thesis especially in Chapter 3.

For *randomized algorithms*, we have a similar definition of the competitive ratio [45]. A randomized online algorithm $\mathcal{A}$ makes some random choices while handling the sequence of the input. Therefore the cost of the randomized online algorithm is a random variable. In addition, depending on the types of the adversaries, we have different types of offline algorithms. An oblivious adversary does not know the actions made by $\mathcal{A}$ on σ. Therefore it cannot make its own movements according to the performances of the online algorithm. We say that $\mathcal{A}$ has an

asymptotic competitive ratio $C(n)$ against an oblivious adversary if there exists a constant b such that for all sequences σ,

$$E[\mathcal{A}(\sigma)] \leq C(|\sigma|) \cdot OPT(\sigma) + b$$

In contrast, for the adversaries who are non-oblivious, for example adversaries can observe the outcome of the online algorithm, and make its own decision accordingly, the competitive ratio then is defined as:

$$E\left[\frac{\mathcal{A}(\sigma)}{C(|\sigma|) \cdot OPT(\sigma)}\right] \leq b$$

Since in this case, $OPT(\sigma)$ is a random variable as well.

Competitive ratio is the key to measure online problems and we use the concept throughout the thesis. For the cow-path problem in Section 2.1, we use the competitive ratio to evaluate the performance of the SmartCow algorithm which is described in Figure 2.1. For the Processor-Allocation problem in Section 2.2, we use a similar concept of the acceleration ratio which is a worst case measure, refer to Definition 3. For the Robots-Search-Rays problem in Section 2.3 which includes our core work of the 1-STRAW problem, we used the competitive ratio as well.

1.3 Our Results and Organization of the Thesis

This thesis is concerned with oblivious robot searches for an immobile target on unbounded real lines. In Chapter 2, we present a survey of three classic online problems, namely the cow-path problem, the Processor-Allocation problem and the Robots-Search-Rays problem. We state the known lower bounds on the competitive ratio for these three problems and compare the common aspects and differences between them. In Chapter 3, we define the problem of One-Robot-Searches-Two-Rays-And-Weighted (1-STRAW) and propose a search strategy. Additionally, we compute both the worst case competitive ratio and the average case competitive ratio given our search strategy. In Section 3.3, we analyze the average case competitive ratio of 1-STRAW in both the un-truncated setting and the truncated setting in terms of the target's location range. We prove a tight lower bound of the worst case competitive ratio and conjecture a lower bound on the average case competitive ratio for the 1-STRAW problem. In Chapter 4, we present the conclusions and the open problems.

Chapter 2

Classic Online Problems

The problem of scheduling is central to many areas of computer science [40]. In this chapter, we demonstrate connections among three typical scheduling problems. The first problem is known as the w-ray cow-path problem [36, 37]. The second involves computing solutions to multiple problems given a limited amount of processors, under the condition that a solution to any one of the problems can be requested at any time. The third problem involves multiple robots searching an unknown environment for a goal. These problems concern *anytime* algorithms whose quality of output improves gradually as the amount of available computation time increases. Such algorithms occur naturally in settings where a computationally intensive problem is addressed under uncertainty with respect to the available computation time.

In the cow-path problem, consider a cow standing at a crossroads (referred to as the origin) with w paths (rays) leading off into unknown territory. On one of the rays there is a grazing field (the goal) at distance (unknown) d from the intersection, and all of the other rays go on forever; unfortunately, the cow does not know that it has found the field until it is standing in it. Clearly, the cow must walk at least distance d to get to the field; if it knows which path to take, the cow will walk exactly distance d. When the cow has no prior knowledge of which ray the field is on, we would like to know how it can find the field while traveling the least distance possible. López-Ortiz [43] proved a firm lower bound of the competitive ratio for the w-ray cow-path problem among all deterministic search strategies. Additionally, Kao *et al.* [36] gave an optimal randomized algorithm for the 2-ray cow-path problem and conjectured a lower bound among all randomized algorithms for the w-ray cow-path problem.

In the Processor-Allocation problem, consider the general setting in which a set of m processors of the identical speed is available for the execution of n problem instances each with a corresponding *contract algorithm*, which is defined as a set of the computation durations for the specific problem instance. For example, given the problem instance p_i, its corresponding contract algorithm is denoted as $\{(p_i, d_1), (p_i, d_2), \ldots (p_i, d_j) \ldots, (p_i, d_n)\}$, which means a processor can be executed for d_j amount of time to get a solution of the problem instance. (p_i, d_j) is termed as a *contract* in the contract algorithm.

The problem we face is how to assign and schedule the executions of the various contract algorithms to the processors in a way that guarantees an efficient *interruptible algorithm*, which is the algorithm that can provide the solutions of the problem instances given the fact that the algorithm can be interrupted at any time to be queried for the problem solutions. In this setting, at query time, the algorithm will report for each problem instance the solution of the corresponding contract of *longest length* (i.e., contract time) which has been completed by query time. Schedules are compared in terms of the *acceleration ratio*, which is a worst-case measure of efficiency. Bernstein *et al.* [13] showed an upper bound of $\frac{n}{m}(\frac{m+n}{n})^{(\frac{m+n}{m})}$ on the acceleration ratio; in addition, they showed that this bound is optimal for a restricted, though natural and intuitive, class of schedules that use a *round robin* and *length-increasing* strategy. Such strategies are known as *cyclic* strategies. López-Ortiz *et al.* [42] proved that this bound is tight among all possible schedules.

In the Robots-Search-Rays problem, m robots search for a target which is located on one of p concurrent rays. We seek search strategies for the robots that minimize the *competitive ratio*, namely the maximum of the ratio of the search cost using the strategy, and the distance from the starting position to the target, over all possible positions of the target. Kao *et al.* [37] gave a hybrid algorithm that achieves an optimal competitive ratio of $m + 2\frac{(p-m+1)^{p-m+1}}{(p-m)^{p-m}}$. For the general problem of m robots and p rays, López-Ortiz and Schuierer [43] showed an optimal strategy that achieves competitive ratio $1 + 2\frac{p-m}{m}(\frac{p}{p-m})^{\frac{p}{m}}$.

It has been observed that the theoretical analysis of geometric searches and robot motion planning is closely linked to the scheduling of heuristics and algorithms for problem solving. This connection was first established by Kao *et al.* [36, 37] for the case of a single searcher in the randomized case, as well as certain multi-searcher scenarios. The work of Bernstein *et al.* [13] drew a similar connection between scheduling contract algorithms and robot searching on a set of rays. Bernstein *et al.* [13] work established the connection only for cyclic schedules. It turns out that interesting parallels can be drawn for the Processor-Allocation problem and the Robots-Search-Rays problem: informally, the rays correspond to problem instances, the robots to processors, and the (unknown) location of the target corresponds to the (also unknown) query time. Moreover, when p=1, the Robots-Search-Rays problem becomes the cow-path problem.

2.1 Cow-path Problem

Search problems are widely studied by computer science researchers [1, 8, 9, 10, 11, 12, 26, 27, 28, 34, 44] and the cow-path problem is one of them. The cow-path has been the subject of intense study under the competitive framework [6, 20, 33, 35, 36], which plays an important role in online problems. The problem has a simple model. A cow is standing at the origin of w rays, which extend from the origin to unlimited distances far away. There is only one grass field (goal) on one of the w rays and the information of the goal is not revealed to the cow. The problem is to minimize the cow's total traveling distance until it finds the grass field.

This problem has various applications, one of which is robot motion planning [7, 16, 17, 18,

25]. For example, a robot searching in a two-dimensional space with obstacles. Whenever the robot runs into an obstacle, it should look for the closest corner of the obstacle to walk around. This problem of avoiding obstacles can be interpreted as a 2-ray cow-path problem, where the goal is the closest corner and rays are the two possible paths, one of which contains the closest corner.

A tight lower bound for general deterministic algorithms for the cow-path problem is known but the randomized search algorithms and the lower bound of the set of general randomized algorithms are relatively less developed than deterministic search strategies. In this section, we list the results of the deterministic search strategies and introduce a randomized algorithm, which was first proposed and shown to be optimal for the 2-ray cow-path problem by Kao *et al.* [36].

Here we denote by $\mathcal{A}$ a deterministic algorithm for the cow-path problem and by g the goal distance $dist(g)$ from the origin. Algorithm $\mathcal{A}$ searches a fixed distance to find the goal, which is denoted as $cost(\mathcal{A}, g)$.

Definition 1 ([36]) *Algorithm $\mathcal{A}$ has competitive ratio c if, for all goal positions g,*

$$cost(\mathcal{A}, g) \leq c * dist(g) + d, \tag{2.1}$$

where c and d are constants that are independent of the goal position g.

We denote randomized algorithms as $\mathcal{R}$. Since the distance explored to find the goal is not a fixed number under randomized algorithms, instead we define $cost(\mathcal{R}, g)$ as a random variable. Therefore we define the competitive ratio of randomized algorithms by the expected value of the random variable $cost(\mathcal{R}, g)$.

Definition 2 ([36]) *Algorithm $\mathcal{R}$ has competitive ratio c if, for all goal positions g,*

$$E[cost(\mathcal{R}, g)] \leq c * dist(g) + d, \tag{2.2}$$

where c and d are constants that are independent of the goal position g.

Particularly, if an algorithm for the cow-path problem has competitive ratio c, then for any goal position g from the origin, the expected search distance that the algorithm explores to find the goal is at most cg plus a small constant.

A Deterministic Algorithm

An optimal deterministic search strategy is a cyclic search strategy with the exponential geometric ratio b. The strategy works as follows. The cow explores the w rays in a fixed cyclic order. Let $b = \dfrac{w}{w-1}$. The sequence of turn points of the cow is given by $x_i = b^i$ for $i = 0, 1, 2, \ldots$. The kth time that the cow returns to the origin it chooses to explore ray ($k \mod w$) up to distance x_k.

Theorem 1 ([43]) *Any deterministic search strategy for the w-ray cow-path problem has a worst case competitive ratio of at least*

$$1 + 2\,(w-1)\left(\frac{w}{w-1}\right)^{w}.$$

In the 2-ray cow-path problem, the optimal deterministic search strategy defined above is named as the *doubling strategy* and it has the worst case competitive ratio of 9, which matched Theorem 1. Additionally, it has been proved that the doubling strategy can achieve an average case competitive ratio of approximately 5.27, which is stated in the theorem as follows.

Theorem 2 ([41]) *Assume that the hiding target selects a random distance d and that it hides using a uniform distribution on the interval* $[d, -d]$. *Then the doubling strategy has an average case competitive ratio of* ≈ 5.27.

A Randomized Algorithm

Kao *et al.* [36] described an algorithm named the SmartCow algorithm, which is a randomized geometric sweep algorithm with *geometric ratio* $r > 1$, which is a fixed constant used to accumulate the cow's search distances in rounds. Without loss of generality, we assume that the w rays are tagged with the integers 0,1,...,w-1. The description of the SmartCow algorithm can be found in Figure 2.1. The analysis of the competitive ratio will be made regarding the constant r.

Restrictions of the SmartCow Algorithm

- The usage of the randomization in the SmartCow algorithm is very limited in this case. The randomization is only used at the beginning of the search, in order to get a random permutation of the w rays and a random *initial search distance*. The algorithm does not require a random number generator after the search begins.
- The goal is located on one of the w rays with the same possibility.

Under the randomized settings, according to Definition 2, we need to consider the expected value of the total searching distance when computing the ratio and Kao *et al.* [36] proved the general representation of the average case competitive ratio given the SmartCow algorithm.

Theorem 3 ([36]) *For any fixed* $r > 1$, *the SmartCow algorithm has the competitive ratio of*

$$1 + \frac{2}{w} \times \frac{1 + r + r^2 + ... + r^{w-1}}{\ln r}.$$

It has been shown that the SmartCow algorithm is an optimal strategy that matches the lower bound of the 2-ray cow-path problem[36].

```
1   σ ← a random permutation of 0,1,2,...,w-1
2   ε ← a random real uniformly chosen from [0,1)
3   d ← r^ε, i ← 0
4   repeat
5     Explore path σ(i) up to distance d
6     If goal not found then return to origin
7     d ← d × r
8     i ← (i + 1) mod w
9   Until
10  goal found
```

Figure 2.1: The SmartCow algorithm proposed in [36]

Theorem 4 ([36]) *For w=2, the optimal competitive ratio is given by*

$$\min_{r>1}\left\{1+\frac{1+r}{\ln r}\right\}$$

Since this ratio is achievable by the SmartCow algorithm, therefore the SmartCow algorithm is optimal.

Note that for the 2-ray cow-path problem, the optimal geometric ratio r^* for the SmartCow algorithm is other than two, i.e. it is 3.59112. Additionally, the lower bound of average case competitive ratio given the SmartCow algorithm is $1+\frac{1+r^*}{\ln r^*}=4.59112$.

Therefore we have shown two lower bounds of the average case competitive ratio under both the deterministic and the randomized settings for the 2-ray cow-path problem. When the geometric ratio is two, the doubling strategy outweighs the SmartCow algorithm in terms of the average case competitive ratio by $\frac{5.32-5.27}{5.32}\approx 1\%$.

In addition, Kao *et al.* [36] conjectured that the SmartCow algorithm is also an optimal randomized search strategy for the general w-ray cow-path problem.

Conjecture 1 ([36]) *The optimal competitive ratio achievable by any algorithm for the w-ray cow-path problem is given by*

$$\min_{r>1}\left(1+\frac{2}{w}\times\frac{1+r+r^2+...+r^{w-1}}{\ln r}\right)$$

Since this is exactly the ratio achievable by SmartCow, under this conjecture, the SmartCow algorithm (with the appropriate minimizing r) is an optimal randomized algorithm.

It is an important open problem to determine the lower bound for randomized algorithms when $w \geq 3$, as stated in Conjecture 1.

2.2 Processor-Allocation Problem

Anytime algorithms [3, 13, 14, 21, 31, 42, 48, 49] are widely used in solving the Processor-Allocation problem. Anytime algorithms are the algorithms that are able to return a partial answer, whose quality depends on the amount of computation they were able to perform. The answer generated by anytime algorithms is an approximation of the correct answer. Anytime algorithms are widely used in solving real-world problems, such as game-playing programs [2, 36, 37], robotics searching and medical diagnosis systems.

There are two different types of anytime algorithms—*interruptible algorithms* and *contract algorithms*. Interruptible algorithms can be stopped at any time during their execution and provide current useful partial results, whereas contract algorithms provide valuable results only when being interrupted after a contract completion. If the contract algorithm is interrupted at the time before a contract completion, there is no result available to be used.

Although contract algorithms are less flexible than interruptible algorithms, it turns out that in terms of implementation, contract algorithms are much simpler. This provides the possibility of working on contract algorithms for online problems and then converting them to interruptible algorithms by applying a standard *black-box* transformation. The Processor-Allocation problem is a typical problem that uses the transformation described above. Consider the scenario as follows:

There are n problem instances and m parallel processors ready to execute the problems. We want to design an *interruptible algorithm*, which at any time, may be interrupted and queried for solutions of each problem instance with its corresponding *longest length* of the contract that has been completed. The problem is how to assign and schedule the executions of the various contract algorithms to the processors in a way that guarantees an efficient interruptible algorithm.

Schedules for the Processor-Allocation problem are evaluated by the *acceleration ratio*, which is a worst-case measure of efficiency. Let us consider the case described as follows in order to perceive what we mean by the *worst case*. Denote the length of the last contract which is completed for problem p_i as d_i^* and processor m_i started a new contract for problem p_i with length d_i at time t, where $d_i > d_i^*$. The worst case acceleration ratio for problem p_i will happen at $t + d_i - \varepsilon$, where ε is an arbitrarily small value, since intuitively, that is when processor m_i is ε-close to complete a longer contract (with better solution) for problem p_i, of which the current solution is still provided by the contract with smaller length (lower quality) of d_i^*.

Definition 3 ([42]) *Given a set P of n problem instances and a set M of m processors of identical speed, the acceleration ratio of a schedule X for P, denoted by $R_{m,n}(X)$ is defined as the smallest value r, with $r \geq 1$ such that for any allowable interruption time t, and any problem $p \in P$, we have that $l_X(p,t) \geq t/r$, where $l_X(p,t)$ is the length of the longest contract for problem p that has been completed by or at time t in X. Then the acceleration ratio for P and a set M of processors of identical speed is defined as*

$$R^*_{m,n} = \inf_X R_{m,n}(X).$$

A schedule X is optimal *if $R_{m,n}(X) = R^*_{m,n}$.*

Informally, the acceleration ratio indicates how much faster the processors in M should be in order to provide a solution of the same quality as an offline algorithm.

Figure 2.2 [42] illustrates an example of a schedule X for 2 problem instances and 4 processors. It is noticeable that the value of $\dfrac{t}{l_X(p,t)}$ peaks just before each contract is completed for each problem instance.

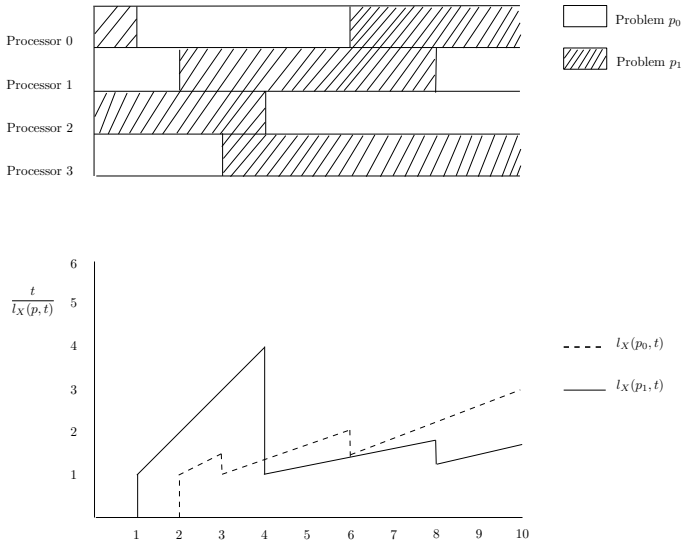


Figure 2.2: The top figure depicts a schedule of contracts for the case of 4 processors and 2 problems, for the first ten time units. The bottom figure depicts the plots of the function $\dfrac{t}{l_X(p,t)}$ for the two problems ($p \in \{p_0, p_1\}$). The acceleration ratio is the maximum value, on the y axis, attained by either curve, and in this example it is equal to 4 [42].

Early research [13] showed that a *cyclic* schedule, as defined below, is a good candidate for an optimal acceleration ratio for the Processor-Allocation problem. The properties are

- Problem-round-robin: The schedule arranges the problems in a round robin manner, which means that given a permutation of problems, the schedule executes each problem in the order according to the permutation of problems and iterates over it.
- Length-increasing: The lengths of contracts are arranged by the schedule in an increasing manner.
- Processor-round-robin: The schedule arranges each problem on a specific processor in a round robin manner, which means that given a permutation of processors, the schedule picks the processor in the order according to the permutation of processors and iterates over it.

