Decoding the Past

Thesis by
Siddharth Jain

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2019
Defended April 1, 2019

Dedicated to my family

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank my advisor Prof. Shuki Bruck. Words are less to express the greatness of Shuki. There are so many things - his ever smiling face, kindness, concern about the well being of his students, creating an atmosphere of a family in the lab by organizing yearly lab alumni workshop, group dinners and many more. Shuki can cheer you up in no time, no matter how low you are feeling in the day. Not only has Shuki inspired me to be a great researcher, but he has also inspired me to be a great mentor, educator and a risk taker. I feel fortunate to have his influence in my life and hope to be able to embed his qualities for my future.

I thank my candidacy and thesis committee members: Prof. P.P. Vaidyanathan, Prof. Babak Hassibi, Prof. Erik Winfree, and Prof. Moshe Schwartz for valuable feedback during my candidacy. I would also like to thank Prof. Vaidyanathan for providing me with an opportunity to TA his Signal Processing class as it brought back good memories from my undergraduate days and the principles that had initially inspired me to pursue research in applied math oriented EE/CS fields.

I want to thank my collaborators - Prof. Moshe Schwartz, Prof. Farzad Farnoud, Prof. Noga Alon, Dr. Netanel Raviv and Bijan Mazaheri for their invaluable help and guidance that has made this thesis possible. My collaboration with Moshe was through Skype calls and the summer visits that he made at Caltech. Moshe's promptness in response and sharpness with problem solving is inspiring. Farzad was like a senior mentor to me during the first half of my PhD. He introduced me to a rich literature connected to the ideas presented in this thesis, taught me how to properly define a research problem, helped me with writing research papers and with career advice for academia. At the same time, he became a nice friend with whom I travelled to conferences and had several dinners and lunches which helped us connect and know about each other's cultures and interests. In the latter half of my PhD, I worked with Netanel and Bijan, without whose support the computational work presented in this thesis wouldn't have been possible. At the same time, Netanel and Bijan also became friends with whom I can share my personal and professional perspectives. I would also like to thank Dr. Eytan Ruppin at National Cancer Institute (NCI) for his valuable feedback on our cancer work. I am also grateful to other members in my lab - Jin, Yue, Wentao, Kordag with whom I didn't have the opportunity to collaborate but made great friendships.

I would like to thank Michelle and Tanya for taking care of all the administrative work regarding conference travel, candidacy and defense scheduling, etc. I would also like to thank Laura and Daniel at ISP for creating a great environment for international students and helping them with their visas and taxes.

I thank all my friends - Abhinav, Sisir, Aditya, Anupama, Ishan, Subbu, Sivani, Srikanth, Chandru, Kavya, Sripriya, Kishore, Neel, Ramya, Akshata, Mannat, Utkarsh, Sumanth, Shashank, Mohit, Anchal, Ehsan, Parham, Wael, Christos, Vikas and Vinamra for providing me with many happy memories and perspectives that have helped me grow both personally as well as professionally. Playing cricket has been a major component of my life. I have been playing and following cricket much before I had gained interest in any form of academic studies. I was fortunate to find a group of cricket enthusiasts here with whom I could discuss international cricket, play the game on weekends and in the process develop friendships. I would especially like to thank Charles Steinhardt who was the main driver in bringing together many Caltech undergrads, graduate students and postdocs interested in cricket and forming a cricket club. We played against other universities and clubs in the area which was educational as well as a lot of fun. I would also like to thank all my cricket team members - Kishore, Sumanth, Neel, Srikanth, Ravi, Vibhor, Shashank, Chanpreet, and Ishwar for their consistent attendance on weekend practice which made our team competitive and playing a lot of fun. Also, I would like to thank Shashank for getting me in touch with cricket clubs in the LA/Orange county area that helped me grow my cricketing skills.

My biggest achievement here was finding my life partner - my wife Prachi. She has been there with me through all the ups and downs of my graduate student life, comforted me during situations when I am unnecessarily stressed, given a positive critique of my ideas and philosophies and helped me refine them by giving other perspectives that I couldn't have thought of, helped me from being an introvert to a person who is more expressive (still work in progress!), given valuable feedback for all my academic presentations, and now also become my cricket-watching partner, sport she had no interest in when we first met. I would also like to thank my in laws for their constant love, support, and cherished memories.

At last, and most importantly, I would like to thank my parents and my sister who have been cheering me on all the way from India. It is tough for parents to send their kids so far. However, when I expressed my interest to pursue graduate school in the west, my mother immediately told me, "Do it," and said "I only care about your happiness." In Indian culture, if your mother is convinced of something, then it is generally safe to assume that your dad is too. My mother is the sole reason that I find myself in a position to write this thesis today. Without the sacrifices she made, I wouldn’t have ended up in a place like IIT Kanpur, where the interest for research started in me. I feel like the luckiest son in the world to have my mother.

ABSTRACT

The human genome is continuously evolving, hence the sequenced genome is a snapshot in time of this evolving entity. Over time, the genome accumulates mutations that can be associated with different phenotypes - like physical traits, diseases, etc. Underlying mutation accumulation is an *evolution channel* (the term *channel* is motivated by the notion of communication channel introduced by Shannon [1] in 1948 and started the area of *Information Theory*), which is controlled by hereditary, environmental, and stochastic factors. The premise of this thesis is to understand the human genome using information theory framework. In particular, it focuses on: (i) the analysis and characterization of the evolution channel using measures of *capacity*, *expressiveness* , *evolution distance*, and *uniqueness* of ancestry and uses these insights for (ii) the design of error correcting codes for DNA storage, (iii) inversion symmetry in the genome and (iv) cancer classification.

