이학 박사 학위논문

Comparison of Cryptanalytic Time Memory Tradeoff Algorithms with Focus on Some Rainbow Variants

(몇 가지 개량된 레인보우 절충기법을 포함한 시간 저장 공간 절충기법 비교)

2016년 2월

서울대학교 대학원

수리과학부

김병일

Abstract

Comparison of Cryptanalytic Time Memory Tradeoff Algorithms with Focus on Some Rainbow Variants

Byoung-Il Kim

Department of Mathematical Sciences

The Graduate School

Seoul National University

Cryptanalytic time memory tradeoff algorithms are tools for inverting one-way functions, and they are used to recover passwords from unsalted password hashes. There are many publicly known tradeoff algorithms, and the rainbow tradeoff algorithm, which is widely believed to be the best tradeoff algorithm, at least among implementers, has been the most popular method.

In this thesis, we provide accurate complexity analyses of the thick rainbow tradeoff algorithm and the non-perfect and perfect table fuzzy rainbow tradeoff algorithms. These are algorithms that have not yet received much attention. Our analyses show that, when the pre-computation cost and the online execution efficiency are both taken into consideration, the perfect table fuzzy rainbow tradeoff can be seen as performing the best among the three algorithms considered and actually even better than the original rainbow tradeoff.

The computational complexities for some time memory data tradeoff methods are also analyzed. The multi-target tradeoffs that we cover are the classical Hellman, distinguished point, and fuzzy rainbow methods, both in their non-perfect and perfect table versions for the latter two methods. We find that their execution complexities are no different from the complexities

of the corresponding single-target algorithms executed under certain matching parameters. As in the single-target case, we conclude that the perfect table fuzzy rainbow tradeoff algorithm is the most preferable among the multi-target tradeoff algorithms we have considered.

Key words: Cryptanalytic Time memory tradeoff, Time memory data tradeoff, Thick rainbow tradeoff, Fuzzy rainbow tradeoff, Perfect table, Non-perfect table

Student Number: 2010–20240

Contents

Chapter 1

Introduction

Cryptanalytic time memory tradeoff algorithms, first introduced by Hellman [14], are useful techniques for inverting generic one-way functions that utilize pre-computed tables. These are used by hackers and law enforcement authorities to recover passwords from stored password hashes. A typical tradeoff algorithm executes a pre-computation phase that requires as much computation as the exhaustive search of all possible inputs, but stores the obtain information in tables whose combined size is much smaller than the size of the complete dictionary. The input corresponding to a given inversion target is recovered in the online phase, whose computational complexity is of much smaller order than that of the exhaustive search of all possible inputs.

The classical Hellman method is the very first work on cryptanalytic tradeoffs and the book [11] states that this was soon followed by Rivest's suggestion called the distinguished point(DP) method. Nowadays, the rainbow table method [26] is most widely believed to be the best tradeoff algorithm.

There is a recent work [17] that gives a noteworthy technique for comparing the performances of various single-target tradeoff algorithms. Different from previous approaches that compared only the online efficiencies of the algorithms, it suggested a method that considers both the pre-computation cost and the online execution behavior. Hence, we can expect this method to be more practically meaningful when discussing whether one tradeoff algorithm is better than another algorithm. This method was used in [16] to

compare the parallel version of the DP tradeoff with other major tradeoff algorithms. This approach was also used [22] on the perfect table versions of single-target tradeoffs to arrive at the conclusion that the perfect rainbow tradeoff outperforms the perfect DP tradeoff [9, 10] under typical situations.

There are some variants of the original rainbow tradeoff that have been designed for use in the multi-target setting. These are the thin rainbow, the thick rainbow, and the fuzzy rainbow tradeoffs, which were first introduced by [3, 5] in 2006. In this thesis, we analyze the online performances of the latter two tradeoff methods and compare them against those of the perfect and non-perfect rainbow tradeoff methods by applying the comparison technique of [17].

The fuzzy rainbow tradeoff appears in works other than the original proposal [3, 5]. Two presentations [24, 25] introduced the algorithm as an integral component of a fully implemented attack on GSM mobile phones, and an elementary analysis of the method was given in [28]. However, it seems that the authors of these works were not aware of the preceding works [3, 5]. In fact, the analysis given by [28] fell short of even the arguments given by [3, 5]. There is also a performance analysis of a practically adapted implementation of the fuzzy rainbow tradeoff [13] that follows the approach of [21] for treating the practical adaptation of the original rainbow tradeoff.

The main contribution of this thesis is the accurate online execution behavior analyses of the thick rainbow tradeoff and both the non-perfect and perfect table fuzzy rainbow tradeoffs that take the effects of false alarms fully into account. We also use the results of our analyses to compare many tradeoff algorithms against each other. Our conclusion is that the perfect fuzzy rainbow tradeoff has the best performance among the three algorithms analyzed and is also advantageous over both the non-perfect and perfect table original rainbow tradeoffs. This fact implies that the perfect fuzzy rainbow tradeoff, which has not yet received widespread recognition, is preferable to any other widely known tradeoff algorithm.

We also find that the non-perfect table fuzzy rainbow tradeoff has better performance than the thick rainbow tradeoff and that it outperforms the original non-perfect rainbow tradeoff. However, we find that it cannot attain

certain high level of efficiencies that can be reached by the original perfect rainbow tradeoff, regardless of its parameter choices. Even for those efficiency levels that can be attained by both algorithms, the non-perfect fuzzy rainbow algorithm can do so with less pre-computation cost than the perfect rainbow tradeoff.

A later part of this thesis provides analyses of some multi-target time memory tradeoffs. The multi-target variants of the tradeoff algorithms are also referred to as the time memory *data* tradeoffs, and these have been used [2, 4, 7, 8, 12] to attack streamciphers and to show that GSM phones are insecure.

In this thesis, we deal with the multi-target versions of the Hellman, DP, and fuzzy rainbow tradeoff algorithms, both in their non-perfect and perfect table methods for the latter two algorithms. It is known through [6] and [3, 5] that straightforward multi-target adaptations of the usual rainbow tradeoff and the thick rainbow tradeoff, respectively, are strictly inferior to other algorithms, and hence, we will not deal with them.

Heuristic arguments can be used to show that the five multi-target tradeoff algorithms that we handle behave comparably in the asymptotic sense. We provide a theoretical treatment of the online execution behaviors of the five time memory data tradeoff algorithms that does not drop any small constant factors. Such an accurate treatment is essential for a practical comparison of the multi-target tradeoffs and allows for one to decide on which algorithm is better than the others.

Our analysis of the multi-target tradeoffs depends heavily on that of the corresponding single-target methods. Based on a detailed understanding of the existing results, we single out just the fundamental arguments and apply them to the five algorithms in the multi-target setting, treating all of them simultaneously. We come to the conclusions that there is no difference between the single-target and multi-target settings in analyzing and comparing the tradeoff algorithms.

The rest of this thesis is organized as follows. In Chapter 2, we review previous results on the well known major tradeoff algorithms and recall the three rainbow variant algorithms that we will be dealing with. Chapter 3

gives full analyses of the online execution behaviors of the three rainbow variants in the single-target setting. The physical memory size required to store the pre-computation tables for each of the analyzed algorithms is discussed in Chapter 4. The results of our analyses are finally used in Chapter 5 to compare the performances of the three rainbow variants with that of the original perfect and non-perfect rainbow tradeoffs. In Chapter 6, we provide the analyses of some multi-target tradeoff algorithms. The experimental data supporting our technical arguments are given in Chapter 7. Finally, this thesis is summarized in Chapter 8. All the contents regarding the non-perfect table and perfect table fuzzy rainbow tradeoffs can be found in [18, 19] and [20], respectively.

Chapter 2

Preliminaries

In this chapter, we review the major tradeoff algorithms and recall some variants of rainbow method, with some fixed notations and terminologies.

Throughout this thesis, the one-way function $f : \mathcal{N} \to \mathcal{N}$ is taken to act on a search space $\mathcal{N}$ of size $\mathtt{N}$, and is always assumed to be a random function. The composition of the one-way function f and the reduction function of i-th color will be written as f_i. The standard parameters for the number of (randomly chosen) starting points per table m_0, the number of chains per table m, the (expected) chain length of a (sub-)matrix t, and the number of tables ℓ will be used. For the rainbow variants, s will be used as the number of colors. The distinguishing property for distinguished points (DPs) will always be assumed to be of probability $\frac{1}{t}$ so that the expected length of a random DP chain is t, when dealing with the DP and fuzzy rainbow tradeoffs. We refer to the collection of all m chains associated with one pre-computation table as a pre-computation matrix.

We use eight symbols for the tradeoff algorithms; $\mathtt{H}$ (classical Hellman), $\mathtt{D}$ (non-perfect DP), $\bar{\mathtt{D}}$ (perfect DP), $\mathtt{R}$ (non-perfect rainbow), $\bar{\mathtt{R}}$ (perfect rainbow), $\mathtt{K}$ (thick rainbow), $\mathtt{F}$ (non-perfect fuzzy rainbow), and $\bar{\mathtt{F}}$ (perfect fuzzy rainbow). We also use the symbol $\mathtt{X}$ for an arbitrary tradeoff algorithm.

The *matrix stopping constants* are defined as $\mathtt{X}_{\mathrm{msc}} = \frac{mt^2}{\mathtt{N}}$ for $\mathtt{X} = \mathtt{H}, \mathtt{D}, \bar{\mathtt{D}}$, $\mathtt{X}_{\mathrm{msc}} = \frac{mt}{\mathtt{N}}$ for $\mathtt{X} = \mathtt{R}, \bar{\mathtt{R}}$, $\mathtt{X}_{\mathrm{msc}} = \frac{mts}{\mathtt{N}}$ for $\mathtt{X} = \mathtt{K}$, and $\mathtt{X}_{\mathrm{msc}} = \frac{mt^2 s}{\mathtt{N}}$ for $\mathtt{X} = \mathtt{F}, \bar{\mathtt{F}}$, which are all assumed to be neither too large nor very close to zero. We use

the notation $\mathtt{X}_{\mathrm{ps}}$ to present the *success probability* for each tradeoff algorithm. The *coverage rate* $\mathtt{X}_{\mathrm{cr}}$ of a matrix for $\mathtt{X} = \mathtt{H}, \mathtt{D}, \bar{\mathtt{D}}, \mathtt{R}, \bar{\mathtt{R}}$ is defined to be the number of distinct points in a matrix of each tradeoff, divided by mt. For $\mathtt{X} = \mathtt{K}, \mathtt{F}, \bar{\mathtt{F}}$, $\mathtt{X}_{\mathrm{cr}}$ is defined a bit differently, and it will be defined later precisely. We also refer to the number of distinct points in the i-th colored sub-matrix of a matrix, divided by mt, as the *coverage rate of the i-th sub-matrix* $\mathtt{X}_{\mathrm{cr},i}$ for $\mathtt{X} = \mathtt{K}, \bar{\mathtt{F}}$.

The work [17] considers two elements to compare various tradeoff algorithms. One is the *pre-computation cost* and the other is the *online efficiency*. Let us define the *pre-computation coefficient* $\mathtt{X}_{\mathrm{pc}}$ and the *tradeoff coefficient* $\mathtt{X}_{\mathrm{tc}}$ for each tradeoff algorithm to be

$$\mathtt{X}_{\mathrm{pc}} = \frac{P}{\mathsf{N}} \text{ and } \mathtt{X}_{\mathrm{tc}} = \frac{TM^2}{\mathsf{N}^2},$$

where P is the pre-computation time complexity, T is the online time complexity, and M is the storage complexity for each algorithm. Since lower value of $\mathtt{X}_{\mathrm{pc}}$ means less pre-computation cost and lower value of $\mathtt{X}_{\mathrm{tc}}$ means higher online efficiency, these two coefficient can be good measures to compare the tradeoff algorithms.

Any implementation of an algorithm that relies on DPs ($\mathtt{X} = \mathtt{D}, \bar{\mathtt{D}}, \mathtt{F}, \bar{\mathtt{F}}$) will set a chain length bound to detect chains falling into loops. We will assume that the chain length bound for each algorithm is sufficiently large, so that the effects from the discarding of chains can be negligible. A more detailed discussion for the chain length bound can be found in [17, 22].

There are two approximation techniques in our analyses of the tradeoff algorithms. The first one is the relation $(1 - \frac{1}{b})^a \approx e^{-\frac{a}{b}}$, which is appropriate when $a = O(b)$. A more detailed statement can be found in [17, Appendix A]. The second one is the approximation of a sum over a large index set into a definite integral. Under any reasonable choice of tradeoff parameters, these approximations will be very accurate whenever we apply them. We will also ignore multiplicative factors of $1 + O\big(\frac{1}{t}\big)$ size.

2.1 Previous Results of Major Algorithms

Some previous results of the classical Hellman, the DP, and the rainbow tradeoffs related to this thesis is introduced in this section. More details and proofs of the results can be found in [17, 22].

2.1.1 Hellman Tradeoff

Recall that a Hellman chain is of the form

$$\mathrm{SP} \xrightarrow{f} \circ \xrightarrow{f} \circ \ \cdots \ \circ \xrightarrow{f} \mathrm{EP},$$

where SP and EP denote the starting and ending point, respectively. In the pre-computation phase, length-t $m(= m_0)$ Hellman chains from m randomly chosen starting points are generated per table, and the collection of m chains is called a pre-computation Hellman matrix.

The coverage rate of a single pre-computation Hellman matrix is computed through the formula

$$\mathtt{H}_{\mathrm{cr}} = \frac{\sqrt{2}}{\sqrt{\mathtt{H}_{\mathrm{msc}}}} \frac{e^{\sqrt{2\mathtt{H}_{\mathrm{msc}}}} - 1}{e^{\sqrt{2\mathtt{H}_{\mathrm{msc}}}} + 1},$$

which is a simple modification of the statement in [23], where $\mathtt{H}_{\mathrm{msc}} = \frac{mt^2}{\mathsf{N}}$ is neither too large nor very close to zero. This formula will be used for the analysis of the thick rainbow tradeoff, since a sub-matrix of a thick rainbow matrix looks like a Hellman matrix.

The success probability of the classical Hellman tradeoff is $\mathtt{H}_{\mathrm{ps}} = 1 - e^{-\mathtt{H}_{\mathrm{cr}}\mathtt{H}_{\mathrm{pc}}}$, and its pre-computation coefficient is defined to be $\mathtt{H}_{\mathrm{pc}} = \frac{mt\ell}{\mathsf{N}}$, which can be rewritten from the form of $\mathtt{H}_{\mathrm{ps}}$ as

$$\mathtt{H}_{\mathrm{pc}} = -\frac{\ln(1 - \mathtt{H}_{\mathrm{ps}})}{\mathtt{H}_{\mathrm{cr}}}.$$

The time memory tradeoff curve for the classical Hellman tradeoff is given

as $TM^2 = \mathtt{H}_{\mathrm{tc}}\mathtt{N}^2$, where the tradeoff coefficient is

$$\mathtt{H}_{\mathrm{tc}} = \Big(\frac{1}{\mathtt{H}_{\mathrm{msc}}} + \frac{1}{6}\Big)\frac{1}{\mathtt{H}_{\mathrm{cr}}^3}\mathtt{H}_{\mathrm{ps}}\big\{\ln(1-\mathtt{H}_{\mathrm{ps}})\big\}^2.$$

One can easily perceive that, when the success probability $\mathtt{H}_{\mathrm{ps}}$ is fixed, both $\mathtt{H}_{\mathrm{pc}}$ and $\mathtt{H}_{\mathrm{tc}}$ are functions of the single variable $\mathtt{H}_{\mathrm{msc}}$.

2.1.2 DP Tradeoff

Recall that the DP tradeoffs use variable length pre-computation and online chains of the form

$$\mathrm{SP} \xrightarrow{f_i} \circ \xrightarrow{f_i} \circ \ \cdots\ \circ \xrightarrow{f_i} \mathrm{DP} = \mathrm{EP},$$

with a preset distinguishing property that defines DPs, where SP and EP denote the starting and ending points, respectively.

For the non-perfect version of the DP tradeoff, $m(=m_0)$ DP chains from m randomly chosen starting points are generated per table, and the collection of m chains is called a non-perfect DP pre-computation matrix, whose average chain length is t.

The coverage rate of a single non-perfect DP matrix is given as

$$\mathtt{D}_{\mathrm{cr}} = \frac{2}{1+\sqrt{1+2\mathtt{D}_{\mathrm{msc}}}},$$

where $\mathtt{D}_{\mathrm{msc}} = \frac{mt^2}{\mathtt{N}}$ is neither too large nor very close to zero. Since the number of distinct entries $|\mathtt{DM}|$ contained in a non-perfect DP matrix $\mathtt{DM}$ satisfies

$$|\mathtt{DM}| = m_{\mathrm{ep}}t,$$

where m_{ep} denotes the number of distinct ending points of $\mathtt{DM}$, it can be obtained that

$$m_{\mathrm{ep}} = \frac{2m}{1+\sqrt{1+2\mathtt{D}_{\mathrm{msc}}}}.$$

As the cases for the Hellman tradeoff, the probability of success of the non-

perfect DP tradeoff is $\mathtt{D}_{\rm ps} = 1 - e^{-\mathtt{D}_{\rm cr}\mathtt{D}_{\rm pc}}$, and the pre-computation coefficient is defined to be $\mathtt{D}_{\rm pc} = \frac{mt\ell}{\mathsf{N}}$, which is also written as

$$\mathtt{D}_{\rm pc} = -\frac{\ln(1-\mathtt{D}_{\rm ps})}{\mathtt{D}_{\rm cr}}.$$

The time memory tradeoff curve for the non-perfect DP tradeoff is given as $TM^2 = \mathtt{D}_{\rm tc}\mathsf{N}^2$, where the tradeoff coefficient is

$$\mathtt{D}_{\rm tc} = \Big(2 + \frac{1}{\mathtt{D}_{\rm msc}}\Big)\frac{1}{\mathtt{D}_{\rm cr}^3}\mathtt{D}_{\rm ps}\big\{\ln(1-\mathtt{D}_{\rm ps})\big\}^2.$$

Both $\mathtt{D}_{\rm pc}$ and $\mathtt{D}_{\rm tc}$ are functions of the single variable $\mathtt{D}_{\rm msc}$, under a fixed succes rate $\mathtt{D}_{\rm ps}$.

For the perfect table DP tradeoff, after generating DP chains from m_0 randomly chosen starting points for a matrix, one retains the longest chain and discard other chains among the chains containing identical ending points. The average chain length of a perfect DP matrix with m chains must be less than t, since the length of discarded chains would be long.

The coverage rate of a single perfect DP matrix is given as

$$\bar{\mathtt{D}}_{\rm cr} = \frac{2}{\bar{\mathtt{D}}_{\rm msc}}\ln\Big(1 + \frac{\bar{\mathtt{D}}_{\rm msc}}{2}\Big),$$

where $\bar{\mathtt{D}}_{\rm msc} = \frac{mt^2}{\mathsf{N}}$ is neither too large nor very close to zero. The probability of success of the perfect DP tradeoff is $\bar{\mathtt{D}}_{\rm ps} = 1 - e^{-\frac{2\bar{\mathtt{D}}_{\rm cr}\bar{\mathtt{D}}_{\rm pc}}{2+\bar{\mathtt{D}}_{\rm msc}}}$, and its pre-computation coefficient $\mathtt{D}_{\rm pc} = (1 + \frac{\bar{\mathtt{D}}_{\rm msc}}{2})\frac{mt\ell}{\mathsf{N}}$ is written as

$$\bar{\mathtt{D}}_{\rm pc} = -\frac{2+\bar{\mathtt{D}}_{\rm msc}}{2\bar{\mathtt{D}}_{\rm cr}}\big\{\ln(1-\bar{\mathtt{D}}_{\rm ps})\big\}.$$

The time memory tradeoff curve is given as $TM^2 = \bar{\mathtt{D}}_{\rm tc}\mathsf{N}^2$, where the tradeoff coefficient is

$$\bar{\mathtt{D}}_{\rm tc} = \Big(1 + \frac{1+0.577\bar{\mathtt{D}}_{\rm msc}}{1+0.451\bar{\mathtt{D}}_{\rm msc}}\frac{\bar{\mathtt{D}}_{\rm msc}}{1+\bar{\mathtt{D}}_{\rm msc}}\Big)\frac{1}{\bar{\mathtt{D}}_{\rm msc}\bar{\mathtt{D}}_{\rm cr}^3}\bar{\mathtt{D}}_{\rm ps}\big\{\ln(1-\bar{\mathtt{D}}_{\rm ps})\big\}^2,$$

and hence, both $\bar{\mathtt{D}}_{\rm pc}$ and $\bar{\mathtt{D}}_{\rm tc}$ are functions of the single variable $\bar{\mathtt{D}}_{\rm msc}$, when

the succes rate requirement $\bar{\mathtt{D}}_{\mathrm{ps}}$ is fixed.

Some results of DP tradeoff will be employed for the non-perfect and perfect fuzzy rainbow tradeoffs, since a fuzzy rainbow matrix is a concatenation of DP sub-matrices.

2.1.3 Rainbow Tradeoff

The rainbow tradeoffs use length-t pre-computation chains of the form

$$\mathrm{SP} \xrightarrow{f_1} \circ \xrightarrow{f_2} \circ \ \cdots \ \circ \xrightarrow{f_t} \mathrm{EP},$$

with t different colors, where SP and EP denote the starting and ending points, respectively. An online chain is generated by applying the one-way functions from one of the colors $1 \leq i \leq t$ to the t-th color.

In the pre-computation phase of the non-perfect table rainbow tradeoff, $m(= m_0)$ rainbow chains from m randomly chosen starting points are generated per table.

The pre-computation coefficient of the non-perfect rainbow tradeoff is computed as

$$\mathtt{R}_{\mathrm{pc}} = \frac{mt\ell}{\mathsf{N}} = \mathtt{R}_{\mathrm{msc}}\ell,$$

and the success probability of that is

$$\mathtt{R}_{\mathrm{ps}} = 1 - \Big(\frac{2}{2+\mathtt{R}_{\mathrm{msc}}}\Big)^{2\ell},$$

where $\mathtt{R}_{\mathrm{msc}} = \frac{mt}{\mathsf{N}}$. Its time memory tradeoff curve is $TM^2 = \mathtt{R}_{\mathrm{tc}}\mathsf{N}^2$, where the tradeoff coefficient is

$$\mathtt{R}_{\mathrm{tc}} = \frac{\ell^3}{(2\ell+1)(2\ell+2)(2\ell+3)}\Big(\big\{(2\ell-1)+(2\ell+1)\mathtt{R}_{\mathrm{msc}}\big\}(2+\mathtt{R}_{\mathrm{msc}})^2 - 4\big\{(2\ell-1)+\ell(2\ell+3)\mathtt{R}_{\mathrm{msc}}\big\}\big(\frac{2}{2+\mathtt{R}_{\mathrm{msc}}}\big)^{2\ell}\Big).$$

Under a fixed success rate requirement $\mathtt{R}_{\mathrm{ps}}$, since each positive integer value of ℓ determines the value of $\mathtt{R}_{\mathrm{msc}}$, both $\mathtt{R}_{\mathrm{pc}}$ and $\mathtt{R}_{\mathrm{tc}}$ are functions of the single

integer variable ℓ.

To make a perfect rainbow table, the number of randomly chosen starting points m_0 should be set so that the number of distinct ending points is m. Some starting and ending point pairs can have duplicate ending points, and only one of these pairs is recorded in the table.

The pre-computation coefficient and the success probability of the perfect rainbow tradeoff are written as

$$\bar{\mathsf{R}}_{\mathrm{pc}} = \frac{2}{2-\bar{\mathsf{R}}_{\mathrm{msc}}}\frac{mt\ell}{\mathsf{N}} = \frac{2\bar{\mathsf{R}}_{\mathrm{msc}}}{2-\bar{\mathsf{R}}_{\mathrm{msc}}}\ell$$

and

$$\bar{\mathsf{R}}_{\mathrm{ps}} = 1 - \exp\Big(-\bar{\mathsf{R}}_{\mathrm{msc}}\ell\Big),$$

respectively, where $\bar{\mathsf{R}}_{\mathrm{msc}} = \frac{mt}{\mathsf{N}}$. The time memory tradeoff curve for that is $TM^2 = \bar{\mathsf{R}}_{\mathrm{tc}}\mathsf{N}^2$, where the tradeoff coefficient is

$$\begin{aligned}\bar{\mathsf{R}}_{\mathrm{tc}} =& \Big(\bar{\mathsf{R}}_{\mathrm{msc}}\ell - \frac{\bar{\mathsf{R}}_{\mathrm{msc}}}{2} + \ell - 2 + \frac{3}{2\ell}\Big) \\ &- \Big(\frac{\bar{\mathsf{R}}_{\mathrm{msc}}^2\ell}{4} + \bar{\mathsf{R}}_{\mathrm{msc}}\ell^2 - \bar{\mathsf{R}}_{\mathrm{msc}}\ell + \bar{\mathsf{R}}_{\mathrm{msc}} + \ell - 2 + \frac{3}{2\ell}\Big)e^{-\bar{\mathsf{R}}_{\mathrm{msc}}\ell}.\end{aligned}$$

As for the non-perfect rainbow tradeoff, it can be easily learn that both $\bar{\mathsf{R}}_{\mathrm{pc}}$ and $\bar{\mathsf{R}}_{\mathrm{tc}}$ are functions of the single integer variable ℓ, when the success rate requirement $\bar{\mathsf{R}}_{\mathrm{ps}}$ is fixed.

The non-perfect and perfect rainbow tradeoff algorithms will be chosen as the comparison targets of the rainbow variants that we are going to analyze, since they are the outstanding algorithms among the major algorithms, as shown in [17, 22].

2.2 Some Rainbow Variants

Barkan et al. [3, 5] introduced some variants of the rainbow tradeoff, which are the thin rainbow, thick rainbow, and fuzzy rainbow tradeoffs. In this thesis, we will analyze and compare the latter two algorithms, with both non-perfect and perfect table versions for the fuzzy rainbow method. Prior

to the analysis, we quickly review each variant and fix some terminologies in this section.