The round-robin schedule of contract lengths $1, a, a^2, ...$ has the acceleration ratio that matches the lower bound for all possible *cyclic* schedules [13, 49], which is stated in the following theorem.

Theorem 5 ([13]) *The optimal acceleration ratio for m processors and n problems is*

$$\left(\frac{n}{m}\right)\left(\frac{m+n}{n}\right)^{\frac{m+n}{m}}$$

for all possible cyclic schedules.

Although a lower bound on the acceleration ratio is known for *cyclic* schedules, the case for general schedules remained open until López-Ortiz *et al.* proved the lower bound proposed in Theorem 5 is true for **all** schedules in [42].

Model Design 1 Model of the Processor-Allocation problem [42]

(p_i, d_i): `contract of length` d_i `for problem` p_i
(p_i, D_i): `the next contract after` (p_i, d_i) `in schedule` X
T_i: `when a processor is about to start contract` (p_i, D_i)
T_j: `when a processor is about to start contract` (p_j, D_j)

The key to López-Ortiz *et al.* [42] proof of the lower bound on the acceleration ratio among all schedules is the *swap* concept. Intuitively, when the schedule is about to choose one contract out of the two to start processing, it chooses the contract with a longer length to process than the one with a shorter length, which is shown in Technique 1.

Another swapping case is shown in Technique 2. It shows that if a contract a, with which the problem instance was executed by the last contract with longer duration, starts earlier than contract b, with which a different problem instance was executed by the last contract with shorter duration, it should be completed later than contract b. That is because we want to balance the value of the acceleration ratios over the two problem instances.

Technique 1 Longer contracts start earlier [42]

Given a schedule X and two problems p_i and p_j for which $d_j < d_i$ and $T_j > T_i$, then either $D_j < D_i$ or we can define a new schedule s.t. $D_j < D_i$, and whose acceleration ratio is no worse than that of the original schedule.

Technique 2 Longer contracts complete later [42]

Let $C_0 = (p_0, D_0)$ be a contract scheduled by X at time T_0 and $C_j = (p_j, D_j)$ be any contract happens after T_0. Then there exists another schedule of no worse acceleration ratio such that if $d_0 \geq d_j$ for a problem $p_j \neq p_0$ then $T_0 + D_0 \geq T_j + D_j$.

A schedule is *normalized* when being constructed using Technique 1 and 2. Additionally, normalized schedules have no worse acceleration ratio than un-normalized schedules. The workflow of the techniques to prove the lower bound acceleration ratio is shown in Figure 2.2.

By normalizing schedules and making use of the results of Gal [28] and Schuierer [46], López-Ortiz *et al.* [42] proved the lower bound for the acceleration ratio for **general** schedules as follows.

Theorem 6 ([42]) *Given n problem instances and m processors, every schedule that simulates an interruptible algorithm using executions of contract algorithms has an acceleration ratio no less than*

$$\left(\frac{n}{m}\right)\left(\frac{m+n}{n}\right)^{\frac{m+n}{m}}.$$

An interesting open problem is to find the optimal randomized algorithms (lower bound) for the Processor-Allocation problem. Instead of having contracts with pre-defined lengths, we can assign contract durations randomly. In addition, the speed of the processors can be different or assigned with random values.

2.3 Robots-Search-Rays Problem

The Robots-Search-Rays problem is a classic and thoroughly studied problem in robotics [4, 24, 29, 32]. In this set of problems, the agents are robots and the space being searched is two-dimensional and consists of rays, which start from an origin and extend to unlimited distances.

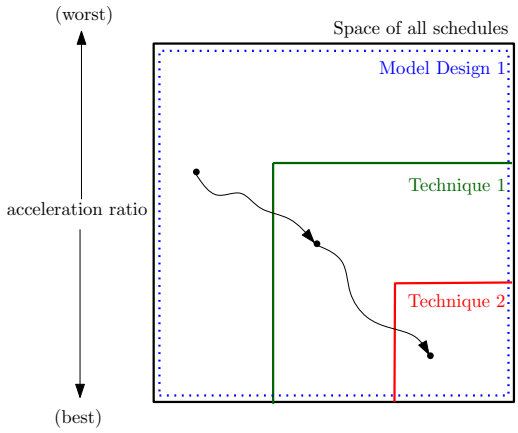


Figure 2.3: Illustration of the techniques for proving the lower bound in the Processor-Allocation problem [42]

Consider m robots standing at the origin of p concurrent rays. On one of the p rays, there is a target g, whose distance from the origin is unknown to the robots. The robot finds the target only when it stands on it. When $m = 1$, the Robots-Search-Rays problem is equivalent to the cow-path problem. Baeza-Yates *et al.* investigated searching on real line [6] and López-Ortiz *et al.* [43] proved the lower bound of the *competitive ratio* for the Robots-Search-Rays problem over all deterministic search strategies.

To formalize the problem, we use the model described in Model Design 2. The competitive ratio is the quotient of the search time over the distance of the target to the origin. Here we consider the robots have same maximal speed, which is without loss of generality, the ratio of unit distance and unit time.

Since our goal is to find the optimal searching strategy with the best competitive ratio, therefore, we need to define the properties which optimal strategies should have. Clearly, having more than one robot searching on the same ray is not optimal since resources are wasted in this case. Additionally, removing the robots' idle periods from the robot schedule never increase the competitive ratio.

Intuitively, when a robot returns to the origin and is ready to start searching another ray, it should choose the ray which is least explored since we want to make every ray searched in a balanced way given the same possibility of containing the target. This rule is defined in Technique 3 and proved to be optimal by López-Ortiz *et al.* [43] among general search strategies.

A strategy satisfying Model Design 3 and Technique 3 is termed a *normalized strategy*. It has the worst case competitive ratio defined as follows:

Model Design 2 Model of the Robots-Search-Rays problem [43]

Unit Speed: All robots have the same maximal speed---
one unit of distance per unit of time
$X_S = (x_0, x_1, \dots)$: robots' turn points
ordered by time
r_i: the ray on which the robot that turns at
x_i is located
T_i: the first time that a robot passes x_i
on ray r_i again after the last time it
was searching the same ray
competitive ratio: $\sup_{i \geq 0} \left\{ \frac{T_i}{x_i} \right\}$
occupied: ray r is occupied at time T if there
is a robot on r at that time
busy: ray r is busy at time T if there
is a robot on r that is moving away
from the origin at that time

Model Design 3 Properties of Optimal Searching Strategies [43]

Let S be a strategy to search on p rays with m robots.
Then there exists a strategy $S2$ with the same competitive
ratio or better such that
(1) At any time t, there is at most one robot on a given ray.
(2) If a robot moves towards the origin on some ray, then it
continues until it has reached the origin.
(3) All robots are moving at all times.

Technique 3 Search Least Explored Ray First [43]

There is an optimal on-line strategy to search on p rays
with m robots that satisfies Model Design 3
such that if a robot is located at the origin at time T,
then it chooses to explore the ray that has been explored
the least among all non-busy rays.

Theorem 7 ([43]) *The worst case competitive ratio of an optimal normalized strategy with turn point sequence* $X = (x_0, x_1, \ldots)$ *is at least*

$$\sup_{k \geq 0} \left\{ 1 + 2 \sum_{i=0}^{k+p-m} x_i^s \Big/ \sum_{i=k-m+1}^{k} x_i^s \right\} \tag{2.3}$$

where p *is the number of rays,* m *is the number of robots and* $X^s = (x_0^s, x_1^s, \ldots)$ *is the sequence of the sorted values of* X *and* $x_i^s := 0$ *if* $i < 0$.

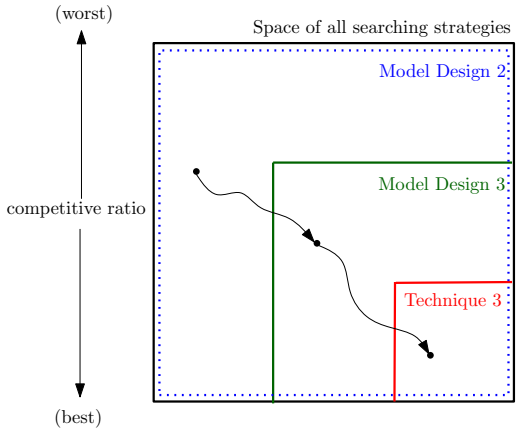


Figure 2.4: Illustration of the techniques for proving the lower bound in the Robots-Search-Rays problem

Note here the restrictions do not mean that only normalized strategies can be optimal. Indeed there are non-normalized strategies which are optimal. However, one can always obtain a normalized strategy from those non-normalized ones with the same or smaller competitive ratio.

By normalizing strategies and making use of the results of Gal [28] and Schuierer [46], López-Ortiz *et al.* [43] proved the lower bound of the competitive ratio for the Robots-Search-Rays problem among all the deterministic search strategies, which is described as follows.

Theorem 8 ([43]) *There is no search strategy for a target on* p *rays using* m *robots with a competitive ratio of less than*

$$1 + 2\left(\frac{p}{m} - 1\right)\left(\frac{p}{p-m}\right)^{p/m}.$$

An optimal exploration algorithm

A classic strategy that achieves a competitive ratio matching the lower bound in Theorem 8 is a cyclic search strategy with the exponential geometric ratio b [43]. The strategy works as follows. The robots explore the rays in a fixed cyclic order. Let $b = (p/(p-m))^{1/m}$. The sequence of turn distances of the robots is given by $x_i = b^i$ for $i = 0, 1, 2, \ldots$. The kth time that robot R returns to the origin it chooses to explore ray $(km + R) \mod p$ up to distance x_{km+R}.

Notice that this optimal strategy is similar to the one for the Processor-Allocation problem, which is cyclic and has contracts assigned with lengths $1, a, a^2, \ldots$. In the next section, we will discuss the connections between these two problems.

2.4 Connections of Three Online Problems

Clearly, the Robots-Search-Rays problem is a generalization of the cow-path problem. Now we will highlight the connections between the Processor-Allocation problem and the Robots-Search-Rays problem.

The connection was first established by Kao *et al.* [36, 37] for the case of a single searcher in the randomized case, as well as certain multi-searcher scenarios. The work of Bernstein *et al.* [13] drew a similar connection between scheduling contract algorithms and robot searching on a set of rays. It turns out that interesting parallels can be drawn for the two problems: informally, the rays correspond to problem instances, the robots to processors, and the (unknown) location of the target corresponds to the (also unknown) query time. Refer to Figure 2.4. By looking into the two problems, we also found the similar intuitions behind the normalization of schedules for the Processor-Allocation problem, and the normalization of search strategies for the Robots-Search-Rays problem.

Note here, it is not a two-direction transformation which means that we can transfer the Processor-Allocation problem to the Robots-Search-Rays problem but we cannot transfer it back. To demonstrate this, we just need to look at one simple example, e.g we can benefit from multiple processors working on contracts for the same problem at the same time. However multiple robots searching the same ray provides no benefit at the same time.

Connections	Processor-Allocation Problem	Robots-Search-Rays Problem
Agent	Processors	Robots
Media	Problem Instances	Rays
Geometric Metric	Contracts	Search Extents
Evaluation Metric	Acceleration Ratio	Competitive Ratio
Normalized Strategy	The normalized strategy pushes forward the completion time by processing the problem with longer contract duration earlier.	The normalized strategy pushes forward the searching process by exploring the ray with smaller searched distance earlier.
Lower Bound	$\left(\frac{n}{m_1}\right)\left(\frac{m_1+n}{n}\right)^{\frac{m_1+n}{m_1}}$	$1+2\left(\frac{p-m_2}{m_2}\right)\left(\frac{p}{p-m_2}\right)^{p/m_2}$

Figure 2.5: Connections of the Processor-Allocation Problem and the Robots-Search-Rays Problem, in order to distinguish the notations, we denote m_1 as the number of processors, n as the number of problems, p as the number of rays and m_2 as the number of robots.

Chapter 3

One Robot Searches Two Rays And Weighted Problem

In this chapter, we study the One-Robot-Searches-Two-Rays-And-Weighted (1-STRAW) problem. Here by *weighted*, we refer to the different probabilities of the target being located on either of the two rays. This problem is new to the Robots-Search-Rays problem and the lower bound of the competitive ratio has never been proved before. In this chapter, we compute and prove a tight lower bound on the worst case competitive ratio and conjecture a lower bound on the average case competitive ratio. This chapter consists of three parts. Firstly, we define the 1-STRAW problem in a formal way and propose a search strategy for this problem. Secondly, we compute the worst case competitive ratio of our proposed search strategy and prove its tight lower bound. Thirdly, we analyze the average case competitive ratio given our search strategy.

3.1 Problem Definition and Search Strategy

3.1.1 Problem Definition

In the 1-STRAW problem, we investigate searching on two concurrent rays for a point target g located at some unknown distance along one of the two rays with different locating probabilities. A robot moving at unit speed searches for the target using a strategy S. The probability of the target located on the first ray is p and the probability of it located on the second ray is q. Therefore we have $p + q = 1$. Without loss of generality, we define $high = \max\{p, q\}$ and $low = \min\{p, q\}$.

Now let us recall the 2-ray cow-path problem, which is equivalent to the problem of One-Robot-Searches-Two-Rays in the unweighted case. Here we define a cyclic search strategy with a geometric ratio b for the 2-ray cow-path problem as S^u, where u stands for *unweighted*. We

define the worst case competitive ratio given the cyclic search strategy as $C_{S^u(w)}$ and we have

$$C_{S^u(w)} = 1 + 2 \cdot \lim_{k\to\infty} \left\{ \frac{\sum_{i=0}^{k+1} b^i}{b^k + \varepsilon} \right\} \tag{3.1}$$

which results in $C_{S^u(w)} = 1 + \frac{2b^2}{b-1}$.

Now, let us compute the worst case competitive ratio $C_{S^u(w)}$ in a different way. We denote the two rays as R_1 and R_2 and the worst case competitive ratio as $C^1_{S^u(w)}$ when the target is found on ray R_1. The worst case competitive ratio is denoted by $C^2_{S^u(w)}$ when the target is found on ray R_2. Since the target selects a location on either one of the two rays with the same probability, we have $C_{S^u(w)} = 0.5 \times C^1_{S^u(w)} + 0.5 \times C^2_{S^u(w)}$. In addition, we know that $C^1_{S^u(w)} = C^2_{S^u(w)} = 1 + \frac{2b^2}{b-1}$. Therefore $C_{S^u(w)} = 1 + \frac{2b^2}{b-1}$.

This gives us an indication of how to compute the worst case competitive ratio for 1-STRAW, which is denoted as $C_{S(w)}$. Here we denote the ray with the probability of *high* as R_H and the ray with the probability of *low* as R_L. In addition, we use $C^H_{S(w)}$ as the worst case competitive ratio when the target is on R_H and $C^L_{S(w)}$ as the worst case competitive ratio when the target is on R_L.

Definition 4 *The worst case competitive ratio $C_S(w)$ for 1-STRAW is*

$$C_S(w) = high \times C^H_{S(w)} + low \times C^L_{S(w)}$$

where $C^H_{S(w)}$ and $C^L_{S(w)}$ represent the worst case competitive ratios on the ray with the probability of high and on the ray with the probability of low respectively.

In the case of the real line, the competitive ratio for a *given target point* g is the distance traversed by the robot divided by the distance from the origin to the target position, that is $C_S(g) = \frac{\widehat{C}_S(g)}{|g|}$, where $\widehat{C}_S(g)$ denotes the total search distance by the robot. Additionally, we are interested in both the worst case and the average case performance under this measure.

Without loss of generality, we can assume that the robot starts from the origin, which simplifies the description of a strategy. In addition, we study the case where the target point is not located within a distance ϵ of the robot starting position. Again, this is a natural restriction, as otherwise the target can hide infinitesimally close to the starting point and on the opposite side of the first search move by the robot, resulting in an unbounded competitive ratio. Therefore, in the study, we set $\epsilon = 1$.

Definition 5 *The average case competitive ratio of a randomized strategy S given the target location range $[d_1, d_2]$ is defined as $C_S(a) = \frac{\int_{d_2-d_1} \widehat{C}(S,g)/|g|dg}{d_2 - d_1}$, where g is the possible target location chosen from $[d_1, d_2]$.*

3.1.2 Proposed Search Strategy

Now that we have the definitions of the 1-STRAW problem and the performance measures of both the worst case competitive ratio (Definition 1) and the average case competitive ratio (Definition 5), we propose a search strategy.

```
1   high ← max {p, 1 − p}
2   low ← min {p, 1 − p}
3   R_H ← 0          /* the ray with probability high */
4   R_L ← 1          /* the ray with probability low */
5   σ ← {R_H, R_L}
6   d_H ← f(p) · b^0
7   d_L ← b^1
8   i ← 0
9   repeat
10      if (i mod 2 == 0)
11          Explore Path σ(0) up to distance d_H
12          if (target is not found)
13              d_H ← d_H × b^2
14              Robot returns to the origin
15      else if (i mod 2 == 1)
16          Explore Path σ(1) up to distance d_L
17          if (target is not found)
18              d_L ← d_L × b^2
19              Robot returns to the origin
20      i + +
21  Until target is found
```

Figure 3.1: Proposed Searching Strategy for the 1-STRAW problem

In this strategy, we inherit the geometric ratio of the cyclic strategy [43]. However, we introduce a ratio function $f(p)$ to accelerate the search distances on the ray with higher probability, which is a function of the probability p. Intuitively, we want the robot to search for longer dis-

tances on the ray with higher probability because there is a better chance that the robot will reach the target there. In the next section, we will obtain the optimal representation of $f(p)$ to get the lower bounds of the worst case and the average case competitive ratio.

3.2 Worst Case Competitive Ratio

In the previous section, we defined the 1-STRAW problem and proposed a search strategy. In this section, we compute its worst case competitive ratio, introduce the representation of the optimal ratio function $f(p)$ and prove the lower bound of the worst case competitive ratio of the 1-STRAW problem.

We observe that the worst case competitive ratio grows bigger as the number of search rounds increases, which means the worst case competitive ratio in the last round is the largest out of all the worst case competitive ratios at all the former rounds, shown in Figure 3.2 and 3.3. The following lemmas prove this point.