The mutational events characterizing this evolution channel can be divided into two categories, namely point mutations and duplications. While evolution through point mutations is *unconstrained*, giving rise to combinatorially many possibilities of what could have happened in the past, evolution through duplications adds constraints limiting the number of those possibilities. Further, more than 50% of the genome has been observed to consist of repeated sequences. We focus on the much constrained form of duplications known as tandem duplications in order to understand the limits of evolution by duplication. Our sequence evolution model consists of a starting sequence called *seed* and a set of tandem duplication rules. We find limits on the diversity of sequences that can be generated by tandem duplications using measures of capacity and expressiveness. Additionally, we calculate bounds on the duplication distance which is used to measure the timing of generation by these duplications. We also ask questions about the uniqueness of seed for a given sequence and completely characterize the duplication length sets where the seed is unique or non-unique. These insights also led us to design error correcting codes for any number of tandem duplication errors that are useful for DNA-storage based applications. For uniform duplication length and duplication length bounded by 2, our designed codes achieve channel capacity. We also define and measure *uncertainty* in decoding when the duplication channel is misinformed. Moreover, we add substitutions to our tandem duplication model and calculate sequence generation diversity for a given budget of substitutions.

We also use our duplication model to explain the inversion symmetry observed in the genome of many species. The inversion symmetry is popularly known as the 2nd Chargaff Rule, according to which in a *single* strand DNA, the frequency of a k-mer is almost the same as the frequency of its reverse complement. The insights gained by these problems led us to investigate the tandem repeat regions in the genome. Tandem repeat regions in the genome can be traced back in time algorithmically to make inference about the effect of the hereditary, environmental and stochastic factors on the mutation rate of the genome. By inferring the evolutionary history of the tandem repeat regions, we show how this knowledge can be used to make predictions about the risk of incurring a mutation based disease, specifically cancer. More precisely, we introduce the concept of mutation profiles that are computed without any comparative analysis, but instead by analyzing the short tandem repeat regions in a single *healthy* genome and capturing information about the individual's evolution channel. Using gradient boosting on data from more than 5,000 TCGA (The Cancer Genome Atlas) cancer patients, we demonstrate that these mutation profiles can accurately distinguish between patients with various types of cancer. For example, the pairwise validation accuracy of the classifier between PAAD (pancreas) patients and GBM (brain) patients is 93%. Our results show that healthy unaffected cells still contain a cancer-specific signal, which opens the possibility of cancer prediction from a healthy genome.

PUBLISHED CONTENT AND CONTRIBUTIONS

1. **Siddharth Jain**, Farzad Farnoud, Jehoshua Bruck
   Capacity and Expressiveness of Genomic Tandem Duplication.
   IEEE Transactions on Information Theory, vol 63, no 10, pp. 1629-1638, October 2017.
   URL: https://ieeexplore.ieee.org/document/7984889
   Contribution: Siddharth Jain participated in the conception of the project, derived the results and wrote the manuscript.

2. **Siddharth Jain**, Farzad Farnoud, Moshe Schwartz, Jehoshua Bruck
   Duplication Correcting Codes for Data Storage in the DNA of a Living Organism.
   IEEE Transactions on Information Theory, vol 63, no 8, pp. 4996-5010, August 2017.
   URL: https://ieeexplore.ieee.org/document/7888471
   Contribution: Siddharth Jain participated in the conception of the project, derived the results and wrote the manuscript.

3. Noga Alon*, Jehoshua Bruck*, Farzad Farnoud*, **Siddharth Jain***
   Duplication Distance to the root for binary sequences.
   IEEE Transactions on Information Theory, vol 63, no 12, pp. 7793-7803, December 2017 (*author list in alphabetical order).
   URL: https://ieeexplore.ieee.org/document/7993073
   Contribution: Siddharth Jain participated in the conception of the project, derived the results and partially contributed to the writing of the manuscript.

4. **Siddharth Jain**, Bijan Mazaheri, Netanel Raviv, Jehoshua Bruck
   Cancer Classification from Healthy DNA using Machine Learning.
   bioRxiv, January 2019.
   DOI: https://doi.org/10.1101/517839
   Contribution: Siddharth Jain participated in the conception of the project, conducted the experiments, implemented the pipeline and wrote the manuscript.

5. **Siddharth Jain**, Bijan Mazaheri, Netanel Raviv, Jehoshua Bruck
   Short Tandem Repeats Information in TCGA is Statistically Biased by Amplification.
   bioRxiv, January 2019.

DOI: https://doi.org/10.1101/518878
Contribution: Siddharth Jain participated in the conception of the project, conducted the experiments, implemented the pipeline and wrote the manuscript.

6. **Siddharth Jain**, Netanel Raviv, Jehoshua Bruck
Attaining the 2nd Chargaff Rule by Tandem Duplications.
In Proceedings of IEEE International Symposium on Information Theory (ISIT), pp. 2241-2245, Vail, Colorado, June 2018.
URL: https://ieeexplore.ieee.org/document/8437526
Contribution: Siddharth Jain participated in the conception of the project, derived the results and wrote the manuscript.