2.2.1 Thick Rainbow Tradeoff

The thick rainbow tradeoff can be viewed as a combination of the classical Hellman tradeoff and the usual rainbow tradeoff. One fixes a positive integer s and generates pre-computation chains of the form

$$\begin{aligned}\mathrm{SP} \xrightarrow{f_1} \circ \cdots \circ \xrightarrow{f_1} \bigcirc \xrightarrow{f_2} \circ \cdots \circ \xrightarrow{f_2} \bigcirc \xrightarrow{f_3} \circ \cdots \\ \cdots \xrightarrow{f_{s-1}} \bigcirc \xrightarrow{f_s} \circ \cdots \circ \xrightarrow{f_s} \mathrm{EP}\end{aligned}$$

which is referred to as a thick rainbow chain. That is, one iterates the one-way function f_i, t times, of a fixed color i, after which the iterations are continued under a different color. Total s colors are used for each pre-computation chain, so that the chain length becomes ts. In another words, each iteration of a rainbow chain is replaced by a length-t Hellman chain, except that the number of colors used by each pre-computation table is s. We assume that the parameters m, t, and s are chosen in such a way that the matrix stopping constant $\mathtt{K}_{\mathrm{msc}} = \frac{mts}{\mathsf{N}}$ is neither too large nor very close to zero.

As in the pre-computation phases of other tradeoff algorithms, $m(= m_0)$ thick rainbow chains are generated for each of the ℓ tables, and only the starting and ending point pairs, sorted according to the ending points, are stored in the tables.

A thick rainbow matrix may be considered as a concatenation of s Hellman sub-matrices. We write $\mathtt{HM}_i$ to refer to the i-th Hellman sub-matrix of a single thick rainbow matrix for $1 \leq i \leq s$. The ending points of one Hellman sub-matrix become the starting points for the next Hellman sub-matrix, and the only difference between $\mathtt{HM}_i$ and a standard Hellman matrix is that $\mathtt{HM}_i$ may contain duplicate starting points, that lead to completely identical chains.

The procedure of the online phase necessitates more clarification. In the first pass, all the s-th Hellman sub-matrices $\mathtt{HM}_s$ of the ℓ pre-computation

matrices are searched in parallel. For $0 \leq j \leq t-1$, every length-j s-th colored chain, for the s-th Hellman sub-matrix $\texttt{HM}_s$ of each pre-computation matrix, is generated through t iterations by recording each point from each iteration. All the alarms induced by possible merges of these $\ell \times t$ online chains are fully resolved before the second pass. The i-th pass is executed only if the correct answer of the target could not be found during the $(i-1)$-th pass. In the i-th pass, all of the $(s-i+1)$-th Hellman sub-matrices $\texttt{HM}_{s-i+1}$ of the ℓ pre-computation matrices are searched in parallel. For each matrix, all the length-j, for $0 \leq j \leq t-1$, Hellman sub-chains of the $(s-i+1)$-th color are created through t iterations by recording each point from each iteration, and the iterations are continued to each length-j sub-chain until s-th color. All induced alarms are handled by regenerating the associated pre-computation chain up to the $(t-j)$-th point of the $(s-i+1)$-th color. The online phase is continued until either the correct answer to the inversion target is found or all the s passes are complete, but all the computation associated with each i-th pass is fully completed even if the correct answer is found during the pass.

In real implementations, the ℓ pre-computation tables of the thick rainbow tradeoff may or may not be processed in parallel, as the implementation of the usual rainbow tradeoff. It is enough to generate the online chains and to deal with all the alarms in serial order during each pass, and this will not give a considerable change to the online execution complexity.

2.2.2 Non-Perfect Table Fuzzy Rainbow Tradeoff

The fuzzy rainbow tradeoff is a combination of the original rainbow tradeoff and the DP tradeoff. Under a fixed distinguishing property of probability $\frac{1}{t}$ and a fixed positive integer s, pre-computation fuzzy rainbow chains of the form

$$\begin{aligned} \text{SP} \xrightarrow{f_1} \circ \cdots \circ \xrightarrow{f_1} \text{DP} \xrightarrow{f_2} \circ \cdots \circ \xrightarrow{f_2} \text{DP} \xrightarrow{f_3} \circ \cdots \\ \cdots \xrightarrow{f_{s-1}} \text{DP} \xrightarrow{f_s} \circ \cdots \circ \xrightarrow{f_s} \text{DP} = \text{EP} \end{aligned}$$

are used. That is, the one-way function iterations are continued under a fixed color until a DP is appeared, and the DP, as the ending point of this DP sub-chain, is used as the starting point of the following DP sub-chain. The color is changed at each intermediate DP until the s-th DP is reached, so that the average chain length becomes ts. In other words, each iteration of a rainbow chain is replaced by a DP chain, except that the number of colors used by each pre-computation table is s and that the expected chain length of each DP sub-chain is t.

In the pre-computation phase of the non-perfect version of the fuzzy rainbow tradeoff, $m(= m_0)$ chains are generated per table, and each pair of the starting and ending points is recorded in each pre-computation table, after being sorted according to the ending points.

We always assume that the parameters m, t, and s, for the non-perfect table fuzzy rainbow tradeoff, are chosen to satisfy that the matrix stopping constant $\mathtt{F}_{\mathrm{msc}} = \frac{mt^2s}{\mathtt{N}}$ is neither too large nor very close to zero. The theoretical arguments of the non-perfect fuzzy rainbow method will be easier to understand when s is assumed to be much smaller than m or t. It will be later verified that the s values of interest will mostly be in the range $15 \sim 100$.

A non-perfect fuzzy rainbow matrix can be regarded as a concatenation of s DP sub-matrices, since the ending points of one DP sub-matrix is used as the starting points for the next DP sub-matrix. When we treat the non-perfect fuzzy rainbow tradeoff, the i-th ($1 \leq i \leq s$) DP sub-matrix will be denoted by $\mathtt{DM}_i$. The only difference between $\mathtt{DM}_i$ and a normal non-perfect DP matrix is that $\mathtt{DM}_i$ may contain duplicate starting points, that bring about fully identical chains. The expected number of distinct starting points and ending points for $\mathtt{DM}_i$ are written as m_{i-1} and m_i, respectively.

To describe the online phase of the non-perfect fuzzy rainbow tradeoff algorithm, the order of online chain creation needs to be clarified. In short, all the tables are processed in parallel, as the online phase of the thick rainbow tradeoff.

In the first pass, for each of the ℓ pre-computation tables, the online chain that starts from the s-th color for the table is generated. That is, all of the s-th DP sub-matrices $\mathtt{DM}_s$ of the ℓ pre-computation matrices are searched first.

Then all the alarms generated by these ℓ online chains are fully resolved. All computation associated with this first pass is fully executed even if the correct answer is found during this process. The second pass is executed only if the first pass did not find the correct answer. In the second pass, the online chains that start from the $(s-1)$-th colors and extend into the s-th colors are generated for all the pre-computation tables, to search on the $(s-1)$-th DP sub-matrix $\mathtt{DM}_{s-1}$ of each pre-computation matrix. The subsequent passes are similarly continued until either the correct answer is found or all the s passes are complete. Note that most of the alarms will come to be false alarms, and in these cases, the regeneration of pre-computation chains may be stopped at the DP of the color from which the online chain was started.

In practice, as the case of the thick rainbow tradeoff, the ℓ pre-computation tables of the non-perfect fuzzy rainbow tradeoff may or may not be processed in parallel. It suffices to generate all the online chains and to treat all the associated alarms in serial order before moving on to the next pass. Our theoretical arguments will be developed disadvantageously, but these will give good approximations of the real situation, unless s is too small.

2.2.3 Perfect Table Fuzzy Rainbow Tradeoff

Since we have already looked at the feature of fuzzy rainbow chains, let us just review the description of the pre-computation and online phases for the perfect fuzzy rainbow algorithm.

The parameter set of m, t, and s for the perfect table fuzzy rainbow tradeoff are chosen with setting that the matrix stopping constant $\bar{\mathtt{F}}_{\mathrm{msc}} = \frac{mt^2s}{\mathtt{N}}$ is neither too large nor very close to zero. As for the non-perfect version, s is assumed to be much smaller than m or t. We will show later that the values of s are typically in the range $30 \sim 150$.

The number of ending points for each perfect fuzzy rainbow table is set to m. That is, for each pre-computation table, sufficiently many pre-computation chains are generated, so that the number of non-merging pre-computation chains can be m. The m starting and ending point pairs are recorded in a pre-computation table after being sorted according to the end-

ing points, and ℓ pre-computation tables are created.

Let us clarify precisely the process of pre-computation phase. The 1-st step of the pre-computation phase begins by the generation of each non-perfect DP sub-matrix $\mathtt{DM}_1$ from m_0 starting points. The chains of $\mathtt{DM}_1$ are sorted according to the ending points, and the groups of merging chains are taken out. From each group of merging chains, the chain with the longest chain segment is retained and other chains are removed. In each i-th step, after the creation of each non-perfect DP sub-matrix $\mathtt{DM}_i$, the chains of $\mathtt{DM}_i$ are sorted according to the ending points, the groups of merging chains are taken out. The chains in each group of merging chains are discarded except the chain with the longest i-th color DP chain segment. The retained (temporary) perfect DP sub-matrix in the i-th step is denoted by $\widetilde{\mathtt{DM}}_i$. Note that the set of ending points from $\widetilde{\mathtt{DM}}_i$ is identical to the set of ending point from $\mathtt{DM}_i$, and these are used as the starting points for the next non-perfect DP sub-matrix $\mathtt{DM}_{i+1}$. For an appropriate choice of m_0, which will be discussed later, the final perfect DP sub-matrix $\widetilde{\mathtt{DM}}_s$ is expected to contain $m = m_s$ non-merging chains. We denote the collection of all DP chains in $\widetilde{\mathtt{DM}}_i$ that eventually reach one of the m DP chains that remain in $\widetilde{\mathtt{DM}}_s$ as $\overline{\mathtt{DM}}_i$. In particular, we have $\overline{\mathtt{DM}}_s = \widetilde{\mathtt{DM}}_s$, and only the points of the perfect DP sub-matrices $\overline{\mathtt{DM}}_i$, which are the components of a perfect fuzzy rainbow matrix, can contribute to the success of inversions.

The above chain removal rule does not rely on the total chain lengths but depends only on the i-th DP chain segment lengths. We chose to work with such a manner, because it allowed the existing results concerning the perfect DP tradeoff [22] to be used for analyzing the perfect fuzzy rainbow tradeoff. However, since some readers may argue against this manner and insist that it is more reasonable to remove chain merges based on the total chain lengths, let us make two comments concerning this matter.

First, we claim that the choice of chain removal rule is not very important for the perfect fuzzy rainbow tradeoff. Note that, for the perfect DP tradeoff, since the chain lengths of a DP matrix form a geometric distribution, the chain removal rule was an important issue. However, the chain lengths of a fuzzy rainbow matrix are expected to form a distribution that very quickly

approaches the normal distribution as s is increased, since the concatenation of multiple DP chains will make an averaging effect. Indeed, it is not difficult to work the distribution out and confirm our claim explicitly. Hence, the variance in the lengths of fuzzy rainbow chains is small, and the impact of choices based on chain lengths on the performance of the perfect fuzzy rainbow tradeoff can only be limited. Furthermore, the averaging effect implies that, except at small i values, the chain removal rule that relies on the i-th DP chain segment lengths is likely to return long chain in overall length.

Second, we have a question whether it is reasonable to retain the longer chains. The choice based on the total chain lengths is widely accepted with the perfect DP tradeoff, because it is expected to achieve higher success rate for the same amount of storage use. However, the method increases the cost of resolving false alarms. Although we strongly believe that the positive effect of choosing longer chains on the success rate is likely to outweigh its negative effects on the cost of resolving false alarms, there is no theoretical proof or experimental support to verify such a claim as yet. A further study concerning this issue would be needed.

The online phase of the perfect fuzzy rainbow tradeoff is essentially the same as that of the non-perfect version, but at most one alarm occurs per table. The pre-computation tables are also processed in parallel. To simulate the parallel treatment in practice, it suffices to generate the online chains and to resolve all the alarms in serial order during each pass, and this modification will not have a noticeable effect on the online execution complexity, if the number of colors s is not too small.

Chapter 3

Analyses of the Three Rainbow Variants

Complexity analyses of the thick rainbow, non-perfect table fuzzy rainbow, and perfect table fuzzy rainbow tradeoff algorithms are given in this chapter. This is the main contribution and the most complicated part of this thesis. Experiments of some technical arguments appeared in this chapter are provided in Chapter 7.

3.1 Thick Rainbow Tradeoff

The online complexity for the thick rainbow tradeoff is analyzed in this section. We also provide the time memory tradeoff curve which takes the effects of false alarms into account. We assume that the parameters m, t, and s, for the thick rainbow tradeoff algorithm, are chosen to satisfy $mts = \mathtt{K}_{\mathrm{msc}}\mathtt{N}$, with a matrix stopping constant $\mathtt{K}_{\mathrm{msc}}$ that is neither too large nor very close to zero.

3.1.1 Probability of Success

The number of one-way function invocations to create all the thick rainbow pre-computation tables is $mts\ell$. Let us define the *pre-computation coefficient*

of the thick rainbow tradeoff to be

$$\mathtt{K}_{\mathrm{pc}} = \frac{mts\ell}{\mathsf{N}}.$$

Since the efforts of sorting the ending points in the tables, which is of $m\ell \log m$ order and is much smaller than the efforts of constructing the tables, can be ignored, we may state $\mathtt{K}_{\mathrm{pc}}\mathsf{N} = mts\ell$ as the pre-computation cost.

We next define the *coverage rate* of a single thick rainbow matrix to be

$$\mathtt{K}_{\mathrm{cr}} = \frac{1}{mts}\big(|\mathtt{HM}_1| + |\mathtt{HM}_2| + \cdots + |\mathtt{HM}_s|\big),$$

where $|\mathtt{HM}_i|$ denotes the expected number of distinct points in the i-th Hellman sub-matrix. This definition does not refer to the total number of distinct points in a whole thick rainbow pre-computation matrix. In spite of this fact, the following proposition presents that the above definition is natural.

Proposition 3.1.1. *The success probability of the thick rainbow tradeoff is*

$$\mathtt{K}_{\mathrm{ps}} = 1 - e^{-\mathtt{K}_{\mathrm{pc}}\mathtt{K}_{\mathrm{cr}}}.$$

Proof. The probability that the thick rainbow tradeoff fails is

$$\begin{aligned} 1 - \mathtt{K}_{\mathrm{ps}} =& \Big(1 - \frac{|\mathtt{HM}_s|}{\mathsf{N}}\Big)^{\ell}\Big(1 - \frac{|\mathtt{HM}_{s-1}|}{\mathsf{N}}\Big)^{\ell} \cdots\cdots \Big(1 - \frac{|\mathtt{HM}_1|}{\mathsf{N}}\Big)^{\ell} \\ \approx& \exp\Big(-\frac{|\mathtt{HM}_s|}{\mathsf{N}}\ell\Big)\exp\Big(-\frac{|\mathtt{HM}_{s-1}|}{\mathsf{N}}\ell\Big)\cdots\cdots\exp\Big(-\frac{|\mathtt{HM}_1|}{\mathsf{N}}\ell\Big). \end{aligned}$$

Thus, $\mathtt{K}_{\mathrm{ps}}$ can be approximately

$$1 - \exp\Big(-\sum_{i=1}^{s}|\mathtt{HM}_i|\frac{\ell}{\mathsf{N}}\Big) = 1 - \exp\Big(-\mathtt{K}_{\mathrm{cr}}\frac{mts\ell}{\mathsf{N}}\Big) = 1 - e^{-\mathtt{K}_{\mathrm{pc}}\mathtt{K}_{\mathrm{cr}}}.$$

□

The ending points in $\mathtt{HM}_i$ become the starting points in $\mathtt{HM}_{i+1}$. Let us use m_{i-1} and m_i to denote the expected numbers of distinct starting points and ending points, respectively, in each Hellman sub-matrix $\mathtt{HM}_i$. In particular,

$m_0 = m$ and m_s are the numbers of distinct starting and ending points, respectively, of a full thick rainbow matrix. We refer to each group of m_i points as the i-th *color boundary points* of a thick rainbow matrix for $0 \leq i \leq s$. We will make good use of the following closed-form formula for m_i.

Lemma 3.1.2. *The number of the i-th color boundary points in a single thick rainbow matrix is*

$$m_i = \frac{2m}{2 + \mathtt{K}_{\mathrm{msc}} \frac{i}{s}},$$

for each $i = 0, 1, \ldots, s$.

Proof. With m starting points, it is known [1, 15] that the number of distinct points x_j expected, that appears at j columns away from the column of starting points, satisfies

$$\frac{x_j}{\mathtt{N}} \approx \frac{1}{\frac{\mathtt{N}}{m} + \frac{j}{2}}.$$

Hence, the number of the i-th color boundary points in a single thick rainbow matrix is

$$\frac{m_i}{\mathtt{N}} \approx \frac{1}{\frac{\mathtt{N}}{m} + \frac{it}{2}},$$

and the claim is obtained from a simple calculation. □

Let us use the notation

$$\mathtt{K}_{\mathrm{cr},i} = \frac{|\mathtt{HM}_i|}{mt},$$

which means the coverage rate of the i-th Hellman sub-matrix of a thick rainbow matrix. Note that this definition allows us to expect $\mathtt{K}_{\mathrm{cr},i}$ to be of $\Theta(1)$ order.

Lemma 3.1.3. *The coverage rate of the Hellman sub-matrix $\mathtt{HM}_i$ is*

$$\mathtt{K}_{\mathrm{cr},i} = \frac{m_{i-1}}{m} \sqrt{\frac{2}{\mathtt{K}_{\mathrm{msc}}} \frac{m}{m_{i-1}} \frac{s}{t}} \frac{e^{\sqrt{2\mathtt{K}_{\mathrm{msc}} \frac{m_{i-1}}{m} \frac{t}{s}}} - 1}{e^{\sqrt{2\mathtt{K}_{\mathrm{msc}} \frac{m_{i-1}}{m} \frac{t}{s}}} + 1}.$$

Proof. Recall from Proposition 21 in [17] (or Section 2.1.1 in this thesis) that the coverage rate of a Hellman matrix $\mathtt{HM}$, with x starting points and chain length of y, is computed as

$$\frac{|\mathtt{HM}|}{xy} = \sqrt{\frac{2\mathsf{N}}{xy^2}} \frac{e^{\sqrt{2\frac{xy^2}{\mathsf{N}}}} - 1}{e^{\sqrt{2\frac{xy^2}{\mathsf{N}}}} + 1}.$$

Thus, the coverage rate of $\mathtt{HM}_i$ is easy to obtain since

$$\mathtt{K}_{\mathrm{cr},i} = \frac{m_{i-1}}{m} \frac{|\mathtt{HM}_i|}{m_{i-1}t}.$$

□

The coverage rate of a thick rainbow matrix is also of $\Theta(1)$ order since

$$\mathtt{K}_{\mathrm{cr}} = \frac{1}{mts} \sum_{i=1}^{s} |\mathtt{HM}_i| = \frac{1}{s} \sum_{i=1}^{s} \mathtt{K}_{\mathrm{cr},i},$$

and hence, unless the requirement of the success probability $\mathtt{K}_{\mathrm{ps}}$ is unrealistically close to 1, the number of tables

$$\ell = \frac{\mathtt{K}_{\mathrm{pc}}}{\mathtt{K}_{\mathrm{msc}}} = -\frac{\ln(1 - \mathtt{K}_{\mathrm{ps}})}{\mathtt{K}_{\mathrm{msc}}\mathtt{K}_{\mathrm{cr}}}$$

for the thick rainbow tradeoff is of $\Theta(1)$ order too.

Proposition 3.1.4. *The coverage rate of a thick rainbow matrix is*

$$\mathtt{K}_{\mathrm{cr}} = \frac{1}{s} \sum_{i=1}^{s} \mathtt{K}_{\mathrm{cr},i}.$$

3.1.2 Online Complexity

Through a carefully computation of the online execution complexity, we obtain the time memory tradeoff curve for the thick rainbow tradeoff in this subsection.

We first write the the probability for each pass of the online phase to be executed.

Lemma 3.1.5. *The probability for the ℓ Hellman sub-matrices $\mathtt{HM}_i$ within the ℓ thick rainbow matrices to be searched for the correct answer to the inversion problem is*

$$\big(1-\mathtt{K}_{\mathrm{ps}}\big)^{\frac{\sum_{k=i+1}^{s}\mathtt{K}_{\mathrm{cr},k}}{s\mathtt{K}_{\mathrm{cr}}}}.$$

Proof. The ℓ Hellman sub-matrices $\mathtt{HM}_i$ of the ℓ thick rainbow matrices will be searched if and only if the correct answer to the inversion target does not belong to the Hellman sub-matrices $\mathtt{HM}_{i+1}, \ldots, \mathtt{HM}_s$ contained in the ℓ thick rainbow matrices. Hence, the probability under consideration is

$$\begin{aligned}
&\Big(1-\frac{|\mathtt{HM}_{i+1}|}{\mathsf{N}}\Big)^{\ell}\Big(1-\frac{|\mathtt{HM}_{i+2}|}{\mathsf{N}}\Big)^{\ell}\cdots\cdots\Big(1-\frac{|\mathtt{HM}_{s}|}{\mathsf{N}}\Big)^{\ell}\\
\approx&\exp\Big(-\mathtt{K}_{\mathrm{msc}}\frac{\ell}{s}\frac{|\mathtt{HM}_{i+1}|}{mt}\Big)\exp\Big(-\mathtt{K}_{\mathrm{msc}}\frac{\ell}{s}\frac{|\mathtt{HM}_{i+2}|}{mt}\Big)\cdots\cdots\exp\Big(-\mathtt{K}_{\mathrm{msc}}\frac{\ell}{s}\frac{|\mathtt{HM}_{s}|}{mt}\Big)\\
=&\exp\Big(-\frac{\mathtt{K}_{\mathrm{pc}}}{s}\sum_{k=i+1}^{s}\mathtt{K}_{\mathrm{cr},k}\Big)=\big(1-\mathtt{K}_{\mathrm{ps}}\big)^{\frac{\sum_{k=i+1}^{s}\mathtt{K}_{\mathrm{cr},k}}{s\mathtt{K}_{\mathrm{cr}}}},
\end{aligned}$$

where the final equality follows from Proposition 3.1.1. □

We next compute the cost of handling alarms that appear during the online phase, with assuming the search on a Hellman sub-matrix $\mathtt{HM}_i$ of a single thick rainbow matrix.

Lemma 3.1.6. *Assume the search on a Hellman sub-matrix $\mathtt{HM}_i$ of a single thick rainbow matrix. The cost of resolving alarms that may be induced by possible merges of the t online chains into this pre-computation matrix is expected to be*

$$\Big((s-i)(i-1)+\frac{s-1}{2}+\frac{1}{6}\Big)\frac{\mathtt{K}_{\mathrm{msc}}}{s}t^2.$$

Proof. Let us assume the generation of the online chain that starts at j columns away from the column of ending points in $\mathtt{HM}_i$. It can be inferred from Proposition 4 in [15] that the expected number of alarm counts for this online chain with the ending points in the thick rainbow pre-computation

matrix is expect to be

$$\frac{m\big((s-i)t+(j+1)\big)}{\mathsf{N}}. \tag{3.1.1}$$

To resolve each alarm from this merge, one must compute $t(i-1)+(t-j)$ iterations of the one-way function. Thus, the cost we want to obtain is

$$\begin{aligned}&\sum_{j=0}^{t-1}\big(t(i-1)+(t-j)\big)\frac{\mathsf{K}_{\mathrm{msc}}\big((s-i)t+(j+1)\big)}{ts}\\ &=\frac{\mathsf{K}_{\mathrm{msc}}}{s}\Big(t^2(s-i)(i-1)+(s-1)\frac{t(t+1)}{2}+\frac{(t+1)(t+2)}{6}\Big),\end{aligned}$$

after calculating the summation. The claimed cost comes out from the above equation by taking the highest term in t. □

Finally, we provide the tradeoff coefficient, which concisely represents the online efficiency of a tradeoff algorithm, with the time memory tradeoff curve for the thick rainbow tradeoff algorithm.

Theorem 3.1.7. *The time memory tradeoff curve for the thick rainbow tradeoff is $TM^2=\mathsf{K}_{\mathrm{tc}}\mathsf{N}^2$, where the tradeoff coefficient is*

$$\begin{aligned}\mathsf{K}_{\mathrm{tc}}=-\frac{\{\ln(1-\mathsf{K}_{\mathrm{ps}})\}^3}{\mathsf{K}_{\mathrm{msc}}\mathsf{K}_{\mathrm{cr}}^3}\frac{1}{s}\sum_{i=1}^{s}\Big(1-\mathsf{K}_{\mathrm{ps}}\Big)^{\frac{\sum_{k=i+1}^{s}\mathsf{K}_{\mathrm{cr},k}}{s\mathsf{K}_{\mathrm{cr}}}}\\ \times\left\{\Big(1-\frac{i}{s}\Big)+\Big((s-i)(i-1)+\frac{s-1}{2}+\frac{1}{6}\Big)\frac{\mathsf{K}_{\mathrm{msc}}}{s^2}\right\}.\end{aligned}$$

Proof. The cost to generate all the t online chains for the i-th Hellman submatrix of a single thick rainbow pre-computation matrix is $(t-1)+(s-i)t\times t\approx(s-i)t^2$ iterations of the one-way function, and the cost of resolving alarms that may be induced by possible merges of these online chains into the pre-computation matrix is already obtained in Lemma 3.1.6.

Noting that it is necessary to multiply these two terms by ℓ to take all the parallel online chains on ℓ thick rainbow pre-computation matrices into

account, the time complexity is

$$T = t^2\ell \sum_{i=1}^{s} \left(1 - \mathtt{K}_{\mathrm{ps}}\right)^{\frac{\sum_{k=i+1}^{s} \mathtt{K}_{\mathrm{cr},k}}{s\mathtt{K}_{\mathrm{cr}}}} \times \left\{(s-i) + \left((s-i)(i-1) + \frac{s-1}{2} + \frac{1}{6}\right)\frac{\mathtt{K}_{\mathrm{msc}}}{s}\right\}.$$

The storage complexity is $M = m\ell$, and the form of the time memory tradeoff curve stated above can be reached by a simple calculation. □

The time complexity T, appearing in the above proof, is bounded from below by

$$T \geq t^2\ell \sum_{i=1}^{s} (1 - \mathtt{K}_{\mathrm{ps}})(s-i) = t^2 \frac{s(s-1)}{2} \ell (1 - \mathtt{K}_{\mathrm{ps}}),$$

and bound from above by

$$T \leq t^2\ell \sum_{i=1}^{s} \left\{\big(s-i+1\big) + \big((s-i)s + s\big)\frac{\mathtt{K}_{\mathrm{msc}}}{s}\right\} = t^2 \frac{s(s+1)}{2} \ell (\mathtt{K}_{\mathrm{msc}} + 1).$$

Since $\ell = \Theta(1)$ from the end of Section 3.1.1, the online time complexity T of the thick rainbow tradeoff must be of $\Theta(t^2 s^2)$ order.