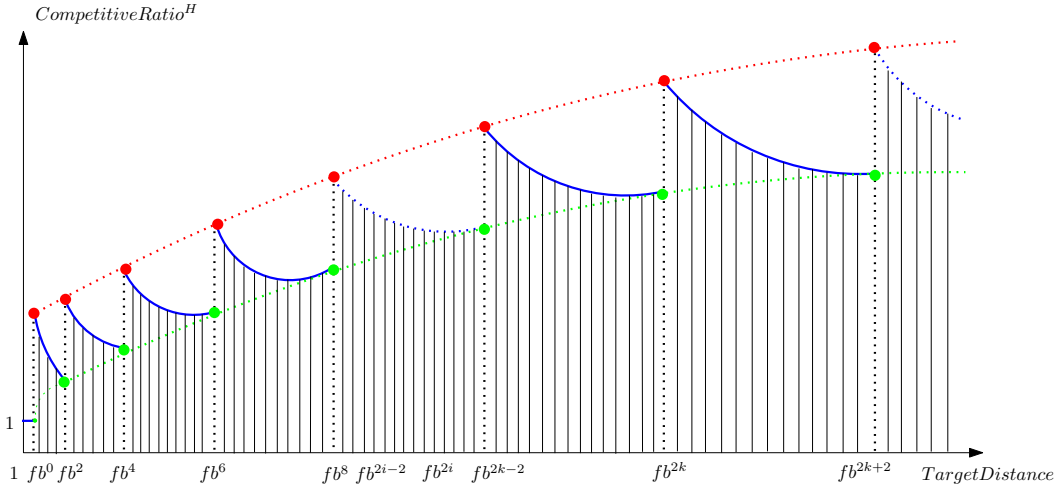


Figure 3.2: General case of C_S^H

Here we have the following settings:

Index i is the accumulator of rounds. $C_{S(w)}^H(i)$ represents the worst case competitive ratio at round $2i+2$ when the target is on R_H. $C_{S(w)}^L(i)$ represents the worst case competitive ratio at round $2i+3$ when the target is on R_L.

Lemma 1 *$C_{S(w)}^H(i)$ is a monotonically increasing function of i.*

Proof. We here introduce a small number ε. The worst case occurs when the target is located just past the end point of the searching distance at round $2i$, which is $f(p) \cdot b^{2i} + \varepsilon$.

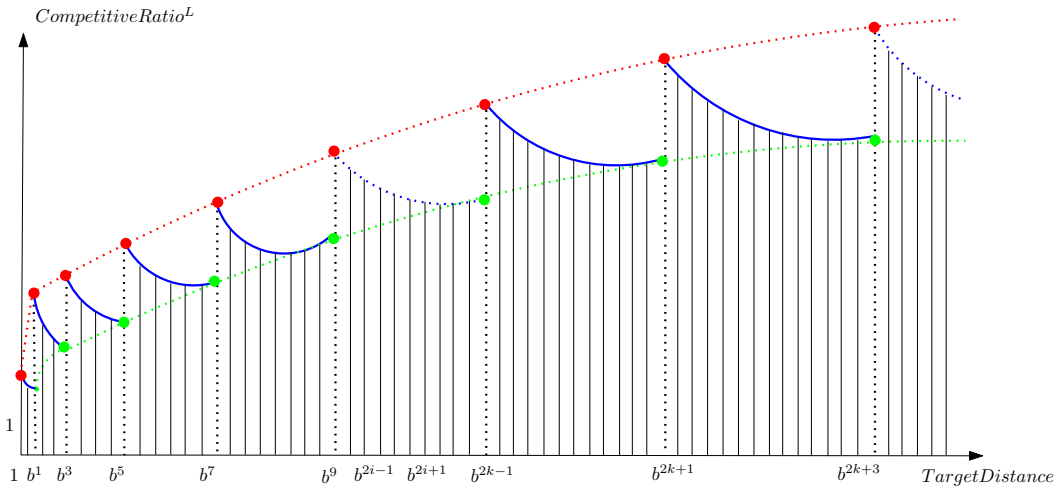


Figure 3.3: General case of C_S^L

Therefore we have

$$C_{S(w)}^H(i) = \frac{2 \cdot \sum_{j=0}^{i} f(p) \cdot b^{2j} + 2 \cdot \sum_{j=0}^{i} b^{2j+1} + (f(p) \cdot b^{2i} + \varepsilon)}{f(p) \cdot b^{2i} + \varepsilon}$$

$$= 1 + \frac{2f(p)(b^{2i+2} - 1) + 2b(b^{2i+2} - 1)}{(b^2 - 1)(f(p) \cdot b^{2i} + \varepsilon)}$$

$$= 1 + \frac{2f(p) + 2b}{b^2 - 1} \cdot \frac{b^2 - \frac{1}{b^{2i}}}{f(p) + \frac{\varepsilon}{b^{2i}}}$$

Recall that ε is arbitrarily small, so we have $\frac{\varepsilon}{b^{2i}} \to 0$ and hence

$$C_{S(w)}^H(i) = 1 + \frac{2f(p) + 2b}{b^2 - 1} \cdot \frac{b^2 - \frac{1}{b^{2i}}}{f(p)}$$

Therefore, $C_{S(w)}^H(i)$ is a monotonically increasing function of i. ∎

Lemma 2 *$C_{S(w)}^L(i)$ is a monotonically increasing function of i.*

Proof. We denote $C^L_{S(w)}(i)$ as the worst case competitive ratio at round $2i+3$ when the target is on R_L. The worst case occurs when the target is located at the closest place to the end point of the searching distance at round $2i+1$, which is $b^{2i+1}+\varepsilon$.

Therefore we have

$$\begin{aligned} C^L_{S(w)}(i) &= \frac{2\cdot\sum_{j=0}^{i+1} f(p)\cdot b^{2j} + 2\cdot\sum_{j=0}^{i} b^{2j+1} + (b^{2i+1}+\varepsilon)}{b^{2i+1}+\varepsilon} \\ &= 1 + \frac{2f(p)\left(b^3 - \frac{1}{b^{2i+1}}\right) + 2b\left(b - \frac{1}{b^{2i+1}}\right)}{(b^2-1)\left(1+\frac{\varepsilon}{b^{2i+1}}\right)} \end{aligned}$$

Since ε is arbitrarily small, so we have $\frac{\varepsilon}{b^{2i+1}} \to 0$ and hence

$$C^L_{S(w)}(i) = 1 + \frac{2f(p)\left(b^3 - \frac{1}{b^{2i+1}}\right) + 2b\left(b - \frac{1}{b^{2i+1}}\right)}{b^2-1}$$

Therefore, $C^L_{S(w)}(i)$ is a monotonically increasing function of i. ∎

The lemmas above allow us to use the worst case competitive ratio at the last search round as the worst case competitive ratio for the 1-STRAW problem.

Now we will compute the worst case competitive ratio for a general function $f(p)$, given the probability p with which the target is located on one of the rays and the geometric ratio b. By the general function $f(p)$, we mean that nothing is known for $f(p)$ other than the fact that $f(p)$ is a function of the probability of p. Later on, we will decide how to choose $f(p)$ in order to get the best efficiency out of the search strategy.

Theorem 9 *The worst case competitive ratio for the 1-STRAW problem given the general function $f(p)$ is:*

$$C_{S(w)} = \begin{cases} \frac{3b^2-1}{b^2-1} + \frac{2b^3}{b^2-1}\left[pf(p) + \frac{1-p}{f(p)}\right] & p \in [0, 0.5] \\ \frac{3b^2-1}{b^2-1} + \frac{2b^3}{b^2-1}\left[(1-p)f(p) + \frac{p}{f(p)}\right] & p \in [0.5, 1] \end{cases}$$

Proof. There are two cases for the location of the target. One is that the target is located on R_H and the worst case competitive ratio is $C^H_{S(w)}$. The second case is that the target is located on R_L and the worst case competitive ratio is $C^L_{S(w)}$. In addition according to Definition 4, we have $C_{S(w)} = high \times C^H_{s(w)} + low \times C^L_{S(w)}$.

The worst case is that the target is located at the point which is ε away from the end point of the last search distance on this ray. For the case that the target is located on R_H, the target distance will be $f(p) \times b^{2k} + \varepsilon$, given our searching strategy. For the case that the target is located on R_L, the target distance will be $b^{2k+1} + \varepsilon$. Now we will compute $C^H_{S(w)}$ and $C^L_{S(w)}$ respectively.

Case 1: Target is on R_H, target distance is $f(p) \times b^{2k} + \varepsilon$

We denote the total distance the robot traveled as D_H and we have

$$\begin{aligned} D_H &= 2 \cdot \sum_{i=0}^{k} f(p) \cdot b^{2i} + (f(p) \cdot b^{2k} + \varepsilon) + 2 \cdot \sum_{i=0}^{k} b^{2i+1} \\ &= f(p) \cdot \frac{2b^{2k+2} - 2}{b^2 - 1} + f(p) \cdot \frac{b^{2k+2} - b^{2k}}{b^2 - 1} + \frac{2b^{2k+3} - 2b}{b^2 - 1} + \varepsilon \\ &= f(p) \cdot \frac{3b^{2k+2} - b^{2k} - 2}{b^2 - 1} + \frac{2b^{2k+3} - 2b}{b^2 - 1} + \varepsilon \end{aligned}$$

Given Definition 1 and Lemma 1, we have

$$\begin{aligned} C^H_{S(w)} &= \lim_{k \to \infty} \left\{ \frac{D_H}{f(p) b^{2k} + \varepsilon} \right\} \\ &= \lim_{k \to \infty} \left\{ \frac{\dfrac{3b^2 - 1 - \dfrac{2}{b^{2k}}}{b^2 - 1} + \dfrac{2b^3 - \dfrac{2b}{b^{2k}}}{(b^2 - 1) \cdot f(p)} + \dfrac{\varepsilon}{f(p) \cdot b^{2k}}}{1 + \dfrac{\varepsilon}{f(p) \cdot b^{2k}}} \right\} \\ &= \frac{3b^2 - 1}{b^2 - 1} + \frac{2b^3}{b^2 - 1} \cdot \frac{1}{f(p)} \end{aligned}$$

Case 2: Target is on R_L, target distance is $b^{2k+1} + \varepsilon$

$$\begin{aligned} D_L &= 2 \cdot \sum_{i=0}^{k+1} f(p) \cdot b^{2i} + 2 \cdot \sum_{i=0}^{k} b^{2i+1} + (b^{2k+1} + \varepsilon) \\ &= f(p) \cdot \frac{2b^{2k+4} - 2}{b^2 - 1} + \frac{3b^{2k+3} - b^{2k+1} - 2b}{b^2 - 1} + \varepsilon \end{aligned}$$

Given Definition 1 and Lemma 2, we have

$$
\begin{aligned}
C^L_{S(w)} &= \lim_{k\to\infty}\left\{\frac{D_L}{b^{2k+1}+\varepsilon}\right\} \\
&= \lim_{k\to\infty}\left\{\frac{f(p)\cdot\dfrac{2b^3-\dfrac{2}{b^{2k+1}}}{b^2-1}+\dfrac{3b^2-\dfrac{2b}{b^{2k+1}}-1}{b^2-1}+\dfrac{\varepsilon}{b^{2k+1}}}{1+\dfrac{\varepsilon}{b^{2k+1}}}\right\} \\
&= \frac{3b^2-1}{b^2-1}+\frac{2b^3}{b^2-1}\cdot f(p)
\end{aligned}
$$

Now we can compute $C_{S(w)} = high \times C^H_{s(w)} + low \times C^L_{S(w)}$.

$$
C_{S(w)} = \frac{3b^2-1}{b^2-1}+\frac{2b^3}{b^2-1}\cdot\{f(p)\cdot low+\frac{high}{f(p)}\}
$$

which is

$$
C_{S(w)} = \begin{cases} \dfrac{3b^2-1}{b^2-1}+\dfrac{2b^3}{b^2-1}\left[pf(p)+\dfrac{1-p}{f(p)}\right] & p\in[0,0.5] \\ \dfrac{3b^2-1}{b^2-1}+\dfrac{2b^3}{b^2-1}\left[(1-p)f(p)+\dfrac{p}{f(p)}\right] & p\in[0.5,1] \end{cases}
$$

This completes the proof. ∎

3.2.1 Lower Bound of Worst Case Competitive Ratio

Representation of Ratio Function

So far we have obtained the expression of the worst case competitive ratio of the 1-STRAW problem given an unknown ratio function $f(p)$. What we do next is to find a proper formula for the ratio function $f(p)$. By proper, we mean that, out of all the possible representations of $f(p)$, the final $f(p)$ we choose minimizes the value of $C_{S(w)}$, given the same geometric ratio b and the same probability p. We additionally prove a theorem which indicates that the ratio function $f(p)$ we choose is the best function out of any possible choice of $f(p)$, in terms of getting the smallest value of worst case competitive ratio for the 1-STRAW problem.

The requirements for the ratio function $f(p)$ are to be considered carefully. Let us examine the search strategy at a higher level. The robot should search a longer distance on the ray with higher probability because the target is more likely to be on this ray. This is where the ratio function $f(p)$ comes in. It is also the reason why there is no need to have a ratio function $f(p)$ for the 2-ray cow-path problem since the two rays have equal probability of having the target

located on them. Therefore, the first requirement for ratio function is $f(p) \geq 1$ (when $p = 0.5$, $f(p) = 1$), which makes the searching distance longer than the ones in the unweighted case.

Another requirement can be observed from the representation of worst case competitive ratio $C_{S(w)}$, which is

$$C_{S(w)} = \frac{3b^2 - 1}{b^2 - 1} + \frac{2b^3}{b^2 - 1} \cdot \left\{ f(p) \cdot low + \frac{high}{f(p)} \right\}$$

Both $f(p) \cdot low$ and $\frac{high}{f(p)}$ should be functions of the probability p with boundaries. This is a straightforward requirement since if either of the function goes to infinity, the worst case competitive ratio will be unbounded, which makes $C_{S(w)}$ meaningless since we desire the smallest possible worst case competitive ratio.

Under the conditions of the two requirements above, we proposed the first candidate of ratio function $f(p)$, which is $f(p) = \frac{high}{low}$. Intuitively, this makes a lot sense. The ratio function $f(p) = \frac{high}{low}$ is bigger than or equal to 1. When $low \to 0$, $f(p) \to \infty$, which means, if the target is on R_H for sure, then the robot will search on R_H to an unbounded distance. This will make the robot get to the target at its first try and the worst case competitive ratio will be 1.

However, if we set $f(p) = \frac{high}{low}$, then we have

$$f(p) \cdot low + \frac{high}{f(p)} = 1$$

Therefore,

$$C_{S(w)} = \frac{3b^2 - 1}{b^2 - 1} + \frac{2b^3}{b^2 - 1} \cdot \left\{ f(p) \cdot low + \frac{high}{f(p)} \right\} = 1 + \frac{2b^2}{b - 1}$$

which cancels out the probability p and is equivalent to the worst case competitive ratio of the 2-ray cow-path problem. This means that, no matter what the probability p is, if we follow the proposed search strategy, we will always get the worst case competitive ratio with the same value as the one in the unweighted case.

The result shows that the *natural* weight function $\frac{high}{low}$ is not correct. So the next question is: does there exist any other representation of $f(p)$, which makes $f(p) \cdot low + \frac{high}{f(p)} \leq 1$?

Ideally, we want to find

$$\min \left\{ f(p) \cdot low + \frac{high}{f(p)} \right\}$$

Now we will introduce a theorem, which takes us one step closer to get the best representation of $f(p)$.

Here we define $g(f(p)) = f(p) \cdot p + \frac{q}{f(p)}$, we propose the theorem described as follows.

Theorem 10 *The optimal ratio function $f(p)$ which makes $g(f(p))$ minimal is*

$$f(p) = \sqrt{\frac{q}{p}}$$

Proof. First, we compute the derivative of $g(f(p))$ over $f(p)$, we have

$$\frac{d}{df(p)} g(f(p)) = p - \frac{q}{f(p)^2} \tag{3.2}$$

Assign Equation 3.2 to zero, we have

$$f(p) = \sqrt{\frac{q}{p}}$$

Now we will prove that $f(p) = \sqrt{\frac{q}{p}}$ is the optimal choice to make $g(f(p))$ minimal.

For any $x \neq \sqrt{\frac{q}{p}}$, we have

$$\begin{aligned} g(x) - g\left(\sqrt{\frac{q}{p}}\right) &= x \cdot p + \frac{q}{x} - p \cdot \sqrt{\frac{q}{p}} - q \cdot \sqrt{\frac{p}{q}} \\ &= \frac{x^2 \cdot p + q}{x} - \frac{x \cdot 2\sqrt{pq}}{x} \\ &= \frac{(\sqrt{p} \cdot x - \sqrt{q})^2}{x} \end{aligned}$$

Since we only consider any $x > 1$, therefore we have

$$g(x) - g\left(\sqrt{\frac{q}{p}}\right) > 0$$

And $f(p) = \sqrt{\frac{q}{p}}$ makes $g(f(p))$ minimal, which value is $2\sqrt{pq}$ ∎

Corollary 1 *$C_{S(w)}$ achieves the smallest value when $f(p) = \frac{\sqrt{high}}{\sqrt{low}}$, given the same choice of geometric ratio b and probability p.*

Proof. Refer to Theorem 10. ∎

Therefore, we have the final representation of the worst case competitive ratio $C_{S(w)}$ for the 1-STRAW problem. Since it is a function of the geometric ratio b and the probability p, we denote the function of the worst case competitive ratio for the 1-STRAW problem as $C_{S(w)}(b, p)$ and it equals to

$$C_{S(w)}(b, p) = \frac{3b^2 - 1}{b^2 - 1} + \frac{4b^3}{b^2 - 1} \cdot \left\{\sqrt{p \cdot (1 - p)}\right\}$$

Lower bound

In this section, we will compute the lower bound of the worst case competitive ratio $C_{S(w)}(b,p)$. The shape of $C_{S(w)}$ is shown in Figure 3.4. Our goal is to find the optimal geometric ratio b given the probability p, which ensures that our proposed search strategy has the smallest worst case competitive ratio. Now that we have the representation of $C_{S(w)}(b,p)$, which makes finding the lower bound easier. What we will do is to compute the derivatives of $C_{S(w)}(b,p)$.

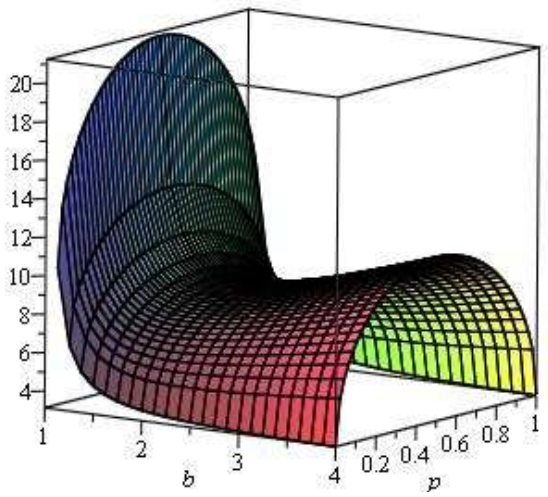


Figure 3.4: $C_{S(w)}(b,p)$

There are two partial derivatives for $C_{S(w)}(b,p)$, which are respectively the partial derivative over p, denoted as $\frac{dC_{S(w)}(b,p)}{dp}$ and the partial derivative over b, denoted as $\frac{dC_{S(w)}(b,p)}{db}$.