7. **Siddharth Jain**, F. Farnoud, M. Schwartz, J. Bruck
Noise and Uncertainty in String-Duplication Systems.
In Proceedings of IEEE International Symposium on Information Theory (ISIT), pp. 3120-3124, Aachen, Germany, June 2017.
URL: https://ieeexplore.ieee.org/document/8007104
Contribution: Siddharth Jain participated in the conception of the project, derived the results and partially contributed to the writing of the manuscript.

8. **Siddharth Jain**, F. Farnoud, M. Schwartz, J. Bruck
Duplication Correcting Codes for DNA Storage in DNA of Living Organism.
In Proceedings of IEEE International Symposium on Information Theory (ISIT), pp. 1028-1032, Barcelona, Spain, July 2016.
URL: https://ieeexplore.ieee.org/document/7541455
Contribution: Siddharth Jain participated in the conception of the project, derived the results and wrote the manuscript.

9. Noga Alon*, J. Bruck*, F. Farnoud*, **Siddharth Jain***
On the Duplication Distance of Binary Strings.
In Proceedings of IEEE International Symposium on Information Theory (ISIT), pp. 260-264, Barcelona, Spain, July 2016 (*author list in alphabetical order).
URL: https://ieeexplore.ieee.org/abstract/document/7541301
Contribution: Siddharth Jain participated in the conception of the project, derived the results and partially contributed to the writing of the manuscript.

10. **Siddharth Jain**, F. Farnoud, J. Bruck
Capacity and Expressiveness of Genomic Tandem Duplication.

In Proceedings of 2015 IEEE International Symposium on Information Theory (ISIT), pp. 1946-1950, Hong Kong, July 2015.

URL: https://ieeexplore.ieee.org/document/7282795

Contribution: Siddharth Jain participated in the conception of the project, derived the results and wrote the manuscript.

TABLE OF CONTENTS

Acknowledgements iv
Abstract vii
Table of Contents xii
List of Illustrations xiv
List of Tables xxii
Chapter I: Introduction 1
1.1 Evolution by Duplications: Mathematical Limits 3
1.2 Applications 6
1.3 Organization 12
Chapter II: Diversity of Tandem Duplication String Systems 14
2.1 Introduction 14
2.2 Preliminaries 18
2.3 Capacity and Expressiveness 20
2.4 Regular Languages for Tandem Duplication String Systems 28
2.5 Conclusion 33
Chapter III: Duplication Distance 35
3.1 Introduction 35
3.2 Bounds on $f(n)$ 39
3.3 Duplication Distance for L-systems 44
3.4 Approximate-duplication distance 48
3.5 Duplication distances for different roots 52
3.6 Open Problems 54
Chapter IV: Uniqueness of Seed and Error Correction 55
4.1 Introduction 55
4.2 Preliminaries 57
4.3 k-Tandem-Duplication Codes 62
4.4 $\leqslant k$-Tandem-Duplication Codes 76
4.5 Uncertainty in Coding for $S_{\leqslant k}$ 84
4.6 Duplication Roots 88
4.7 Conclusion 92
Chapter V: String Duplication Systems with Substitutions 94
5.1 Introduction 94
5.2 Preliminaries 94
5.3 Capacity and Expressiveness in S_k^p 96
5.4 Discussion 104
Chapter VI: Inversion Symmetry by Tandem Duplications 105
6.1 Motivation for the Model 107
6.2 Boundary effect 108
6.3 Results and Discussions 110

6.4 Experimental Findings 118
6.5 Conclusion 119
Chapter VII: Cancer Classification from Healthy DNA 121
7.1 Introduction 121
7.2 Cancer Genomics 122
7.3 Results 123
7.4 Discussion 126
7.5 Methods 127
Chapter VIII: Future Work 145
8.1 Repeat regions and disease prediction 145
8.2 Interspersed Repeats 146
8.3 Mathematical Models of Evolution 147
Bibliography 149

LIST OF ILLUSTRATIONS

Number *Page*

1.1 Slipped Strand Mispairing Mutation. Here a tandem repeat with 12 CAG repeats is converted to a repeat with 24 CAG repeats due to slipped strand mispairing that happens during DNA replication. Figure obtained from [5]. 2

1.2 Shipman et al. Nature 2017, used CRISPR-Cas to store this gif inside a bacterial DNA. 7

1.3 Information embedded in DNA can get corrupted by mutational errors over generations due to DNA replication that occurs during cell division. 7

1.4 Tandem duplication error channel where any number of tandem duplication errors can occur corrupting the embedded information in the DNA. 8

1.5 Erwin Chargaff, the scientist behind the discovery of the inversion symmetry in a single strand DNA, known as the 2nd Chargaff Rule. [Source: Wikipedia] 9

1.6 A tandem repeat region in Chromosome 1: 136200-137288 in the reference human genome. The highlighted columns show the locations in the repeats with mismatches caused due to point mutations such as insertion, deletion and substitution. 10

1.7 A normal cell becomes a tumor cell due to the accumulation of driver mutations. [Source: [51]] 11

1.8 The Cancer Genome Atlas (TCGA) contains DNA and RNA information about individuals from 33 different cancer types. [Source: [55]] 12

2.1 The finite automaton for the systems $S = (\{0,1\}, 01, \mathcal{T}^{tan}_{\leqslant k})$, where $k \geqslant 2$, including the system of Example 2.1. Notation used here is described in detail in Section 2.2. 16