As the tag end of this subsection, we provide the expected number of table lookups required during the online phase. The table lookup time strongly depends on the implementation condition. Note that table lookups could be ignored when the pre-computation tables are in fast memory, since the cost of table lookups could be insignificant compared with the cost of one-way function computations. We will take this approach and not refer to the following proposition in the rest of this thesis. However, in practice, table lookups can become an obstructive factor to the algorithm performance.

Proposition 3.1.8. *The online processing of the thick rainbow tradeoff is expected to require*

$$t\ell \sum_{i=1}^{s} \left(1 - \mathtt{K}_{\mathrm{ps}}\right)^{\frac{\sum_{k=i+1}^{s} \mathtt{K}_{\mathrm{cr},k}}{s\mathtt{K}_{\mathrm{cr}}}}$$

lookups to the thick rainbow tables.

Proof. The probability for the ℓ Hellman sub-matrices $\mathtt{HM}_i$ of the ℓ pre-computation thick rainbow matrices to be searched is given by Lemma 3.1.5, and such case requires $t \times \ell$ table lookups since each generation of t families of ℓ parallel online chains call for a single table lookup per pre-computation table. Thus, the expected number of total table lookups is obtained as stated. □

Since the table lookup count stated by the above proposition is upper bounded by $st\ell$ and lower bounded by $st\ell(1 - \mathtt{K}_{\mathrm{ps}})$, we can easily verify that the number of table lookups made by the online phase of the thick rainbow tradeoff is of $\Theta(ts)$ order, which is much smaller than the online complexity $T = \Theta(t^2 s^2)$.

3.2 Non-Perfect Table Fuzzy Rainbow Tradeoff

In this section, we analyze the online complexity for the non-perfect table fuzzy rainbow tradeoff. This content can be also found in [18, 19]. The parameters m, t, and s, for the non-perfect fuzzy rainbow tradeoff, are chosen to satisfy $mt^2 s = \mathtt{F}_{\mathrm{msc}}\mathtt{N}$, with a matrix stopping constant $\mathtt{F}_{\mathrm{msc}}$ that is neither too large nor very close to zero.

3.2.1 Probability of Success

Consider each DP sub-matrix $\mathtt{DM}_i$ in a non-perfect fuzzy rainbow matrix. Let us use m_{i-1} and m_i to denote the expected numbers of distinct starting and ending points, respectively, in $\mathtt{DM}_i$. Note that, especially, $m_0 = m$ and m_s are the numbers of distinct starting and ending points, respectively, of the whole non-perfect fuzzy rainbow matrix. We refer to each group of m_i points as the i-th *color boundary points* of a non-perfect fuzzy rainbow matrix for

$0 \leq i \leq s$. Some previous results for the non-perfect DP tradeoff allow us to find a recurrence form for m_i.

Lemma 3.2.1. *Assume that the number of colors s is sufficiently large. Then the recurrence formula for the number of the i-th color boundary points has approximately the form $m_{i+1} = m_i\big(1 - \frac{\mathtt{F}_{\mathrm{msc}}}{2s}\frac{m_i}{m}\big)$ for $i = 0, 1, ..., s-1$.*

Proof. It is known from Lemma 2 in [22] (or Section 2.1.2 in this thesis) that the number of distinct ending points in a DP matrix with m starting points is

$$\frac{2m}{1+\sqrt{1+2\mathtt{D}_{\mathrm{msc}}}}.$$

We can replace m as m_i and $\mathtt{D}_{\mathrm{msc}}$ as $\frac{m_i t^2}{\mathsf{N}}$ for each $\mathtt{DM}_i$ in a non-perfext fuzzy rainbow matrix. Thus,

$$m_{i+1} = m_i\frac{2}{1+\sqrt{1+2\frac{m_i t^2}{\mathsf{N}}}} = m_i\Big\{1 - \frac{1}{2}\frac{m_i t^2}{\mathsf{N}} + O\Big(\big(\frac{m_i t^2}{\mathsf{N}}\big)^2\Big)\Big\}.$$

Replacing $\frac{t^2}{\mathsf{N}}$ as $\frac{\mathtt{F}_{\mathrm{msc}}}{ms}$, m_{i+1} can be approximately

$$m_i\Big(1 - \frac{\mathtt{F}_{\mathrm{msc}}}{2s}\frac{m_i}{m}\Big).$$

□

We use the Euler method for difference equations to obtain a closed-form formula for m_i which is easy to utilize.

Lemma 3.2.2. *After applying the Euler method for difference equations, the number of the i-th color boundary points in a single non-perfect fuzzy rainbow matrix is*

$$m_i = \frac{2m}{2 + \mathtt{F}_{\mathrm{msc}}\frac{i}{s}},$$

for each $i = 0, 1, \ldots, s$ when the number of colors s is sufficiently large.

Proof. Let us rewrite the recursive formula for m_i of Lemma 3.2.1 as a difference equation

$$\frac{m_{i+1}}{m} - \frac{m_i}{m} = -\frac{\mathtt{F}_{\mathrm{msc}}}{2s}\left(\frac{m_i}{m}\right)^2.$$

Applying the Euler method, then we can write the above equation as a differential equation

$$y' = -\frac{\mathtt{F}_{\mathrm{msc}}}{2s}y^2,$$

with the initial condition $y(0) = \frac{m_0}{m} = 1$. The unique solution for this equation is

$$y(x) = \frac{2}{2 + \frac{\mathtt{F}_{\mathrm{msc}}}{s}x},$$

and hence, we obtain

$$\frac{m_i}{m} = \frac{2}{2 + \frac{\mathtt{F}_{\mathrm{msc}}}{s}i},$$

as we claimed. □

The *coverage rate* of a non-perfect fuzzy rainbow matrix is defined as

$$\mathtt{F}_{\mathrm{cr}} = \frac{1}{mts}\big(|\mathtt{DM}_1| + |\mathtt{DM}_2| + \cdots + |\mathtt{DM}_s|\big),$$

where $|\mathtt{DM}_i|$ denotes the expected number of distinct points in the i-th DP sub-matrix. Note that this definition does not signify the total number of distinct points expected in a whole non-perfect fuzzy rainbow matrix.

Using the following lemma, We can compute the partial coverage of DP sub-matrices from $(i+1)$-th color to s-th color in a single non-perfect fuzzy rainbow matrix.

Lemma 3.2.3. *When the number of colors s is sufficiently large, the partial*

sum $\frac{m_{i+1}+m_{i+2}+\cdots+m_s}{ms}$ *can be approximately*

$$\frac{2}{\mathtt{F}_{\mathrm{msc}}} \ln\Big(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}}\Big)$$

for $i = 0, 1, \ldots, s-1$.

Proof. The sum

$$\sum_{k=i+1}^{s} \frac{m_k}{m}\frac{1}{s} = \sum_{k=i+1}^{s} \frac{2}{2+\mathtt{F}_{\mathrm{msc}}\frac{k}{s}}\frac{1}{s}$$

can be approximately

$$\int_{\frac{i}{s}}^{1} \frac{2}{2+\mathtt{F}_{\mathrm{msc}}x}dx = \frac{2}{\mathtt{F}_{\mathrm{msc}}}\Big(\ln(2+\mathtt{F}_{\mathrm{msc}}x)\Big|_{\frac{i}{s}}^{1}\Big) = \frac{2}{\mathtt{F}_{\mathrm{msc}}} \ln\Big(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}}\Big)$$

since we assume that s is sufficiently large. □

The partial coverage of DP sub-matrices from $(i+1)$-th color to s-th color in a single non-perfect fuzzy rainbow matrix is

$$\frac{|\mathtt{DM}_{i+1}| + \cdots + |\mathtt{DM}_s|}{mt(s-i)} = \frac{s}{s-i}\frac{m_{i+1}+\cdots+m_s}{ms} = \frac{s}{s-i}\frac{2}{\mathtt{F}_{\mathrm{msc}}} \ln\Big(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}}\Big). \tag{3.2.1}$$

The coverage for a non-perfect fuzzy rainbow matrix comes out from the special case of Eq. (3.2.1) (or Lemma 3.2.3) for $i = 0$.

Proposition 3.2.4. *When the number of colors* s *is sufficiently large, the coverage rate for a single non-perfect fuzzy rainbow matrix is*

$$\mathtt{F}_{\mathrm{cr}} = \frac{2}{\mathtt{F}_{\mathrm{msc}}} \ln\Big(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2}\Big).$$

The arguments that we give so far in this subsection is valid only if the number of colors s is sufficiently large. Note that we use two techniques of approximation, the Euler method for a differential equation and the change

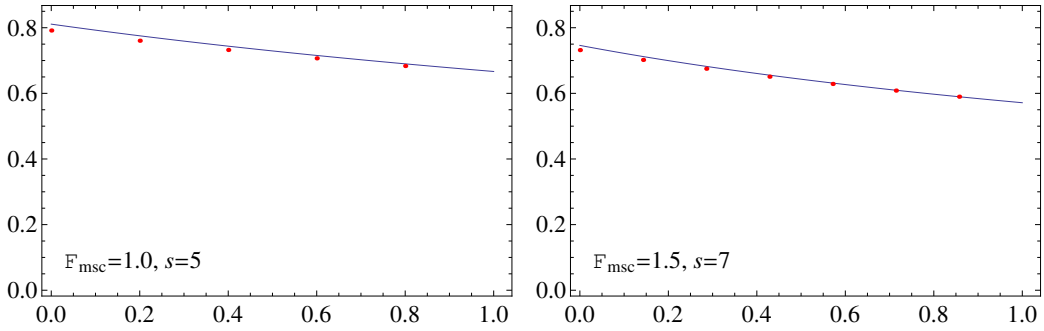

Figure 3.1: Comparisons between closed-form formula (*line*) and iteratively computed values (*dots*) of the partial coverage rate at small s. (x-axis:$\frac{i}{s}$; y-axis: $\frac{s}{s-i} \times \frac{1}{s}\sum_{k=i+1}^{s} \frac{m_k}{m}$)

from a summation to a definite integral. However, those can be still valid even if s is small.

Recall the iteration formula for the number of the i-th color boundary points

$$\frac{m_{i+1}}{m} = \frac{m_i}{m} \frac{2}{1+\sqrt{1+2\frac{\mathrm{F_{msc}}}{s}\frac{m_i}{m}}} \quad \text{with} \quad \frac{m_0}{m} = 1. \tag{3.2.2}$$

We have compared

$$\frac{s}{s-i} \times \frac{1}{s} \sum_{k=i+1}^{s} \frac{m_k}{m}, \tag{3.2.3}$$

where $\frac{m_k}{m}$ were iteratively computed through Eq. (3.2.2), and the last formula of Eq. (3.2.1), at some small s values in Figure 3.1.

Figure 3.1 shows that Lemma 3.2.2, Eq. (3.2.1), and Proposition 3.2.4 are accurate even at those very small s values. Since it will become clear in Section 5.2 that the s values of interest will be somewhat larger than those small values, our arguments will be treated as being valid for all s values of interest.

Before giving an expression for the probability of success of the non-perfect fuzzy rainbow tradeoff, we define the *pre-computation coefficient* of the algorithm. The pre-computation cost to construct all the pre-computation tables is expected to be $mts\ell$, so that the pre-computation coefficient is

defined as

$$\mathtt{F}_{\mathrm{pc}} = \frac{mts\ell}{\mathsf{N}},$$

when the effort of sorting the ending points, which is of $m\ell \log m$ order, is ignored.

Note that the non-perfect table fuzzy rainbow tradeoff is unsuccessful if and only if the correct answer of the target does not exist in every DP sub-matrix on each of ℓ fuzzy rainbow matrices. Since the probability that the correct answer is in the DP sub-matrix $\mathtt{DM}_i$ is $\frac{|\mathtt{DM}_i|}{\mathsf{N}}$ on a single non perfect fuzzy rainbow matrix, we can obtain the success probability of the non-perfect fuzzy rainbow tradeoff through this fact.

Proposition 3.2.5. *The success probability of the non-perfect table fuzzy rainbow tradeoff is*

$$\mathtt{F}_{\mathrm{ps}} = 1 - e^{-\mathtt{F}_{\mathrm{pc}}\mathtt{F}_{\mathrm{cr}}}.$$

Proof. The probability that the non-perfect table fuzzy rainbow tradeoff fails is

$$\begin{aligned} 1 - \mathtt{F}_{\mathrm{ps}} =& \Big(1 - \frac{|\mathtt{DM}_s|}{\mathsf{N}}\Big)^{\ell} \Big(1 - \frac{|\mathtt{DM}_{s-1}|}{\mathsf{N}}\Big)^{\ell} \cdots\cdots \Big(1 - \frac{|\mathtt{DM}_1|}{\mathsf{N}}\Big)^{\ell} \\ =& \Big(1 - \mathtt{F}_{\mathrm{msc}} \frac{m_s}{mts}\Big)^{\ell} \Big(1 - \mathtt{F}_{\mathrm{msc}} \frac{m_{s-1}}{mts}\Big)^{\ell} \cdots\cdots \Big(1 - \mathtt{F}_{\mathrm{msc}} \frac{m_1}{mts}\Big)^{\ell}. \end{aligned}$$

Thus, $\mathtt{F}_{\mathrm{ps}}$ can be approximately

$$1 - \prod_{i=1}^{s} \exp\Big(-\mathtt{F}_{\mathrm{msc}} \frac{\ell}{t} \frac{m_i}{ms}\Big) = 1 - \exp\Big(-\mathtt{F}_{\mathrm{msc}} \frac{\ell}{t} \sum_{i=1}^{s} \frac{m_i}{ms}\Big) = 1 - e^{-\mathtt{F}_{\mathrm{pc}}\mathtt{F}_{\mathrm{cr}}}.$$

□

Combining Proposition 3.2.4 and Proposition 3.2.5, we can express $\mathtt{F}_{\mathrm{pc}}$ in terms of $\mathtt{F}_{\mathrm{ps}}$ and $\mathtt{F}_{\mathrm{msc}}$ through a simple calculation.

Corollary 3.2.6. $\mathtt{F}_{\mathrm{pc}} = -\dfrac{\mathtt{F}_{\mathrm{msc}} \ln(1 - \mathtt{F}_{\mathrm{ps}})}{2 \ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2})}$.

Proposition 3.2.4 shows the coverage rate $\mathtt{F}_{\mathrm{cr}}$ as a function of the single variable $\mathtt{F}_{\mathrm{msc}}$, and Proposition 3.2.5 and Corollary 3.2.6 show that any set of

parameters m, t, s and ℓ that attains the success rate $\mathtt{F}_{\mathrm{ps}}$ must satisfy the relation

$$\frac{\ell}{t} = \frac{\mathtt{F}_{\mathrm{pc}}}{\mathtt{F}_{\mathrm{msc}}} = \frac{\{-\ln(1-\mathtt{F}_{\mathrm{ps}})\}}{\mathtt{F}_{\mathrm{msc}}\mathtt{F}_{\mathrm{cr}}}. \tag{3.2.4}$$

Thus, when any set of parameters is restricted to that attains a fixed requirement $\mathtt{F}_{\mathrm{ps}}$ on the success rate, the ratio $\frac{\ell}{t}$ is also regarded as a function of the single variable $\mathtt{F}_{\mathrm{msc}}$.

Note that, since we are working with parameters for which $\mathtt{F}_{\mathrm{msc}}$ is of $\Theta(1)$ order, Proposition 3.2.4 implies that

$$\mathtt{F}_{\mathrm{cr}} = 1 - \frac{1}{2}\Big(\frac{\mathtt{F}_{\mathrm{msc}}}{2}\Big) + \frac{1}{3}\Big(\frac{\mathtt{F}_{\mathrm{msc}}}{2}\Big)^2 - \frac{1}{4}\Big(\frac{\mathtt{F}_{\mathrm{msc}}}{2}\Big)^3 + \cdots$$

is also of $\Theta(1)$ order. Hence, the relation (3.2.4) implies that ℓ and t are of similar order, unless the success rate requirement $\mathtt{F}_{\mathrm{ps}}$ is unrealistically close to 1.

3.2.2 Online Complexity

The online execution complexity for the non-perfect fuzzy rainbow tradeoff is obtained in this subsection through a carefully computation of the average case complexity rather than the worst case complexity.

We start by evaluating the probability for each step of the online phase to be executed.

Lemma 3.2.7. *The probability for the ℓ i-th DP sub-matrices $\mathtt{DM}_i$ within the ℓ non-perfect fuzzy rainbow matrices to be searched for the correct answer to the inversion problem is*

$$\Big(1-\mathtt{F}_{\mathrm{ps}}\Big)^{\frac{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}})}{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2})}}.$$

Proof. The ℓ i-th DP sub-matrices $\mathtt{DM}_i$ of the ℓ non-perfect fuzzy rainbow matrices will be searched if and only if there is no correct answer of the inversion target in $\mathtt{DM}_{i+1}$, $\ldots$, $\mathtt{DM}_s$ of the all ℓ non-perfect fuzzy rainbow

matrices. Thus, the probability that is stated above is

$$\begin{aligned}
&\Big(1-\frac{|\mathtt{DM}_{i+1}|}{\mathsf{N}}\Big)^{\ell}\Big(1-\frac{|\mathtt{DM}_{i+2}|}{\mathsf{N}}\Big)^{\ell}\cdots\cdots\Big(1-\frac{|\mathtt{DM}_{s}|}{\mathsf{N}}\Big)^{\ell}\\
\approx&\exp\Big(-\mathtt{F}_{\mathrm{msc}}\frac{\ell}{t}\frac{|\mathtt{DM}_{i+1}|}{mts}\Big)\exp\Big(-\mathtt{F}_{\mathrm{msc}}\frac{\ell}{t}\frac{|\mathtt{DM}_{i+2}|}{mts}\Big)\cdots\cdots\exp\Big(-\mathtt{F}_{\mathrm{msc}}\frac{\ell}{t}\frac{|\mathtt{DM}_{s}|}{mts}\Big)\\
=&\exp\Big(-\mathtt{F}_{\mathrm{pc}}\sum_{k=i+1}^{s}\frac{|\mathtt{DM}_{k}|}{mts}\Big).
\end{aligned}$$

Lemma 3.2.3, Proposition 3.2.4, Proposition 3.2.5 and the fact $|\mathtt{DM}_k| = m_k t$ give the result, as

$$\exp\Big(\frac{\ln(1-\mathtt{F}_{\mathrm{ps}})}{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2})}\ln\big(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}}\big)\Big)=\Big(1-\mathtt{F}_{\mathrm{ps}}\Big)^{\frac{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}})}{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2})}}.$$

□

Our next goal is to express the cost of resolving alarms that appear during the online phase. We assume the search on a DP sub-matrix $\mathtt{DM}_i$ of a single non-perfect fuzzy rainbow matrix, and we treat the case of possible merges at the i-th color and the case of possible merges strictly after the i-th color separately.

The alarms need to be handled separately for each pre-computation chain that merges with the online chain. The resolving of an alarm associated with one pre-computation chain requires the same amount of work regardless of whether or not the pre-computation and online chains merge with other pre-computation chains. Hence, the total cost of resolving alarms is m times of the cost associated with a single pre-computation chain. In the following two lemmas, we will focus on the possible merge between the online chain and a single randomly generated pre-computation chain.

The first lemma gives the cost of dealing with alarms from immediate merges at the i-th color.

Lemma 3.2.8. *Assume the search on a DP sub-matrix* $\mathtt{DM}_i$ *of a single non-perfect fuzzy rainbow matrix. The cost of resolving alarms that may be induced*

by possible merges of the online chain into the pre-computation matrix at the i-th color segment is expected to be $(i+1)\frac{\mathtt{F}_{\mathrm{msc}}}{s}t$.

Proof. The probability for a single fuzzy rainbow pre-computation chain to contain the i-th colored DP sub-chain of length k is $\left(1-\frac{1}{t}\right)^{k-1}\frac{1}{t}$. The probability for the online chain to merge with this DP sub-chain of length k of the pre-computation chain at the i-th color segment is $\sum_{j=1}^{\infty}\left(1-\frac{1}{t}-\frac{k}{\mathsf{N}}\right)^{j-1}\frac{k}{\mathsf{N}}$. The number of the one-way function iterations to resolve this alarm is expected to be $(i-1)t+k$ iterations. The total cost of resolving alarms is m times of the cost associated with single pre-computation chain, hence, the total cost is

$$\begin{aligned}
&m\sum_{k=1}^{\infty}\left\{\left(1-\frac{1}{t}\right)^{k-1}\frac{1}{t}\cdot\sum_{j=1}^{\infty}\left(1-\frac{1}{t}-\frac{k}{\mathsf{N}}\right)^{j-1}\frac{k}{\mathsf{N}}\cdot\big((i-1)t+k\big)\right\}\\
=\ &m\sum_{k=1}^{\infty}\left\{\left(1-\frac{1}{t}\right)^{k-1}\frac{1}{t}\cdot\frac{k}{\mathsf{N}}\frac{1}{\frac{1}{t}+\frac{k}{\mathsf{N}}}\cdot\big((i-1)t+k\big)\right\}.
\end{aligned}$$

This can be approximated by

$$\begin{aligned}
&m\sum_{k=1}^{\infty}\left\{\left(1-\frac{1}{t}\right)^{k-1}\frac{1}{t}\cdot\frac{k}{\mathsf{N}}t\cdot\big((i-1)t+k\big)\right\}\\
\approx\ &m\frac{t^3}{\mathsf{N}}\sum_{k=1}^{\infty}\left\{e^{-\frac{k-1}{t}}\cdot\frac{k}{t}\cdot\big((i-1)+\frac{k}{t}\big)\cdot\frac{1}{t}\right\}\\
\approx\ &\mathtt{F}_{\mathrm{msc}}\frac{t}{s}\int_0^{\infty}e^{-u}u\big((i-1)+u\big)du=(i+1)\frac{\mathtt{F}_{\mathrm{msc}}}{s}t.
\end{aligned}$$

□

The cost of dealing with alarms at strictly later colors, assuming the generation of an online chain that starts from the i-th color, is given next.

Lemma 3.2.9. *Assume the search on a DP sub-matrix $\mathtt{DM}_i$ of a single non-perfect fuzzy rainbow matrix. The cost of resolving alarms that may be induced by possible merges of the online chain into the pre-computation matrix strictly after the i-th color is expected to be $i(s-i)\frac{\mathtt{F}_{\mathrm{msc}}}{s}t$.*

Proof. Note that a single fuzzy rainbow pre-computation chain is a concatenation of DP sub-chains, and so the online chain is. Hence, it is essential to compute the probability that a random DP chain merges with another random DP chain.

The probability for a random DP chain to be of length k is $\left(1-\frac{1}{t}\right)^{k-1}\frac{1}{t}$. The probability for another random DP chain to merge with this DP chain of length k is $\sum_{j=1}^{\infty}\left(1-\frac{1}{t}-\frac{k}{\mathsf{N}}\right)^{j-1}\frac{k}{\mathsf{N}}$. Thus, the probability for a random DP chain to merge with another random DP chain can be approximately

$$\begin{aligned}
&\sum_{k=1}^{\infty}\left\{\left(1-\frac{1}{t}\right)^{k-1}\frac{1}{t}\cdot\sum_{j=1}^{\infty}\left(1-\frac{1}{t}-\frac{k}{\mathsf{N}}\right)^{j-1}\frac{k}{\mathsf{N}}\right\}\\
=&\sum_{k=1}^{\infty}\left(1-\frac{1}{t}\right)^{k-1}\frac{1}{t}\cdot\frac{k}{\mathsf{N}}\frac{1}{\frac{1}{t}+\frac{k}{\mathsf{N}}}\approx\sum_{k=1}^{\infty}\left(1-\frac{1}{t}\right)^{k-1}\frac{1}{t}\cdot\frac{k}{\mathsf{N}}t\\
\approx&\frac{t^2}{\mathsf{N}}\sum_{k=1}^{\infty}e^{-\frac{k-1}{t}}\cdot\frac{k}{t}\cdot\frac{1}{t}\approx\frac{\mathsf{F}_{\mathrm{msc}}}{ms}\int_0^{\infty}e^{-u}u\,du=\frac{\mathsf{F}_{\mathrm{msc}}}{ms}.
\end{aligned}$$

From the above result, it is inferred that the probability for the online chain to merge with a single fuzzy rainbow pre-computation chain at the h-th color with $h>i$ is $\left(1-\frac{\mathsf{F}_{\mathrm{msc}}}{ms}\right)^{h-i}\frac{\mathsf{F}_{\mathrm{msc}}}{ms}$, and hence, the probability for the online chain to merge with a pre-computation chain strictly after the i-th color is the sum of that probabilities for $i+1\le h\le s$, that is,

$$\sum_{h=i+1}^{s}\left(1-\frac{\mathsf{F}_{\mathrm{msc}}}{ms}\right)^{h-i}\frac{\mathsf{F}_{\mathrm{msc}}}{ms}=\left(1-\frac{\mathsf{F}_{\mathrm{msc}}}{ms}\right)\left\{1-\left(1-\frac{\mathsf{F}_{\mathrm{msc}}}{ms}\right)^{s-i}\right\}.$$

The number of the one-way function iterations to resolve this alarm is expected to be it iterations. The m times of the cost associated with single pre-computation chain is

$$m\cdot it\cdot\left(1-\frac{\mathsf{F}_{\mathrm{msc}}}{ms}\right)\left\{1-\left(1-\frac{\mathsf{F}_{\mathrm{msc}}}{ms}\right)^{s-i}\right\},$$

and this can be approximated by

$$m\cdot it\cdot(s-i)\frac{\mathsf{F}_{\mathrm{msc}}}{ms}=i(s-i)\frac{\mathsf{F}_{\mathrm{msc}}}{s}t$$

as desired, since

$$\begin{aligned}
&1-\Big(1-\frac{\mathtt{F}_{\mathrm{msc}}}{ms}\Big)^{s-i} \\
&\quad = (s-i)\frac{\mathtt{F}_{\mathrm{msc}}}{ms} - \binom{s-i}{2}\Big(\frac{\mathtt{F}_{\mathrm{msc}}}{ms}\Big)^2 + \cdots - (-1)^{s-i}\binom{s-i}{s-i}\Big(\frac{\mathtt{F}_{\mathrm{msc}}}{ms}\Big)^{s-i} \\
&\quad \approx (s-i)\frac{\mathtt{F}_{\mathrm{msc}}}{ms}.
\end{aligned}$$

□

The two components of the cost of resolving alarms for the non-perfect fuzzy rainbow tradeoff have been obtained, and we are ready to state the tradeoff coefficient.