We will compute the partial derivative over p first since it is more straightforward. Because we have

$$C_{S(w)}(b,p) = \frac{3b^2-1}{b^2-1} + \frac{4b^3}{b^2-1} \cdot \left\{ \sqrt{p \cdot (1-p)} \right\}$$

when we compute the partial derivative over p, we regard b as a constant. We introduce two constants c_1 and c_2 with $c_1 = \frac{3b^2-1}{b^2-1}$ and $c_2 = \frac{2b^3}{b^2-1}$. Therefore the function $C_{S(w)}(b,p)$ can be re-written as

$$C_{S(w)}(p) = c_1 + c_2 \cdot 2\sqrt{p \cdot (1-p)}$$

We denote $g(p) = 2\sqrt{p \cdot (1-p)}$, as it is shown in Figure 3.5. Since c_1 and c_2 are constants and $C_{S(w)}(p) = c_1 + c_2 \cdot 2\sqrt{p \cdot (1-p)}$, therefore $C_{S(w)}(p)$ has the same symmetric shape as $g(p)$.

Given

$$\frac{d}{dp}C_{S(w)} = \frac{2b^3(1-2p)}{(b^2-1)\sqrt{p\cdot(1-p)}}$$

when $\frac{dC_{S(w)}}{dp} = 0$, we have $p = 0.5$. Therefore the maximum worst case competitive ratio is achieved at $p = 0.5$ and

$$C_{S(w)}(b, 0.5) = \frac{2b^3 + 3b^2 - 1}{b^2 - 1}$$

Because of the symmetry of $C_{S(w)}(p)$, the minimum worst case competitive ratio is achieved at either $p = 0$ or $p = 1$ and

$$C_{S(w)}(b, 0) = C_{S(w)}(b, 1) = \frac{3b^2 - 1}{b^2 - 1}$$

Intuitively, this is as expected. When the probability of the target located on one ray is close to 1, which means almost surely that the target is on this ray, then our proposed strategy will make the robot search on this ray first for a long distance, which ensures that the robot finds the target as soon as possible, thus making the worst case competitive ratio as small as possible. When the target is located on either of the rays with the same probability, then our strategy results in an unbiased search and the worst case competitive ratio will be the highest among all cases with non-equal probabilities.

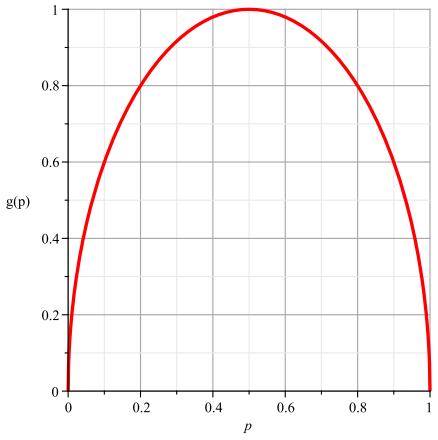


Figure 3.5: $g(p) = 2\sqrt{p\cdot(1-p)}$

Now let us consider the partial derivative of $C_{S(w)}(b, p)$ over b. In this case, the probability p is regarded as a constant. We have

$$\frac{dC_{S(w)}(b,p)}{db} = \frac{6b}{b^2-1} - \frac{2b(3b^2-1)}{(b^2-1)^2} + \frac{12b^2\sqrt{p\cdot(1-p)}}{b^2-1} - \frac{8b^4\sqrt{p\cdot(1-p)}}{(b^2-1)^2}$$

When $\frac{dC_{S(w)}(b,p)}{db} = 0$, we have

$$\sqrt{p \cdot (1-p)} = \frac{1}{b^3 - 3b}$$

Therefore, given the equation of $\sqrt{p \cdot (1-p)} = \frac{1}{b^3 - 3b}$, we can represent p as $y(b)$. In addition, we can compute $C_{S(w)}(b, y(b))$ as

$$\begin{aligned} C_{S(w)}(b, y(b)) &= \frac{3b^2 - 1}{b^2 - 1} + \frac{4b^3}{b^2 - 1} \cdot \sqrt{p \cdot (1-p)} \\ &= 3 + \frac{6}{b^2 - 3} \end{aligned}$$

Therefore,

$$C_{S(w)}(b, y(b)) = 3 + \frac{6}{b^2 - 3} \tag{3.3}$$

In order to distinguish this special b which makes $\frac{dC_{S(w)}(b,p)}{db} = 0$, we denote this b as b^*. Now we will prove that this $C_{S(w)}(b^*, y(b^*))$ is the minimal value out of any possible $C_{S(w)}(b, p)$, given the same p.

Theorem 11 *$C_{S(w)}(b^*, y(b^*))$ is the smallest value out of any $C_{S(w)}(b, y(b^*))$.*

Proof.

$$\begin{aligned} \because C_{S(w)}(b^*, y(b^*)) &= 3 + \frac{6}{(b*)^2 - 3} \\ C_{S(w)}(b, y(b^*)) &= \frac{3b^2 - 1}{b^2 - 1} + \frac{4b^3}{b^2 - 1} \cdot \sqrt{p \cdot (1-p)} \\ &= \frac{3b^2 - 1}{b^2 - 1} + \frac{4b^3}{b^2 - 1} \cdot \frac{1}{(b^*)^3 - 3b^*} \end{aligned}$$

$\therefore C_{S(w)}(b^*, y(b^*)) - C_{S(w)}(b, y(b^*))$

$$\begin{aligned} &= 3 + \frac{6}{(b^*)^2 - 3} - \left[\frac{3b^2 - 1}{b^2 - 1} + \frac{4b^3}{b^2 - 1} \cdot \frac{1}{(b^*)^3 - 3b^*} \right] \\ &= \frac{2}{[(b^*)^3 - 3b^*] \cdot (b^2 - 1)} \cdot \left[(2b^* + b^*)b^2 - (b^*)^2 \cdot b^* - 2b^2 \cdot b \right] \\ &= \frac{2}{[(b^*)^3 - 3b^*] \cdot (b^2 - 1)} \cdot \Big\{ (b^* - b) \cdot [b(b+b) - b^*(b^* + b)] \Big\} \end{aligned}$$

There are two cases when $b \neq b^*$

Case 1: if $b > b^*$, then $b^* - b < 0, b(b+b) - b^*(b^*+b) > 0$

Therefore, $C_{S(w)}(b^*, y(b^*)) - C_{S(w)}(b, y(b^*)) < 0$,

$\therefore C_{S(w)}(b^*, y(b^*)) < C_{S(w)}(b, y(b^*))$

Case 2: if $b < b^*$, then $b^* - b > 0, b(b+b) - b^*(b^*+b) < 0$

Therefore, $C_{S(w)}(b^*, y(b^*)) - C_{S(w)}(b, y(b^*)) < 0$,

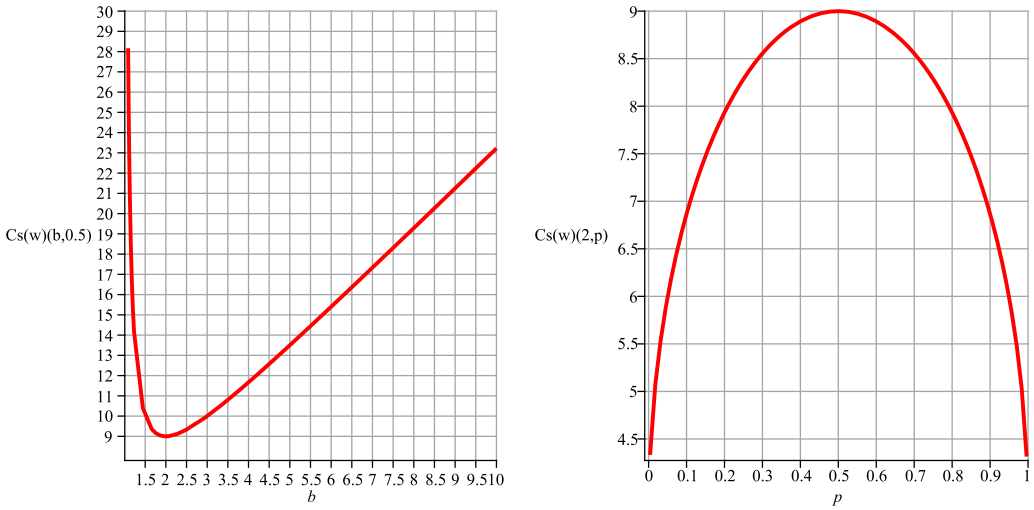
$\therefore C_{S(w)}(b^*, y(b^*)) < C_{S(w)}(b, y(b^*))$

$C_{S(w)}(b^*, y(b^*))$ is the smallest value out of any $C_{S(w)}(b, y(b^*))$. ∎

Now given the probability p, we can get the optimal b to reach the lower bound of $C_{S(w)}(b, p)$. The equation is

$$\sqrt{p \cdot (1-p)} = \frac{1}{b^3 - 3b}$$

And the lower bound of the worst case competitive ratio is $C_{S(w)}(b, y(b)) = 3 + \frac{6}{b^2 - 3}$, shown in Equation 3.3.



(a) $C_{S(w)}(b, 0.5)$ (b) $C_{S(w)}(2, p)$

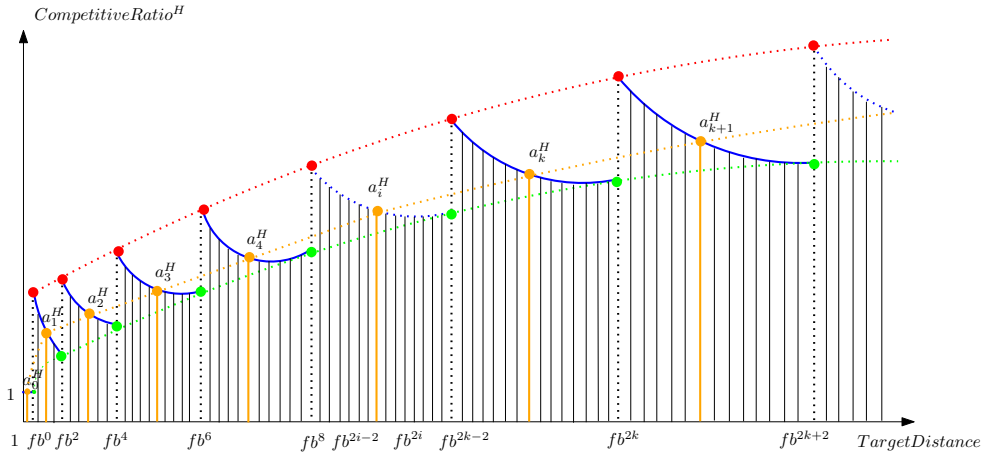
Figure 3.6: $C_{S(w)}$ given the specific probability p and the geometric ratio b

Also, refer to Figure 3.4 and Figure 3.6. When the probability $p = 0.5$, the optimal b equals to 2. The result matches with the lower bound of the worst case competitive ratio for the 2-ray cow-path problem, which is 9. Note here, when $p = 0.5$, our proposed search strategy is equivalent to the doubling strategy.

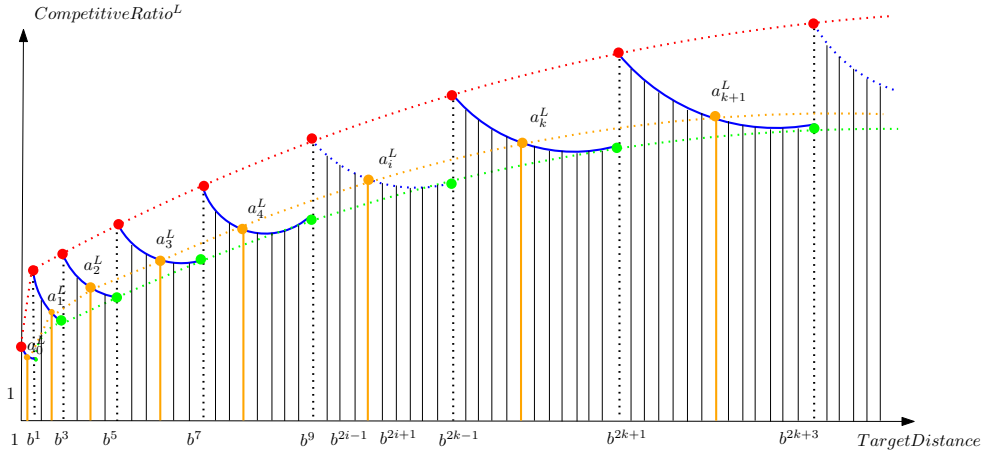
3.3 Average Case Competitive Ratio

3.3.1 Un-truncated Average Case Competitive Ratio

In this section, we will compute the average case competitive ratio for the 1-STRAW problem. Different than the worst case competitive ratio, the average case competitive ratio is determined by any possible target location since we want to get the average value of all the competitive ratios, see in Figure 3.7.



(a) General case of C_S^H with the average competitive ratio



(b) General case of C_S^L with the average competitive ratio

Figure 3.7: General case of C_S^H and C_S^L with the average competitive ratio

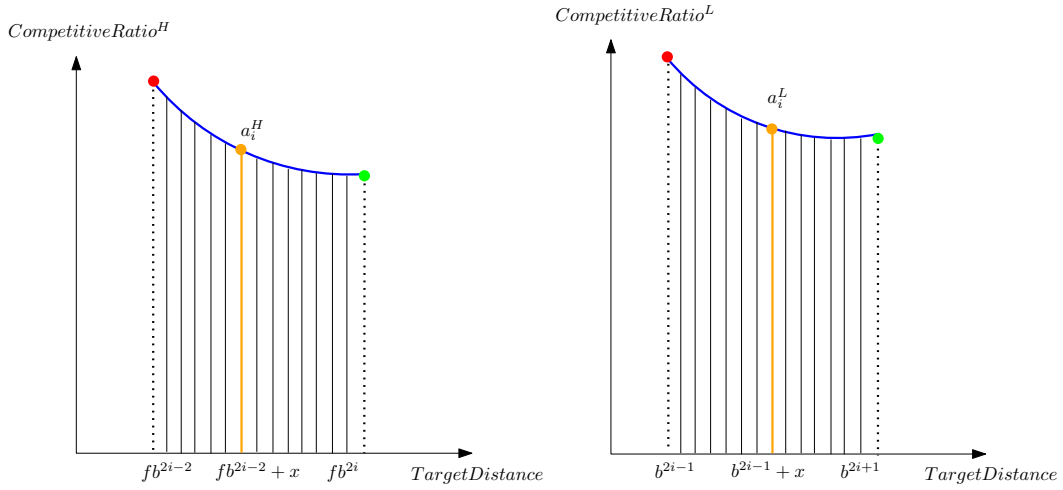
We assume that there is no limitation on the range of the target location, which means that it is possible that the target is located at an unbounded distance to the origin. We call this case *un-truncated*.

The average case competitive ratio under this setting is named the *un-truncated average case competitive ratio*, denoted as $C_{S(a)}$. We will introduce three models to compute the un-truncated average case competitive ratio and analyze the lower bound of $C_{S(a)}$. To compute the average case competitive ratio, we will use the formula as follows.

$$C_{S(a)} = high \times C^H_{S(a)} + low \times C^L_{S(a)} \tag{3.4}$$

where $C^H_{S(a)}$ is the average case competitive ratio when the target is on R_H and $C^L_{S(a)}$ is the average case competitive ratio when the target is on R_L.

First, we will introduce two lemmas which give the properties of $C_{S(a)}$. Refer to Figure 3.8a and Figure 3.8b. To simplify the descriptions, instead of using the notation of *round*, we use the notation of *term*. Note here the *term* does not equal to the *round*. We name the un-truncated average case competitive ratio for term i (which is the $(2i)$th searching round) as a^H_i on R_H. Similarly, we denote the un-truncated average case competitive ratio for term i (which is the $(2i)$st searching round) as a^L_i on R_L.