2.2 Finite automaton for $S = (\{0,1,2\}, 012, \mathcal{T}^{tan}_{\leqslant 3})$. 21

2.3 Finite automaton for $S = (\Sigma, a_1a_2a_3\cdots a_m, \mathcal{T}^{tan}_{\leqslant 3})$. 30

2.4 Finite automaton for $S = (\{0,1,2\}, 012, \mathcal{T}^{tan}_{\leqslant 2})$. The regular expression $R = 0^+1^+(0^+1^+)^*2^+(1^+2^+)^*$. 33

3.1 $\frac{f(n,m)}{n-m}$ for $3 \leqslant m < n \leqslant 32$. 43

3.2 $f_{10}(n)$ and $f_{101}(n)$ for $1 \leqslant n \leqslant 32$. 52

4.1 The graph $\mathcal{G}_3'$ producing the set of ternary irreducible words $\text{Irr}_{3;\leqslant 3}$. Vertices without edges were removed as well. 82

5.1 The derivation process of (5.2). Solid arrows represent tandem duplication. Dashed arrows represent tandem duplication with at least one point mutation. The process proceeds from top to bottom, and left to right. 101

5.2 Bounds on $\mathsf{cap}(S_k^p(s))$ with $\Sigma = \mathbb{Z}_4$, $k = 2$, and $p(t) = \alpha kt$. The improved lower bound of Lemma 5.4 with (a) $\beta = 1$, (b) $\beta = 1.25$, (c) $\beta = 1.5$, as well as (d) the lower bound of Theorem 5.3 and (e) the upper bound of Theorem 5.3. 103

6.1 Red and green segments represent unbalanced and balanced segments in v_i respectively. Each $a_j n_i (1 \leqslant j \leqslant q_i + 1)$ represents a unbalanced segment. Each $\mu_j n_i (1 \leqslant j \leqslant q_i)$ represents a balanced segment. Note $q_i \leqslant i$. 111

6.2 Addition Operation: All the symbols of y_i are unbalanced in v_i and the symbol before and after y_i are also unbalanced in v_i. $y_i y_i^*$ is added as a new balanced segment in v_{i+1}. Note that the count of balanced segments is 1 more than the count of balanced segments in v_i. 112

6.3 Modification Operation: Some symbols of y_i are balanced in v_i. Hence $y_i y_i^*$ in v_{i+1} is a modification of the previously created balanced segments in v_i. Note that the count of balanced segments in v_{i+1} has not increased from v_i. In fact in this particular instance, it has decreased. 113

6.4 Number of generations (g) needed such that $\frac{N_g}{n_g} = (1 - \epsilon)$ for $\epsilon = 0.00005$, $|v_0| = 10000$ and $d_i = d \; \forall i \geqslant 0$. 115

6.5 Variation of $\Delta_{(10,g)}$ vs g for d = 50, 100, 150, 200, 250, 300, 350, 400, 450, 500. and $|v| = 10000$. We can see that a larger value of d diminishes the boundary effects. 116

6.6 Comparison of R_X^k value calculated experimentally in chromsomes X, 14, 17 and 21 (solid lines) with those obtained by reverse complement tandem (dashed lines) and interspersed (dotted lines) duplications for different duplication lengths d. 119

7.1 **(a)** Two different evolution histories for the tandem repeat region *ACGTACGTACATAGAT* with pattern length 4 and repeat region length 16. In History 1, only 2 point mutations were needed (marked with green and red respectively). In History 2, 3 point mutations were needed: 1 marked with green and 2 marked with red. In our approach, we would consider History 1 to be more likely as it involves lesser number of point mutations. Therefore, for this tandem repeat region we have that $m = 2$ and $d = 4$. **(b)** The workflow of our algorithm. In Part A a classifier is trained based on the mutation profiles generated from the healthy DNA of cancerous individuals. This part in only performed once per training set. In Part B, the resulting classifier is applied over a given genome to assess an individual's inclination of developing different cancers. 132

7.2 The matrix on the left contains the pairwise validation accuracies of classifiers built by using 3843 blood-derived normal DNA samples covering 11 different cancer types. These cancer types are TCGA-SKCM (skin), PAAD (pancreas), STAD (stomach), BLCA (bladder), PRAD (prostate), LGG (brain_lgg), LUAD (lung), THCA (thyroid), LUSC (lung_sq), HNSC (head_neck), GBM (brain). Each cell in the accuracy seriation matrix represents the average validation accuracy of the binary pairwise classifiers. Each pairwise classifier between cancer X and cancer Y (X≠Y) was constructed using 4-fold cross-validation with patients of each cancer type. These accuracies can be interpreted as distances. The darker the cell, farther are the cancers being compared. The darker rows corresponding to brain, skin and pancreas are indicative of the presence of cancer-type signal in the blood-derived normal (healthy) DNA of cancer patients. The diagonal entries in the seriation matrix represent the accuracies when half of the patients of cancer X were labeled 0 and half of the patients of the same cancer X were labeled 1. As one can expect, the average test accuracy for such classifier should be around 50%. The value in the cell corresponding to the row “pancreas” and the column “prostate” signifies that an average of 74% of the people were correctly classified in each validation pass. The matrix on the right, contains the sensitivity/specificity values. Each cell in the sensitivity/specificity seriation matrix represents the sensitivity value when the row cancer is considered positive and the column cancer is considered negative. It can also be regarded as specificity when the row cancer is considered negative and the column cancer is considered positive. Sensitivity is defined as $TP/(TP + FN)$ and specificity is defined as $TN/(TN + FP)$, where TP = True Positive, FP = False Positive, TN = True Negative, FN = False Negative. A value of .77 in the row “prostate” and the column “pancreas” means that 77% of the prostate patients in the test set were truly classified as prostate type (sensitivity when prostate is considered positive). A value of .73 in the row “pancreas” and the column “prostate” means that 73% of the pancreas patients in the test set were truly classified as pancreas type (specificity when prostate is considered positive). The seriation ordering is obtained by solving TSP (i.e., the Travelling Salesman Problem) exhaustively [117]. 133