Theorem 3.2.10. *The time memory tradeoff curve for the non-perfect table fuzzy rainbow tradeoff is* $TM^2 = \mathtt{F}_{\mathrm{tc}}\mathsf{N}^2$*, where the tradeoff coefficient is*

$$\begin{aligned}
\mathtt{F}_{\mathrm{tc}} = &-\frac{\{\ln(1-\mathtt{F}_{\mathrm{ps}})\}^3}{\mathtt{F}_{\mathrm{msc}}\mathtt{F}_{\mathrm{cr}}^3} \\
&\times \frac{1}{s}\sum_{i=1}^{s}\Big(1-\mathtt{F}_{\mathrm{ps}}\Big)^{\frac{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}})}{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2})}}\left\{\big(1-\frac{i-1}{s}\big)+\big((i+1)+i(s-i)\big)\frac{\mathtt{F}_{\mathrm{msc}}}{s^2}\right\}.
\end{aligned}$$

Proof. The cost to generate an online chain for a single non-perfect fuzzy rainbow pre-computation matrix that starts from the i-th color is expected to be $\big(s-i+1\big)t$ iterations of the one-way function, and the cost of resolving alarms that may be induced by possible merges of this online chain into the pre-computation matrix is expected to be the sum $\big((i+1)+i(s-i)\big)\frac{\mathtt{F}_{\mathrm{msc}}}{s}t$ of two results of Lemma 3.2.8 and Lemma 3.2.9. Note that it is necessary to multiply these two terms by ℓ to take ℓ online chains into account. Hence, the time complexity is

$$T = t\ell\sum_{i=1}^{s}\Big(1-\mathtt{F}_{\mathrm{ps}}\Big)^{\frac{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}})}{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2})}}\Big\{\big(s-i+1\big)+\big((i+1)+i(s-i)\big)\frac{\mathtt{F}_{\mathrm{msc}}}{s}\Big\}.$$

The storage complexity is $M = m\ell$, and the form of the time memory tradeoff curve stated above can be reached by a simple calculation. □

Note that the time complexity T, appearing in the above proof, is bounded from below by

$$T \geq t\ell \sum_{i=1}^{s} (1 - \mathtt{F}_{\mathrm{ps}})(s - i + 1) = t\ell \frac{s(s+1)}{2}(1 - \mathtt{F}_{\mathrm{ps}}),$$

and bound from above by

$$T \leq t\ell \sum_{i=1}^{s} \left\{ \big(s-i+1\big) + \big(1+s(s-i+1)\big)\frac{\mathtt{F}_{\mathrm{msc}}}{s} \right\} = t\ell \left\{ \frac{s(s+1)}{2}(\mathtt{F}_{\mathrm{msc}}+1) + \mathtt{F}_{\mathrm{msc}} \right\}.$$

Since $\ell = \Theta(t)$ from the last paragraph in Section 3.2.1, the online time complexity T of the non-perfect fuzzy rainbow tradeoff must be of $\Theta(t^2 s^2)$ order.

Before ending this section, we give the expected number of table lookups during the online phase of the non-perfect fuzzy rainbow tradeoff algorithm.

Proposition 3.2.11. *The online processing of the non-perfect fuzzy rainbow tradeoff is expected to require*

$$\ell \sum_{i=1}^{s} \left(1 - \mathtt{F}_{\mathrm{ps}}\right)^{\frac{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}})}{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2})}}$$

lookups to the non-perfect fuzzy rainbow tables.

Proof. The probability for ℓ online chains that start from the i-th colors of ℓ non-perfect fuzzy rainbow pre-computation matrices to be generated is given by Lemma 3.2.7, and such case requires ℓ table lookups. Thus, the expected number of total table lookups is obtained as stated. □

Since the table lookup count stated by the above proposition is bounded from above by $s\ell$ and bounded from below by $s\ell(1 - \mathtt{F}_{\mathrm{ps}})$, we can easily verify that the number of table lookups made by the online phase of the non-perfect fuzzy rainbow tradeoff is of $\Theta(ts)$ order, which is much smaller than the online complexity $T = \Theta(t^2 s^2)$.

3.3 Perfect Table Fuzzy Rainbow Tradeoff

The online efficiency of the perfect table fuzzy rainbow tradeoff is analyzed in this section. This content can be also found in [20]. The parameters m, t, and s, for the perfect fuzzy rainbow tradeoff, are chosen to satisfy $mt^2s = \bar{\mathtt{F}}_{\mathrm{msc}}\mathtt{N}$, with a matrix stopping constant $\bar{\mathtt{F}}_{\mathrm{msc}}$ that is neither too large nor very close to zero.

3.3.1 Probability of Success

Let us recall a non-perfect fuzzy rainbow matrix with m_0 starting points and its non-perfect DP sub-matrices $\mathtt{DM}_i$. We have already seen in Lemma 3.2.2 that the number m_i of the i-th color boundary points in $\mathtt{DM}_i$ has the closed-form approximation

$$m_i = \frac{2m_0}{2 + \mathtt{F}_{\mathrm{msc}}\frac{i}{s}}, \tag{3.3.1}$$

where $\mathtt{F}_{\mathrm{msc}} = \frac{m_0t^2s}{\mathtt{N}}$ is the matrix stopping constant for the non-perfect fuzzy rainbow matrix.

Let us rewrite Eq. (3.3.1) in terms of the perfect fuzzy rainbow tradeoff parameters for relevant uses.

Lemma 3.3.1. *To create a perfect fuzzy rainbow matrix containing m non-merging chains, one must expect to generate $m_0 = \frac{2m}{2-\bar{\mathtt{F}}_{\mathrm{msc}}}$ chains. Furthermore, the number of i-th color boundary points in $\mathtt{DM}_i$ of a non-perfect fuzzy rainbow matrix with m_0 starting points is expected to be*

$$m_i = \frac{2m}{(2 - \bar{\mathtt{F}}_{\mathrm{msc}}) + \bar{\mathtt{F}}_{\mathrm{msc}}\frac{i}{s}},$$

for $i = 0, 1, \ldots, s$.

Proof. From Lemma 3.2.2, it is easy to obtain that a non-perfect fuzzy rainbow matrix with m_0 starting points contains $m_s = \frac{2m_0}{2+\mathtt{F}_{\mathrm{msc}}}$ non-merging chains, where $\mathtt{F}_{\mathrm{msc}} = \frac{m_0t^2s}{\mathtt{N}}$. Since $m_0 = \frac{2+\mathtt{F}_{\mathrm{msc}}}{2}m$ and $\mathtt{F}_{\mathrm{msc}} = \frac{m_0t^2s}{\mathtt{N}} = \frac{2+\mathtt{F}_{\mathrm{msc}}}{2}\frac{mt^2s}{\mathtt{N}} =$

$\frac{2+\mathtt{F}_{\mathrm{msc}}}{2}\bar{\mathtt{F}}_{\mathrm{msc}}$, we have

$$\bar{\mathtt{F}}_{\mathrm{msc}} = \frac{2\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}}, \tag{3.3.2}$$

which is equivalent to

$$\mathtt{F}_{\mathrm{msc}} = \frac{2\bar{\mathtt{F}}_{\mathrm{msc}}}{2-\bar{\mathtt{F}}_{\mathrm{msc}}}. \tag{3.3.3}$$

Combining $m_0 = \frac{2+\mathtt{F}_{\mathrm{msc}}}{2}m$ and Eq. (3.3.3), then we can obtain the first claim.

As for the second claim,

$$m_i = \frac{2m_0}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}} = \frac{2}{2+\frac{2\bar{\mathtt{F}}_{\mathrm{msc}}}{2-\bar{\mathtt{F}}_{\mathrm{msc}}}\frac{i}{s}}\frac{2m}{2-\bar{\mathtt{F}}_{\mathrm{msc}}} = \frac{2m}{(2-\bar{\mathtt{F}}_{\mathrm{msc}})+\bar{\mathtt{F}}_{\mathrm{msc}}\frac{i}{s}}$$

from Lemma 3.2.2 and Eq. (3.3.3). □

Both Eq. (3.3.2) and Eq. (3.3.3) imply that $\bar{\mathtt{F}}_{\mathrm{msc}} < 2$ is always satisfied. One may predict that taking $\bar{\mathtt{F}}_{\mathrm{msc}}$ very close to 2 makes bad parameter choices. Later, it will be observed in Section 5.2 that $\bar{\mathtt{F}}_{\mathrm{msc}}$ is bounded sufficiently away from 2 for any meaningful parameters and that the pre-computation requirement grows unrealistically large as $\bar{\mathtt{F}}_{\mathrm{msc}}$ approaches 2.

Our next goal is to obtain a formula for the *coverage rate* of a perfect fuzzy rainbow matrix, which is defined to be

$$\bar{\mathtt{F}}_{\mathrm{cr}} = \frac{1}{mts}\big(|\overline{\mathtt{DM}}_1| + |\overline{\mathtt{DM}}_2| + \cdots + |\overline{\mathtt{DM}}_s|\big).$$

Let us just focus on $\widetilde{\mathtt{DM}}_i$, the retained perfect DP sub-matrix with m_i ending points in the i-th step during the pre-computation phase. For each $\widetilde{\mathtt{DM}}_i$, we use notation $\bar{\mathtt{f}}_i = \frac{m_i t^2}{\mathtt{N}}$, and one can easily compute that

$$\bar{\mathtt{f}}_i = \frac{1}{s}\,\frac{2\bar{\mathtt{F}}_{\mathrm{msc}}}{(2-\bar{\mathtt{F}}_{\mathrm{msc}})+\bar{\mathtt{F}}_{\mathrm{msc}}\frac{i}{s}}. \tag{3.3.4}$$

For convenience, we use Eq. (3.3.4) for all non-negative integer i, even if $i > s$. Note that since $\bar{\mathtt{F}}_{\mathrm{msc}}$ is bounded away from 2 for all practical parameter sets, we may assume $\bar{\mathtt{f}}_i$ to be of $\Theta\big(\frac{1}{s}\big)$ order.

From Eq. (3.3.4), one can also obtain two equalities,

$$\frac{m_{i-1}}{m_i} = \frac{\bar{\mathtt{f}}_{i-1}}{\bar{\mathtt{f}}_i} \tag{3.3.5}$$

and

$$1 + \frac{\bar{\mathtt{f}}_i}{2} = \frac{\bar{\mathtt{f}}_i}{\bar{\mathtt{f}}_{i+1}}, \tag{3.3.6}$$

and these equalities will be also used later.

We next define the coverage rate $\bar{\mathtt{F}}_{\mathrm{cr},i}$ of $\widetilde{\mathtt{DM}}_i$ to be $\frac{|\widetilde{\mathtt{DM}}_i|}{m_i t}$, where $|\widetilde{\mathtt{DM}}_i|$ denotes the number of distinct points expected in $\widetilde{\mathtt{DM}}_i$. Recall from [22] (or Section 2.1.2) that the coverage rate of a perfect DP matrix $\bar{\mathtt{D}}_{\mathrm{cr}}$ is

$$\bar{\mathtt{D}}_{\mathrm{cr}} = \frac{2}{\bar{\mathtt{D}}_{\mathrm{msc}}} \ln\left(1 + \frac{\bar{\mathtt{D}}_{\mathrm{msc}}}{2}\right) \tag{3.3.7}$$

where $\bar{\mathtt{D}}_{\mathrm{msc}} = \frac{mt^2}{\mathtt{N}}$ is the matrix stopping constant of the perfect DP tradeoff with m ending points. Through the above Eq. (3.3.7), we can easily obtain the following lemma for $\bar{\mathtt{F}}_{\mathrm{cr},i}$.

Lemma 3.3.2. *The coverage rate of a retained DP sub-matrix $\widetilde{\mathtt{DM}}_i$ in the i-th step during the pre-computation phase is*

$$\bar{\mathtt{F}}_{\mathrm{cr},i} = \frac{2}{\bar{\mathtt{f}}_i} \ln\left(1 + \frac{\bar{\mathtt{f}}_i}{2}\right).$$

Now, let us look at $\overline{\mathtt{DM}}_i$, the i-th perfect DP sub-matrix with m ending points of a perfect fuzzy rainbow matrix. One can wonder the coverage rate of $\overline{\mathtt{DM}}_i$, $\frac{|\overline{\mathtt{DM}}_i|}{mt}$, which might be a great help to obtain tractable formula for $\bar{\mathtt{F}}_{\mathrm{cr}}$. Fortunately, it is no necessity for obtaining a new formula for the coverage rate of $\overline{\mathtt{DM}}_i$. Theoretically,

$$\frac{|\widetilde{\mathtt{DM}}_i|}{m_i t} = \frac{|\overline{\mathtt{DM}}_i|}{mt}, \tag{3.3.8}$$

because m chains of $\overline{\mathtt{DM}}_i$ can be considered as randomly chosen m chains from

m_i chains of $\widetilde{\mathtt{DM}_i}$. Thus, we can express $\bar{\mathtt{F}}_{\mathrm{cr}}$ in terms of s and $\bar{\mathtt{F}}_{\mathrm{cr},i}$, as

$$\bar{\mathtt{F}}_{\mathrm{cr}} = \frac{1}{mts}\sum_{i=1}^{s}|\overline{\mathtt{DM}}_i| = \frac{1}{s}\sum_{i=1}^{s}\frac{|\overline{\mathtt{DM}}_i|}{mt} = \frac{1}{s}\sum_{i=1}^{s}\bar{\mathtt{F}}_{\mathrm{cr},i}.$$

Note that both $\bar{\mathtt{F}}_{\mathrm{cr},i}$ and $\bar{\mathtt{F}}_{\mathrm{cr}}$ are of $\Theta(1)$ order.

Proposition 3.3.3. *The coverage rate of a perfect fuzzy rainbow matrix is*

$$\bar{\mathtt{F}}_{\mathrm{cr}} = \frac{1}{s}\sum_{i=1}^{s}\bar{\mathtt{F}}_{\mathrm{cr},i}.$$

We are now ready to get the following proposition for the success probability of the perfect fuzzy rainbow tradeoff.

Proposition 3.3.4. *The success probability of the perfect table fuzzy rainbow tradeoff is*

$$\bar{\mathtt{F}}_{\mathrm{ps}} = 1 - \exp\Big(-\bar{\mathtt{F}}_{\mathrm{msc}}\bar{\mathtt{F}}_{\mathrm{cr}}\frac{\ell}{t}\Big).$$

Proof. The probability that the perfect table fuzzy rainbow tradeoff fails is

$$\begin{aligned}1-\bar{\mathtt{F}}_{\mathrm{ps}} =& \Big(1-\frac{|\overline{\mathtt{DM}}_s|}{\mathtt{N}}\Big)^{\ell}\Big(1-\frac{|\overline{\mathtt{DM}}_{s-1}|}{\mathtt{N}}\Big)^{\ell}\cdots\cdots\Big(1-\frac{|\overline{\mathtt{DM}}_1|}{\mathtt{N}}\Big)^{\ell}\\ =& \Big(1-\bar{\mathtt{F}}_{\mathrm{msc}}\frac{\bar{\mathtt{F}}_{\mathrm{cr},s}}{ts}\Big)^{\ell}\Big(1-\bar{\mathtt{F}}_{\mathrm{msc}}\frac{\bar{\mathtt{F}}_{\mathrm{cr},s-1}}{ts}\Big)^{\ell}\cdots\cdots\Big(1-\bar{\mathtt{F}}_{\mathrm{msc}}\frac{\bar{\mathtt{F}}_{\mathrm{cr},1}}{ts}\Big)^{\ell}.\end{aligned}$$

Thus, $\bar{\mathtt{F}}_{\mathrm{ps}}$ can be approximately

$$1-\prod_{i=1}^{s}\exp\Big(-\bar{\mathtt{F}}_{\mathrm{msc}}\frac{\ell}{t}\frac{\bar{\mathtt{F}}_{\mathrm{cr},i}}{s}\Big) = 1-\exp\Big(-\bar{\mathtt{F}}_{\mathrm{msc}}\frac{\ell}{t}\sum_{i=1}^{s}\frac{\bar{\mathtt{F}}_{\mathrm{cr},i}}{s}\Big) = 1-e^{-\bar{\mathtt{F}}_{\mathrm{msc}}\bar{\mathtt{F}}_{\mathrm{cr}}\frac{\ell}{t}}.$$

□

This proposition shows that any set of parameters m, t, s, ℓ with $0 < \bar{\mathtt{F}}_{\mathrm{msc}} < 2$ that attains any success rate requirement $\bar{\mathtt{F}}_{\mathrm{ps}}$ must satisfy the relation

$$\frac{\ell}{t} = \frac{\{-\ln(1-\bar{\mathtt{F}}_{\mathrm{ps}})\}}{\bar{\mathtt{F}}_{\mathrm{msc}}\bar{\mathtt{F}}_{\mathrm{cr}}}. \tag{3.3.9}$$

Note that this implies that $\frac{\ell}{t} = \Theta(1)$, unless the success probability requirement $\bar{\mathtt{F}}_{\mathrm{ps}}$ is unrealistically close to 1. In other words, ℓ and t are of the same order.

Now, we define the *pre-computation coefficient* of the perfect table fuzzy rainbow tradeoff algorithms. The number of one-way function invocations requiered to create each temporary sub-matrix $\widetilde{\mathtt{DM}_i}$ from its m_{i-1} distinct starting points is $m_{i-1}t$ expected, since each DP chain is expected to be of length t on average. The effort of sorting the ending points of $\widetilde{\mathtt{DM}_i}$, so that duplicates can be removed and the distinct starting points for the next sub-matrix are obtained, is of $m \log m$ order. This is much smaller than the effort of generating the sub-matrix, and can be ignored. Taking account of the ℓ tables, the pre-computation cost is expected to be $(m_0 + m_1 + \cdots + m_{s-1})t\ell$, and the pre-computation coefficient is defined to be

$$\bar{\mathtt{F}}_{\mathrm{pc}} = \frac{(m_0 + m_1 + \cdots + m_{s-1})t\ell}{\mathsf{N}}.$$

Corollary 3.3.5. $\bar{\mathtt{F}}_{\mathrm{pc}} = \dfrac{\{-\ln(1 - \bar{\mathtt{F}}_{\mathrm{ps}})\}}{\bar{\mathtt{F}}_{\mathrm{msc}}\bar{\mathtt{F}}_{\mathrm{cr}}} \displaystyle\sum_{i=0}^{s-1} \bar{\mathtt{f}}_i.$

Proof. It is obvious by applying two equations Eq. (3.3.4) and Eq. (3.3.9) to

$$\bar{\mathtt{F}}_{\mathrm{pc}} = \frac{\ell}{t} \sum_{i=0}^{s-1} \frac{m_i t^2}{\mathsf{N}}.$$

□

3.3.2 Online Complexity

Having full knowledge of previous subsection, we gain the online execution complexity. We will focus on the average case complexity, rather than the worst case complexity.

We first assess how likely each pass of the online phase is to be executed.

Lemma 3.3.6. *The probability for the ℓ i-th DP sub-matrices $\overline{\mathtt{DM}_i}$ within the ℓ perfect fuzzy rainbow matrices to be searched for the correct answer to the*

inversion problem is

$$\left(1-\bar{\mathtt{F}}_{\mathrm{ps}}\right)^{\frac{\sum_{k=i+1}^{s}\bar{\mathtt{F}}_{\mathrm{cr},k}}{s\bar{\mathtt{F}}_{\mathrm{cr}}}}.$$

Proof. The ℓ i-th DP sub-matrices $\overline{\mathtt{DM}}_i$ of the ℓ perfect fuzzy rainbow matrices will be searched if and only if there is no correct answer of the inversion target in $\overline{\mathtt{DM}}_{i+1}$, ..., $\overline{\mathtt{DM}}_s$ of the all ℓ perfect fuzzy rainbow matrices. Hence, the probability under consideration is

$$\begin{aligned}
&\left(1-\frac{|\overline{\mathtt{DM}}_{i+1}|}{\mathsf{N}}\right)^{\ell}\left(1-\frac{|\overline{\mathtt{DM}}_{i+2}|}{\mathsf{N}}\right)^{\ell}\cdots\cdots\left(1-\frac{|\overline{\mathtt{DM}}_{s}|}{\mathsf{N}}\right)^{\ell}\\
\approx&\exp\left(-\bar{\mathtt{F}}_{\mathrm{msc}}\frac{\ell}{t}\frac{|\overline{\mathtt{DM}}_{i+1}|}{mts}\right)\exp\left(-\bar{\mathtt{F}}_{\mathrm{msc}}\frac{\ell}{t}\frac{|\overline{\mathtt{DM}}_{i+2}|}{mts}\right)\cdots\cdots\exp\left(-\bar{\mathtt{F}}_{\mathrm{msc}}\frac{\ell}{t}\frac{|\overline{\mathtt{DM}}_{s}|}{mts}\right)\\
=&\exp\left(-\bar{\mathtt{F}}_{\mathrm{msc}}\frac{\ell}{t}\frac{1}{s}\sum_{k=i+1}^{s}\bar{\mathtt{F}}_{\mathrm{cr},k}\right)=\left(1-\bar{\mathtt{F}}_{\mathrm{ps}}\right)^{\frac{\sum_{k=i+1}^{s}\bar{\mathtt{F}}_{\mathrm{cr},k}}{s\bar{\mathtt{F}}_{\mathrm{cr}}}},
\end{aligned}$$

where the final equality follows from an application of Proposition 3.3.4. □

Next aim is to obtain the cost of dealing with alarms that occur during the online phase. This is the most technical part because of very delicate random function arguments.

As for the non-perfect fuzzy rainbow tradeoff, we assume the search on a DP sub-matrix $\overline{\mathtt{DM}}_i$ of a single perfect fuzzy rainbow matrix, and we treat the case of possible merges at the i-th color and the case of possible merges strictly after the i-th color separately. Prior to compute the cost of dealing with alarms, the probability for an online chain to merge into a fuzzy rainbow matrix needs to be considered first.

Lemma 3.3.7. *The probability for an online DP sub-chain segment of the k-th color not to merge into the non-perfect DP sub-matrix $\mathtt{DM}_k$ is $\frac{\bar{\mathtt{f}}_{k+2}}{\bar{\mathtt{f}}_k}$. The probability for an online chain that starts from the i-th color not to merge into the* perfect *fuzzy rainbow pre-computation matrix is $\prod_{k=i}^{s}\frac{\bar{\mathtt{f}}_{k+2}}{\bar{\mathtt{f}}_k}=\frac{\bar{\mathtt{f}}_{s+1}}{\bar{\mathtt{f}}_i}\frac{\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i+1}}$.*

Proof. For the first claim, an online DP sub-chain segment of the i-th color will not merge into the non-perfect DP sub-matrix $\mathtt{DM}_i$ with probability

$$\sum_{k=1}^{\infty}\left(1-\frac{1}{t}-\frac{|\mathtt{DM}_i|}{\mathsf{N}}\right)^{k-1}\left(\frac{1}{t}-\frac{m_i}{\mathsf{N}}\right)=\frac{\frac{1}{t}-\frac{m_i}{\mathsf{N}}}{\frac{1}{t}+\frac{|\mathtt{DM}_i|}{\mathsf{N}}}\approx\frac{1}{1+\frac{m_it^2}{\mathsf{N}}},$$

since the last approximation is supported by that $\frac{m_i}{\mathsf{N}} = \Theta\big(\frac{1}{t^2 s}\big)$ and $\frac{|\mathtt{DM}_i|}{\mathsf{N}} = \Theta\big(\frac{1}{ts}\big)$. After applying Eq. (3.3.4), it can be expressed as

$$\frac{1}{1+\bar{\mathtt{f}}_i} = \frac{(2-\bar{\mathsf{F}}_{\mathrm{msc}}) + \bar{\mathsf{F}}_{\mathrm{msc}}\frac{i}{s}}{(2-\bar{\mathsf{F}}_{\mathrm{msc}}) + \bar{\mathsf{F}}_{\mathrm{msc}}\frac{i+2}{s}} = \frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}.$$

Next, an online chain that starts from the i-th color does not merge into the perfect fuzzy rainbow pre-computation matrix if and only if Any of its DP sub-chain segments does not merge into the corresponding sub-matrices of $\overline{\mathtt{DM}}_i$, $\overline{\mathtt{DM}}_{i+1}$, …, $\overline{\mathtt{DM}}_s$, and this happens if and only if none of the DP sub-chain segments merges into the corresponding non-perfect sub-matrices of $\mathtt{DM}_i$, $\mathtt{DM}_{i+1}$, …, $\mathtt{DM}_s$. Hence, the second claimed probability is

$$\frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\frac{\bar{\mathtt{f}}_{i+3}}{\bar{\mathtt{f}}_{i+1}}\cdots\frac{\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_s} = \frac{\bar{\mathtt{f}}_{s+1}}{\bar{\mathtt{f}}_i}\frac{\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i+1}}.$$

□

The cost of resolving an alarm from a merge strictly after the i-th color is given in the following lemma, with assuming the search on a DP sub-matrix $\overline{\mathtt{DM}}_i$.

Lemma 3.3.8. *Assume the search on a DP sub-matrix $\overline{\mathtt{DM}}_i$ of a single perfect fuzzy rainbow pre-computation matrix. The cost of dealing with an alarm that may be induced by a possible merge of the online chain into the pre-computation matrix strictly after the i-th color is expected to be*

$$\Big\{\frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\Big(1-\frac{\bar{\mathtt{f}}_{s+1}}{\bar{\mathtt{f}}_{i+1}}\frac{\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i+2}}\Big)+\Big(1-\frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\Big)\Big(1-\frac{\bar{\mathtt{f}}_s}{\bar{\mathtt{f}}_i}\Big)\Big\}\Big(\sum_{k=1}^{i}\bar{\mathsf{F}}_{\mathrm{cr},k}\Big)t$$

iterations of the one-way function.