(a) C^H_S at term i (b) C^L_S at term i

Figure 3.8: The ith term of C^H_S and C^L_S

Lemma 3 *a^H_i is a monotonically increasing function of i, where $i = 1, 2, \ldots, \infty$.*

Proof. In the ith term, if the target is located at $f(p) \cdot b^{2i-2} + x$, where $x \in [0, f(p) \cdot (b^{2i} - b^{2i-2})]$, refer to Figure 3.8a, the un-truncated average case competitive ratio is:

$$a^H_i = \frac{1}{f(p) \cdot (b^{2i} - b^{2i-2})} \cdot \int_0^{f(p)\cdot(b^{2i}-b^{2i-2})} \left\{ 1 + \frac{2 \cdot \sum_{j=0}^{i-1} f(p) \cdot b^{2j} + 2 \cdot \sum_{j=0}^{i-1} b^{2j+1}}{f(p) \cdot b^{2i-2} + x} \right\} dx$$

$$= \frac{1}{f(p)\cdot(b^{2i}-b^{2i-2})}\cdot x\Big|_0^{f(p)\cdot(b^{2i}-b^{2i-2})} + \frac{2(f(p)+b)\cdot(b^{2i}-1)}{f(p)\cdot b^{2i-2}\cdot(b^2-1)^2}\cdot\ln(f(p)\cdot b^{2i-2}+x)\Big|_0^{f(p)\cdot b^{2i-2}(b^2-1)}$$
$$= 1 + \frac{2(f(p)+b)\cdot(b^{2i}-1)}{f(p)\cdot b^{2i-2}\cdot(b^2-1)^2}\cdot[2i\cdot\ln b - (2i-2)\ln b]$$

$$\therefore a_i^H = 1 + \frac{4\ln b(f(p)+b)}{f(p)\cdot(b^2-1)^2}\cdot\left[b^2 - \frac{1}{b^{2i-2}}\right] \tag{3.5}$$

When i increases, $\left[b^2 - \frac{1}{b^{2i-2}}\right]$ increases, therefore a_i^H increases. ∎

Lemma 4 *a_i^L is a monotonically increasing function of i, where $i = 1, 2, \ldots, \infty$.*

Proof. In the ith term, if the target distance is located at $b^{2i-1}+x$, where $x \in [0, b^{2i+1}-b^{2i-1}]$, refer to Figure 3.8b, the un-truncated average case competitive ratio is:

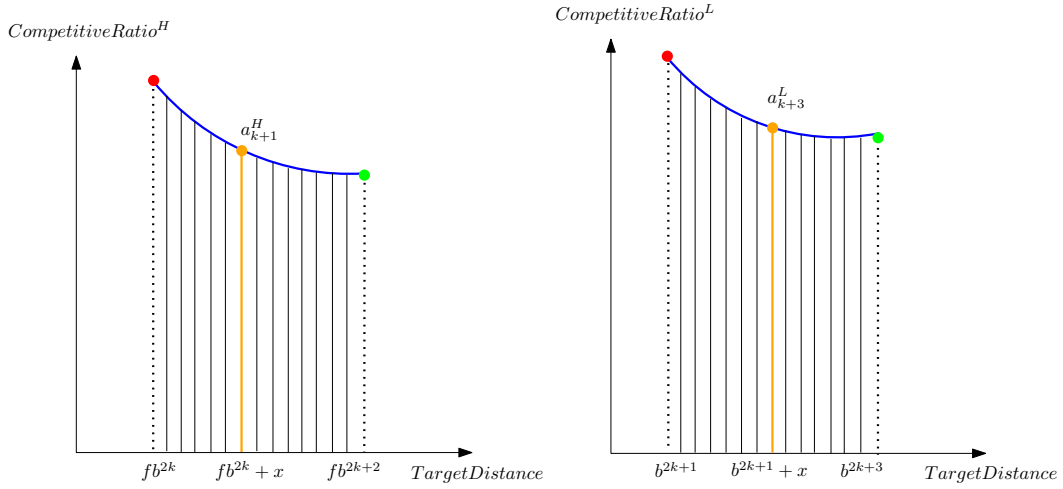
$$a_i^L = \frac{1}{b^{2i+1}-b^{2i-1}}\cdot\int_0^{b^{2i+1}-b^{2i-1}}\left\{1 + \frac{2\cdot\sum_{j=0}^{i} f(p)\cdot b^{2j} + 2\cdot\sum_{j=0}^{i-1} b^{2j+1}}{b^{2i-1}+x}\right\}dx$$
$$= \frac{1}{b^{2i+1}-b^{2i-1}}\cdot x\Big|_0^{b^{2i+1}-b^{2i-1}} + \frac{2f(p)\cdot(b^{2i+2}-1)+2b(b^{2i}-1)}{b^{2i-1}(b^2-1)^2}\cdot\ln(b^{2i-1}+x)\Big|_0^{b^{2i+1}-b^{2i-1}}$$
$$= 1 + \frac{4\ln b\cdot\left[f(p)(b^{2i+2}-1)+b(b^{2i}-1)\right]}{(b^2-1)^2\cdot b^{2i-1}}$$

$$\therefore a_i^L = 1 + \frac{4\ln b\cdot\left[f(p)(b^3-\frac{1}{b^{2i-1}})+b(b-\frac{1}{b^{2i-1}})\right]}{(b^2-1)^2} \tag{3.6}$$

When i increases, $\left[f(p)(b^3-\frac{1}{b^{2i-1}})+b(b-\frac{1}{b^{2i-1}})\right]$ increases, therefore a_i^L increases. ∎

Last term of the un-truncated Average Case Competitive Ratio

According to Lemma 3 and Lemma 4, we know that the last term of the un-truncated average case competitive ratio is the largest. We will compute the largest value given that the target distance can be unbounded. To be consistent with the proofs for the worst case competitive ratio, we will compute the $(i+1)$st term for both R_H and R_L, shown in Figure 3.9a and Figure 3.9b.



(a) Last term of C_S^H (b) Last term of C_S^L

Figure 3.9: Last term of C_S^H and C_S^L

Theorem 12 *The last term of the un-truncated average case competitive ratio $C_{S(a)_{last}}$ for the 1-STRAW problem given the general function $f(p)$ is:*

$$C_{S(a)_{last}} = \begin{cases} 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} + \frac{4\ln b \cdot b^3}{(b^2-1)^2}\left[pf(p) + \frac{1-p}{f(p)}\right] & p \in [0, 0.5] \\ 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} + \frac{4\ln b \cdot b^3}{(b^2-1)^2}\left[(1-p)f(p) + \frac{p}{f(p)}\right] & p \in [0.5, 1] \end{cases}$$

Proof. There are two cases in terms of the location of the target.

Case 1: Target is on R_H, target distance is $f(p) \times b^{2k} + x$, $x \in [0, f(p)b^{2k+2} - f(p)b^{2k}]$, refer to Figure 3.9a.

$$\begin{aligned} a_{k+1}^H &= \frac{1}{f(p)\cdot(b^{2k+2}-b^{2k})} \cdot \int_0^{f(p)\cdot(b^{2k+2}-b^{2k})} \left\{1 + \frac{2\cdot\sum_{i=0}^{k} f(p)\cdot b^{2i} + 2\cdot\sum_{i=0}^{k} b^{2i+1}}{f(p)\cdot b^{2k} + x}\right\} dx \\ &= 1 + \frac{4\ln b(f(p)+b)}{f(p)\cdot(b^2-1)^2} \cdot \left[b^2 - \frac{1}{b^{2k}}\right] \end{aligned}$$

Because

$$C_{S(a)_{last}}^H = \lim_{k\to\infty}\left\{a_{k+1}^H\right\}$$

Therefore we have

$$C_{S(a)_{last}}^H = 1 + \frac{4\ln b \cdot b^2 \cdot (f(p)+b)}{f(p)\cdot(b^2-1)^2} = 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} \cdot [\frac{b}{bf(p)} + 1]$$

Case 2: Target is on R_L, target distance is $b^{2k+1}+x$, $x \in [0, b^{2k+3}-b^{2k+1}]$, refer to Figure 3.9b.

$$a_{k+1}^L = \frac{1}{b^{2k+3}-b^{2k+1}} \cdot \int_0^{b^{2k+3}-b^{2k+1}} \left\{ 1 + \frac{2 \cdot \sum_{i=0}^{k+1} f(p) \cdot b^{2i} + 2 \cdot \sum_{i=0}^{k} b^{2i+1}}{b^{2k+1}+x} \right\} dx$$

$$= 1 + \frac{4 \ln b \cdot \left[f(p)(b^3 - \frac{1}{b^{2k+1}}) + b(b - \frac{1}{b^{2k+1}}) \right]}{(b^2-1)^2}$$

Because

$$C_{S(a)_{last}}^L = \lim_{k \to \infty} \left\{ a_{k+1}^L \right\} = 1 + \frac{4 \ln b \cdot b^2}{(b^2-1)^2} \cdot [f(p) \cdot b + 1]$$

$$\because C_{S(a)_{last}} = high \times C_{S(a)_{last}}^H + low \times C_{S(a)_{last}}^L$$

We have $C_{S(a)_{last}} = 1 + \frac{4 \ln b \cdot b^2}{(b^2-1)^2} + \frac{4 \ln b \cdot b^3}{(b^2-1)^2} \times [low \times f(p) + \frac{high}{f(p)}]$, which is

$$C_{S(a)_{last}} = \begin{cases} 1 + \frac{4 \ln b \cdot b^2}{(b^2-1)^2} + \frac{4 \ln b \cdot b^3}{(b^2-1)^2} \left[pf(p) + \frac{1-p}{f(p)} \right] & p \in [0, 0.5] \\ 1 + \frac{4 \ln b \cdot b^2}{(b^2-1)^2} + \frac{4 \ln b \cdot b^3}{(b^2-1)^2} \left[(1-p)f(p) + \frac{p}{f(p)} \right] & p \in [0.5, 1] \end{cases}$$

This completes the proof. ∎

Arithmetic Mean of the un-truncated Average Case Competitive Ratio

Now we need to emphasize the difference between the worst case competitive ratio and the average case competitive ratio in terms of computing the lower bound. In the previous section, we have two lemmas similar to Lemma 3 and Lemma 4. We used the last term worst case competitive ratio for the 1-STRAW problem to compute the lower bound of $C_{S(w)}$. The reason is because our goal is $\min \left\{ \max \left\{ C_{S(w)} \right\} \right\}$. However, for the average case competitive ratio, our goal is to minimize the average value of the average case competitive ratios of all the terms, which is denoted as $\text{AVE}\left\{ C_{S(a)} \right\}$. Therefore we desire

$$\min \left\{ \text{AVE} \left\{ C_{S(a)} \right\} \right\}$$

The first option of computing the average value of the competitive ratio is to calculate the arithmetic mean of a_i, which is denoted as

$$\text{AVE}_{arithmetic} = \frac{\sum_{0}^{k} a_i}{k+1}$$

Theorem 13 *The arithmetic mean of the un-truncated average case competitive ratio $C_{S(a)_{arithmetic}}$ for the 1-STRAW problem given the general function $f(p)$ is:*

$$C_{S(a)_{arithmetic}} = \begin{cases} 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} + \frac{4\ln b \cdot b^3}{(b^2-1)^2}\left[pf(p) + \frac{1-p}{f(p)}\right] & p \in [0, 0.5] \\ 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} + \frac{4\ln b \cdot b^3}{(b^2-1)^2}\left[(1-p)f(p) + \frac{p}{f(p)}\right] & p \in [0.5, 1] \end{cases}$$

Proof. When the target is on R_H, we name the arithmetic mean of the average case as $\text{AVE}^H_{arithmetic}$. Whereas when the target is on R_L, we use $\text{AVE}^L_{arithmetic}$ as the arithmetic mean of the average case. We have

$$\text{AVE}^H_{arithmetic} = \frac{\sum_{i=0}^{k} a_i^H}{k+1}$$

and

$$\text{AVE}^L_{arithmetic} = \frac{\sum_{i=0}^{k} a_i^L}{k+1}$$

Case 1: Target is on R_H, refer to Figure 3.7a.

First, we need to know the value of a_0^H, in which case the target distance is between $[1, f(p)\cdot b^0]$. The robot will find the target at the first round of searching and the competitive ratio will always be 1. Therefore, $a_0^H = 1$. Given $a_0^H = 1$ and Equation 3.5, we have

$$\begin{aligned} \text{AVE}^H_{arithmetic} &= \frac{\sum_{i=0}^{k} a_i^H}{k+1} = \frac{a_0^H + \sum_{i=1}^{k} a_i^H}{k+1} \\ &= \frac{1 + \sum_{i=1}^{k}\left\{1 + \frac{4\ln b(f(p)+b)}{f(p)\cdot(b^2-1)^2}\cdot\left[b^2 - \frac{1}{b^{2i-2}}\right]\right\}}{k+1} \\ &= 1 + \frac{4\ln b \cdot b^2(f(p)+b)}{f(p)\cdot(b^2-1)^2}\cdot\frac{1}{1+\frac{1}{k}} - \frac{b^2}{b^2-1}\cdot\frac{1-\frac{1}{b^{2k}}}{k+1} \end{aligned}$$

Additionally, we have

$$
\begin{aligned}
C^{H}_{S(a)_{arithmetic}} &= \lim_{k\to\infty}\left\{\mathrm{AVE}^{H}_{arithmetic}\right\} \\
&= \lim_{k\to\infty}\left\{1+\frac{4\ln b\cdot b^2(f(p)+b)}{f(p)\cdot(b^2-1)^2}\cdot\frac{1}{1+\frac{1}{k}}-\frac{b^2}{b^2-1}\cdot\frac{1-\frac{1}{b^{2k}}}{k+1}\right\} \\
&= 1+\frac{4\ln b\cdot b^2(f(p)+b)}{f(p)\cdot(b^2-1)^2}
\end{aligned}
$$

Case 2: Target is on R_L, refer to Figure 3.7b.

First, we need to know the value of a^L_0, in which case the target distance is between $[1, b^1]$. According to the definition of the competitive ratio, we have

$$
a^L_0=\frac{1}{b-1}\cdot\int_0^{b-1}\left\{1+\frac{2f(p)\cdot b^0}{1+x}\right\}dx=1+\frac{2f(p)}{b-1}\cdot\ln(x+1)\Big|_0^{b-1}=1+\frac{2\ln b\cdot f(p)}{b-1}
$$

Given $a^L_0=1+\dfrac{2\ln b\cdot f(p)}{b-1}$ and Equation 3.6, we have

Therefore

$$
\mathrm{AVE}^{L}_{arithmetic}=\frac{\sum_{i=0}^{k}a^L_i}{k+1}
$$

$$
\begin{aligned}
&= \frac{1+\frac{2\ln b\cdot f(p)}{b-1}+\sum_{i=1}^{k}\left\{1+\frac{4\ln b\cdot\left[f(p)(b^3-\frac{1}{b^{2i-1}})+b(b-\frac{1}{b^{2i-1}})\right]}{(b^2-1)^2}\right\}}{k+1} \\
&= 1+\frac{2\ln bf(p)}{b-1}\cdot\frac{1}{k+1}+\frac{4\ln b\cdot b^2(1+bf(p))}{(b^2-1)^2}\cdot\frac{1}{1+\frac{1}{k}}-\frac{4\ln b\cdot b(f(p)+b)}{(b^2-1)^2}\cdot\frac{b^2}{b^2-1}\cdot\frac{1-\frac{1}{b^{2k}}}{k+1}
\end{aligned}
$$

Additionally, we have

$$\begin{aligned}
C^L_{S(a)_{arithmetic}} &= \lim_{k\to\infty}\left\{\text{AVE}^L_{arithmetic}\right\} \\
&= \lim_{k\to\infty}\left\{1+\frac{2\ln bf(p)}{b-1}\cdot\frac{1}{k+1}+\frac{4\ln b\cdot b^2(1+bf(p))}{(b^2-1)^2}\cdot\frac{1}{1+\frac{1}{k}}\right\} \\
&\quad -\lim_{k\to\infty}\left\{\frac{4\ln b\cdot b(f(p)+b)}{(b^2-1)^2}\cdot\frac{b^2}{b^2-1}\cdot\frac{1-\frac{1}{b^{2k}}}{k+1}\right\} \\
&= 1+\frac{4\ln b\cdot b^2(1+bf(p))}{(b^2-1)^2}
\end{aligned}$$

Because we know

$$\begin{aligned}
C_{S(a)_{arithmetic}} &= high\times C^H_{S(a)_{arithmetic}} + low\times C^L_{S(a)_{arithmetic}} \\
&= high\times\left[1+\frac{4\ln b\cdot b^2(f(p)+b)}{f(p)\cdot(b^2-1)^2}\right]+low\times\left[1+\frac{4\ln b\cdot b^2(1+bf(p))}{(b^2-1)^2}\right] \\
&= 1+\frac{4\ln b\cdot b^2}{(b^2-1)^2}+\frac{4\ln b\cdot b^3}{(b^2-1)^2}\times[low\times f(p)+\frac{high}{f(p)}]
\end{aligned}$$

which is

$$C_{S(a)_{arithmetic}} = \begin{cases} 1+\frac{4\ln b\cdot b^2}{(b^2-1)^2}+\frac{4\ln b\cdot b^3}{(b^2-1)^2}\left[pf(p)+\frac{1-p}{f(p)}\right] & p\in[0,0.5] \\ 1+\frac{4\ln b\cdot b^2}{(b^2-1)^2}+\frac{4\ln b\cdot b^3}{(b^2-1)^2}\left[(1-p)f(p)+\frac{p}{f(p)}\right] & p\in[0.5,1] \end{cases}$$

This completes the proof. ∎

Weighted Mean of the un-truncated Average Case Competitive Ratio

Now we compute the weighted mean of the un-truncated average case competitive ratio. Here we define the weight as the length difference between the endings of two consecutive searching rounds on the same ray. Therefore we denote $len^H_i = f(p)\cdot b^{2i} - f(p)\cdot b^{2i-2}$ and $len^L_i = b^{2i+1}-b^{2i-1}$ when $i\geq 1$. And the weighted mean of the un-truncated average case competitive ratio is defined as

$$\text{AVE}^H_{weight} = \frac{\sum_{i=0}^{k}\left\{a^H_i\times len^H_i\right\}}{len^H_0+len^H_1+len^H_2+\cdots+len^H_k}$$

and

$$\text{AVE}^L_{weight} = \frac{\sum_{i=0}^{k}\left\{a^L_i\times len^L_i\right\}}{len^L_0+len^L_1+len^L_2+\cdots+len^L_k}$$

Theorem 14 *The weighted mean of the un-truncated average case competitive ratio $C_{S(a)_{weight}}$ for the 1-STRAW problem given the general function $f(p)$ is:*

$$C_{S(a)_{weight}} = \begin{cases} 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} + \frac{4\ln b \cdot b^3}{(b^2-1)^2}\left[pf(p) + \frac{1-p}{f(p)}\right] & p \in [0, 0.5] \\ 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} + \frac{4\ln b \cdot b^3}{(b^2-1)^2}\left[(1-p)f(p) + \frac{p}{f(p)}\right] & p \in [0.5, 1] \end{cases}$$

Proof. Same as the arithmetic mean of average case, we will consider two situations ($C^H_{S(a)_{weight}}$ and $C^L_{S(a)_{weight}}$) respectively and then combine them using the formula

$$C_{S(a)_{weight}} = high \times C^H_{S(a)_{weight}} + low \times C^L_{S(a)_{weight}}$$

Case 1: Computation of $C^H_{S(a)_{weight}}$

$\because len_i^H = f(p) \cdot b^{2i} - f(p) \cdot b^{2i-2}$, $a_i^H = 1 + \frac{4\ln b(f(p)+b)}{f(p)\cdot(b^2-1)^2} \cdot \left[b^2 - \frac{1}{b^{2i-2}}\right]$ when $i \geq 1$ and $a_0^H = 1$, therefore we have

$$\begin{aligned} \text{AVE}^H_{weight} &= \frac{\sum_{i=0}^{k}\left\{a_i^H \times len_i^H\right\}}{len_0^H + len_1^H + len_2^H + \cdots + len_k^H} \\ &= \frac{1 + f(p)\cdot(b^2-1)\sum_{i=1}^{k}\left\{b^{2i-2} \times \left[1 + \frac{4\ln b(f(p)+b)}{f(p)\cdot(b^2-1)^2} \cdot (b^2 - \frac{1}{b^{2i-2}})\right]\right\}}{f(p)\cdot b^{2k} - 1} \\ &= 1 + \frac{2 - f(p)}{f(p)\cdot b^{2k} - 1} + \frac{4\ln b \cdot b^2(f(p)+b)}{(b^2-1)^2} \cdot \frac{1 - \frac{1}{b^{2k}}}{f(p) - \frac{1}{b^{2k}}} \\ &\quad - \frac{4\ln b(f(p)+b)}{b^2-1} \cdot \frac{\frac{k}{b^{2k}}}{f(p) - \frac{1}{b^{2k}}} \end{aligned}$$

$$\because C^H_{S(a)_{weight}} = \lim_{k\to\infty}\left\{\text{AVE}^H_{weight}\right\}$$

$$\begin{aligned}\therefore C^H_{S(a)_{weight}} &= \lim_{k\to\infty}\left\{1+\frac{2-f(p)}{f(p)\cdot b^{2k}-1}+\frac{4\ln b\cdot b^2(f(p)+b)}{(b^2-1)^2}\cdot\frac{1-\frac{1}{b^{2k}}}{f(p)-\frac{1}{b^{2k}}}\right\}\\ &\quad-\lim_{k\to\infty}\left\{\frac{4\ln b(f(p)+b)}{b^2-1}\cdot\frac{\frac{k}{b^{2k}}}{f(p)-\frac{1}{b^{2k}}}\right\}\\ &= 1+\frac{4\ln b\cdot b^2(f(p)+b)}{f(p)\cdot(b^2-1)^2}\end{aligned}$$