7.3 These seriation matrices show 4-fold validation accuracy and sensitivity/specificity for the four main clusters of cancers in Figure 7.2 generated using 3843 blood-derived normal samples. Class 1 = (brain), Class 2 = (skin), Class 3 = (pancreas), Class 4 = (stomach, bladder, prostate, brain_lgg, lung, thyroid, lung_sq, head_neck). . . . 134

7.4 Mean and standard deviations for the cancer classification profiles of individuals in Class 1, Class 2, Class 3 and Class 4 (viewing left to right). To generate these profiles, we a trained multiclassifier on all four classes of cancers using gradient boosting. We then used this multiclassifier to obtain cancer classification profiles for a different set of individuals reported the average results for each cancer class. Class 1 individuals show a high probability for Class 1 cancers. Class 2 and Class 3 individuals also show a higher probability for their respective classes, but with a slightly weaker signal. Class 4 individuals show similar probabilities for Class 2, Class 3 and Class 4. 134

7.5 The matrix on the left contains the pairwise validation accuracies of classifiers built by using 4561 unamplified samples covering both blood-derived and solid tissue normal DNA of 12 different cancer types. These cancer types are TCGA-SKCM (skin), PAAD (pancreas), STAD (stomach), BLCA (bladder), PRAD (prostate), LGG (brain_lgg), LUAD (lung), THCA (thyroid), LUSC (lung_sq), HNSC (head_neck), GBM (brain), KIRC (kidney). 135

7.6 Classifier accuracies and sensitivity/specificity when training and testing is done using amplified samples only. The cancers covered here are TCGA-GBM (brain), TCGA-OV (ovary) and TCGA-LAML (leukemia). There is a strong signal distinguishing all these cancer types. We speculate that TCGA-KIRC would behave similarly to TCGA-GBM, since these cancers are similar in Figure 7.5. 135

7.7 Mean and standard deviation for the cancer classification profiles of individuals belonging to TCGA-GBM (brain), TCGA-OV (ovary) and TCGA-LAML (leukemia). These classification profiles are obtained by building a multiclassifier using gradient boosting. The multiclassifier is obtained by training *only* on amplified healthy samples of brain, ovary and leukemia cancers on TCGA. The testing is done on a separate set of amplified samples for these cancers. It can be seen that brain, ovary and leukemia cancer patients in the test set are showing a higher probability for the respective cancer using their healthy DNA mutation profile. 136

7.8 A comparison of classifiers trained on mutation profiles restricted to known driver-gene regions and those restricted to non-driver-gene regions. Darker cells represent a larger difference in the testing accuracies of the classifier, indicated by the scale. This experiment was performed on unamplified (left) and amplified (right) samples separately. The uneven coloring suggests that some cancer-type signals exist more primarily in driver-gene regions than others. 136

7.9 Cancer classification profile for Class 4 individuals when the multiclassifier was trained only using Class 1, Class 2 and Class 3 samples mentioned in Figure 7.3. This shows a stronger association of Class 4 with Class 2 and Class 3 than Class 1. However the high standard deviation for Class 1 also means that some individuals in Class 4 have a stronger association with Class 1 than Class 2 and Class 3. 138

7.10 These seriation matrices show 4-fold validation accuracy and sensitivity/specificity for the four main clusters of cancers in Figure 7.5 where all the 4561 unamplified samples were used. Class 1 = (brain, kidney), Class 2 = (skin), Class 3 = (pancreas), Class 4 = (stomach, bladder, prostate, brain_lgg, lung, thyroid, lung_sq, head_neck). . . . 138

7.11 Mean and standard deviations for the cancer classification profiles of individuals in Class 1, Class 2, Class 3 and Class 4 (viewing left to right). To generate these profiles, we a trained multi-classifier on all four classes of cancers using gradient boosting. We then used this multi-classifier to obtain cancer classification profiles for a different set of individuals reported the average results for each cancer class. Class 1 individuals show a high probability for Class 1 cancers. Class 2 and Class 3 individuals also show a higher probability for their respective classes, but with a slightly weaker signal. Class 4 individuals show similar probabilities for Class 2, Class 3 and Class 4. 139

7.12 Cancer classification profile for Class 4 individuals when the multi-classifier was trained only using Class 1, Class 2 and Class 3 samples shown in Figure 7.10. This shows a stronger association of Class 4 with Class 2 and Class 3 than Class 1. However the high standard deviation for Class 1 also means that some individuals in Class 4 have a stronger association with Class 1 than Class 2 and Class 3. 140

7.13 Accuracies and sensitivity/specificity for the pairwise classifiers when only the copy number (d) information was used in the mutation profile for the 3843 WXS blood-derived unamplified samples mentioned in Table 7.1A. Notice that like Figure 7.2, we again find strong distinguishing signals for brain, pancreas and skin cancers as can be seen by the darker rows corresponding to these cancers. 140