Proof. Note that we want to acquire the probability for an online chain that starts from the i-th color to merge into the perfect fuzzy rainbow pre-computation matrix without merging into the sub-matrix $\overline{\mathtt{DM}}_i$. Such a case could come about through the following two separate events. First, the i-th color online DP sub-chain segment could not merge into the non-perfect

sub-matrix $\mathtt{DM}_i$, but the remainder part of the online chain that starts from the ending DP of the i-th color segment could merge into the perfect pre-computation matrix. Second, the i-th color online DP sub-chain could merge into $\mathtt{DM}_i$, without merging into $\overline{\mathtt{DM}}_i$. In this event, the ending DP of the i-th color DP sub-chain is one of the m_i ending points of $\mathtt{DM}_i$, but the ending point of the full online chain is not one of the m_s ending points of the perfect pre-computation matrix

For the first event, the probability for the i-th color DP sub-chain segment not to merge into $\mathtt{DM}_i$ is $\frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}$, and the probability for the remainder part of the online chain to merge into the perfect pre-computation matrix is $1-\frac{\bar{\mathtt{f}}_{s+1}}{\bar{\mathtt{f}}_{i+1}}\frac{\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i+2}}$, by Lemma 3.3.7. Note that we regarded the remainder part of the chain as still a random chain. Hence, the probability for the first event to happen is

$$\frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\Big(1-\frac{\bar{\mathtt{f}}_{s+1}}{\bar{\mathtt{f}}_{i+1}}\frac{\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i+2}}\Big).$$

For the second event, the probability for the i-th color DP sub-chain segment of the online to merge into $\mathtt{DM}_i$ is $1-\frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}$, and the probability for this online chain not to merge into $\overline{\mathtt{DM}}_i$ is $1-\frac{m_s}{m_i}=1-\frac{\bar{\mathtt{f}}_s}{\bar{\mathtt{f}}_i}$. Thus, the probability for the second event to happen is

$$\Big(1-\frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\Big)\Big(1-\frac{\bar{\mathtt{f}}_s}{\bar{\mathtt{f}}_i}\Big).$$

The probability for a merge to appear strictly after the i-th color is the sum of two probabilities.

Finally, to resolve the alarm from the merge, it is expected to require $(\bar{\mathtt{F}}_{\mathrm{cr},1}+\cdots+\bar{\mathtt{F}}_{\mathrm{cr},i})t$ iterations of the one-way function, since the average chain length of DP sub-chains in each i-th perfect DP sub-matrix $\overline{\mathtt{DM}}_i$ is $\bar{\mathtt{F}}_{\mathrm{cr},i}t$.

The claimed cost is a simple product of the merge probability and the number of iterations. □

We next provides the cost of resolving an alarm by a merge within the i-th color.

Lemma 3.3.9. *Assume the search on a DP sub-matrix $\overline{\mathtt{DM}}_i$ of a single perfect*

fuzzy rainbow pre-computation matrix. The cost of dealing with an alarm that may be induced by a possible merge of the online chain into the pre-computation matrix at the i-th color segment is expected to be approximately

$$\Big(1-\frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\Big)\frac{\bar{\mathtt{f}}_s}{\bar{\mathtt{f}}_i}\Big(1+\sum_{k=1}^{i}\bar{\mathtt{F}}_{\mathrm{cr},k}\Big)t$$

iterations of the one-way function.

Proof. Arguments given in the proof of Lemma 3.3.8 also gives that the probability for a merge to appear at the i-th color segment into the perfect DP sub-matrix $\overline{\mathtt{DM}}_i$ is

$$\Big(1-\frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\Big)\frac{\bar{\mathtt{f}}_s}{\bar{\mathtt{f}}_i}.$$

Since there exists at most one merge of the online chain into the pre-computation matrix, it only remains to figure out how many iterations of the one-way function are required to resolve such a merge.

An alarm will require the associated pre-computation chain to be regenerated at least up to the start of the i-th DP sub-chain, and this is expected to cost $(\bar{\mathtt{F}}_{\mathrm{cr},1}+\cdots+\bar{\mathtt{F}}_{\mathrm{cr},i-1})t$ iterations of the one-way function. The required number of iterations for the additional i-th color segment must be treated more carefully. It must be expected to cost larger than $\bar{\mathtt{F}}_{\mathrm{cr},i}t$ since longer pre-computation chains are more likely to be involved in merges.

It can be inferred from Lemma 12 of [17] that the number of false alarms and the cost of resolving alarms on the processing of a single non-perfect DP table are $\mathtt{D}_{\mathrm{msc}}$ and $2\mathtt{D}_{\mathrm{msc}}t$, respectively, where $\mathtt{D}_{\mathrm{msc}}$ is the matrix stopping constant of the non-perfect DP matrix. Hence, we can conclude that, during the processing of a single non-perfect DP table, each alarm calls for $2t$ iterations of the one-way function to resolve, on average.

Now, since Eq. (3.3.5) and Eq. (3.3.6) imply that $\frac{m_i}{m_{i-1}}=1-\Theta\big(\frac{1}{2s}\big)$, we know that only a small portion of $\mathtt{DM}_i$ is discarded in creating $\widetilde{\mathtt{DM}}_i$, so that the DP matrices $\mathtt{DM}_i$ and $\widetilde{\mathtt{DM}}_i$ must be similar in their distributions of chain lengths. Moreover, since the selection of $\overline{\mathtt{DM}}_i$ from $\widetilde{\mathtt{DM}}_i$ does not affect the distribution of chain lengths, it seems to be reasonable to expect a merge of

the online chain within the i-th color segment into either $\mathtt{DM}_i$ or $\overline{\mathtt{DM}}_i$ to both call for approximately $2t$ iterations of the i-th colored one-way function, on average.

On the other hand, the average chain length of a non-perfect DP matrix is t, whereas that of $\overline{\mathtt{DM}}_i$ is $\bar{\mathtt{F}}_{\mathrm{cr},i}t$. Since the average chain length may be understood as a concise representation of the distribution of chain lengths, one might expect $2\bar{\mathtt{F}}_{\mathrm{cr},i}t$, which take the average chain length of $\overline{\mathtt{DM}}_i$ into account, to be a better approximation of the additional work than $2t$. In any case, since Lemma 3.3.2 implies that $\bar{\mathtt{F}}_{\mathrm{cr},i} = 1 - \Theta\big(\frac{1}{4s}\big)$, we know that $2t$ and $2\bar{\mathtt{F}}_{\mathrm{cr},i}t$ are very close to each other. In view of this similarity, we *choose* to take $(1+\bar{\mathtt{F}}_{\mathrm{cr},i})t$ as the number of extra i-th color iterations required, since this will make our later formulas look slightly simpler.

The claimed cost is a simple product of $\big(1 - \frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\big)\frac{\bar{\mathtt{f}}_s}{\bar{\mathtt{f}}_i}$ and $(\bar{\mathtt{F}}_{\mathrm{cr},1} + \cdots + \bar{\mathtt{F}}_{\mathrm{cr},i-1})t + (1 + \bar{\mathtt{F}}_{\mathrm{cr},i})t$. □

After combining Lemma 3.3.8 and Lemma 3.3.9, the cost of dealing with an alarm that may be induced by an online chain generated from the i-th color is

$$\Big\{\Big(1 - \frac{\bar{\mathtt{f}}_{s+1}}{\bar{\mathtt{f}}_i}\frac{\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i+1}}\Big)\Big(\sum_{k=1}^{i}\bar{\mathtt{F}}_{\mathrm{cr},k}\Big) + \big(1 - \frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\big)\frac{\bar{\mathtt{f}}_s}{\bar{\mathtt{f}}_i}\Big\}t. \tag{3.3.10}$$

One might have something to be clarified for this cost because of the proof of Lemma 3.3.9, but later, our experiments given in Section 7.3 verify that Eq. (3.3.10) is highly accurate.

After piecing all the argument so far together, we can derive the time memory tradeoff curve for the perfect fuzzy rainbow tradeoff.

Theorem 3.3.10. *The time memory tradeoff curve for the perfect table fuzzy*

rainbow tradeoff is $TM^2 = \bar{\mathtt{F}}_{\mathrm{tc}}\mathtt{N}^2$*, where the tradeoff coefficient is*

$$\bar{\mathtt{F}}_{\mathrm{tc}} = -\frac{\{\ln(1-\bar{\mathtt{F}}_{\mathrm{ps}})\}^3}{\bar{\mathtt{F}}_{\mathrm{msc}}\bar{\mathtt{F}}_{\mathrm{cr}}^3} \times \frac{1}{s}\sum_{i=1}^{s}\big(1-\bar{\mathtt{F}}_{\mathrm{ps}}\big)^{\frac{\sum_{k=i+1}^{s}\bar{\mathtt{F}}_{\mathrm{cr},k}}{s\bar{\mathtt{F}}_{\mathrm{cr}}}} \left(\begin{array}{l} \dfrac{s-i+1}{s} + \dfrac{1}{s}\big(1-\dfrac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\big)\dfrac{\bar{\mathtt{f}}_s}{\bar{\mathtt{f}}_i} \\ \quad + \dfrac{1}{s}\Big(1-\dfrac{\bar{\mathtt{f}}_{s+1}}{\bar{\mathtt{f}}_i}\dfrac{\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i+1}}\Big)\Big(\sum_{k=1}^{i}\bar{\mathtt{F}}_{\mathrm{cr},k}\Big) \end{array} \right).$$

Proof. The cost to generate an online chain that starts from the i-th color is expected to be $\big(s-i+1\big)t$ iterations of the one-way function, and the cost of resolving alarms that may be induced by a possible merge of this online chain into the pre-computation matrix is given in Eq. (3.3.10). Note that it is necessary to multiply these two terms by ℓ to take ℓ online chains into account. Hence, the online time complexity can be stated as

$$T = t\ell\sum_{i=1}^{s}\big(1-\bar{\mathtt{F}}_{\mathrm{ps}}\big)^{\frac{\sum_{k=i+1}^{s}\bar{\mathtt{F}}_{\mathrm{cr},k}}{s\bar{\mathtt{F}}_{\mathrm{cr}}}} \left(\begin{array}{l} (s-i+1) + \big(1-\dfrac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\big)\dfrac{\bar{\mathtt{f}}_s}{\bar{\mathtt{f}}_i} \\ \quad + \Big(1-\dfrac{\bar{\mathtt{f}}_{s+1}}{\bar{\mathtt{f}}_i}\dfrac{\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i+1}}\Big)\Big(\sum_{k=1}^{i}\bar{\mathtt{F}}_{\mathrm{cr},k}\Big) \end{array} \right) \quad (3.3.11)$$

The storage complexity is $M = m\ell$, and the time memory tradeoff curve stated above can be easily reached by a simple calculation. □

As the cases for the online time complexities of the thick rainbow and non-perfect fuzzy rainbow tradeoffs, it is easy to argue that T is of $\Theta(t^2s^2)$ order since $\ell = \Theta(t)$ from Eq. (3.3.9).

We lastly give the following proposition for the number of table lookups expected during the online phase of the perfect fuzzy rainbow tradeoff, and its proof is almost identical to that of Lemma 3.2.11.

Proposition 3.3.11. *The online processing of the perfect fuzzy rainbow tradeoff is expected to require*

$$\ell\sum_{i=1}^{s}\big(1-\bar{\mathtt{F}}_{\mathrm{ps}}\big)^{\frac{\sum_{k=i+1}^{s}\bar{\mathtt{F}}_{\mathrm{cr},k}}{s\bar{\mathtt{F}}_{\mathrm{cr}}}}$$

lookups to the perfect fuzzy rainbow tables.

Proof. The probability for ℓ online chains that start from the i-th colors of ℓ perfect fuzzy rainbow pre-computation matrices to be generated is given by Lemma 3.3.6, and such case requires ℓ table lookups. Thus, the expected number of total table lookups is obtained as stated. □

As for the non-perfect case, one can easily verify that this is of $\Theta(ts)$ order, which is much smaller than the online computational complexity $T = \Theta(t^2s^2)$.

Chapter 4

Storage Optimization

There is a further discussion concerning the storage complexity M to prepare algorithm comparison. The storage complexity M for each algorithm appearing in the previous chapter just refers to the total number of starting point and ending point pair entries to be stored in the pre-computation tables. Practically speaking, the real physical memory size should be considered not only with the number of entries but also with the number of bits needed per each entry, since, for each tradeoff algorithm, the number of bits per entry required to record in the tables can be different.

One can easily think that it requires $2 \log \mathsf{N}$ bits to store each starting point and ending point pair entry. However, there are several techniques to store each entry of each algorithm more efficiently. One can record each starting point in $\log m_0$ bits rather than $\log \mathsf{N}$ bits where m_0 is the number of starting points that are initially used for creating a matrix of a specific tradeoff algorithm. This is possible by utilizing the nature of random functions and consecutive starting points [1, 8, 9]. For the ending points, there are three existing methods that are generally used for the time memory tradeoff algorithms. More details for these methods than what is briefly explained in the followings can be found in [17, 22].

The first method can be applicable only for the tradeoffs with the usage of DPs, such as DP tradeoffs and fuzzy rainbow tradeoffs [8, 27]. It is not necessary to store the distinguished parts of the ending points. For example,

when the distinguishing property is of probability $\frac{1}{t}$, $\log t$ bits of each ending point can be removed without any loss of information.

The second technique is the index table technique [8]. This technique let one remove almost $\log m$ most significant bits of each ending point without losing any information, in the situation that pre-computation tables are sorted according to the ending points.

The last one is the truncation method of ending points [5, 8]. Each ending point is truncated to a certain length before storing the entries in the table, and this can reduce the memory size requirement. However, it loses some ending point information, and hence, it causes another kind of false alarm and more cost of resolving alarms. In the following section, we will gain the degree of ending point truncation that allows its side effects to be negligible, for the three rainbow variants analyzed in the previous chapter.

4.1 The Degree of Ending Point Truncation

The extra cost induced by ending point truncation-related alarms depends on how many truncate the ending points. Thus, it is essential to find proper degree of truncation not to be meaningful as compared with the online time complexity. We offer the degrees of truncation for the thick rainbow, non-perfect fuzzy rainbow and perfect fuzzy rainbow tradeoff algorithms in this section.

4.1.1 Thick Rainbow Tradeoff

Let us fix an ending point truncation method with setting truncation match probability of $\frac{1}{r}$, i.e., we assume that the truncated outcomes of two independently and randomly chosen ending points will be identical with probability $\frac{1}{r}$. We can obtain the truncation match probability of $\frac{1}{r}$ by retaining $\log r$ bits of each ending point.

The extra cost induced by truncation-related alarms for the thick rainbow tradeoff algorithm is stated in the following proposition.

Proposition 4.1.1. *Assume the use of the ending point truncation method with the truncated match probability set to* $\frac{1}{r}$*. Then, during the online phase of the thick rainbow tradeoff, one can expected to observe*

$$\frac{m}{r}\, t^2 \ell \sum_{i=1}^{s} \Big(i - \frac{1}{2}\Big) \Big(1 - \mathtt{K}_{\mathrm{ps}}\Big)^{\frac{\sum_{k=i+1}^{s} \mathtt{K}_{\mathrm{cr},k}}{s\mathtt{K}_{\mathrm{cr}}}}$$

extra invocations of the one-way function induced by truncation-related alarms.

Proof. Consider an online chain that starts at j columns away from the column of ending points in $\mathtt{HM}_i$ of a thick rainbow pre-computation matrix. Through an argument similar to that appearing in the proof of Lemma 3.1.6, we can infer that the number of ending points in the pre-computation matrix that do *not* cause normal alarms with the online chain is expect to be

$$m - m\frac{(s-i)t + (j+1)}{\mathsf{N}}.$$

The probability for a non-merging online chain to bring about a truncated match with any one of the truncated ending points is $\frac{1}{r}$. To resolve each of these truncation-related alarms, $t(i-1)+(t-j)$ iterations of the one-way function is required. Hence, the cost of resolving the truncation-related alarms that may be induced by possible truncated matches of the t online chains generated for the i-th Hellman sub-matrix $\mathtt{HM}_i$ into the pre-computation matrix is

$$\sum_{j=0}^{t-1} \big(t(i-1) + (t-j)\big)\Big\{1 - \frac{(s-i)t + (j+1)}{\mathsf{N}}\Big\}\frac{m}{r}.$$

Since $\frac{(s-i)t+(j+1)}{\mathsf{N}} = O\big(\frac{1}{m}\big)$, we may approximate the above to

$$\sum_{j=0}^{t-1} \big(t(i-1) + (t-j)\big)\frac{m}{r} = \frac{m}{r}\Big(t^2(i-1) + \frac{t(t+1)}{2}\Big) \approx \frac{m}{r}t^2\Big(i - \frac{1}{2}\Big).$$

Taking the ℓ pre-computation matrices into account and recalling Lemma 3.1.5, the cost of dealing with truncation-related alarms can be written as we stated. □

One can easily find that the cost stated in the proposition is upper bounded by

$$\frac{m}{r}\, t^2 \ell \sum_{i=1}^{s} i = \frac{m}{r}\, t^2 \ell\, \frac{s(s+1)}{2}$$

and lower bounded by

$$\frac{m}{r}\, t^2 \ell \sum_{i=1}^{s} (i-1)\,(1-\mathtt{K}_{\mathrm{ps}}) = \frac{m}{r}\, t^2 \ell\, \frac{s(s-1)}{2}(1-\mathtt{K}_{\mathrm{ps}}).$$

Hence, the added cost of dealing with truncation-related alarms is of $\Theta\big(t^2 s^2 \frac{m}{r}\big)$ order in comparison with that the online time complexity T is of $\Theta(t^2 s^2)$ order.

We can infer from these two complexity orders that, if $\frac{m}{r}$ is a sufficiently small fraction, then the added cost of handling truncation-related alarms will be insignificant in comparison to the time complexity T for the algorithm without the use of ending point truncation. In other words, the side effects of ending point truncation can be ignored if each of the truncated ending points contains slightly more than $\log m$ bits of information. Furthermore, even the remaining effective $\log m$ bits of each ending point can mostly be removed through the index table technique, without any loss of information.

In summary, each starting point of the thick rainbow tradeoff can be recorded in $\log m_0 = \log m$ bits and each ending point can be recorded in a small number ε of bits. Hence, each entry of the thick rainbow tradeoff can be recorded in $\log m + \varepsilon$ bits.

4.1.2 Non-Perfect Table Fuzzy Rainbow Tradeoff

Let us fix an ending point truncation method with match probability of $\frac{1}{r}$. Since every ending point is a DP, the match probability of no truncation is of $\frac{1}{r} = \frac{t}{\mathsf{N}}$. Note that, in this case, the way to obtain the truncation match probability of $\frac{1}{r}$ is to retain $\log r$ bits of each ending point except the distinguishing part. Note that whether the distinguishing part is also retained does not affect the match probability.

The following proposition, which has already been published in [19], states the extra cost induced by truncation-related alarms for the non-perfect table fuzzy rainbow tradeoff.

Proposition 4.1.2. *Assume the use of the ending point truncation method with the truncated match probability set to* $\frac{1}{r}$*. Then, during the online phase of the non-perfect fuzzy rainbow tradeoff, one can expected to observe*

$$\frac{m}{r}\, t\ell \sum_{i=1}^{s} i \left(1-\mathtt{F}_{\mathrm{ps}}\right)^{\frac{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}})}{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2})}}$$

extra invocations of the one-way function induced by truncation-related alarms.

Proof. Consider an online chain that starts from the i-th color. Through an argument similar to that appearing in the proof of Lemma 3.2.9, we can deduce that the probability for the online chain *not* to merge into any single fixed pre-computation is

$$1-\sum_{h=i}^{s}\left(1-\frac{\mathtt{F}_{\mathrm{msc}}}{ms}\right)^{h-i}\frac{\mathtt{F}_{\mathrm{msc}}}{ms} \approx 1-(s-i+1)\frac{\mathtt{F}_{\mathrm{msc}}}{ms}.$$

Unless $\frac{1}{r} \approx \frac{t}{\mathtt{N}}$, the probability for the non-merging two chains to bring about a truncated match is $\frac{1}{r}$. Hence, the probability for an online chain that starts from the i-th color to cause a truncation-related alarm is

$$\left\{1-(s-i+1)\frac{\mathtt{F}_{\mathrm{msc}}}{ms}\right\}\frac{1}{r}.$$

Each of these truncation-related alarms is expected to require it iterations of the one-way function to resolve. Taking the ℓm pre-computation chains into account and recalling Lemma 3.2.7, the cost of dealing with truncation-related alarms can be written as

$$\ell m \sum_{i=1}^{s}\left\{1-(s-i+1)\frac{\mathtt{F}_{\mathrm{msc}}}{ms}\right\}\frac{1}{r}\cdot it\cdot\left(1-\mathtt{F}_{\mathrm{ps}}\right)^{\frac{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2+\mathtt{F}_{\mathrm{msc}}\frac{i}{s}})}{\ln(\frac{2+\mathtt{F}_{\mathrm{msc}}}{2})}}.$$

It is enough to observe that $(s-i+1)\frac{\mathtt{F}_{\mathrm{msc}}}{ms} = O\left(\frac{1}{m}\right)$ to claim that our stated

formula is an accurate approximation. □

The cost stated in the proposition is bounded from above by

$$\frac{m}{r}\, t\ell \sum_{i=1}^{s} i = \frac{m}{r}\, t\ell\, \frac{s(s+1)}{2}$$

and bounded from below by

$$\frac{m}{r}\, t\ell \sum_{i=1}^{s} i\,(1-\mathtt{F}_{\mathrm{ps}}) = \frac{m}{r}\, t\ell\, \frac{s(s+1)}{2}(1-\mathtt{F}_{\mathrm{ps}}).$$

Hence, the added cost of dealing with truncation-related alarms is of $\Theta\big(t^2 s^2 \frac{m}{r}\big)$ order in comparison with that the online time complexity T is of $\Theta(t^2 s^2)$ order. Hence, the additional cost induced by the truncation method can be stamped out to a negligible level through retaining slightly more than $\log m$ bits of information for each ending point. Of course, if a truncated ending point still contains bits that can be recovered from the DP definition, they may also be removed without any loss of information. Moreover, by the index table method, almost $\log m$ bits of each ending point can be also removed without any loss of information. Thus, we can conclude that each entry of the non-perfect table fuzzy rainbow tradeoff can be recorded in $\log m + \varepsilon$ bits, where ε is a small positive integer.

4.1.3 Perfect Table Fuzzy Rainbow Tradeoff

With still assuming same situation as the above for the non-perfect version, we state the extra cost incurred by truncation-related alarms for the perfect fuzzy rainbow tradeoff, which has already been provided in [20].

Proposition 4.1.3. *Assume the use of the ending point truncation method in which the probability for two truncated randomly chosen DPs to be identical is $\frac{1}{r}$. Then, during the online phase of the perfect fuzzy rainbow tradeoff, one*

can expected to observe

$$\frac{m}{r}\, t\ell \sum_{i=1}^{s} \Big(\frac{\bar{\mathtt{f}}_{s+1}\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i}\bar{\mathtt{f}}_{i+1}} \sum_{k=1}^{i} \bar{\mathtt{F}}_{\mathrm{cr},k} \Big) \big(1 - \bar{\mathtt{F}}_{\mathrm{ps}}\big)^{\frac{\sum_{k=i+1}^{s} \bar{\mathtt{F}}_{\mathrm{cr},k}}{s\bar{\mathtt{F}}_{\mathrm{cr}}}}$$

extra invocations of the one-way function induced by truncation-related alarms.

Proof. The probability for an online chain *not* to merge into the perfect fuzzy rainbow pre-computation matrix is given by Lemma 3.3.7. The probability for a non-merging online chain to cause a truncation-related alarm with any one of the truncated ending points is $\frac{1}{r}$ and there are m of these ending points in a matrix, each of which could require separate treatment.

Each truncation-related alarm is expected to require $(\bar{\mathtt{F}}_{\mathrm{cr},1} + \cdots + \bar{\mathtt{F}}_{\mathrm{cr},i})t$ iterations of the one-way function to resolve. Taking the ℓ pre-computation matrices into account and recalling Lemma 3.3.6, the claimed formula can be obtained. □

Since $\frac{\bar{\mathtt{f}}_{s+1}\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i}\bar{\mathtt{f}}_{i+1}} = \Theta(1)$ and $\sum_{k=1}^{i} \bar{\mathtt{F}}_{\mathrm{cr},k} = \Theta(i)$, the cost stated in the above can be easily checked to be of $\Theta\big(t^2 s^2 \frac{m}{r}\big)$ order, as for the non-perfect case. One can show, either by comparing this against $T = \Theta(t^2 s^2)$, that the additional cost of resolving alarms induced by the ending point truncation technique can be ignored if each truncated ending point contains slightly more than $\log m$ bits of information.

The arguments appearing in Section 4.1.2 for the non-perfect version may then be repeated, almost word for word, to conclude that each entry of the perfect table fuzzy rainbow tradeoff can be stored in $\log m_0 + \varepsilon$ bits, where ε is a small positive integer.

Chapter 5

Comparison of Algorithms

With our analyses so far, we will compare the three rainbow variants in this chapter. We also recall the non-perfect and perfect original rainbow tradeoff to compare with the three rainbow variants, since the results of [17, 22] show that they are preferable to the other major algorithms.

We follow the comparison technique of [17] to compare algorithms. Let us give a brief account of the technique. We first fix several success rate requirements $\mathtt{X}_{\mathrm{ps}}$, and then we display the graphs for the algorithms, each of which consists of a horizontal axis for the pre-computation coefficients $\mathtt{X}_{\mathrm{pc}}$ and a vertical axis for the tradeoff coefficients $\mathtt{X}_{\mathrm{tc}}$. We use the symbols $\mathtt{X} = \mathtt{K}, \mathtt{F}, \bar{\mathtt{F}}, \mathtt{R}, \bar{\mathtt{R}}$ for the parameters and the complexities as in $m_{\mathtt{K}}$, $\ell_{\bar{\mathtt{R}}}$, $T_{\mathtt{F}}$, and so on.

Some contents of this chapter, concerning the non-perfect and perfect fuzzy rainbow tradeoff algorithms, can be also found in [19, 20].

5.1 Adjustment Factors for Tradeoff Coefficients

The storage complexity M for each tradeoff algorithm signifies just the total number of the starting and ending point pair entries to be recorded in the tables, not the practical storage size. Thus, the tradeoff coefficients $\mathtt{X}_{\mathrm{tc}} = \frac{T_{\mathtt{X}} M_{\mathtt{X}}^2}{\mathsf{N}}$

cannot be applied directly to compare the tradeoffs. Since the number of bits to store each entry may vary with the tradeoff algorithm, each tradeoff coefficient should be adjusted. In this section, we offer the adjustment factors for the tradeoff coefficients $\mathtt{X}_{\mathrm{tc}}$ of the tradeoffs that we will compare.