Case 2: Computation of $C^L_{S(a)_{weight}}$

$\because len^L_i = b^{2i+1}-b^{2i-1}$, $a^L_i = 1+\dfrac{4\ln b\cdot\left[f(p)(b^3-\frac{1}{b^{2i-1}})+b(b-\frac{1}{b^{2i-1}})\right]}{(b^2-1)^2}$ when $i\geq 1$ and $a^L_0 = 1+\dfrac{2\ln b\cdot f(p)}{b-1}$, therefore we have

$$\begin{aligned}\text{AVE}^L_{weight} &= \frac{\sum_{i=0}^{k}\left\{a^L_i\times len^L_i\right\}}{len^L_0+len^L_1+len^L_2+\cdots+len^L_k}\\ &= \frac{a^L_0+\sum_{i=1}^{k}\left\{a^L_i\times(b^{2i+1}-b^{2i-1})\right\}}{b^{2k+1}-1}\\ &= 1+\frac{\frac{2\ln b\cdot f(p)}{b-1}+2-b}{b^{2k+1}-1}+\frac{4\ln b\cdot b^2\cdot(f(p)\cdot b+1)}{(b^2-1)^2}\cdot\frac{1-\frac{b}{b^{2k+1}}}{1-\frac{1}{b^{2k+1}}}\\ &\quad-\frac{4\ln b\cdot(f(p)+b)}{b^2-1}\cdot\frac{\frac{k}{b^{2k+1}}}{1-\frac{1}{b^{2k+1}}}\end{aligned}$$

$$\because C^L_{S(a)_{weight}} = \lim_{k\to\infty}\left\{\text{AVE}^L_{weight}\right\}$$

$$\therefore C^L_{S(a)_{weight}} = \lim_{k\to\infty}\left\{1+\frac{\frac{2\ln b\cdot f(p)}{b-1}+2-b}{b^{2k+1}-1}+\frac{4\ln b\cdot b^2\cdot(f(p)\cdot b+1)}{(b^2-1)^2}\cdot\frac{1-\frac{b}{b^{2k+1}}}{1-\frac{1}{b^{2k+1}}}\right\}$$

$$-\lim_{k\to\infty}\left\{\frac{4\ln b\cdot(f(p)+b)}{b^2-1}\cdot\frac{\frac{k}{b^{2k+1}}}{1-\frac{1}{b^{2k+1}}}\right\}$$

$$=1+\frac{4\ln b\cdot b^2(f(p)\cdot b+1)}{(b^2-1)^2}$$

Because we have

$$\begin{aligned}C_{S(a)_{weight}} &= high\times C^H_{S(a)_{weight}} + low\times C^L_{S(a)_{weight}}\\ &= high\times\left[1+\frac{4\ln b\cdot b^2(f(p)+b)}{f(p)\cdot(b^2-1)^2}\right]+low\times\left[1+\frac{4\ln b\cdot b^2(1+bf(p))}{(b^2-1)^2}\right]\\ &= 1+\frac{4\ln b\cdot b^2}{(b^2-1)^2}+\frac{4\ln b\cdot b^3}{(b^2-1)^2}\times\left[low\times f(p)+\frac{high}{f(p)}\right]\end{aligned}$$

which is

$$C_{S(a)} = \begin{cases} 1+\frac{4\ln b\cdot b^2}{(b^2-1)^2}+\frac{4\ln b\cdot b^3}{(b^2-1)^2}\left[pf(p)+\frac{1-p}{f(p)}\right] & p\in[0,0.5]\\ 1+\frac{4\ln b\cdot b^2}{(b^2-1)^2}+\frac{4\ln b\cdot b^3}{(b^2-1)^2}\left[(1-p)f(p)+\frac{p}{f(p)}\right] & p\in[0.5,1]\end{cases}$$

This completes the proof. ∎

We computed three measures of the un-truncated average case competitive ratio, namely $C_{S(a)_{last}}$, $C_{S(a)_{arithmetic}}$ and $C_{S(a)_{weight}}$. In addition, the values of these three representations of average case competitive ratio are the same. Therefore, we have the observation.

Observation 1 *Note that $C_{S(a)_{last}} = C_{S(a)_{arithmetic}} = C_{S(a)_{weight}}$ and the last few terms of the un-truncated average case competitive ratio dominate the whole picture of the un-truncated average case competitive ratio, see Figure 3.7a and Figure3.7b.*

Since we observed that the last few terms of the un-truncated average case competitive ratio dominate over the previous terms, we conjecture that while the robot searches the target to an unbounded distance using our proposed search strategy, the un-truncated average case competitive ratio will converge to a limit, which is the last term of the un-truncated average case competitive ratio.

Conjecture 2 *By following our proposed search strategy, the un-truncated average case competitive ratio for the 1-STRAW problem converges to a limit when the robot searches the target to an unbounded distance.*

Lower Bounds of the un-truncated Average Case Competitive Ratio

Since the three un-truncated average case competitive ratios have the same values, we will denote the ratio as $C_{S(a)}$ and analyze the lower bound of $C_{S(a)}(b,p)$, which is denoted as a function of the geometric ratio b and the probability p. Because we have

$$C_{S(a)}(b,p) = 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} + \frac{4\ln b \cdot b^3}{(b^2-1)^2} \times (low \cdot f(p) + \frac{high}{f(p)})$$

Therefore, according to Theorem 10, the optimal representation of $C_{S(a)}(b,p)$ is

$$C_{S(a)}(b,p) = 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} + \frac{8\ln b \cdot b^3}{(b^2-1)^2} \cdot \sqrt{p(1-p)}$$

which is shown in Figure 3.10.

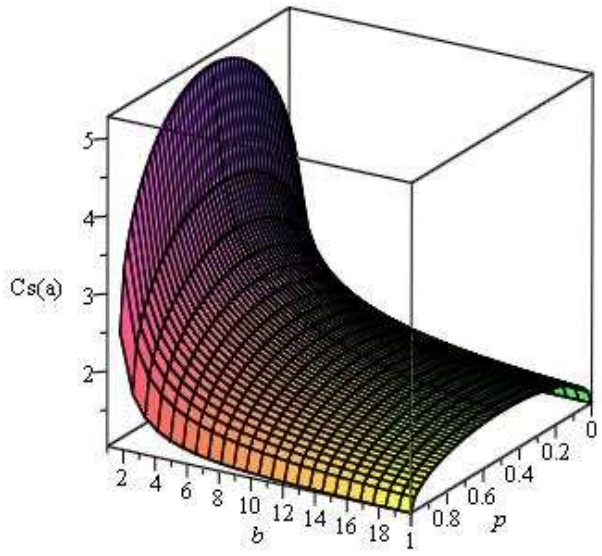


Figure 3.10: $C_{S(a)}(b,p)$

Here we consider the two partial derivatives of $C_{S(a)}(b,p)$, which are $\frac{dC_{S(a)}}{dp}$ and $\frac{dC_{S(a)}}{db}$, to get the insight into the properties of $C_{S(a)}$.

When we regard b as a constant and p as a parameter $\left(\frac{dC_{S(a)}(b,p)}{dp}\right)$, the representation of the un-truncated average case competitive ratio can be written as

$$C_{S(a)}(p) = 1 + c_1 + c_2 \cdot 2\sqrt{p(1-p)}$$

where $c_1 = \frac{4\ln b \cdot b^2}{(b^2-1)^2}$ and $c_2 = \frac{4\ln b \cdot b^3}{(b^2-1)^2}$.

Refer to Figure 3.5. When $p = 0.5$, $C_{S(a)}(0.5)$ is maximized, with value

$$C_{S(a)}(b, 0.5) = 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} + \frac{4\ln b \cdot b^3}{(b^2-1)^2}$$

which is shown in Figure 3.11.

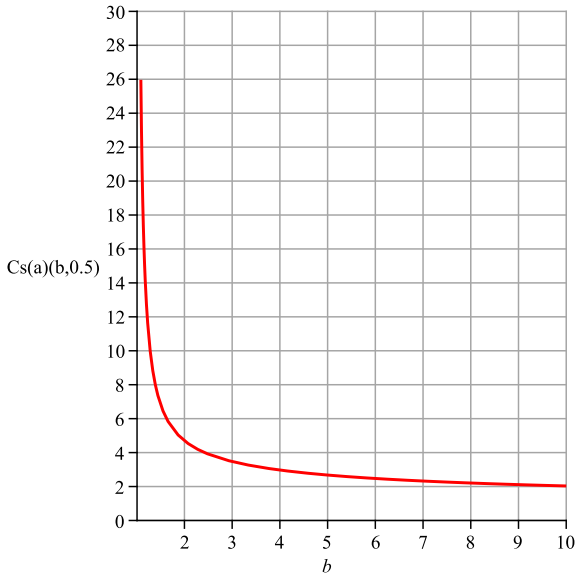


Figure 3.11: $C_{S(a)}(b, 0.5)$

In addition,when $p = 0$ or $p = 1$, $C_{S(a)}(0)$ or $C_{S(a)}(1)$ is the min, which is

$$C_{S(a)}(b, 0.5) = 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2}$$

Since we are seeking a lower bound of any possible average case competitive ratio, we desire to find the minimal value of the maximum of the un-truncated average case competitive ratio,

which is $C_{S(a)}(b, 0.5)$. Now let us consider the derivative of $C_{S(a)}(b, 0.5)$, which is

$$\begin{aligned}\frac{dC_{S(a)}(b,0.5)}{db} &= \frac{4b}{(b^2-1)^2} + \frac{8\ln b \cdot b}{(b^2-1)^2} - \frac{16\ln b \cdot b^3}{(b^2-1)^3} \\ &+ \frac{4b^2}{(b^2-1)^2} + \frac{12\ln b \cdot b^2}{(b^2-1)^2} - \frac{16\ln b \cdot b^4}{(b^2-1)^3} \\ &= \frac{4b}{(b^2-1)^3} \cdot \left[(b^3+b^2-b-1)(1-\ln b) - \ln b(b^2+4b+3)\right]\end{aligned}$$

As we can see from $\frac{dC_{S(a)}(b,0.5)}{db}$, when b grows bigger than e, then we have $\ln b > 1$ and $\frac{dC_{S(a)}(b,0.5)}{db} < 0$, which means $C_{S(a)}(b, 0.5)$ decreases as b grows. Therefore the optimal b for the un-truncated average case competitive ratio is unbounded.

Now let us regard p as a constant and b as a parameter. We have

$$C_{S(a)}(b) = 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} + c \cdot \frac{4\ln b \cdot b^3}{(b^2-1)^2}$$

where $c = 2\sqrt{p(1-p)}$. Obviously, $C_{S(a)}(b)$ decreases while b increases. Therefore, the optimal b in this case for the un-truncated average case competitive ratio is also unbounded.

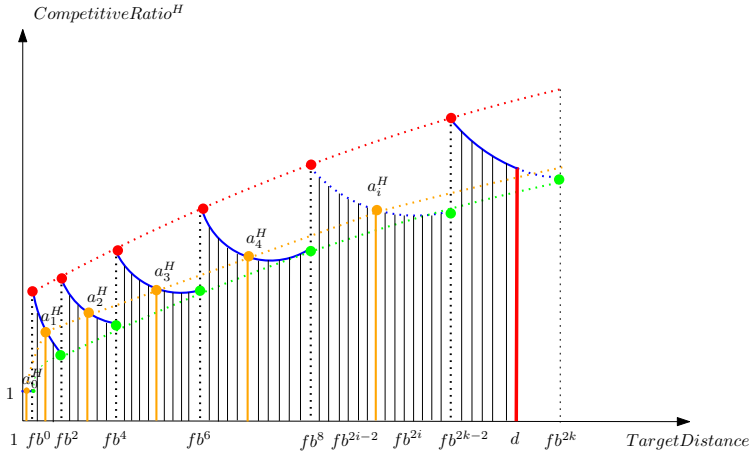
Now if we consider the problem at a high level, the un-truncated average case competitive ratio is achieved when the robot can always finish its searching in every round given that the target was not found at that round. Moreover, the target's location range is $[-\infty, +\infty]$. It makes sense that the optimal b is unbounded in this case since the robot can always reach the target given a sufficiently large b. However if the target is located in a restricted range, then the optimal b is bounded. We will consider the truncated case in the next section.

3.3.2 Truncated Average Case Competitive Ratio

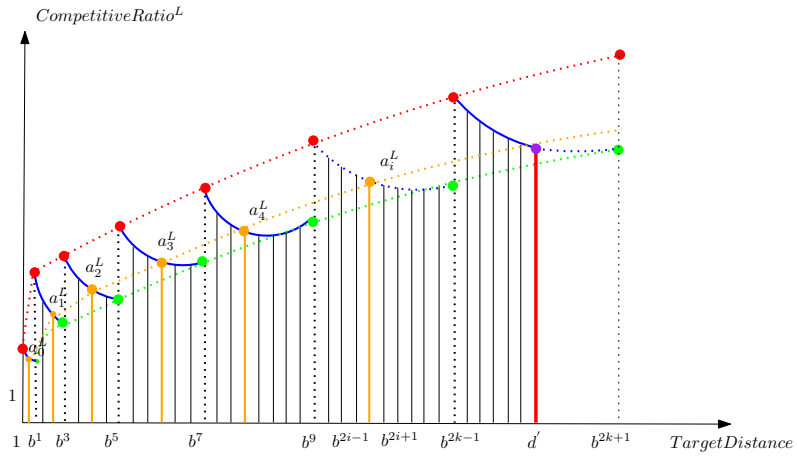
In this section, we will compute the truncated average case competitive ratio for the 1-STRAW problem. By *truncated*, we mean that the target is located in a bounded interval $[-d', +d]$, see in Figure 3.12a and Figure 3.12b. The signs of $+$ and $-$ represent robot searchings on R_H and R_L respectively.

Additionally, we have $f(p) \cdot b^{2k-2} \leq d \leq f(p) \cdot b^{2k}$ and $b^{2k-1} \leq d' \leq b^{2k+1}$. Here we introduce a parameter t to represent the *ratio* between the target boundary and the search distance closest to the target boundary given the search strategy. Therefore we have $d = t \cdot f(p) \cdot b^{2k-2}$, $d' = t \cdot b^{2k-1}$ and $t \in [1, b^2]$. Next, we will compute the truncated average case competitive ratio. Our setting here is that the target truncates the searching distance with the same ratio t on both R_H and R_L.

Now we will compute the truncated average case competitive ratio for the 1-STRAW problem, denoted by $C^*_{S(a)}$, given the general ratio function $f(p)$.



(a) The Truncated Competitive Ratio $C_S^{H^*}$



(b) The Truncated Competitive Ratio $C_S^{L^*}$

Figure 3.12: The Truncated Competitive Ratio of $C_S^{H^*}$ and $C_S^{L^*}$

Theorem 15 *The truncated average case competitive ratio $C^*_{S(a)}$ for the 1-STRAW problem given the general function $f(p)$ is:*

$$C^*_{S(a)} = \begin{cases} 1 + \dfrac{4\ln b \cdot b^2}{(b^2-1)^2} \cdot \dfrac{1}{t} + \dfrac{2b^2}{b^2-1} \cdot \dfrac{\ln t}{t} \\ \quad + \left[\dfrac{4\ln b \cdot b^3}{(b^2-1)^2} \cdot \dfrac{1}{t} + \dfrac{2b^3}{b^2-1} \cdot \dfrac{\ln t}{t}\right] \cdot \left[pf(p) + \dfrac{1-p}{f(p)}\right] & p \in [0, 0.5] \\ 1 + \dfrac{4\ln b \cdot b^2}{(b^2-1)^2} \cdot \dfrac{1}{t} + \dfrac{2b^2}{b^2-1} \cdot \dfrac{\ln t}{t} \\ \quad + \left[\dfrac{4\ln b \cdot b^3}{(b^2-1)^2} \cdot \dfrac{1}{t} + \dfrac{2b^3}{b^2-1} \cdot \dfrac{\ln t}{t}\right] \cdot \left[(1-p)f(p) + \dfrac{p}{f(p)}\right] & p \in [0.5, 1] \end{cases}$$

Proof. We denote the truncated average case competitive ratio $\text{AVE}^{H}_{truncated}$ when the target is located within $[f(p)\cdot b^{2k-2}, d]$ on R_H. Similarly, $\text{AVE}^{L}_{truncated}$ denotes the truncated average case competitive ratio when the target is located at $[b^{2k-1}, d']$ on R_L. Additionally, we denote

$$C^{H^*}_{S(a)} = \lim_{k\to\infty}\left\{\text{AVE}^{H}_{truncated}\right\}$$

$$C^{L^*}_{S(a)} = \lim_{k\to\infty}\left\{\text{AVE}^{L}_{truncated}\right\}$$

and we have the formula

$$C^{*}_{S(a)} = high \times C^{H^*}_{S(a)} + low \times C^{L^*}_{S(a)}$$

Next we will compute $C^{H^*}_{S(a)}$ and $C^{L^*}_{S(a)}$ respectively.