7.14 Accuracies and sensitivity/specificity for the pairwise classifiers when only the error number (m) information was used in the mutation profile for the 3843 WXS blood-derived unamplified samples mentioned in Table 7.1A. Notice that like Figure 7.2, we again find strong distinguishing signals for brain and skin cancers as can be seen by the darker rows corresponding to these cancers. However unlike Figure 7.2, the signal for pancreas cancer is not as strong. 141

7.15 Accuracies and sensitivity/specificity for the pairwise classifiers when only the copy number (d) information was used in the mutation profile for the 4561 WXS unamplified samples that comprised both blood-derived and solid tissue normal type DNA as mentioned in Table 7.1A. Notice that like Figure 7.5, we again find strong distinguishing signals for (brain,kidney), pancreas and skin cancers as can be seen by the darker rows corresponding to these cancers. 141

7.16 Accuracies and sensitivity/specificity for the pairwise classifiers when only the error number (m) information was used in the mutation profile for the 4561 WXS unamplified samples that comprised both blood-derived and solid tissue normal type DNA as mentioned in Table 7.1A. Notice that like Figure 7.5, we again find strong distinguishing signals for (brain,kidney) and skin cancers as can be seen by the darker rows corresponding to these cancers. However, unlike Figure 7.5, the signal for pancreas cancer is not as strong. 142

7.17 Accuracies and sensitivity/specificity for the pairwise classifiers when only the copy number(d) information was used in the mutation profile for the 525 WXS amplified samples mentioned in Table 7.1B. Notice that like Figure 7.6, we again find strong distinguishing signals for all the three cancers. 142

7.18 Accuracies and sensitivity/specificity for the pairwise classifiers when only the error number (m) information was used in the mutation profile for the 525 WXS amplified samples mentioned in Table 7.1B. Notice that like Figure 7.6, we again find strong distinguishing signals for all the three cancers. 143

7.19 Accuracies for the pairwise classifiers when hipSTR was used for STR detection. hipSTR detects short tandem repeats with pattern lengths 6 or less. We used **–min-reads** 1 **–def-stutter-model** setting. Even though hipSTR requires further trimming of mutation profiles to tandem repeats with pattern lengths $\leqslant 6$, we were still able to detect cancer-specific signals. The differences from Figure 7.5 in Figure 7.19 (for example: prostate and brain, prostate and kidney, skin and pancreas) can be attributed to the fact that there are tandem repeat regions with pattern lengths greater than 6 that contained critical cancer-specific information. Further, we only used STRs that were commonly detected by hipSTR in all the samples being analyzed. Therefore, the STRs that were not detected in some samples were not used in this analysis. 144

LIST OF TABLES

Number *Page*

2.1 Capacity values tandem duplication string systems $(\Sigma, s, \mathcal{T}^{tan}_{\leqslant k})$. Here $x, y \in \Sigma^*$, and $a, b, c \in \Sigma$ are distinct. 18

2.2 Expressiveness of tandem duplication string systems $\left(\Sigma, s, \mathcal{T}^{tan}_{\leqslant k}\right)$. . . . 18

2.3 Expressiveness of tandem duplication string systems $(\Sigma, s, \mathcal{T}^{tan}_{\leqslant k})$. . . . 28

2.4 Capacity and Expressiveness for different tandem duplication string systems $(\Sigma, s, \mathcal{T}^{tan}_{\leqslant k})$. 29

3.1 $f(n)$ for $1 \leqslant n \leqslant 32$. 37

4.1 A list of all overlapping squares of length at most 3 (up to a permutation of the alphabet symbols) 79

7.1 **(A)** Number of unamplified healthy samples used for each cancer type in the study showing the number of blood derived normal and solid tissue normal samples. In total, the number of blood derived healthy samples are 3874 and the tissue derived healthy samples are 687. The sample information is provided in Supplementary Files 1-12. **(B)** Number of amplified healthy samples used for each cancer type in the study showing the number of blood derived normal and solid tissue normal samples. In total, the number of blood derived healthy samples are 331 and the tissue derived healthy samples are 194. The sample information is provided in Supplementary Files 13-15. 137

Chapter 1

INTRODUCTION

The human genome is an incredibly rich source of information, but relevant insights remain trapped within a puzzle of billions of base pairs. The genome has evolved through a series of mutational events spanning generations, giving rise to tremendous diversity between individuals. We believe that each individual's genome is a realization of a *distinct evolution channel*, which is a function of hereditary, environmental, and stochastic factors. By observing an individual's genome we only see the *effects* of this underlying evolution channel.

Genomic studies such as Linkage and Genome Wide Association Studies (GWAS) attempted to understand the genome by viewing it as a *time-independent source* and finding variants or "spelling mistakes" by comparing genomes of individuals with different phenotypes [2, 3]. This approach led to the discovery of several genes and pathways, but could not explain many properties of the observed phenotypes known as the issue of "missing heritability" [4]. A different approach would be to use the genomic information to *trace back in time* to characterize how the genome is mutating. This brings us to our first question:- *Does an individual's genome contain markers which can retrace the evolutionary history of the genome?* In other words, can we use the genome to quantify a *signature* of mutation activity? A signature indicating high mutation activity may, for example, suggest that the genome is prone to developing high risk of a certain disease. Intuitively, the extracted signature is a count of the *accumulated* mutations providing access to the *time-dependent* traits in the genome. Our approach is to first capture and quantify this signature independently, not associating DNA with any phenotype. This raises the second question: *Can we capture an individual's evolution channel independently, without relying on comparisons with other genomes?* In other words, can we assign a *persona* to each individual's genome? Answering this question requires an understanding of the evolution of diverse genomes through different mutations and then identifying markers in the genome that *encode* a *decodable* evolutionary history.