Recall the end result in each subsection of Section 4.1. The numbers of bits to record each entry of the thick rainbow, the non-perfect fuzzy rainbow, and the perfect fuzzy rainbow tradeoff are $\log m_{\mathtt{K}} + \varepsilon$, $\log m_{\mathtt{F}} + \varepsilon$, and $\log(\frac{2}{2-\bar{\mathtt{F}}_{\mathrm{msc}}} m_{\bar{\mathtt{F}}}) + \varepsilon$, respectively. We also recall from [17, 22] that the number of bits per entry of the non-perfect rainbow tradeoff is $\log m_{\mathtt{R}} + \varepsilon$, and that of the perfect rainbow tradeoff is $\log(\frac{2\ell_{\bar{\mathtt{R}}}}{2\ell_{\bar{\mathtt{R}}} + \ln(1-\bar{\mathtt{R}}_{\mathrm{ps}})} m_{\bar{\mathtt{R}}}) + \varepsilon$. The adjustment factor for each $\mathtt{X}_{\mathrm{tc}}$ may be the square of each number of bits per entry because of $M_{\mathtt{X}}^2$ in $\mathtt{X}_{\mathrm{tc}}$. Since ε bits for each algorithm correspond to the part that stay after the ending point truncation and index table method, ε for the tradeoffs can be shared. Throughout the rest of this thesis, we always assume $\varepsilon = 8$ since it can be acceptable value through the arguments of Section 4.

Now, the requirements that $T_{\mathtt{R}} \approx T_{\bar{\mathtt{R}}} \approx T_{\mathtt{K}} \approx T_{\mathtt{F}} \approx T_{\bar{\mathtt{F}}}$ and $M_{\mathtt{R}} \approx M_{\bar{\mathtt{R}}} \approx M_{\mathtt{K}} \approx M_{\mathtt{F}} \approx M_{\bar{\mathtt{F}}}$ give the relations

$$t_{\mathtt{R}} \approx t_{\bar{\mathtt{R}}} \approx t_{\mathtt{K}} s_{\mathtt{K}} \approx t_{\mathtt{F}} s_{\mathtt{F}} \approx t_{\bar{\mathtt{F}}} s_{\bar{\mathtt{F}}}$$
$$\text{and} \quad m_{\mathtt{R}} \approx m_{\bar{\mathtt{R}}} \approx m_{\mathtt{K}} \approx m_{\mathtt{F}} t_{\mathtt{F}} \approx m_{\bar{\mathtt{F}}} t_{\bar{\mathtt{F}}}$$

by using the facts $\ell_{\mathtt{F}} \approx t_{\mathtt{F}}$, $\ell_{\bar{\mathtt{F}}} \approx t_{\bar{\mathtt{F}}}$, and $\ell_{\mathtt{R}} \approx \ell_{\bar{\mathtt{R}}} \approx 1$. We next set new parameters m_* and t_* such that $m_* t_*^2 \approx \mathsf{N}$ which satisfy $m_* t_* \approx m_{\mathtt{R}}$ and $t_* \approx t_{\mathtt{R}}$. Then it can be easily found the relations

$$m_{\mathtt{R}} \approx m_{\bar{\mathtt{R}}} \approx m_{\mathtt{K}} \approx m_*^{\frac{1}{2}} \mathsf{N}^{\frac{1}{2}} \tag{5.1.1}$$
$$m_{\mathtt{F}} \approx m_* s_{\mathtt{F}}, \tag{5.1.2}$$
$$\text{and} \qquad m_{\bar{\mathtt{F}}} \approx m_* s_{\bar{\mathtt{F}}}. \tag{5.1.3}$$

Using the relations (5.1.1), (5.1.2), and (5.1.3), we can finally write the adjusted tradeoff coefficients as

$$\mathtt{K}_{\mathrm{atc}} = \Big(\frac{3}{2\log \mathsf{N}}\Big)^2 (\frac{1}{2}\log \mathsf{N} + \frac{1}{2}\log m_* + \varepsilon)^2\, \mathtt{K}_{\mathrm{tc}}, \tag{5.1.4}$$

$$\mathtt{F}_{\mathrm{atc},s} = \Big(\frac{3}{2\log \mathsf{N}}\Big)^2 (\log s + \log m_* + \varepsilon)^2\, \mathtt{F}_{\mathrm{tc},s}, \tag{5.1.5}$$

$$\bar{\mathtt{F}}_{\mathrm{atc},s} = \Big(\frac{3}{2\log \mathsf{N}}\Big)^2 \Big(\log \frac{2}{2-\bar{\mathtt{F}}_{\mathrm{msc}}} + \log s + \log m_* + \varepsilon\Big)^2\, \bar{\mathtt{F}}_{\mathrm{tc},s}, \tag{5.1.6}$$

$$\mathtt{R}_{\mathrm{atc}} = \Big(\frac{3}{2\log \mathsf{N}}\Big)^2 \Big(\frac{1}{2}\log \mathsf{N} + \frac{1}{2}\log m_* + \varepsilon\Big)^2\, \mathtt{R}_{\mathrm{tc}}, \tag{5.1.7}$$

$$\bar{\mathtt{R}}_{\mathrm{atc}} = \Big(\frac{3}{2\log \mathsf{N}}\Big)^2 \Big(\log \frac{2\ell_{\bar{\mathtt{R}}}}{2\ell_{\bar{\mathtt{R}}} + \ln(1-\bar{\mathtt{R}}_{\mathrm{ps}})} + \frac{1}{2}\log \mathsf{N} + \frac{1}{2}\log m_* + \varepsilon\Big)^2\, \bar{\mathtt{R}}_{\mathrm{tc}}. \tag{5.1.8}$$

The factor $(\frac{3}{2\log \mathsf{N}})^2$ is just for taking the adjustment factor to $\Theta(1)$ order under the typical situation of parameters. We have also put parameter s for the each fuzzy rainbow tradeoff on the position of subscript of the adjust tradeoff coefficient.

5.2 Some Observations concerning Fuzzy Rainbow Tradeoffs

Prior to comparison of the tradeoff algorithms, we need a further discussion for the (non-perfect and perfect) fuzzy rainbow tradeoffs. Let us look at Figure 5.1 of two boxes, each of which shows the $\mathtt{X}_{\mathrm{pc}}$-$\mathtt{X}_{\mathrm{atc}}$ curves under a fixed success rate. We chose three definite values of s for each figure.

The first one illustrates three $\mathtt{F}_{\mathrm{pc}}$-$\mathtt{F}_{\mathrm{atc}}$ curves with setting $\log m_* + \varepsilon = 13+8 = 21$, which is typical situation for $\mathsf{N} = 39$-bits, and success probability of 90%. The second one displays three $\bar{\mathtt{F}}_{\mathrm{pc}}$-$\bar{\mathtt{F}}_{\mathrm{atc}}$ curves with setting $\log m_* + \varepsilon = 25 + 8 = 33$, which is for $\mathsf{N} = 75$-bits, and 95% success rate.

Each curve was drawn until its lowest point, since the part of curve that would be plotted on the right of the lowest point gives the parameter sets with higher pre-computation costs and lower online efficiencies than the parameter set of the lowest point. One may think that a curve, that has lower

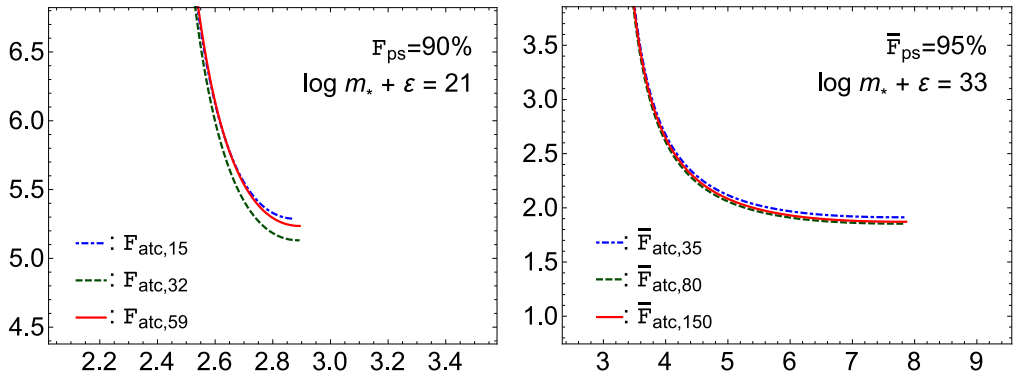

Figure 5.1: The adjusted tradeoff coefficients in relation to corresponding pre-computation coefficients (x-axis: $\mathtt{X}_{pc}$; y-axis: $\mathtt{X}_{atc}$)

lowest point than the lowest point of another curve, is lower than that curve. However, two curves may intersect each other at a higher value of $\mathtt{X}_{atc}$.

Even so, we focus on the lowest points and their slightly left parts of the curves, since higher efficiency part is more attractive in practice. Now, for each fixed $\mathtt{X}_{ps}$ and $\log m_* + \varepsilon$, we say s for the non-perfect and perfect fuzzy rainbow tradeoffs to be optimal if

$$\min_{\mathtt{F}_{msc}} \left\{\mathtt{F}_{atc,s}\right\} = \min_{i \geq 1, \mathtt{F}_{msc}} \left\{\mathtt{F}_{atc,i}\right\} \quad \text{and} \quad \min_{\bar{\mathtt{F}}_{msc}} \left\{\bar{\mathtt{F}}_{atc,s}\right\} = \min_{i \geq 1, \bar{\mathtt{F}}_{msc}} \left\{\bar{\mathtt{F}}_{atc,i}\right\},$$

respectively.

The optimal values of s are given in Table 5.1 for the non-perfect tradeoff and Table 5.2 for the perfect tradeoff, for several fixed success rate requirements and $\log m_* + \varepsilon$ values. For the perfect fuzzy rainbow tradeoff, the value of $\bar{\mathtt{F}}_{msc}$ that gives the minimum $\bar{\mathtt{F}}_{atc,s}$ value is also listed under each optimal s. It can be seen from two tables that the optimal s value becomes larger as the success rate requirement is increased and also as the number of bits per table entry becomes larger.

There is an intuitive reason for the movement of optimal s value related to the success rate requirement. A higher s value allows for earlier termination of the online phase upon an encounter with the correct answer. Early exits from the online phase is less common under low success rates, and the importance

Table 5.1: The optimal s for the non-perfect fuzzy rainbow tradeoff, at various success probabilities and $\log m_* + \varepsilon$ values.

$\log m_* + \varepsilon$	25%	50%	75%	90%	95%	99%	99.5%	99.9%
18	15	17	20	26	30	40	45	56
19	15	18	21	27	31	42	47	59
20	16	18	22	28	33	44	49	62
21	17	19	23	29	34	46	51	64
22	17	20	24	31	35	48	53	67
23	18	21	25	32	37	50	55	69
24	19	21	26	33	38	51	57	72
25	19	22	27	34	40	53	59	74
26	20	23	28	35	41	55	61	77
27	21	24	29	36	42	57	63	79
28	21	25	30	38	44	59	65	82
29	22	25	31	39	45	60	67	84
30	23	26	32	40	46	62	69	87
31	23	27	33	41	48	64	72	89
32	24	28	34	42	49	66	74	92
33	25	28	35	43	50	68	76	94
34	25	29	36	45	52	70	78	97
35	26	30	37	46	53	71	80	99

Table 5.2: The Optimal s for the perfect fuzzy rainbow tradeoff, at various success probabilities and $\log m_* + \varepsilon$ values. The $\bar{\mathrm{F}}_{\mathrm{msc}}$ value listed below each s gives the minimum $\bar{\mathrm{F}}_{\mathrm{atc},s}$ value.

$\log m_* + \varepsilon$		50%	75%	90%	95%	99%	99.5%	99.9%
18	s	34	38	43	48	60	66	79
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.6880	1.6882	1.6846	1.6813	1.6697	1.6647	1.6531
19	s	36	40	46	50	63	68	83
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.6999	1.6996	1.6967	1.6921	1.6810	1.6754	1.6644
20	s	37	42	48	53	65	71	86
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7095	1.7103	1.7071	1.7030	1.6910	1.6858	1.6747
21	s	39	44	50	55	68	74	89
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7198	1.7202	1.7167	1.7126	1.7008	1.6955	1.6843
22	s	41	46	52	57	71	77	93
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7294	1.7293	1.7256	1.7215	1.7101	1.7046	1.6936
23	s	43	48	54	59	73	80	96
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7382	1.7379	1.7339	1.7298	1.7183	1.7133	1.7021
24	s	45	50	56	62	76	83	100
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7465	1.7459	1.7418	1.7381	1.7264	1.7214	1.7105
25	s	47	51	58	64	79	86	103
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7542	1.7527	1.7492	1.7454	1.7340	1.7290	1.7180
26	s	49	53	60	66	81	89	106
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7615	1.7598	1.7562	1.7523	1.7410	1.7362	1.7252
27	s	51	55	63	69	84	91	110
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7684	1.7665	1.7633	1.7593	1.7478	1.7427	1.7322
28	s	52	57	65	71	87	94	113
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7740	1.7728	1.7695	1.7655	1.7543	1.7492	1.7387
29	s	54	59	67	73	89	97	116
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7801	1.7788	1.7754	1.7713	1.7602	1.7554	1.7448
30	s	56	61	69	75	92	100	120
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7859	1.7845	1.7809	1.7769	1.7661	1.7612	1.7508
31	s	58	63	71	78	95	103	123
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7914	1.7898	1.7862	1.7825	1.7716	1.7668	1.7564
32	s	60	65	73	80	97	106	126
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.7966	1.7949	1.7913	1.7875	1.7767	1.7720	1.7618
33	s	62	67	75	82	100	109	130
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.8015	1.7998	1.7961	1.7923	1.7817	1.7771	1.7670
34	s	64	69	78	84	103	111	133
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.8062	1.8044	1.8010	1.7969	1.7865	1.7818	1.7718
35	s	66	71	80	87	105	114	136
	$\bar{\mathrm{F}}_{\mathrm{msc}}$	1.8106	1.8088	1.8053	1.8015	1.7909	1.7864	1.7765

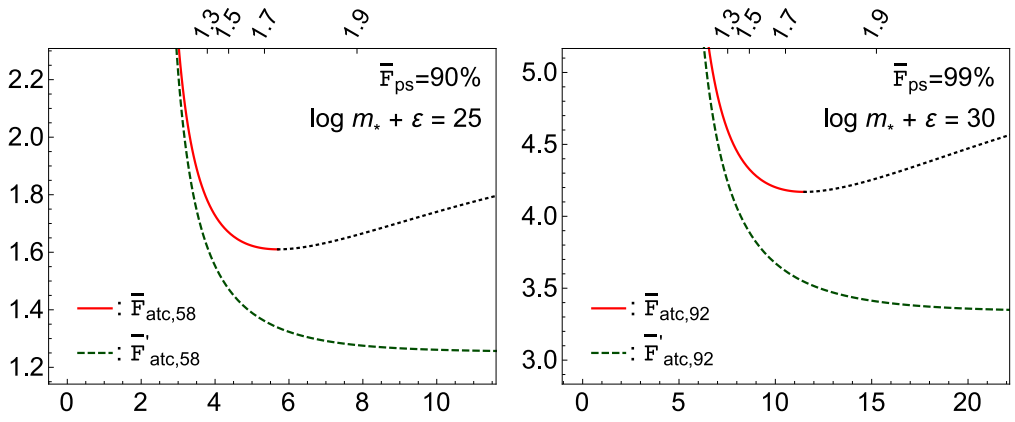

Figure 5.2: The adjusted tradeoff coefficient $\bar{\mathrm{F}}_{\mathrm{atc},s}$ and its simplified version $\bar{\mathrm{F}}'_{\mathrm{atc},s}$ in relation to $\bar{\mathrm{F}}_{\mathrm{pc},s}$ (bottom: $\bar{\mathrm{F}}_{\mathrm{pc},s}$; left: $\bar{\mathrm{F}}_{\mathrm{atc},s}$ and $\bar{\mathrm{F}}'_{\mathrm{atc},s}$; top: $\bar{\mathrm{F}}_{\mathrm{msc}}$)

of higher s value increases as the success rate requirement is increased. As for the movement of optimal s value concerning the number of bits per table entry, it could be from a natural scaling effect that follows the general increase in search space size associated with the increase in $\log m_*$.

Let us now return to the definition of the adjusted tradeoff coefficient. The work [22] had ignored the $\log \frac{2\ell_{\bar{\mathrm{R}}}}{2\ell_{\bar{\mathrm{R}}}+\ln(1-\bar{\mathrm{R}}_{\mathrm{ps}})}$ term from Eq. (5.1.8) in comparing the perfect rainbow tradeoff with the other major tradeoff algorithms. This was justified in their work based on the observation that the term remains bounded above by a small positive number for practical parameters that do not call for large pre-computation cost.

The same argument can be applied to the case of the perfect fuzzy rainbow tradeoff, and the, we can see that the $\log \frac{2}{2-\bar{\mathrm{F}}_{\mathrm{msc}}}$ term of Eq. (5.1.6) is likewise bounded above for parameters that are reasonable in view of pre-computation cost. Hence, we could consider the possibility of using the adjusted tradeoff coefficient defined as

$$\bar{\mathrm{F}}'_{\mathrm{atc},s} = \left(\frac{3}{2\log \mathsf{N}}\right)^2 \big(\log s + \log m_* + \varepsilon\big)^2. \tag{5.2.1}$$

This definition could be favorable to the previous definition Eq. (5.1.6), in view of simplicity.

Figure 5.2 shows the effect of removing the $\log \frac{2}{2-\bar{\mathtt{F}}_{\mathrm{msc}}}$ term from Eq. (5.1.6). The curves for $\bar{\mathtt{F}}_{\mathrm{atc},s}$ and $\bar{\mathtt{F}}'_{\mathrm{atc},s}$ are certainly different from each other, and the definition of (5.2.1) cannot be justified to use.

From Table 5.2, it can be easily seen that any reasonable choice of parameters will mostly satisfy $\bar{\mathtt{F}}_{\mathrm{msc}} \leq 1.8$, and this implies that $\log \frac{2}{2-\bar{\mathtt{F}}_{\mathrm{msc}}} \leq 3.32193$. However, we can also recognize from Table 5.2 that the values of $\log s$ and the values of $\log m_* + \varepsilon$ are not very large. Hence, unlike the case of the perfect rainbow tradeoff, the bound of $\log \frac{2}{2-\bar{\mathtt{F}}_{\mathrm{msc}}}$ is not small enough to be ignored in comparison to the other terms.

There is one more comment that is closely connected to the above discussion. Combining the bound $\bar{\mathtt{F}}_{\mathrm{msc}} \leq 1.8$ with Corollary 3.3.5, we can state the bound

$$\bar{\mathtt{F}}_{\mathrm{pc}} \leq -\frac{\ln(1-\bar{\mathtt{F}}_{\mathrm{ps}})}{\bar{\mathtt{F}}_{\mathrm{msc}}\bar{\mathtt{F}}_{\mathrm{cr}}} \frac{2\bar{\mathtt{F}}_{\mathrm{msc}}}{2-\bar{\mathtt{F}}_{\mathrm{msc}}} \leq -\frac{\ln(1-\bar{\mathtt{F}}_{\mathrm{ps}})}{\Theta(1)}\, 10,$$

concerning the pre-computation coefficient. Hence, pre-computation cost will be bounded by $\Theta(\mathsf{N})$ for all practical parameters, unless the success rate requirement is unrealistically close to 1.

5.3 Comparison

In this section, we finally compare the tradeoff algorithms that we have analyzed with the non-perfect and perfect rainbow tradeoff algorithms. We take pre-computation and online behavior into account by imitating the craft of [17, 22].

Figure 5.3 shows the comparison graphs of $\mathtt{X}_{\mathrm{pc}}$-$\mathtt{X}_{\mathrm{atc}}$ curves of the five tradeoff algorithms with a small space size. The typical situation of $\mathsf{N} = 39$-bits was taken, and three fixed high success probabilities were given. The values of parameter s for the non-perfect and perfect fuzzy rainbow tradeoffs were the optimal values from Table 5.1 and Table 5.2.

The comparisons under the situation of a large space size $\mathsf{N} = 75$-bits are provided in Figure 5.4. As before, we used parameters that are typically

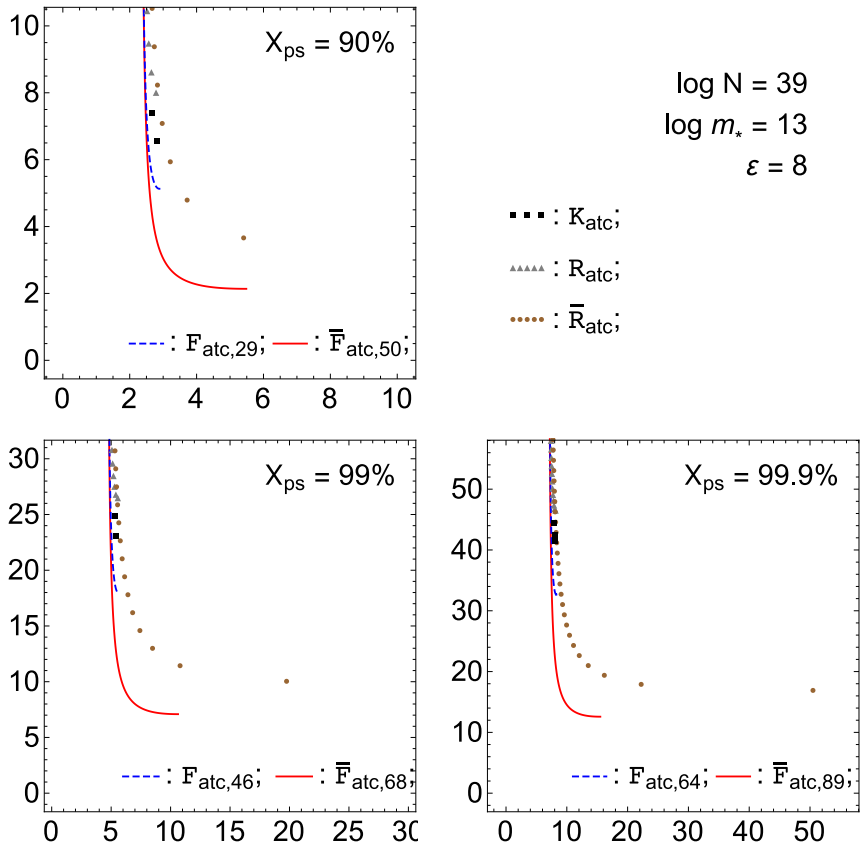

Figure 5.3: The $\mathtt{X}_{\mathrm{pc}}$-$\mathtt{X}_{\mathrm{atc}}$ comparison graphs for the thick rainbow, non-perfect fuzzy rainbow, perfect fuzzy rainbow, non-perfect rainbow, and perfect rainbow, at various success rates under the typical situation for $\mathsf{N} = 39$-bits (bottom: $\mathtt{X}_{\mathrm{pc}}$; left: $\mathtt{X}_{\mathrm{atc}}$)

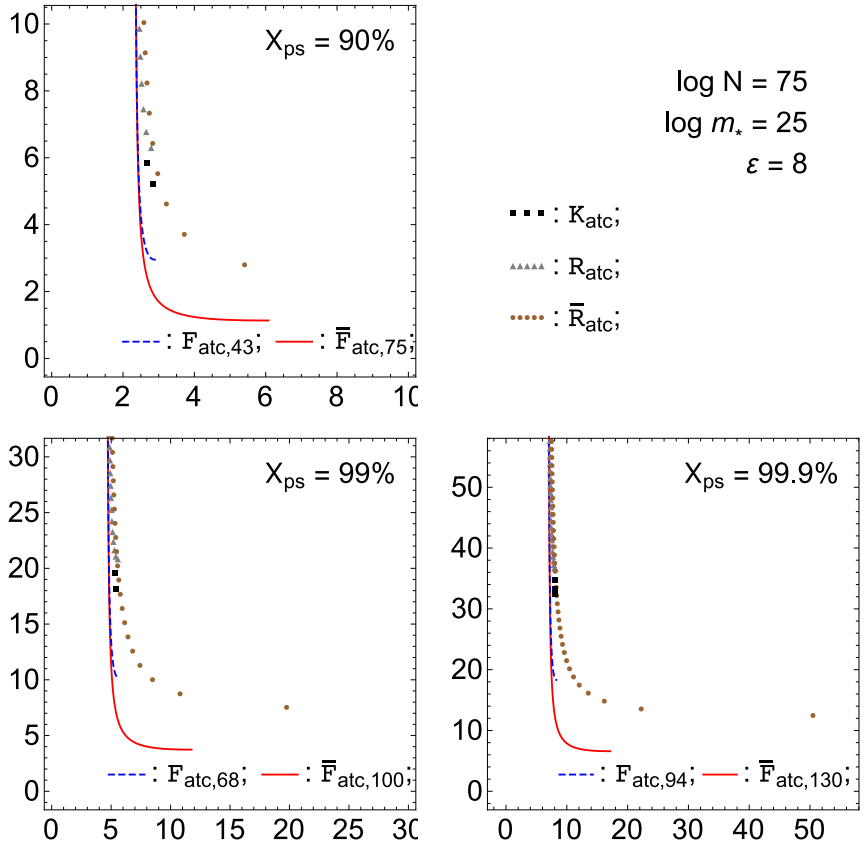

Figure 5.4: The $\mathtt{X}_{\mathrm{pc}}$-$\mathtt{X}_{\mathrm{atc}}$ comparison graphs for the thick rainbow, non-perfect fuzzy rainbow, perfect fuzzy rainbow, non-perfect rainbow, and perfect rainbow, at various success rates under the typical situation for $\mathsf{N} = 75$-bits (bottom: $\mathtt{X}_{\mathrm{pc}}$; left: $\mathtt{X}_{\mathrm{atc}}$)

considered during theoretical analyses of the tradeoff algorithms.

In all the figures, the arcs of the perfect fuzzy rainbow tradeoff are located closer to the lower left corner than those of all the other algorithms, and that means, the perfect fuzzy rainbow method can give higher online efficiency through lower pre-computation cost than any other tradeoff even the original perfect rainbow tradeoff. Thus, the perfect table fuzzy rainbow tradeoff algorithm is always prefer to other tradeoff algorithms.

The performance of the non-perfect fuzzy rainbow tradeoff is not particularly good as much as the perfect version. The non-perfect fuzzy rainbow

tradeoff has always better ability than the thick rainbow tradeoff and the original non-perfect rainbow tradeoff, but it cannot reach the level of a certain high online efficiency of the perfect rainbow tradeoff. Nevertheless, at the same online efficiency levels, the non-perfect fuzzy rainbow tradeoff demands lower cost of pre-computation than the perfect rainbow method.