Case 1: Computation of $C^{H^*}_{S(a)}$

As we can see from Figure 3.12a, there are three ranges to include in the computation of $\text{AVE}^{H}_{truncated}$. The first range is from $[1, f(p)\cdot b^0]$ and in this range, $a^H_0 = 1$. The second range is from $[f(p)\cdot b^0, f(p)\cdot b^{2k-2}]$ and in the second range, the robot returns to the origin after completes the search distance in the current round given the fact that the target is not found. The third range is from $[f(p)\cdot b^{2k-2}, d]$ and robot finds the target in the third range. Therefore we have

$$\begin{aligned}
\text{AVE}^{H}_{truncated} = \frac{1}{d-1}\cdot\Bigg\{&(f(p)-1)\\
&+\sum_{i=1}^{k-1}\int_0^{f(p)\cdot b^{2i}-f(p)\cdot b^{2i-2}}\left\{1+\frac{2\sum_{j=0}^{i-1}f(p)\cdot b^{2j}+2\sum_{j=0}^{i-1}b^{2j+1}}{f(p)\cdot b^{2i-2}+x}\right\}dx\\
&+\int_0^{d-f(p)\cdot b^{2k-2}}\left\{1+\frac{2\sum_{i=0}^{k-1}f(p)\cdot b^{2i}+2\sum_{i=0}^{k-1}b^{2i+1}}{f(p)\cdot b^{2k-2}+x}\right\}dx\Bigg\}
\end{aligned}$$

$$\because \int_0^{f(p)\cdot b^{2i}-f(p)\cdot b^{2i-2}}\left\{1+\frac{2\sum_{j=0}^{i-1}f(p)\cdot b^{2j}+2\sum_{j=0}^{i-1}b^{2j+1}}{f(p)\cdot b^{2i-2}+x}\right\}dx$$

$$= \int_0^{f(p)\cdot b^{2i} - f(p)\cdot b^{2i-2}} \left\{ 1 + \frac{2f(p)\cdot \dfrac{b^{2i}-1}{b^2-1} + 2b\cdot \dfrac{b^{2i}-1}{b^2-1}}{f(p)\cdot b^{2i-2} + x} \right\} dx$$

$$= [f(p)\cdot b^{2i} - f(p)\cdot b^{2i-2}] + \frac{4\ln b(f(p)+b)\cdot(b^{2i}-1)}{b^2-1}$$

$$\therefore \mathrm{AVE}^H_{truncated} = \frac{1}{d-1}\cdot\Bigg\{(f(p)-1)$$
$$+ \sum_{i=1}^{k-1}\left\{[f(p)\cdot b^{2i} - f(p)\cdot b^{2i-2}] + \frac{4\ln b(f(p)+b)\cdot(b^{2i}-1)}{b^2-1}\right\}$$
$$+ \int_0^{d-f(p)\cdot b^{2k-2}} \left\{ 1 + \frac{2\sum_{i=0}^{k-1} f(p)\cdot b^{2i} + 2\sum_{i=0}^{k-1} b^{2i+1}}{f(p)\cdot b^{2k-2} + x} \right\} dx\Bigg\}$$
$$= 1 + \frac{1}{d-1}\cdot\frac{4\ln b\cdot(f(p)+b)}{b^2-1}\cdot\left[\frac{b^2(b^{2k-2}-1)}{b^2-1} - k + 1\right]$$
$$+ \frac{1}{d-1}\cdot\frac{2(f(p)+b)\cdot(b^{2k}-1)}{b^2-1}\cdot\Big[\ln d - \ln(f(p)\cdot b^{2k-2})\Big]$$

$$\because d = t\cdot f(p)\cdot b^{2k-2}$$

$$\therefore \mathrm{AVE}^H_{truncated} = 1 + \frac{4\ln b\cdot(f(p)+b)\cdot b^2}{(b^2-1)^2}\cdot\frac{1-\dfrac{1}{b^{2k-2}}}{t\cdot f(p) - \dfrac{1}{b^{2k-2}}}$$
$$- \frac{4\ln b\cdot(f(p)+b)}{b^2-1}\cdot\frac{k-1}{t\cdot f(p)\cdot b^{2k-2} - 1}$$
$$+ \frac{2(f(p)+b)\cdot\ln t}{b^2-1}\cdot\frac{b^2 - \dfrac{1}{b^{2k-2}}}{t\cdot f(p) - \dfrac{1}{b^{2k-2}}}$$

$$\because C^{H^*}_{S(a)} = \lim_{k\to\infty}\left\{\mathrm{AVE}^H_{truncated}\right\}$$

$$\therefore C^{H^*}_{S(a)} = 1 + \frac{4\ln b\cdot b^2\cdot\left(1+\dfrac{b}{f(p)}\right)}{(b^2-1)^2}\cdot\frac{1}{t} + \frac{2b^2\left(1+\dfrac{b}{f(p)}\right)}{b^2-1}\cdot\frac{\ln t}{t}$$

Case 2: Computation of $C_{S(a)}^{L^*}$

As we can see from Figure 3.12b, there are three ranges to include in the computation of $C_{S(a)}^{L^*}$. The first range is from $[1, b]$. The second range is from $[b, b^{2k-1}]$ and in this range, the robot returns to the origin after completes the search distance in the current round given the fact that the target is not found. The third range is from $[b^{2k-1}, d^{'}]$ and robot finds the target in the third range. So we have

$$\begin{aligned}
\text{AVE}_{truncated}^{L} = \frac{1}{d^{'}-1} \cdot \Bigg\{ & \int_0^{b-1} \left\{ 1 + \frac{2f(p)+1}{1+x} \right\} dx \\
& + \sum_{i=1}^{k-1} \int_0^{b^{2i+1}-b^{2i-1}} \left\{ 1 + \frac{2\sum_{j=0}^{i} f(p) \cdot b^{2j} + 2\sum_{j=0}^{i-1} b^{2j+1}}{b^{2i-1}+x} \right\} dx \\
& + \int_0^{d^{'}-b^{2k-1}} \left\{ 1 + \frac{2\sum_{i=0}^{k} f(p) \cdot b^{2i} + 2\sum_{i=0}^{k-1} b^{2i+1}}{b^{2k-1}+x} \right\} dx \Bigg\} \\
\because \int_0^{b^{2i+1}-b^{2i-1}} & \left\{ 1 + \frac{2\sum_{j=0}^{i} f(p) \cdot b^{2j} + 2\sum_{j=0}^{i-1} b^{2j+1}}{b^{2i-1}+x} \right\} dx \\
& = (b^{2i+1} - b^{2i-1}) + \left[2f(p) \cdot \frac{b^{2i+2}-1}{b^2-1} + 2b \cdot \frac{b^{2i}-1}{b^2-1} \right] \cdot 2\ln b
\end{aligned}$$

$$\begin{aligned}
\therefore \text{AVE}_{truncated}^{L} = \frac{1}{d^{'}-1} \cdot \Bigg\{ & (b-1) + (2f(p)+1) \cdot \ln(1+x) \Big|_0^{b-1} \\
& + \sum_{i=1}^{k-1} \left\{ (b^{2i+1} - b^{2i-1}) + \left[2f(p) \cdot \frac{b^{2i+2}-1}{b^2-1} + 2b \cdot \frac{b^{2i}-1}{b^2-1} \right] \cdot 2\ln b \right\} \\
& + \int_0^{d^{'}-b^{2k-1}} \left\{ 1 + \frac{2f(p) \cdot \frac{b^{2k+2}-1}{b^2-1} + 2b \cdot \frac{b^{2k}-1}{b^2-1}}{b^{2k-1}+x} \right\} dx \Bigg\}
\end{aligned}$$

$$= 1 + \frac{\ln b \cdot (2f(p)+1)}{d^{'}-1} + \frac{4\ln b \cdot b^4 \cdot f(p)}{(b^2-1)^2} \cdot \frac{b^{2k-2}-1}{d^{'}-1} + \frac{4\ln b \cdot b^3}{(b^2-1)^2} \cdot \frac{b^{2k-2}-1}{d^{'}-1}$$
$$- \frac{4\ln b \cdot f(p)}{b^2-1} \cdot \frac{k-1}{d^{'}-1} - \frac{4\ln b \cdot b}{b^2-1} \cdot \frac{k-1}{d^{'}-1}$$
$$+ \left[\frac{2f(p)}{b^2-1} \cdot \frac{b^{2k+2}-1}{d^{'}-1} + \frac{2b}{b^2-1} \cdot \frac{b^{2k}-1}{d^{'}-1}\right] \cdot (\ln(d^{'}) - \ln(b^{2k-1}))$$

$$\because d^{'} = t \cdot b^{2k-1}$$

$$\therefore \mathrm{AVE}^{L}_{truncated} = 1 + \frac{\ln b \cdot (2f(p)+1)}{t \cdot b^{2k-1}-1} + \frac{4\ln b \cdot b^4 \cdot f(p)}{(b^2-1)^2} \cdot \frac{1-\frac{1}{b^{2k-2}}}{t \cdot b - \frac{1}{b^{2k-2}}}$$
$$+ \frac{4\ln b \cdot b^3}{(b^2-1)^2} \cdot \frac{1-\frac{1}{b^{2k-2}}}{t \cdot b - \frac{1}{b^{2k-2}}} - \frac{4\ln b \cdot f(p)}{b^2-1} \cdot \frac{k-1}{t \cdot b^{2k-1}-1} - \frac{4\ln b \cdot b}{b^2-1} \cdot \frac{k-1}{t \cdot b^{2k-1}-1}$$
$$+ \left[\frac{2f(p)}{b^2-1} \cdot \frac{b^3 - \frac{1}{b^{2k-1}}}{t - \frac{1}{b^{2k-1}}} + \frac{2b}{b^2-1} \cdot \frac{b - \frac{1}{b^{2k-1}}}{t - \frac{1}{b^{2k-1}}}\right] \cdot \ln t$$

$$\because C^{L^*}_{S(a)} = \lim_{k\to\infty} \left\{ \mathrm{AVE}^{L}_{truncated} \right\}$$
$$\therefore C^{L^*}_{S(a)} = 1 + \frac{4\ln b \cdot b^2 \cdot (b \cdot f(p)+1)}{(b^2-1)^2} \cdot \frac{1}{t} + \frac{2b^2 \cdot (b \cdot f(p)+1)}{b^2-1} \cdot \frac{\ln t}{t}$$

Now we can compute $C^{*}_{S(a)}$ which is $high \times C^{H^*}_{s(a)} + low \times C^{L^*}_{S(a)}$.

$$C^{*}_{S(a)} = high \times C^{H^*}_{s(a)} + low \times C^{L^*}_{S(a)}$$
$$= 1 + \frac{4\ln b \cdot b^2}{(b^2-1)^2} \cdot \frac{1}{t} + \frac{2b^2}{b^2-1} \cdot \frac{\ln t}{t}$$
$$+ \left[\frac{4\ln b \cdot b^3}{(b^2-1)^2} \cdot \frac{1}{t} + \frac{2b^3}{b^2-1} \cdot \frac{\ln t}{t}\right] \cdot \left(\frac{high}{f(p)} + low \cdot f(p)\right)$$

This completes the proof. ∎

According to Corollary 1, the optimal function $f(p)$ is $\frac{\sqrt{high}}{\sqrt{low}}$. Therefore we have the representation of $C^{*}_{S(a)}(b,p,t)$, which is denoted as the function of the geometric ratio b and the probability p and the ratio t, defined as follows.

$$C^{*}_{S(a)}(b,p,t) = 1 + \frac{4\ln b \cdot b^2 \cdot [1+2b \cdot \sqrt{p(1-p)}]}{(b^2-1)^2} \cdot \frac{1}{t} + \frac{2 \cdot b^2[1+2b \cdot \sqrt{p(1-p)}]}{b^2-1} \cdot \frac{\ln t}{t} \qquad (3.7)$$

Theorem 16 *The maximum of* $C^*_{S(a)}(b,p,t)$ *is*

$$C^*_{S(a)}(b,p) = \max\left\{C^*_{S(a)(b,p,t)}\right\} = 1 + \frac{2\cdot b^2[1+2b\cdot\sqrt{p(1-p)}]}{b^2-1}\cdot e^{\frac{2\ln b}{b^2-1}-1}$$

when $t = e^{1-\frac{2\ln b}{b^2-1}}$.

Proof.

Here we regard b and p as constants and we set $c_1^* = \frac{4\ln b\cdot b^2\cdot[1+2b\cdot\sqrt{p(1-p)}]}{(b^2-1)^2}$ and $c_2^* = \frac{2\cdot b^2[1+2b\cdot\sqrt{p(1-p)}]}{b^2-1}$. Therefore c_1^* and c_2^* are taken as constants as well. Given Equation 3.7, we have

$$C^*_{S(a)}(t) = 1 + c_1^*\cdot\frac{1}{t} + c_2^*\cdot\frac{\ln t}{t}$$

Now we will compute $C^*_{S(a)}{}'(t)$

$$\frac{dC^*_{S(a)(t)}}{dt} = -c_1^*\cdot\frac{1}{t^2} - c_2^*\cdot\frac{\ln t}{t^2} + c_2^*\cdot\frac{1}{t^2} = 0$$

We have the result that when $t = e^{1-\frac{c_1^*}{c_2^*}}$, $\frac{dC^*_{S(a)(t)}}{dt} = 0$.

Now we will divide all possible values of t to three categories, which is

$$\begin{cases} t_1 : [1, e^{1-\frac{c_1^*}{c_2^*}}) \\ t^* = e^{1-\frac{c_1^*}{c_2^*}} \\ t_2 : (e^{1-\frac{c_1^*}{c_2^*}}, b^2] \end{cases}$$

and we will prove that both $C^*_{S(a)}(t_1)$ and $C^*_{S(a)}(t_2)$ are smaller than $C_{S(a)}(t^*)$.

Case 1: $t_1 \in [1, e^{1-\frac{c_1^*}{c_2^*}})$

$$\because t_1 < t^* = e^{1-\frac{c_1^*}{c_2^*}}$$

therefore we set $t_1 = t^*\cdot e^{-\varepsilon} = e^{1-\frac{c_1^*}{c_2^*}-\varepsilon}, \varepsilon > 0$.

$$\begin{aligned} C_{S(a)}(t^*) - C_{S(a)}(t1) &= c_1^*\cdot\frac{1}{t^*} + c_2^*\cdot\frac{\ln t^*}{t^*} - c_1^*\cdot\frac{1}{t_1} - c_2^*\cdot\frac{\ln t_1}{t_1} \\ &= c_1^*\cdot\frac{1}{t^*} + c_2^*\cdot\frac{1-\frac{c_1^*}{c_2^*}}{t^*} - c_1^*\cdot\frac{1}{t_1} - c_2^*\cdot\frac{1-\frac{c_1^*}{c_2^*}-\varepsilon}{t_1} \\ &= c_2^*\cdot\frac{1}{t^*}\cdot(1-\frac{1-\varepsilon}{e^{-\varepsilon}}) \end{aligned}$$

$$\because 1-\varepsilon < e^{-\varepsilon}$$
$$\therefore C_{S(a)}(t^*) - C_{S(a)}(t1) > 0$$

Case 2: $t_1 \in (e^{1-\frac{c_1^*}{c_2^*}}, b^2]$

$$\because t_2 > t^* = e^{1-\frac{c_1^*}{c_2^*}}$$

therefore we set $t_1 = t^* \cdot e^{\varepsilon} = e^{1-\frac{c_1^*}{c_2^*}+\varepsilon}, \varepsilon > 0$.

$$\begin{aligned} C_{S(a)}(t^*) - C_{S(a)}(t2) &= c_1^* \cdot \frac{1}{t^*} + c_2^* \cdot \frac{\ln t^*}{t^*} - c_1^* \cdot \frac{1}{t_2} - c_2^* \cdot \frac{\ln t_2}{t_2} \\ &= c_1^* \cdot \frac{1}{t^*} + c_2^* \cdot \frac{1-\frac{c_1^*}{c_2^*}}{t^*} - c_1^* \cdot \frac{1}{t_2} - c_2^* \cdot \frac{1-\frac{c_1^*}{c_2^*}+\varepsilon}{t_2} \\ &= c_2^* \cdot \frac{1}{t^*} \cdot (1 - \frac{1+\varepsilon}{e^{\varepsilon}}) \end{aligned}$$

$$\because 1+\varepsilon < e^{\varepsilon}$$
$$\therefore C_{S(a)}(t^*) - C_{S(a)}(t2) > 0$$

Therefore, we proved that

$$C_{S(a)}(t^*) = 1 + \frac{2 \cdot b^2[1 + 2b \cdot \sqrt{p(1-p)}]}{b^2-1} \cdot e^{\frac{2\ln b}{b^2-1}-1} \tag{3.8}$$

is maximum when $t^* = e^{1-\frac{2\ln b}{b^2-1}}$ out of any possible $t \in [1, b^2]$. ∎

The graph of $C^*_{S(a)}(b,p)$ is shown in Figure 3.13.

Lower Bound of Truncated Average Case Competitive Ratio

In what follows, we compute the lower bound of the truncated average case competitive ratio $C^*_{S(a)}(b,p)$. Refer to Figure 3.13, we know visually that given b as a constant, function $C^*_{S(a)}(p)$ has the same shape as $g(p) = 2\sqrt{p \cdot (1-p)}$. In addition, given p as a constant, function $C^*_{S(a)}(b)$ decreases in the beginning and once passes a specific value of b, $C^*_{S(a)}(b)$ increases. Therefore, we compute $\frac{dC^*_{S(a)}(b,p)}{dp}$ and $\frac{dC^*_{S(a)}(b,p)}{db}$ respectively to get the observations above proved theoretically.