The premise of this thesis is to understand the human genome using information theory framework. In particular, it focuses on:

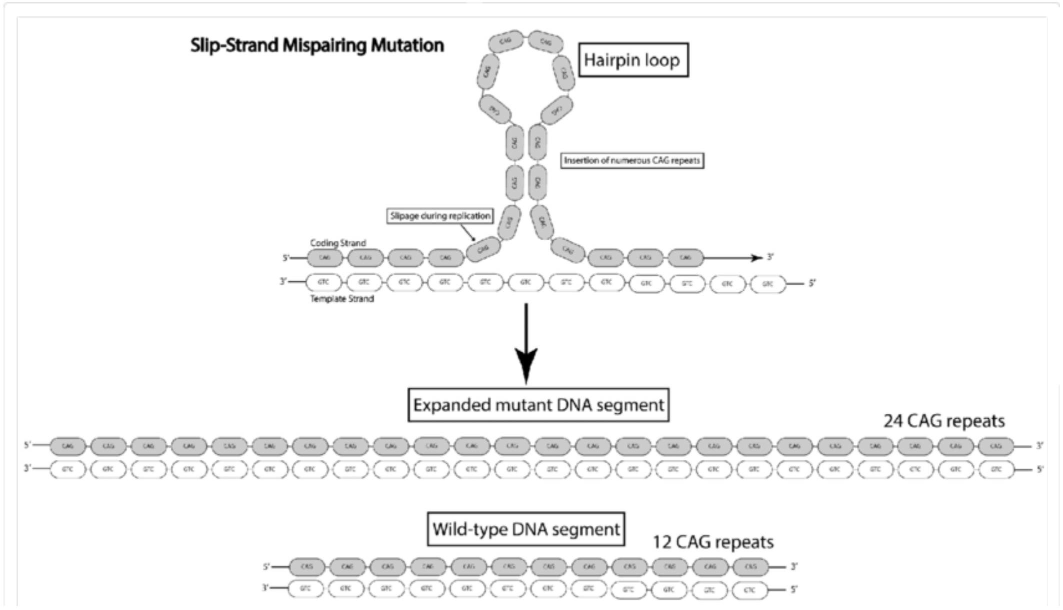

Slip-Strand Mispairing Mutation
Hairpin loop
Insertion of numerous CAG repeats
Slipage during replication
Coding Strand
Template Strand
Expanded mutant DNA segment
24 CAG repeats
Wild-type DNA segment
12 CAG repeats

Figure 1.1: Slipped Strand Mispairing Mutation. Here a tandem repeat with 12 CAG repeats is converted to a repeat with 24 CAG repeats due to slipped strand mispairing that happens during DNA replication. Figure obtained from [5].

1. Analysis and characterization of the evolution channel using *capacity*, *expressiveness*, *evolution distance* and *uniqueness* of ancestry.
2. Use the insights gained above for
    - the design of error correcting codes for DNA storage.
    - understanding the inversion symmetry observed in the genome.
    - classifying different cancer-types using blood-derived healthy DNA.

More than 50% of the human genome consists of repeated sequences [6]. It has been conjectured that the repeat regions store many hidden evolution clues [6], and decoding those clues might hold the key to cracking the accumulation of mutations in a DNA. Two important types of common repeats are (i) interspersed repeats and (ii) tandem repeats. Interspersed repeats are caused by transposons. A transposon, also known as a jumping gene, is a segment of DNA that can copy or cut and paste itself into new positions of the genome. Tandem repeats are caused by slipped-strand mispairings [7]. Slipped-strand mispairings occur when one DNA strand in the duplex becomes misaligned with the other (See Figure 1.1).

1.1 Evolution by Duplications: Mathematical Limits

A simple model dictates that genome evolution is governed by point mutations, such as substitutions, insertions, and deletions. Observationally, however, the genome is found to also have undergone many duplications [8]; and it is questionable whether point mutations are the dominant contributor to evolution. Mathematically, it is simple to understand genetic and genomic dissimilarity—or diversity between individuals—based on point mutations. However, it is non-trivial to determine how duplications contribute to genomic diversity. Duplications in DNA are transposon and replication slippage driven [6]. They are known as interspersed and tandem duplications respectively. An example of interspersed and tandem duplication of the substring TC of *duplication length* 2 would be $Interspersed : AG\underline{TC}GAT \rightarrow AG\underline{TC}GA\underline{TC}T$, $\quad Tandem : AG\underline{TC}GAT \rightarrow AG\underline{TCTC}GAT$. Mathematically, tandem duplication is an interspersed duplication with an additional constraint of duplicating next to the original. Therefore, only analyzing the evolution of sequences by tandem duplications will provide us insights into the evolutionary limits of the duplication process.

We asked three questions concerning the *diversity* of new sequences, *history* of evolution and the *uniqueness* of the ancestor given evolution by tandem duplications. These three questions are connected, while diversity measures the number of sequences that can be generated from a given starting sequence called *seed*, timing of evolution measures the minimum number of steps required to generate a given sequence from a seed. Finding duplication processes for which a given sequence has a unique seed is equivalent to solving if for any two seeds there is no intersection in the set of sequences generated by them. In our sequence evolution model, a set of tandem duplication operations, is applied successively on the seed to generate new sequences. More formally, *String Duplication Systems* that consist of a starting sequence called *seed* and a duplication rule were introduced in [9] to model these operations. Example 1.1 below illustrates an instance that can be generated by a tandem duplication string system.