The thick rainbow tradeoff might be preferable to the non-perfect rainbow tradeoff since the thick rainbow tradeoff can attain better online execution capabilities at similar levels of pre-computation complexity. However, the best capability of the thick rainbow tradeoff is still worse than those of other three algorithms.

As a matter of fact, the curve of the non-perfect fuzzy rainbow tradeoff and that of the perfect version in each figure intersect at a point of low pre-computation level. Hence, the non-perfect fuzzy rainbow can have better execution behavior than the perfect fuzzy rainbow tradeoff, at the same pre-computation cost of certain low pre-computation range. However, the range gives awfully bad online efficiency to be used in practice.

In summary, among the three rainbow variants, we conclude that the perfect table fuzzy rainbow tradeoff is the best algorithm, the non-perfect table version follows next, and the thick rainbow tradeoff algorithm is the worse one.

Chapter 6

Time Memory Data Tradeoff Algorithms

The execution behaviors of some time memory tradeoff algorithms for multi-targets are analyzed in this chapter. We cover the classical Hellman, non-perfect and perfect DP, and non-perfect and perfect fuzzy rainbow tradeoff methods. We will not handle the multi-target versions of the original rainbow tradeoffs and the thick rainbow tradeoff, since they are known as inferior algorithms to the others [3, 5, 6].

6.1 Algorithms

Let us first review the five tradeoff algorithms for single-target setting with focusing on the online phase. Recall the online phases of the single-target algorithms in Algorithm 1 and Algorithm 2 in a roughly manner. Algorithm 1 presents the online phases of the classical Hellman tradeoff and the non-perfect and perfect table DP tradeoffs, and Algorithm 2 presents the online phases of the non-perfect and perfect table fuzzy rainbow tradeoffs.

The details of the inner-most loops of Algorithm 1 and Algorithm 2 are actually more complicated and varies for each tradeoff algorithm. However, they are not important here, and we just focus on the frames of the algorithms. Note that, when setting $s = 1$ in Algorithm 2, we can regard Algo-

Algorithm 1: Online phase of the Hellman and DP tradeoffs for single-target setting.

```
for j = 1 to ℓ do
    generate the online chain for the j-th table;
    resolve alarm if it occurs;
    terminate if the answer is found;
end
```

Algorithm 2: Online phase of the fuzzy rainbow tradeoffs for single-target setting.

```
for i = s to 1 do
    for j = 1 to ℓ do
        generate the online chain for the j-th table that starts from the
        i-color;
        resolve alarm if it occurs;
        terminate if the answer is found;
    end
end
```

rithm 2 identically as Algorithm 1 since the outer-most loop of Algorithm 2 is removed.

Now, let us consider the online phases of the tradeoff algorithms for D multi-targets with Algorithm 3. Algorithm 3 is suitable to the fuzzy rainbow tradeoff algorithms, but when setting $s = 1$ as in the above paragraph, this can be also applicable to the classical Hellman and DP tradeoff algorithms. Setting $D = 1$ in Algorithm 3 essentially removes the inner-most loop and reduces this to Algorithm 2, and to Algorithm 1 with taking $s = 1$. Hence, if Algorithm 3 is regarded as a family of tradeoff algorithms with each value of the parameter D corresponding to a different tradeoff, each single-target tradeoff is then just a particular case of the corresponding family of tradeoff algorithms. In the remainder of this chapter, we will treat only Algorithm 3, and any argument made for Algorithm 3 may be also applicable to the classical Hellman and DP tradeoffs by taking $s = 1$.

Note that there is no such generally accepted practice for the placement of the k-loop in Algorithm 3, the loop corresponding to the inversion targets.

Algorithm 3: Online phase for D multi-target setting.

```
for i = s to 1 do
    for j = 1 to ℓ do
        for k = 1 to D do
            generate the online chain associated with the k-th target for
            the j-th table that starts from the i-color;
            resolve alarm if it occurs;
            terminate if the answer is found;
        end
    end
end
```

However, when the pre-computation tables are too large to be fully loaded into fast memory, it is more advantageous to handle frequent changes of the targets than to access the tables. Thus, our placement given in Algorithm 3 is quite practical.

6.2 Analysis

In this section, we focus on the computational complexity of Algorithm 3 to analyze and compare the performances of the five time memory data tradeoff algorithms. For the classical Hellman and DP tradeoff algorithms, it is implicitly assumed that the parameter $s = 1$ and the i-loop is removed. We use the usual notation for the search space size N, the pre-computation time complexity P, the online time complexity T, and the storage complexity M. For a tradeoff algorithm with D targets, we define the *pre-computation coefficient* of the algorithm to be PD/N and the *tradeoff coefficient* to be TM^2D^2/N^2.

Let us write $\mathtt{M}_i$ to denote the i-th sub-matrix for the j-th pre-computation matrix and $|\mathtt{M}_i|$ to denote the number of distinct points expected in $\mathtt{M}_i$. Note that each value of $|\mathtt{M}_i|$ is a function of the parameters m, t, s, and the color index i, and hence, this is independent from the specific table and the inversion target. One can easily obtain by using this notation that the probability $P_{i,j,k}$ for the inner-most loop with certain indices i, j, and k to operate on

the online phase is

$$P_{i,j,k} = \prod_{h=i+1}^{s} \left(1 - \frac{|\mathtt{M}_h|}{N}\right)^{\ell D} \left(1 - \frac{|\mathtt{M}_i|}{N}\right)^{(j-1)D} \left(1 - \frac{|\mathtt{M}_i|}{N}\right)^{k-1}. \tag{6.2.1}$$

We next use C_i to denote the cost associated with the inner-most loop corresponding to the indices i, j, and k that includes both the cost of online chain creation and the cost of resolving alarms, when assuming the operation of the loop. As the value $|\mathtt{M}_i|$, C_i is also a function of the parameters m, t, s, and the color index i.

Now, we can compute the online time complexity T for Algorithm 3 as

$$\begin{aligned} T &= \sum_{i=1}^{s} \sum_{j=1}^{\ell} \sum_{k=1}^{D} P_{i,j,k} C_i \\ &= \sum_{i=1}^{s} \left\{ \prod_{h=i+1}^{s} \left(1 - \frac{|\mathtt{M}_h|}{N}\right)^{\ell D} \frac{N}{|\mathtt{M}_i|} \left\{1 - \left(1 - \frac{|\mathtt{M}_i|}{N}\right)^{\ell D}\right\} C_i \right\}, \end{aligned}$$

which is a function of the parameters m, t, s, ℓ, and the number of targets D. We can also find that, if the product ℓD is regarded just as one parameter, T could be regarded as a function of the parameters m, t, s, and ℓD.

This finding motivates us to introduce a new notion. Let us fix any one of the classical Hellman, non-perfect DP, perfect DP, non-perfect fuzzy rainbow, and perfect fuzzy rainbow tradeoffs, and consider a set of parameters m, t, s, and ℓ' for this tradeoff with single-target setting. Next, consider the corresponding set of parameters m, t, s, and $\ell = \frac{\ell'}{D}$ for the same tradeoff with D multi-target setting. Now, we refer to these two sets of parameters as *matching parameter sets* for the single-target and multi-target versions of the same tradeoff algorithm. In other words, the matching parameter sets for a single-target tradeoff and the corresponding multi-target tradeoff differ from each other only in their numbers of tables, with the number of tables for the single-target tradeoff being equal to the product of the number of tables and the number of targets for the multi-target tradeoff.

When executed under matching parameter sets, the success probability of a multi-target tradeoff is exactly the same as that of corresponding single-

target tradeoff since

$$1 - \prod_{i=1}^{s} \left(1 - \frac{|\mathtt{M}_i|}{N}\right)^{\ell D} = 1 - \prod_{i=1}^{s} \left(1 - \frac{|\mathtt{M}_i|}{N}\right)^{\ell'}.$$

It is also easily see that, when executed under matching parameter sets, the product of the number of targets D and the pre-computation complexity P of a multi-target tradeoff is equal to the pre-computation complexity of corresponding single-target tradeoff, and so the case of the storage complexity M is, since both complexities are linear in the number of tables. Through the results so far, we can finally arrive at the following theorem.

Theorem 6.2.1. *When executed under matching parameter sets, the online time complexity, the success probability, the pre-computation coefficient and the tradeoff coefficient of a single-target tradeoff and those of the corresponding multi-target tradeoff are equal, respectively.*

Note that storage optimization techniques in Chapter 4 do not rely on the number of targets D. Thus, we can apply the same comparison method of [17, 22] to the multi-target version of the five tradeoff algorithms. In particular, as we have seen in Chapter 5, we can conclude that the perfect table fuzzy rainbow multi-target tradeoff algorithm outperforms the other four algorithms.

Chapter 7

Experiments

Experimental results that support some theoretical findings are given in this chapter. In all cases, the results matched our theory very well. The contents appearing in Section 7.2 and Section 7.3 can be also found in [19] and [20], respectively.

The key to ciphertext mapping, under a randomly generated fixed plaintext, of AES-128 was used as the one-way function for all the tests. Distinct randomly generated plaintexts were used to change the one-way function for independence between multiple tests. Truncations of 128-bit ciphertexts to binary strings of a certain fixed length and zero-extensions of these to 128-bit keys were used to restrict the search space to a manageable size. The reduction functions were set to constant XOR-ing operations, with the binary expression of i used as the i-th color constant. For all the experiments of two fuzzy rainbow tradeoffs, the parameter t was taken to be an integer power of 2, and the distinguishing property was set to check whether $\log t$ least significant bits were zero.

7.1 Thick Rainbow Tradeoff

Our first experiment for the thick rainbow tradeoff algorithm testifies to Lemma 3.1.2. We first fixed the two parameter sets of m, t, s, and N, and for each parameter set, we made ten thick rainbow matrices, each of which

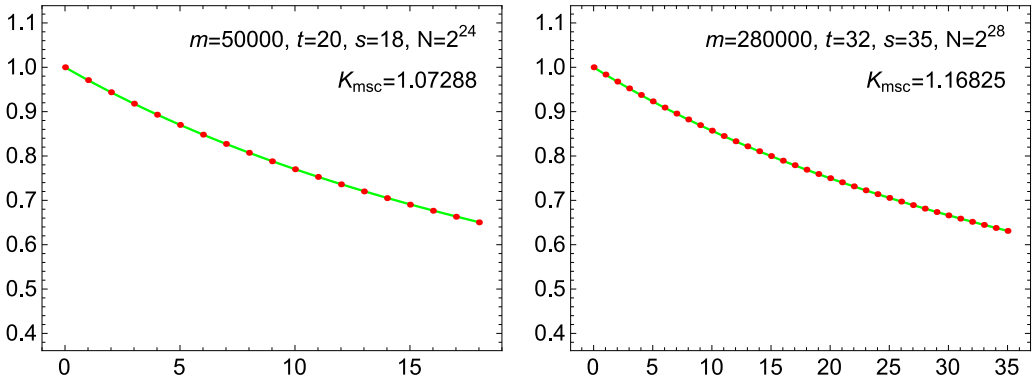

Figure 7.1: Averaged number of i-th color boundary points in a thick rainbow matrix (*arc*: theory; *dots*: test; x-axis: i; y-axis: $\frac{m_i}{m}$)

Table 7.1: Experimental verification of Lemma 3.1.3.

$m = 30000$, $t = 16$, $s = 20$, $\mathsf{N} = 2^{23}$									
i	1	3	5	7	10	13	15	17	20
test $= \lvert\mathtt{HM}_i\rvert$	413011.	393559.	376103.	359736.	337900.	318639.	306857.	296120.	281225.
theory $= \mathtt{K}_{\mathrm{cr},i}mt$	418073.	398149.	380045.	363521.	341272.	321597.	309696.	298647.	283479.
test/theory	0.98789	0.98847	0.98963	0.98959	0.99012	0.99080	0.99083	0.99154	0.99205

were generated from m starting points. During the generation of each matrix, all the numbers of i-th color boundary points were recorded. For each i, the average of ten numbers of i-th color boundary points m_i was computed.

Figure 7.1 gives the test results of our two parameter sets. In each figure, the arc represents the theoretical result of Lemma 3.1.2, and the dots are the averaged m_i, divided by m, from our experiment. Clearly, our theory corresponds with the experiment very well, in spite of only ten repetitions for each test. We can conclude that Lemma 3.1.2 predicts the number of i-th color boundary points very accurately.

Second, we verified Lemma 3.1.3, the coverage rate of the i-th Hellman sub-matrix $\mathtt{HM}_i$ in a single thick rainbow matrix. For a chosen parameter set, we generated 100 thick rainbow matrices, and the number of distinct points $\lvert\mathtt{HM}_i\rvert$ in the sub-matrix $\mathtt{HM}_i$ of each thick rainbow matrix was counted for several selected values of i.

The averaged numbers counted is presented in Table 7.1. These were com-

Table 7.2: Experimental verification of the number of alarm counts for the online chain that starts at j columns away from the column of ending points in $\mathtt{HM}_i$.

$m = 1000000$, $t = 40$, $s = 30$, $\mathtt{N} = 2^{30}$; 200 tables × 5000 online chains per table												
i	1			10			20			30		
j	39	20	1	39	20	1	39	20	1	39	20	1
test	1119014	1100956	1084019	783853	765784	746492	408685	390533	373952	36952	19711	1836
theory	1117587.	1099891.	1082197.	782311.	764616.	746921.	409782.	392087.	374392.	37253.	19558.	1863.
test/theory	1.0013	1.0010	1.0017	1.0020	1.0015	0.9994	0.9973	0.9960	0.9988	0.9919	1.0078	0.9857

pared with the numerical values from our theory, which can be calculated by $\mathtt{K}_{\mathrm{cr},i}mt$. As one can see, our test results are near by our theoretical expectation in error by less than 1.3%.

The third experiment targeted the accuracy of Eq. (3.1.1) in the proof of Lemma 3.1.6. That is, we verified the number of alarm counts

$$\frac{m\big((s-i)t+(j+1)\big)}{\mathtt{N}}$$

for the online chain that starts at j columns away from the column of ending points in the i-th Hellman sub-matrix $\mathtt{HM}_i$ of a single thick rainbow matrix. A parameter set was chosen, and total 200 thick rainbow tables were made. For each table, we generated 5000 online chains for each pair of selected i's and j's, and all the alarms were counted.

The experimental figures for Eq. (3.1.1) are given in Table 7.2. The theoretical numbers listed in Table 7.2 were computed by multiplying the number 200 of created thick rainbow tables and the number 5000 of generated online chains per table to Eq. (3.1.1). We can find that the experimental values are close to the theoretical values and the all errors are less than 1.5%.

7.2 Non-Perfect Table Fuzzy Rainbow Tradeoff

The first test for the non-perfect fuzzy rainbow tradeoff algorithm verifies Lemma 3.2.2. This test lends weight to the validity of Lemma 3.2.3, Propo-

Table 7.3: Experimental verification of Lemma 3.2.2 for a small s.

$m = 2000$, $t = 2^{13}$, $s = 7$, $\mathsf{N} = 2^{40}$								
i	0	1	2	3	4	5	6	7
test	2000.0	1891.4	1791.0	1698.8	1618.8	1548.6	1483.8	1423.8
Eq. (7.2.1)	2000.0	1890.9	1792.8	1704.2	1623.7	1550.4	1483.2	1421.6
Lemma 3.2.2	2000.0	1885.0	1782.4	1690.5	1607.5	1532.4	1463.9	1401.3
test/Eq. (7.2.1)	1.0000	1.0003	0.9990	0.9968	0.9970	0.9989	1.0004	1.0016
test/Lemma 3.2.2	1.0000	1.0034	1.0048	1.0049	1.0070	1.0106	1.0136	1.0161

sition 3.2.4, and Lemma 3.2.7.

After fixing parameters m, t, s, and the space size N, we generated a matrix with m distinct starting points. In the beginning, the first DP sub-matrix $\mathtt{DM}_1$ was initially generated. After sorting these color boundary DPs, we discarded any duplicates of the DPs, and the number of remainder was recorded. Next, we applied the same work to the second DP sub-matrix $\mathtt{DM}_2$, and the number of remaining set of color boundary DPs was recorded. The same processes were continued for s colors.

A small number of the chains that did not reach a DP within the suitably large chain length bound of $15t$ were discarded during each process. We created ten fuzzy rainbow matrices, and the number of each i-th color boundary points was averaged separately.

Test results for the small $s = 7$ case are given in Table 7.3. It also verifies that the results is very close to the original theoretical values given by the iterative formula

$$m_{i+1} = \frac{2m_i}{1 + \sqrt{1 + 2^{\frac{\mathrm{F}_{\mathrm{msc}}}{s}} \frac{m_i}{m}}} \quad \text{with} \quad m_0 = m. \tag{7.2.1}$$

The numerical values, which come out from the closed-form formula for m_i given by Lemma 3.2.2, are slightly further away from the experimental results, but all the errors are by less than 2% at the worst.

The results of tests where more practical values of s are used are given in Figure 7.2. In order to present the larger data set for the larger s more effectively, we summarized the results as graphs. The lines of two graphs

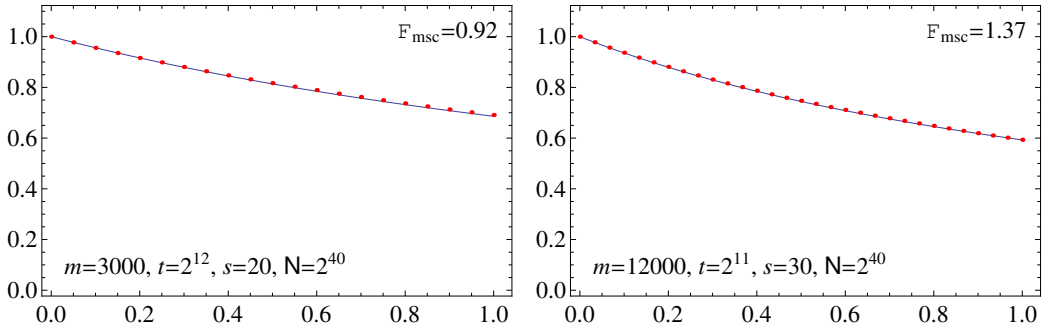

Figure 7.2: Number of i-th color boundary points in a non-perfect fuzzy rainbow matrix (*line*: theory; *dots*: test; x-axis: $\frac{i}{s}$; y-axis: $\frac{m_i}{m}$)

represent the theory given by Lemma 3.2.2 and the dots correspond to the experimental data. Each dot gives the number of the i-th color boundary points, which is averaged over ten tests. In spite of few test repetitions, it is evident that the results match Lemma 3.2.2 very well.

Our second experiment is designed to verify an argument concerning the cost of resolving alarms, which is a technical claim made during the proof of Lemma 3.2.9. That is the probability for an online chain that starts from the i-th color to merge with a pre-computation chain strictly after the i-th color as $(s-i)\frac{\mathtt{F}_{\mathrm{msc}}}{ms} = (s-i)\frac{t^2}{\mathsf{N}}$.

The experimental proof of this claim was performed as follows. Multiple pre-computation chains were generated, and their color boundary DPs were recorded. Since these chains were regarded as individual pre-computation chains rather than as members of a pre-computation matrix, no sorting was done. After fixing a starting color index i, online chains were generated to the ending points. The first DP of the online chain, i.e., the DP that ends the i-th colored sub-chain, in addition to the terminating DP, were recorded. For each online chain, we did a linear search over the collection of pre-computation chains for matching ending points. Whenever a collision was found, we compared the first DP of the online chain against the corresponding color boundary DP of the colliding pre-computation chain to check whether the merge occur within the i-th color sub-chain or strictly after the i-th color, and occurrences of only the latter type of collisions were counted. Online chains

Table 7.4: Tests for the probability of merge between an online chain that starts from the i-th color and a pre-computation chain. For each test, the number of merges at colors strictly after the i-th color are listed.

$t = 2^{12}$, $s = 20$, $\mathsf{N} = 2^{40}$, # of pre-computation chains = 500000							
i		1	4	8	12	16	19
# of online chains		26315	31250	41666	62500	125000	500000
# of merges	test	3753142	3763624	3795469	3841206	3847142	3728423
	theory	3814583	3814697	3814636	3814697	3814697	3814697
test/theory		0.9839	0.9866	0.9950	1.0069	1.0085	0.9774

$t = 2^{11}$, $s = 30$, $\mathsf{N} = 2^{40}$, # of pre-computation chains = 500000							
i		1	6	12	18	24	29
# of online chains		17241	20833	27777	41666	83333	500000
# of merges	test	966734	954600	948527	964926	979533	948478
	theory	953653	953659	953648	953659	953671	953674
test/theory		1.0137	1.0010	0.9946	1.0118	1.0271	0.9946

that start at different color indices were tested for merges against a common large collection of pre-computation chains.

Since we do not need the role of matrix stopping rule in this test, we were free to use any large number of pre-computation chains, and hence, test repetitions were not needed. Two sets of tests, corresponding to different parameter choices, were executed.

The results are given in Table 7.4. The theoretical counts of merges in the table were computed by multiplying the number of pre-computation chains and the number of online chains generated for the index i to the probability $(s-i)\frac{t^2}{\mathsf{N}}$. Larger number of chains were used when testing shorter online chains, since merges are seen less often with these chains and a smaller number of merges would bring about unreliable data. The experimentally obtained merge counts for the two sets of tests are close to our theoretical findings.

7.3 Perfect Table Fuzzy Rainbow Tradeoff

There are four separate experiments that support the theoretical predictions for the perfect fuzzy rainbow tradeoff algorithm in this section. The first experiment gives the accuracy level of the approximation claimed by Lemma 3.3.1. The subsequent two experiments verify the correctness of two of our logical arguments that lie hidden within the proofs of technical lemmas. The final experiment verifies that our claim of online computational complexity is correct and serves as an overall checkup of our theory.

Let us first check the accuracy of Lemma 3.3.1 through experiments. Recall that Lemma 3.3.1 is equivalent to Lemma 3.2.2, which deals with the non-perfect fuzzy rainbow tradeoff. Note that we have already confirmed the correctness of Lemma 3.2.2 through tests in the previous section, even for small s values. However, the parameter sets used there are such that the $\mathsf{F}_{\mathrm{msc}} = \frac{m_0 t^2 s}{\mathsf{N}}$ values are 0.92 and 1.37, which correspond to $\bar{\mathsf{F}}_{\mathrm{msc}}$ values of 0.63 and 0.81, respectively. Table 5.2 implies that parameter sets for the perfect version should belong to the rough range $1.5 \leq \bar{\mathsf{F}}_{\mathrm{msc}} \leq 1.8$. Hence, we cannot depend on the previous test results to claim that Lemma 3.3.1 is sufficiently accurate to use with the perfect table case analysis.

Two of the test results are given by Figure 7.3. After fixing parameters m, t, s, and the space size N, we computed m_0 through Lemma 3.3.1, and generated the fuzzy rainbow matrix from m_0 starting points. A small number of chains that did not reach a DP within the chain length bound of $15t$, at various colors, were discarded without replacing them with new chains. We generated ten fuzzy rainbow matrices for each parameter set, and the numbers of i-th color boundary points were recorded and averaged separately for each i.

The lines of four boxes in Figure 7.3 represent the theory given by Lemma 3.3.1 and the dots correspond to the tested data. Each dot gives the number of the i-th color boundary points, which is averaged over ten tests.

The two left-hand side boxes of Figure 7.3 give the test results for the worst parameter set we had experimented. This is the most inaccurate situation as compared with our theory, but even in this case, it is shown in

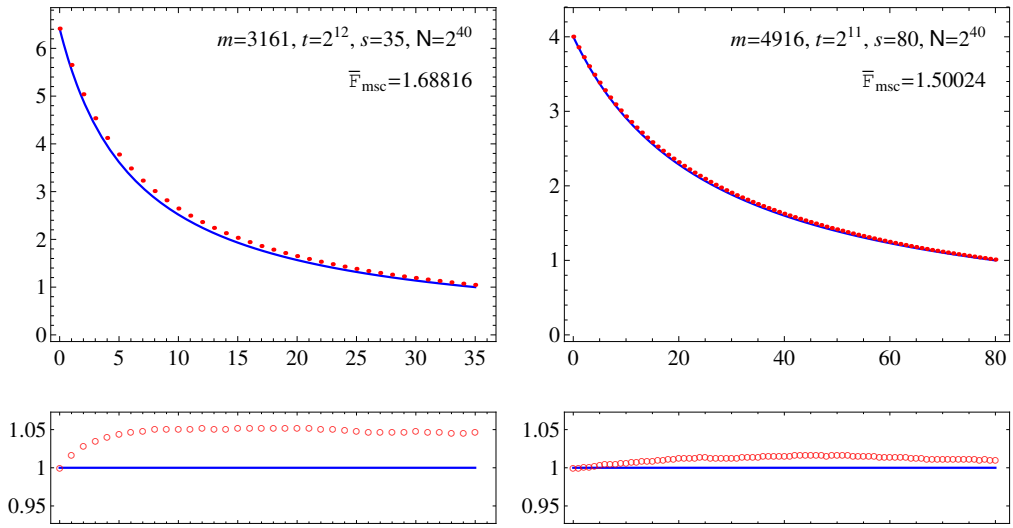

Figure 7.3: Number of i-th color boundary points in a fuzzy rainbow matrix (*line*: theory; *dots*: test; *circles*: test/theory; x-axis: i; y-axis: $\frac{m_i}{m}$ and test/theory ratio)

the bottom box that the largest inaccuracy of Lemma 3.3.1 is by approximately 5%. The two right-hand side boxes give the test results for a parameter set whose $(s, \bar{\mathtt{F}}_{\mathrm{msc}})$ pair does not necessarily derive optimal efficiency for any success rate. The right-hand side boxes present the situation that one would be experiencing in practice, since the considerations concerning precomputation cost will make parameter sets of sub-optimal online efficiency more practical. Although not perfectly accurate, the results match our theory practically well, and hence, we may conclude that the approximation in Lemma 3.3.1 is sufficiently accurate to use in real world.