Firstly, we regard b as a constant and denote

$$a_1^* = \frac{2b^2}{b^2-1} \cdot e^{\frac{2\ln b}{b^2-1}-1}$$

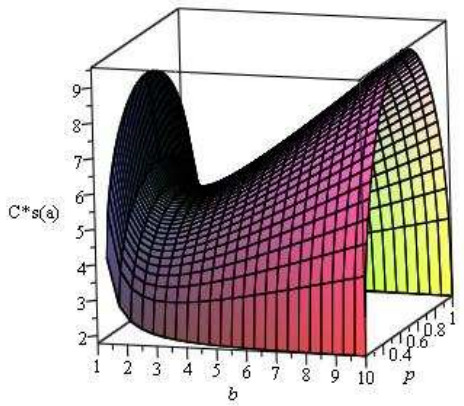


Figure 3.13: Truncated Average Case Competitive Ratio $C^*_{S(a)}(b,p)$

$$a_2^* = \frac{2b^3}{b^2-1} \cdot e^{\frac{2\ln b}{b^2-1}-1}$$

Therefore, according to Equation 3.8, we have

$$C^*_{S(a)}(p) = 1 + a_1^* + a_2^* \cdot 2\sqrt{p\cdot(1-p)}$$

In addition, we have

$$\frac{dC^*_{S(a)}(b,p)}{dp} = a_2^* \cdot \frac{1-2p}{\sqrt{p\cdot(1-p)}} \tag{3.9}$$

Since the function $g(p) = 2\sqrt{p\cdot(1-p)}$ as shown in Figure 3.5, therefore function $C^*_{S(a)}(p)$ has the same shape as $g(p)$. Given the shape of $C^*_{S(a)}(p)$ and Equation 3.9, we have that when $p = 0.5$, the value of $C^*_{S(a)}(0.5)$ is maximum, which equals to

$$C^*_{S(a)}(0.5) = 1 + \frac{2b^2}{b-1} \cdot e^{\frac{2\ln b}{b^2-1}-1}$$

When $p = 0$ or $p = 1$, either $C^*_{S(a)}(0)$ or $C^*_{S(a)}(1)$ is minimum, which equals to

$$C^*_{S(a)}(0) = C^*_{S(a)}(1) = 1 + \frac{2b^2}{b^2-1} \cdot e^{\frac{2\ln b}{b^2-1}-1}$$

Now we will regard p as a constant and we have

$$\frac{C^*_{S(a)}(b,p)}{db} == e^{\frac{2\ln b}{b^2-1}-1} \cdot \left(1 + 2b\cdot\sqrt{p\cdot(1-p)}\right) \cdot \frac{4b^2}{b^2-1} \times \left\{ \frac{\sqrt{p\cdot(1-p)}}{1+2b\cdot\sqrt{p\cdot(1-p)}} - \frac{2b\cdot\ln b}{(b^2-1)^2} \right\}$$

$$\because e^{\frac{2\ln b}{b^2-1}-1} \cdot \left(1 + 2b \cdot \sqrt{p \cdot (1-p)}\right) \cdot \frac{4b^2}{b^2-1} > 0$$

$\therefore \frac{dC^*_{S(a)}(b,p)}{db} = 0$ *if and only if* $\frac{\sqrt{p \cdot (1-p)}}{1 + 2b \cdot \sqrt{p \cdot (1-p)}} - \frac{2b \cdot \ln b}{(b^2-1)^2} = 0$. Therefore we have

$$\sqrt{p \cdot (1-p)} = \frac{2b \cdot \ln b}{(b^2-1)^2 - 4b^2 \cdot \ln b} \tag{3.10}$$

The solution of Equation 3.10 is

$$b = e^{RootOf\left(2Ze^Z + 4Z(e^Z)^2\sqrt{p\cdot(1-p)} - \sqrt{p\cdot(1-p)}(e^Z)^4 + 2\sqrt{p\cdot(1-p)}(e^Z)^2 - \sqrt{p\cdot(1-p)}\right)} \tag{3.11}$$

which makes $\frac{dC^*_{S(a)}(b,p)}{db} = 0$.

Here we introduce a conjecture on the lower bound of the truncated average case competitive ratio as follows.

Conjecture 3 *The optimal geometric ratio b which leads to the lower bound of $C^*_{S(a)}$ given the probability p is defined as $e^{RootOf\left(2Ze^Z + 4Z(e^Z)^2\sqrt{p\cdot(1-p)} - \sqrt{p\cdot(1-p)}(e^Z)^4 + 2\sqrt{p\cdot(1-p)}(e^Z)^2 - \sqrt{p\cdot(1-p)}\right)}$, where the function RootOf is a place holder for representing all the roots of an equation in one variable and it checks the validity of its arguments and is expressed in a single-argument canonical form, obtained by making the argument primitive and expressing the RootOf in terms of the global variable Z.*

Comparisons between the 1-STRAW problem and the SmartCow algorithm

In this part, we compare the lower bound of the truncated average case competitive ratio for the 1-STRAW problem with the lower bound of SmartCow algorithm. Since the SmartCow algorithm only applies to the unweighted case, therefore, the truncated average case competitive ratio we are using here is

$$C^*_{S(a)}(b, 0.5) = 1 + \frac{2b^2}{b-1} \cdot e^{\frac{2\ln b}{b^2-1}-1}$$

Firstly, we will demonstrate the similarities and differences between our strategy for the 1-STRAW problem and SmartCow algorithm. Refer to [36] and Figure 3.1.2. We have the following comparison table, see in Table 3.1. Both the competitive ratios in the table are for the average cases.

Comparisons		$1-STRAW$		$SmartCow$
Similarities	1	Two Rays	1	Two Paths
	2	One Robot	2	One Cow
	3	Geometric Ratio: $b > 1$	3	Geometric Ratio: $r > 1$
	4	Randomization Initially: $f(p) \cdot b^0, p \in U[0,1]$	4	Randomization Initially: $d \leftarrow r^\epsilon, \epsilon \in U[0,1]$
Differences	1	Weighted Case: $high, low, R_H, R_L$	1	Un-weighted Case: $p = q = 0.5$
	2	First Searched Ray: R_H	2	First Searched Path: Random Permutation on Two Paths
	3	Initial Searching Distances: $R_H : f(p) \cdot b^0$ $R_L : b$	3	Initial Searching Distances: $R_1 : r^\epsilon$ $R_2 : r^{\epsilon+1}$
	4	Accumulations: $R_H : f(p) \cdot b^{2i}$ $R_L : b^{2i+1}$	4	Accumulations: $R_1 : r^{\epsilon+i}$ $R_2 : r^{\epsilon+i+1}$
	5	Competitive Ratio: $C^*_{S(a)}(b, 0.5) = 1 + \frac{2b^2}{b-1} \cdot e^{\frac{2lnb}{b^2-1}-1}$	5	Competitive Ratio: $C^{smart}_{S(a)} = 1 + \frac{1+r}{lnr}$

Table 3.1: Comparisons of the 1-STRAW problem and the SmartCow algorithm

In addition, we computed the truncated average case competitive ratio $C^*_{S(a)}(b, 0.5)$ given

different geometric ratio b and the average case competitive ratio $C^{smart}_{S(a)}$ given various values of r. Without loss of generality, we use b as geometric ratio for both algorithms and produced a value table for both cases, see in Table 3.2.

	b	$C^*_{S(a)}(b, 0.5) = 1 + \frac{2b^2}{b-1} \cdot e^{\frac{2lnb}{b^2-1}-1}$	$C^{smart}_{S(a)}(b) = 1 + \frac{1+b}{lnb}$
1	1.5	7.3342	7.1658
2	1.7	6.3259	6.0883
3	1.9	5.8262	5.5182
4	2.0	5.6718	5.3280
5	2.5	5.3463	4.8197
6	3.0	5.3574	4.6409
7	3.5	5.5046	4.5921
8	3.59112	5.53998	4.59112
9	4.0	5.7207	4.6067
10	4.5	5.9769	4.6567

Table 3.2: Comparisons on the lower bounds of the 1-STRAW problem and the SmartCow algorithm

Here we can see from Table 3.2 that the optimal geometric ratio b for the 1-STRAW problem lies between $[2.5, 3.0]$ and the optimal value of the truncated average case competitive ratio $C^*_{S(a)}$ can be obtained between $[5.3463, 5.3574]$. Whereas, the optimal geometric ratio b of the SmartCow algorithm is 3.59112 and the lower bound of $C^{smart}_{S(a)}$ is 4.59112. Moreover, given the same b, $C^{smart}_{S(a)}$ is slightly smaller than $C^*_{S(a)}$.

The lower bound of the average case competitive ratio given our search strategy, therefore, is less than $\frac{5.3574 - 4.6409}{4.6409} = 15.4388\%$ worse than the lower bound of the average case competitive ratio obtained by the SmartCow algorithm , given the same geometric ratio b under this unweighted setting. The reason is because that the SmartCow algorithm is an optimal randomized algorithm and our search strategy for the 1-STRAW problem is in essence a deterministic algorithm.

However, the SmartCow algorithm only applies to the unweighted case. In the 1-STRAW problem, our proposed search strategy outweighs the SmartCow algorithm. To illustrate this point, let us consider a simple case where $p = 1$. In our search strategy, the robot will choose the ray with the probability of 1 to search for an unbounded distance, which ensures that the robot will find the target at its first try and the competitive ratio is 1, which is the optimal value. However, given the same settings, the SmartCow algorithm will make the robot to pick one of the two rays randomly which means that there is a half chance that the robot will choose the ray with the probability of 0 to search at the first round and will return to the origin after it fails to find the target. This guarantees that the competitive ratio after the first round given our search strategy is better than the one obtained by the SmartCow algorithm. Therefore, under weighted settings, our search strategy is the better choice than the SmartCow algorithm.

Chapter 4

Conclusions

In this thesis, we surveyed three classic online problems, which are the cow-path problem, the Processor-Allocation problem and the Robots-Search-Rays problem, and described the connections between these three problems.

In our core work, which is the One-Robot-Searches-Two-Rays-And-Weighted problem under the Robots-Search-Rays problem set, Chapter 2 Section 2.3, we proposed a new search strategy and an optimal ratio function $f(p)$ for the strategy given. We computed the representation of the worst case competitive ratio $C_S(w)$ given the optimal $f(p) = \sqrt{\frac{high}{low}}$, which is

$$C_{S(w)}(b,p) = \frac{3b^2-1}{b^2-1} + \frac{4b^3}{b^2-1} \cdot \left\{ \sqrt{p \cdot (1-p)} \right\}$$

In addition, we proved a tight lower bound on the worst case competitive ratio, which is $3 + \frac{6}{b^2-3}$ when equation $\sqrt{p \cdot (1-p)} = \frac{1}{b^3-3b}$ is fulfilled. This means that given the equation and any probability p, we can get the optimal b^* to reach the lower bound of $C_S(w) = 3 + \frac{6}{b^{*2}-3}$. The result is equivalent to the result of the doubling strategy, since when $p = 0.5$, the optimal b equals to two and the minimal worst case competitive ratio is 9. This shows our search strategy is no worse than the doubling strategy [41].

Moreover, we studied the average case competitive ratio for the 1-STRAW problem in two settings. We first assumed the target can be located on the two rays without boundaries. In this setting, we obtained the conjecture that the average case competitive ratio, as the robot searches towards to an unbounded distance, will converge to a limit, shown in Conjecture 2. Also we have the result that the optimal geometric ratio b is unbounded, which led us to study the truncated average case competitive ratio. In the truncated setting, the target is located in the range of $[-d', +d]$. Under this setting, we obtained the representation of the truncated average

case competitive ratio, which is

$$C^*_{S(a)} = 1 + \frac{2 \cdot b^2[1 + 2b \cdot \sqrt{p(1-p)}]}{b^2 - 1} \cdot e^{\frac{2lnb}{b^2-1}-1}$$

And we conjecture the lower bound of the truncated average case competitive ratio, which is shown in Conjecture 3.

Additionally, we compared our search strategy and its performances with Kao *et al.*'s SmartCow algorithm under the unweighted setting. It shows that the average case competitive ratio given our search strategy is less than $[15.4388\% - \epsilon]$ worse than that obtained by SmartCow algorithm [36], since in essence, our search strategy is a deterministic algorithm whereas the SmartCow algorithm is an optimal randomized search strategy. However, under the weighted settings, our search strategy outweighs the SmartCow algorithm.

Last but not least, there are some open problems raised in our research. The first open problem is to prove Conjecture 3. Secondly, the design and the computation of the lower bound of the randomized algorithms for the 1-STRAW problem remain unknown. Moreover, we know there are some connections between the Processor-Allocation problem and the Robots-Search-Rays problem in the unweighted case. We are interested in researching on the possible connections between the two problems in the weighted case as well. Additionally, we would like to consider more general case, which are the k-STRAW problem and the k-SnRAW (k-Robots-Search-n-Rays-And-Weighted) problem.

Bibliography

[1] S. Alpern and S. Gal. *The Theory of Search Games and Rendezvous*. Kluwer Academic Publishers, 2003. 5

[2] I. Althöfer. A symbiosis of man and machine beats grandmaster timoshchenko. *Journal of the International Computer Chess Association.*, 20(1):187, 1997. 9

[3] S. Angelopoulos, A. López-Ortiz, and A. Hamel. Optimal scheduling of contract algorithms with soft deadlines. In *AAAI*, pages 868–873, 2008. 9

[4] D. Angluin, J. Westbrook, and W. Zhu. Robot navigation with range queries. In *STOC*, pages 469–478, 1996. 12

[5] E. M. Arkin, H. Meijer, J. S. B. Mitchell, D. Rappaport, and S. Skiena. Decision trees for geometric models. *Int. J. Comput. Geometry Appl.*, 8(3):343–364, 1998. 1

[6] R. A. Baeza-Yates, J. C. Culberson, and G. J.E. Rawlins. Searching in the plane. *Information and Computation*, 106(2):234–252, 1991. 5, 13

[7] E. Bar-Eli, P. Berman, A. Fiat, and P. Yan. Online navigation in a room. *J. Algorithms*, 17:319–341, 1994. 5

[8] A. Beck. On the linear search problem. *Israel Journal of Math.*, 2:221–228, 1964. 5

[9] A. Beck. More on the linear search problem. *Israel Journal of Math.*, 3:61–70, 1965. 5

[10] A. Beck and D. Newman. Yet more on the linear search problem. *Israel Journal of Math.*, 8:419–429, 1970. 5

[11] A. Beck and P. Warren. The return of the linear search problem. *Israel Journal of Math.*, 10:169–183, 1972. 5

[12] R. Bellman. Problem 63-9*: An optimal search. *SIAM Review*, 5(2):274, 1963. 5

[13] D. S. Bernstein. Contract algorithms and robots on rays: Unifying two scheduling problems. In *IJCAI*, pages 1211 – 1217, 2003. 5, 9, 10, 11, 16

[14] D. S. Bernstein, T. J. Perkins, and S. Zilberstein. Scheduling contract algorithms on multiple processors. In *AAAI/IAAI 2002*, pages 702–706, 2002. 9

[15] M. Betke, R. L. Rivest, and M. Singh. Piecemeal learning of an unknown environment. *Machine Learning*, 18(2-3):231–254, 1995. 1

[16] A. Blum and P. Chalasani. An online algorithm for improving performance in navigation. *SIAM J. Comput.*, 29(6):1907–1938, 2000. 5

[17] A. Blum, P. Raghavan, and B. Schieber. Navigating in unfamiliar geometric terrain. *SIAM J. Comput.*, 26(1):110–137, 1997. 5

[18] K-F. Chan and T. W. Lam. An on-line algorithm for navigating in an unknown environment. *International Journal of Computational Geometry and Applications*, 3:227–244, 1993. 5

[19] R. E. Tarjan D. D. Sleator and R. Endre. Self-adjusting binary search trees. *J. ACM*, 32(3):652–686, 1985. iii, 2

[20] A. Datta and Ch. Icking. Competitive searching in a generalized street. *Comput. Geom.*, 13(2):109–120, 1999. 5

[21] T. Dean and M. S. Boddy. An analysis of time-dependent planning. In *AAAI*, pages 49–54, 1988. 9

[22] R. Dorrigiv. *Alternative Measures for the Analysis of Online Algorithms*. PhD thesis, Department of Computer Science, University of Waterloo, 2010. iii, 2

[23] Reza Dorrigiv and Alejandro López-Ortiz. A survey of performance measures for on-line algorithms. *SIGACT News*, 36(3):67–81, 2005. 2

[24] G. Dudek, K. Romanik, and S. Whitesides. Localizing a robot with minimum travel. *SIAM J. Comput.*, 27(2):583–604, 1998. 12

[25] Computational Geometry Impact Task Force. Application challenges to computational geometry. Technical report, Princeton University, 1996. 6

[26] S. Gal. A general search game. *Israel Journal of Math.*, 12:32–45, 1972. 5

[27] S. Gal. Minimax solutions for linear search problems. *SIAM Journal of Applied Math.*, 27(1):17–30, 1974. 5

[28] S. Gal. *Search Games*. Academic Press, 1980. 5, 12, 15

[29] L. Guibas, R. Motwani, and P. Raghavan. The robot localization problem in two dimensions. In *SODA*, pages 259–268, 1992. 12

[30] E. Horowitz, D. D. Sleator, and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28(2):202–208, 1985. iii, 2

[31] E. Horvitz. Reasoning under varying and uncertain resource constraints. In *AAAI*, pages 111–116, 1988. 9

[32] S. Hutchinson. Exploiting visual constraints in robot motion planning. In *IEEE Conference on Robotics and Automation*, pages 1722–1727, 1991. 12

[33] Ch. Icking and R. Klein. Searching for the kernel of a polygon: A competitive strategy. In *Symposium on Computational Geometry*, pages 258–266, 1995. 5

[34] Ch. Icking, R. Klein, and E. Langetepe. How to find a point on a line within a fixed distance. Informatik-Bericht 220, Fernuni Hagen, November 1997. 5

[35] B. Kalyanasundaram and K. Pruhs. A competitive analysis of nearest neighbor based algorithms for searching unknown scenes. In *STACS*, pages 147–157. Springer-Verlag, 1992. 5

[36] M. Y. Kao, J. H. Reif, and S. R. Tate. Searching in an unknown environment: an optimal randomized algorithm for the cow-path problem. *Inf. Comput.*, 131(1):63–79, 1996. iii, ix, 4, 5, 6, 7, 8, 9, 16, 55, 59

[37] M.Y Kao, Y. Ma, M. Sipser, and Y. Yin. Optimal constructions of hybrid algorithms. *CoRR*, cs.DM/0101028, 2001. 4, 5, 9, 16

[38] A. R. Karlin, M. S. Manasse, L. Rudolph, and D. D. Sleator. Competitive snoopy caching. *Algorithmica*, 3:77–119, 1988. iii, 2

[39] R. Klein. Walking an unknown street with bounded detour. *Comput. Geom. Theory Appl.*, 1, 1992. 1

[40] Y. Liu and A. López-Ortiz. *manuscript*, 2010. 4

[41] A. López-Ortiz. *On-line Searching on Bounded and Unbounded Domains*. PhD thesis, Department of Computer Science, University of Waterloo, 1996. iii, 1, 7, 58

[42] A. López-Ortiz, S. Angelopoulos, and A. M. Hamel. Optimal scheduling of contract algorithms for anytime problems. In *AAAI*. American Association for Artificial Intelligence, 2006. ix, 5, 9, 10, 11, 12, 13

[43] A. López-Ortiz and S. Schuierer. On-line parallel heuristics, processor scheduling and robot searching under the competitive framework. *Theoretical Computer Science*, 310(1-3):527 – 537, 2004. 4, 5, 7, 13, 14, 15, 16, 20

[44] A. López-Ortiz and G. Sweet. Parallel searching on a lattice. In *CCCG*, pages 125–128, 2001. 5

[45] R. Karp G. Tardos S. Ben-David, A. Borodin and A. Wigderson. On the power of randomization in on-line algorithms. *Algorithmica*, 11(1):2–14, 1994. 2

[46] S. Schuierer. Lower bounds in on-line geometric searching. *Computational Geometry: Theory and Applications*, 18(1):37–53, 2001. 12, 15

[47] T. Kameda X. Deng and C. Papadimitriou. How to learn an unknown environment i: The rectilinear case. *J. ACM*, 45(2):215–245, 1998. 1

[48] S. Zilberstein. Using anytime algorithms in intelligent systems. *AI Magazine*, 17(3):73–83, 1996. 9

[49] S. Zilberstein, F. Charpillet, and P. Chassaing. Real-time problem-solving with contract algorithms. In *IJCAI*, pages 1008–1015, 1999. 9, 11