The second experiment concerns a crucial argument that the selection process of DP sub-chains $\overline{\mathtt{DM}}_i$ from among the DP sub-chains $\widetilde{\mathtt{DM}}_i$ is irrelevant to the lengths of the DP sub-chains in $\widetilde{\mathtt{DM}}_i$. This argument is equivalent to the equation

$$\frac{|\widetilde{\mathtt{DM}}_i|}{m_i t} = \frac{|\overline{\mathtt{DM}}_i|}{m_s t} \tag{7.3.1}$$

that already appeared in Eq. (3.3.8). This also allows us to claim in the proof of Lemma 3.3.8 that it requires $(\bar{\mathtt{F}}_{\mathrm{cr},1} + \cdots + \bar{\mathtt{F}}_{\mathrm{cr},i})t$ iterations of the one-way

Table 7.5: Experimental confirmation that selection of $\overline{\mathtt{DM}}_i$ from $\widetilde{\mathtt{DM}}_i$ does not disturb average chain length.

$m_0 = 14000$, $t = 2^{13}$, $s = 8$, $\mathsf{N} = 2^{40}$; $m_s = 3604$, $\frac{m_s t^2 s}{\mathsf{N}} = 1.7598$								
i	1	2	3	4	5	6	7	8
$\frac{m_i}{m_s}$	2.9270	2.3391	1.9370	1.6338	1.4159	1.2461	1.1105	1.0000
$\frac{\lvert\widetilde{\mathtt{DM}}_i\rvert}{m_i t}$	0.8649	0.8895	0.9090	0.9157	0.9293	0.9370	0.9446	0.9550
$\frac{\lvert\overline{\mathtt{DM}}_i\rvert}{m_s t}$	0.8698	0.8923	0.9130	0.9144	0.9289	0.9369	0.9455	0.9550
$\frac{\lvert\widetilde{\mathtt{DM}}_i\rvert}{m_i t} / \frac{\lvert\overline{\mathtt{DM}}_i\rvert}{m_s t}$	0.9947	0.9970	0.9956	1.0014	1.0005	1.0002	0.9991	1.0000

$m_0 = 40000$, $t = 2^{12}$, $s = 15$, $\mathsf{N} = 2^{40}$; $m_s = 7821$, $\frac{m_s t^2 s}{\mathsf{N}} = 1.7901$								
i	1	3	5	7	9	11	13	15
$\frac{m_i}{m_s}$	4.1159	2.8944	2.2258	1.8053	1.5029	1.2921	1.1257	1.0000
$\frac{\lvert\widetilde{\mathtt{DM}}_i\rvert}{m_i t}$	0.8942	0.9218	0.9376	0.9472	0.9567	0.9615	0.9634	0.9702
$\frac{\lvert\overline{\mathtt{DM}}_i\rvert}{m_s t}$	0.8916	0.9223	0.9373	0.9445	0.9575	0.9604	0.9630	0.9702
$\frac{\lvert\widetilde{\mathtt{DM}}_i\rvert}{m_i t} / \frac{\lvert\overline{\mathtt{DM}}_i\rvert}{m_s t}$	1.0029	0.9995	1.0004	1.0029	0.9992	1.0011	1.0004	1.0000

function to regenerate a pre-computation chain up to the i-th color. This is a new argument that had not appeared in any of the previous works, and it would be necessary to verify the claim experimentally.

From the formula of Lemma 3.3.2, we can expect each value of both sides of Eq. (7.3.1) to be of $1-\Theta\left(\frac{1}{4s}\right)$ order, implying that there are more dropped chains within each DP sub-matrix for smaller s values. Thus, to increase the chances of discovering possible errors in testing Eq. (7.3.1), one would choose parameters with not only large ratios of $\frac{m_i}{m_s}$ but also small s values. Since the formula of Lemma 3.3.2 is more accurate for larger s values of interest, it is more appropriate to experiment our claim directly through Eq. (7.3.1), rather than through the formula of Lemma 3.3.2.

The results for $s = 8$ and $s = 15$ values are given in Table 7.5. The other parameters m_0 and t were chosen so that the corresponding experimentally obtained $\bar{\mathsf{F}}_{\mathrm{msc}} = \frac{m_s t^2 s}{\mathsf{N}}$ values would be large with a large ratio of $\frac{m_0}{m_s}$, while still falling within our range of interest. Each test was repeated ten times, and all figures in Table 7.5 including the m_s and $\frac{m_s t^2 s}{\mathsf{N}}$ values, except the

Table 7.6: Experimental verification of theoretically obtained expected cost of resolving a possible alarm associated with an online chain that starts from the i-th color.

$m_0 = 32000$, $t = 2^{10}$, $s = 70$, $\mathsf{N} = 2^{38}$; $\mathrm{Avg}(m_s) = 6179.5$, $\frac{\mathrm{Avg}(m_s)\, t^2 s}{\mathsf{N}} = 1.6501$								
i	1	10	20	30	40	50	60	70
test	981.91	9144.6	16700.	21868.	23781.	21544.	14513.	1653.9
Eq. (3.3.10)	975.69	9094.8	16575.	21636.	23470.	21279.	14263.	1626.6
Eq. (7.3.2)&(7.3.3)	978.26	9136.9	16696.	21850.	23759.	21588.	14499.	1656.6
test/(3.3.10)	1.0064	1.0055	1.0076	1.0107	1.0132	1.0125	1.0176	1.0168
test/(7.3.2)&(7.3.3)	1.0037	1.0008	1.0002	1.0008	1.0009	0.9980	1.0010	0.9984

$m_0 = 27000$, $t = 2^{11}$, $s = 100$, $\mathsf{N} = 2^{40}$; $\mathrm{Avg}(m_s) = 4461.4$, $\frac{\mathrm{Avg}(m_s)\, t^2 s}{\mathsf{N}} = 1.7019$								
i	1	10	20	30	50	70	90	100
test	1963.2	18961.	36171.	50616.	67819.	64260.	32306.	3462.6
Eq. (3.3.10)	1970.9	18878.	35910.	50239.	67358.	63441.	31745.	3377.4
Eq. (7.3.2)&(7.3.3)	1974.3	18932.	36069.	50539.	67955.	64167.	32181.	3427.2
test/(3.3.10)	0.9961	1.0044	1.0073	1.0075	1.0069	1.0129	1.0177	1.0252
test/(7.3.2)&(7.3.3)	0.9944	1.0016	1.0028	1.0015	0.9980	1.0014	1.0039	1.0103

values of parameters m_0, t, s, and N, are averaged over ten repetitions. In particular, the stated $\frac{|\widetilde{\mathtt{DM}}_i|}{m_i t} / \frac{|\widetilde{\mathtt{DM}}_i|}{m_s t}$ values are averages and not the simple ratios of the two averages appearing above these values. The experimental data shows that Eq. (7.3.1) is correct strictly.

Third experiment is designed to verify the accuracy of Eq. (3.3.10), which means the cost of dealing with a possible alarm that may be induced by an online chain generated from the i-th color. This was of particular interest because the formula depended on Lemma 3.3.9, which was stated as an approximation. This verification would also raise our confidence level in the extremely delicate random function arguments during the proof of Lemma 3.3.8.

We give the results of third experiment for two separate parameter sets in Table 7.6. We generated 50 pre-computation tables from the specified m_0-many starting points for each set. For each pre-computation table and each fixed starting color i, we generated sufficiently many online chains so as to observe approximately 2000 merges. The test results are in very good matching up with Eq. (3.3.10).

We also added another row of theoretical predictions for the alarm cost

in Table 7.6. Tracing back through the proofs of Lemma 3.3.7, Lemma 3.3.8, and Lemma 3.3.9, and referring to Lemma 3.3.1 and Lemma 3.3.2, it can be easily seen that Eq. (3.3.10) is a simplified expression for

$$t\left\{\left(\sum_{k=1}^{i}\frac{2}{\frac{m_k t^2}{\mathsf{N}}}\ln\left(1+\frac{\frac{m_k t^2}{\mathsf{N}}}{2}\right)\right)\left(1-\prod_{k=i}^{s}\frac{1}{1+\frac{m_k t^2}{\mathsf{N}}}\right)+\left(1-\frac{1}{1+\frac{m_i t^2}{\mathsf{N}}}\right)\frac{m_s}{m_i}\right\}. \tag{7.3.2}$$

Since the formula of Lemma 3.3.1 is an approximated formula from the iterative formula

$$m_{i+1}=m_i\frac{2}{1+\sqrt{1+\frac{2m_i t^2}{\mathsf{N}}}}, \tag{7.3.3}$$

the theoretical predictions through Eq. (7.3.2) and Eq. (7.3.3) could lead higher accuracy level. Certainly, we can see in Table 7.6 that the theoretical predictions through Eq. (7.3.2) and Eq. (7.3.3) are even more accurate with the experiment data than the predictions of Eq. (3.3.10).

Our final test for the perfect fuzzy rainbow tradeoff is for the verification of Eq. (3.3.11), which gives the online time complexity of the algorithm. Since this experiment is meant to be an overall sanity check of the theory for the perfect table fuzzy rainbow tradeoff algorithm, practical s values were used.

After choosing each parameter set m, t, s, and ℓ, we computed m_0 from Lemma 3.3.1 and created ℓ pre-computation tables from m_0 starting points on each pre-computation phase. We next generated 10000 random inversion targets and executed the online phase. Experimental results corresponding to three separate parameter sets are given in Table 7.7. During both the pre-computation and the online phases, any chain that did not reach a DP within the chain length bound of $15t$, at each color, was discarded. The cost associated with such discarded chains is included in the tested online complexities.

In Test-1, noting that $\log m = 21.0$, we truncated each ending point to the length of $\log|\mathrm{ep}| = 26$ bits with allowing for extra 5.0 bits of information. A small fraction of the ending points were made identical to each other, and we carefully processed all the identical truncated ending points during the online phase. Since it is reasonable for the $\log m = 21.0$ situation to remove

Table 7.7: Experimental verification of the online time complexity for the perfect table fuzzy rainbow tradeoff algorithm given by Eq. (3.3.11).

$\mathsf{N} = 2^{40}$	m	t	s	ℓ	$\bar{\mathsf{F}}_{\mathrm{msc}}$	m_0	$\log m_0$	$\log m$	$\log m_\star$
Test-1	2078517	2^7	56	173	1.73445	15654415	23.9	21.0	15.2
Test-2	489178	2^8	56	477	1.63281	2664424	21.3	18.9	13.1
Test-3	1258291	2^7	80	396	1.50000	5033162	22.3	20.3	13.9

	theory		test				reasonable ε
	$\bar{\mathsf{F}}_{\mathrm{ps}}$	$T/t\ell s$	$\log\lvert\mathrm{sp}\rvert$	$\log\lvert\mathrm{ep}\rvert$	$\bar{\mathsf{F}}_{\mathrm{ps}}$	$T/t\ell s$	
Test-1	0.9002	12.3627	24	26	0.9048	12.6569	$8 = 3.0 + 5.0$
Test-2	0.9501	9.0359	22	24	0.9571	9.1363	$8 = 2.9 + 5.1$
Test-3	0.9900	6.8393	23	28	0.9907	6.8207	$11 = 3.3 + 7.7$

18 bits for the index table technique, it would allow each ending point to be recorded in $\varepsilon = 3.0 + 5.0 = 8$ bits. However, in order to implement easier, we did not do that, and just allocated 3 bytes to the starting points and 4 bytes to the ending points. Since $\log m_\star = \log \frac{m}{s} = 15.2$, we are handling the $\log m_\star + \varepsilon \approx 23$ situation, and we can check through Table 5.2 that Test-1 used a parameter set for the 90% success rate is close to optimal in the light of the online phase.

Test-2 implementation stored 24 bits of each ending point, and when we use a 16-bit index, this situation would correspond to $\log m_\star + \varepsilon = 13.1 + 24 - 16 = 21.1$. We used $s = 56$ which is close to the optimal value of s for this situation to reach a 95% success rate. However, we selected the parameters m and t so that the $\bar{\mathsf{F}}_{\mathrm{msc}}$ value is smaller than what is optimal for the online phase in this situation. Thus, the parameter set of Test-2 may be more implementable in view of lower pre-computation cost.

The parameter set for Test-3 was also picked to be more practical than to be optimal for the online phase. One difference with Test-2 is that we truncated the ending points to be a slightly longer length. It could be checked that our theory of the cost of dealing with alarms was more accurate for this condition than Test-1 and Test-2.

The experimental results and our theory for all the three tests in Table 7.7 fit together well. In practice, since the average m_i values from the

tests must be slightly larger than the theory of Lemma 3.3.1, it brings about a higher success rate than expected. While the higher success rate reduces the cost of creating the online chains, the application of ending point truncation method increases the cost of dealing with alarms. The results in Table 7.7 has been already affected by the combination of the two opposite effects. We have verified through separate computations that substituting Eq. (7.3.3) for Lemma 3.3.1 derives even better predictions of at least the costs of generating the online chains. Therefore, the small discrepancies between theory and test results in Table 7.7 are due to the accuracy limitations of Lemma 3.3.1 rather than to oversights in any other theoretical arguments.

7.4 Time Memory Data Tradeoff Algorithms

An outcome of Theorem 6.2.1 is that the explicit formulas for the online time complexities of the single-target tradeoffs can easily be understood to be those of the corresponding multi-target tradeoffs. It suffices to replace every table count ℓ appearing in those formulas with the product of table count and target count being used by the multi-target method.

We have tested the correctness of the above claim for the non-perfect DP and the perfect fuzzy rainbow tradeoff methods. For each of the two tradeoff methods, we experimented two separate parameter sets, with both small and large target sets, comparing the experimentally obtained online time complexities with the multi-target versions of theoretical online time complexities that were stated by [17] and [20] for the single-target setting. Each test used a parameter set for a fixed success rate requirement that is close to optimal in view of the online phase. During both the pre-computation and the online phases, we discarded any chain that did not reach a DP within the chain length bound of $15t$. The computational cost associated with these discarded chains is included in the tested online complexities.

The multi-target versions of the online time complexities for the non-

Table 7.8: Experimental verification for the multi-target version of the online time complexity.

Non-perfect table DP tradeoff						
	N	$m(=m_0)$	t	$\mathtt{D}_{\mathrm{msc}}$	ℓ	D
Test-1	40 bits	147325	2^{11}	0.562	645	32
Test-2	48 bits	9428796	2^{12}	0.562	5	9490

	test		theory		test/theory	
	$\mathtt{D}_{\mathrm{ps}}$	T/t^2	$\mathtt{D}_{\mathrm{ps}}$	T/t^2	$\mathtt{D}_{\mathrm{ps}}$	T/t^2
Test-1	0.9890	4.6354	0.9900	4.5973	0.9990	1.0083
Test-2	0.9944	4.6207	0.9950	4.6205	0.9994	1.0000

Perfect table fuzzy rainbow tradeoff								
	N	m_0	m	t	s	$\bar{\mathtt{F}}_{\mathrm{msc}}$	ℓ	D
Test-3	39 bits	5575647	808190	2^7	71	1.7101	175	2
Test-4	42 bits	163001257	26455104	2^6	68	1.6754	3	69

	test		theory		test/theory	
	$\bar{\mathtt{F}}_{\mathrm{ps}}$	$T/ts\ell D$	$\bar{\mathtt{F}}_{\mathrm{ps}}$	$T/ts\ell D$	$\bar{\mathtt{F}}_{\mathrm{ps}}$	$T/ts\ell D$
Test-3	0.9907	6.3688	0.9900	6.3915	1.0007	0.9964
Test-4	0.9942	4.9625	0.9950	4.9195	0.9992	1.0087

perfect DP tradeoff and the perfect fuzzy rainbow tradeoff are

$$T_{\mathtt{D}} = (1+2\mathtt{D}_{\mathrm{msc}})t\sum_{i=1}^{\ell D}\big(1-\frac{\mathtt{D}_{\mathrm{cr}}mt}{\mathsf{N}}\big)^{i-1} = \frac{\mathtt{D}_{\mathrm{ps}}(1+2\mathtt{D}_{\mathrm{msc}})}{\mathtt{D}_{\mathrm{cr}}\mathtt{D}_{\mathrm{msc}}}t^2$$

and

$$T_{\bar{\mathtt{F}}} = t\ell D\sum_{i=1}^{s}\big(1-\bar{\mathtt{F}}_{\mathrm{ps}}\big)^{\frac{\sum_{k=i+1}^{s}\bar{\mathtt{F}}_{\mathrm{cr},k}}{s\bar{\mathtt{F}}_{\mathrm{cr}}}}\left(\begin{array}{l}(s-i+1)+\big(1-\frac{\bar{\mathtt{f}}_{i+2}}{\bar{\mathtt{f}}_i}\big)\frac{\bar{\mathtt{f}}_s}{\bar{\mathtt{f}}_i}\\ +\Big(1-\frac{\bar{\mathtt{f}}_{s+1}}{\bar{\mathtt{f}}_i}\frac{\bar{\mathtt{f}}_{s+2}}{\bar{\mathtt{f}}_{i+1}}\Big)\Big(\sum_{k=1}^{i}\bar{\mathtt{F}}_{\mathrm{cr},k}\Big)\end{array}\right),$$

respectively. The results are displayed in Table 7.8, and for all cases, the experimental results tie in with the theory.

Chapter 8

Conclusion

In this thesis, we analyzed the online capabilities of three rainbow variants, which are the thick rainbow, the non-perfect fuzzy rainbow, and the perfect fuzzy rainbow tradeoff algorithms. We adopted some existing results from [15, 17, 22, 23] to obtain the computational complexities. The accurate online time complexity of each tradeoff, which includes the full cost of treating false alarms, was also obtained during this process, and some of our technical arguments were experimentally verified. We offered the time memory tradeoff curves for the three algorithms together with their corresponding tradeoff coefficients.

We also discussed the efficient utilization of memory space. Several techniques [1, 5, 8, 9, 27] for optimizing the storage of the rainbow variants were considered. In particular, we computed the additional cost of dealing with alarms that are related the truncation method of reducing ending point storage [5, 8]. As a result, we obtained the number of bits required by each tradeoff algorithm to store each starting and ending point pair in the pre-computation table.

We compared the three variants of the rainbow tradeoff, using the aforementioned material. We followed the method of [17, 22] that takes both the cost of pre-computation and the online execution complexity into consideration. The original versions of non-perfect and perfect rainbow method were also compared against the three rainbow variants, since they had the best

performance among the well known algorithms. The comparison revealed that the perfect fuzzy rainbow method is superior to the other tradeoffs. Roughly speaking, performance of the non-perfect fuzzy rainbow tradeoff is better than that of the thick rainbow tradeoff. However, neither of these two algorihms can achieve the online efficiency of the original perfect rainbow method.

The multi-target versions of the classical Hellman, the non-perfect and perfect DP, the non-perfect and perfect fuzzy rainbow tradeoffs were also analyzed. We demonstrated that there is virtual no difference between the multi-target setting and the corresponding single-target setting in their complexities and coefficients. Thus, the comparisons made among the single-target tradeoffs are still valid for the the multi-target tradeoffs, and hence, the perfect fuzzy rainbow tradeoff is the best tradeoff algorithm even as a time memory data tradeoff method.

Bibliography

[1] G. Avoine, P. Junod, P. Oechslin, Characterization and improvement of time-memory trade-off based on perfect tables. *ACM Trans. Inform. Syst. Secur.*, **11**(4), 17:1–17:22 (2008). Preliminary version presented at INDOCRYPT 2005.

[2] S. H. Babbage, Improved "exhaustive search" attacks on stream ciphers. In *European Convention on Security and Detection*, IEE Conference Publication, No. 408 (IEE, London, 1995), pp. 161–166.

[3] E. P. Barkan, *Cryptanalysis of Ciphers and Protocols*. Ph.D. Thesis, Technion—Israel Institute of Technology, March 2006.

[4] E. Barkan, E. Biham, N. Keller, Instant ciphertext-only cryptanalysis of GSM encrypted communication. *Advances in Cryptology—CRYPTO 2003*, LNCS **2729**, (Springer, 2003), pp. 600–616.

[5] E. Barkan, E. Biham, A. Shamir, Rigorous bounds on cryptanalytic time/memory tradeoffs. In *Advances in Cryptology—CRYPTO 2006*, LNCS **4117**, (Springer, 2006), pp. 1–21.

[6] A. Biryukov, S. Mukhopadhyay, P. Sarkar, Improved time-memory trade-offs with multiple data. In *SAC 2005*, LNCS **3897**, (Springer-Verlag, 2006), pp. 110–127.

[7] A. Biryukov, A. Shamir, Cryptanalytic time/memory/data tradeoffs for stream ciphers, In *Advances in Cryptology—ASIACRYPT 2000*. LNCS **1976**, (Springer, 2000), pp. 1–13.

[8] A. Biryukov, A. Shamir, D. Wagner, Real time cryptanalysis of A5/1 on a PC. In *FSE 2000*, LNCS **1978**, (Springer, 2001), pp. 1–18.

[9] J. Borst, *Block Ciphers: Design, Analysis, and Side-Channel Analysis*. Ph.D. Thesis, Katholieke Universiteit Leuven, September 2001.

[10] J. Borst, B. Preneel, J. Vandewalle, On the time-memory tradeoff betweeen exhaustive key search and table precomputation. In *Proceedings of the 19th Symposium on Information Theory in the Benelux*, WIC, 1998.

[11] D. E. Denning, *Cryptography and Data Security* (Addison-Wesley, 1982).

[12] J. Dj. Golić, Cryptanalysis of alleged A5 stream cipher. In *Advances in Cryptology—EUROCRYPT '97*, LNCS **1233**, (Springer, 1997), pp. 239–255.

[13] M. Haghighi, M. Dakhilalian, A practical time complexity analysis of fuzzy rainbow tradeoff. In *Information Security and Cryptology* (ISCISC), 2014 11th International ISC Conference.

[14] M. E. Hellman, A cryptanalytic time-memory trade-off. *IEEE Trans. on Infor. Theory*, **26**, (1980), pp. 401–406.

[15] J. Hong, The cost of false alarms in hellman and rainbow tradeoffs. Des. Codes Cryptogr., **57**, (2010), pp. 293–327.

[16] J. Hong, G. W. Lee, D. Ma, Analysis of the parallel distinguished point tradeoff. In *Progress in Cryptogloy—INDOCRYPT 2011*, LNCS **7107**, (Springer, 2011), pp. 161–180.

[17] J. Hong, S. Moon, A comparison of cryptanalytic tradeoff algorithms. *J. Cryptology*, **26**(4), (2013), pp. 559–637.

[18] B.-I. Kim, J. Hong, Analysis of the non-perfect table fuzzy rainbow tradeoff. In *ACISP 2013*, LNCS **7959**, (Springer, 2013), pp. 347–362.

[19] B.-I. Kim, J. Hong, Analysis of the non-perfect table fuzzy rainbow tradeoff. Cryptology ePrint Archive, Report 2012/612.

[20] B.-I. Kim, J. Hong, Analysis of the perfect table fuzzy rainbow tradeoff. *J. Appl. Math.*, **2014**, (2014), Article ID 765394.

[21] J. W. Kim, J. Hong, K. Park, Analysis of the rainbow tradeoff algorithm used in practice. Cryptology ePrint Archive, Report 2013/591.

[22] G. W. Lee, J. Hong, A comparison of perfect table cryptanalytic tradeoff algorithms. To appear in a future volume of *Des. Codes Cryptogr.*, Online July 2015. `http://dx.doi.org/10.1007/s10623-015-0116-0`

[23] D. Ma, J. Hong, Success probability of the hellman trade-off. Inf. Process. Lett., **109**(7), (2009), pp 347–351.

[24] K. Nohl, Attacking Phone Privacy. Presented at *Black Hat USA 2010*, Las Vegas, July 2010.

[25] K. Nohl, C. Paget, GSM-SRSLY? Presented at *26th Chaos Communication Congress* (26C3), Berlin, December 2009.

[26] P. Oechslin, Making a faster cryptanalytic time-memory trade-off. In *Advances in Cryptology—CRYPTO 2003*, LNCS **2729**, (Springer, 2003) pp. 617–630.

[27] F.-X. Standaert, G. Rouvroy, J.-J. Quisquater, J.-D. Legat, A time-memory tradeoff using distinguished points: new analysis & FPGA results. In *Cryptographic Hardware and Embedded Systems—CHES 2002*, LNCS **2523**, (Springer, 2003), pp. 593–609.

[28] F. van den Broek, E. Poll, A comparison of time-memory trade-off attacks on stream ciphers. In *Progress in Cryptology—AFRICACRYPT 2013*, LNCS **7918**, (Springer, 2013), pp. 406–423.

국문초록

암호 해독적 시간 저장 공간 절충기법은 단방향 함수의 역상을 구하는 도구로, 솔트 없이 생성된 패스워드 해시로부터 패스워드를 복원하는 데에 사용된다. 공공연히 알려져 있는 여러 절충기법 알고리즘 중에서, 적어도 구현자들에게는 가장 좋은 알고리즘으로 널리 여겨지고 있는 무지개 절충기법이 현재까지 가장 대중적인 알고리즘이다.

본 학위논문에서는 짙은 무지개 절충기법, 중복이 제거되지 않은 테이블을 이용한 탁한 무지개 절충기법, 그리고 중복이 제거된 테이블을 이용한 탁한 무지개 절충기법의 정확한 수행복잡도를 분석한다. 이 절충기법들은 모두 그동안 많은 주목을 받아오지 못했던 알고리즘들이다. 분석의 결과로, 사전계산 비용과 온라인 수행복잡도를 모두 고려할 경우, 중복이 제거된 테이블을 이용한 탁한 무지개 절충기법이 세 개의 절충기법 중 가장 좋은 성능을 보이며, 기존의 무지개 절충기법 보다도 훨씬 뛰어나다는 것을 알 수 있었다.

또한, 몇 가지 시간 저장 공간 정보 절충기법(다중 표적 절충기법)들의 수행복잡도도 분석하였다. 본 학위논문에서 다루는 다중 표적 절충기법들은 Hellman의 기법, 특이점 절충기법, 탁한 무지개 절충기법 등으로, 특이점 절충기법과 탁한 무지개 절충기법은 중복이 제거되지 않은 테이블을 이용한 방법과 중복이 제거된 테이블을 이용한 방법을 모두 다루었다. 결과적으로 다중 표적 절충기법의 수행복잡도는 적절한 파라미터 세팅 하에 수행된 단일 표적 절충기법의 그것과 차이가 없음을 알 수 있었다. 따라서 중복이 제거된 테이블을 이용한 탁한 무지개 절충기법이 다중 표적 절충기법으로서도 가장 뛰어난 알고리즘이라는 결론을 내릴 수 있었다.

주요어휘: 암호 해독적 시간 저장 공간 절충기법, 시간 저장 공간 정보 절충기법, 짙은 무지개 절충기법, 탁한 무지개 절충기법, 중복이 제거된 테이블, 중복이 제거되지 않은 테이블

학번: 2010–20240