Example 1.1. $Seed = ACGT$
$Duplication\ Rule$ = any substring of length at most 2 can be tandemly duplicated.
<u>Instance</u>

$$AC\underline{GT} \rightarrow AC\underline{G}TGT \rightarrow A\underline{C}GGTGT \rightarrow AC\underline{CG}GTGT \rightarrow ACCGCGGTGT.$$

The underlined substring represents the part of the string that is duplicated at the current step. Note, only substrings of length 1 or 2 are tandemly duplicated.

Diversity

Generation by tandem duplications has been studied in the past in [10–12]. However the main concern of these works is to determine the place of tandem duplication rules in the Chomsky hierarchy of formal languages. One measure to evaluate diversity is *capacity* which can be described as the limit on the number of sequences that can be generated by successive tandem duplications. For tandem duplication string systems, the authors in [13] show that for a *fixed* duplication length the capacity is 0. Further, they find a lower bound on the capacity of these systems, when duplications of all lengths are allowed. We evaluate *exact* capacity for tandem duplication string systems, where we restrict the maximum size of the substring being tandemly duplicated to a certain *finite* length. Using regular languages and Perron-Frobenius theory [14, 15], in [16] we found *exact* capacity for tandem duplication string systems for *any* seed and *any* alphabet where the duplication length is bounded by 2 and 3 respectively. These results indicated that even by restricting the duplication lengths to a set of constant values, exponential number of sequences can be generated for any alphabet size, indicating that enough sequence diversity is possible by successive tandem duplications. Though our capacity results indicated exponential sequences can be generated by tandem duplications, there are sequences that cannot be generated by tandem duplications. Hence, motivated by the fact that small portions of genome called exomes are known to be responsible for protein expression, we investigated if *full* exomic diversity is possible by tandem duplications, meaning whether or not each possible sequence can be generated as substring of some larger sequence. We were able to completely solve this question for all seeds, all alphabet sizes and any finite bound on the duplication length. We found that the binary and ternary systems are *fully expressive* even by restricting the maximum duplication lengths to 2 and 4 respectively; however, from quaternary and onwards, the systems are not, as long as all the duplication lengths are finite [16]. A major contributor to this striking contrast between these two alphabet classes is the result by Axel Thue [17], according to which, for ternary and higher alphabets, there exist repeat-free or *squarefree* sequences for every length. Hence, repeat-free exomes are difficult to generate by tandem duplications. These findings are described in detail in Chapter 2.

History of Evolution

Investigating this generative model of duplication made us ask a related question that could give insights into the upper and lower bounds on the number of tandem duplication steps required to generate a given sequence from a seed. We defined *duplication distance* to measure the number of tandem duplication steps. Duplication distance is the minimum number of tandem duplications needed to generate a sequence from its seed. Example 1.2 below shows two different histories for the generation of $ACACCAAC$ from AC.

Example 1.2. *Seed* = AC, *Duplication Rule* = any substring can be tandemly duplicated.
Below, we show two histories that are possible for the generation of $S = ACACCAAC$ from AC.
History 1:

$$\underline{AC} \to A\underline{CAC} \to ACACC\underline{A}C \to ACACCAAC.$$

History 2:

$$\underline{AC} \to AC\underline{A}C \to A\underline{C}AAC \to \underline{AC}CAAC \to ACACCAAC.$$

History 1 requires 3 steps and history 2 requires 4 steps for generation. For sequence $S = ACACCAAC$, it can be verified that history 1 indeed is the *shortest* and hence the duplication distance of $ACACCAAC$ from AC is 3.

Intuitively, the evolution of larger sequences from smaller sequences should be much faster by duplications than the evolution by only point mutations. However, to our surprise we found that it is not much faster and almost all length n binary sequences (set of probability 1) required $\Theta(n)$ [18] tandem duplication steps to evolve, even if *unbounded* duplication lengths were allowed. This result is very similar in nature to Asymptotic Equipartition property (AEP) which is a well known property in Information Theory literature. Roughly, this also means that *short duplication lengths play a major role in generating capacity*. We also calculated the duplication distance for a number of classes of binary sequences. For de-Bruijn sequences, we show that the duplication distance is $\Omega(\frac{n}{\log n})$. For binary sequences described by Lindenmayer Systems [19], we found the duplication distance to be $\Theta(\log n)$. We also defined the approximate duplication distance that allows tandem duplications with imperfect or approximate blocks, i.e. at most β fraction of symbols can differ in the duplicated block from the original block. We found that for $\beta < 1/2$, the approximate duplication distance is $\Theta(n)$, however for $\beta > 1/2$, the duplication distance is $\Theta(\log n)$. The case of $\beta = 0.5$ is still open. We describe these results in detail in Chapter 3.

Insight: In Chapter 2, we observe that full generation capacity can be achieved for a binary tandem duplication string system when the duplication length is at most 2. For this system, the duplication distance is trivially $\Theta(n)$ for any binary string of length n. It is surprising to observe in Theorem 3.1 in Chapter 3 that the duplication distance is $\Theta(n)$ for *almost all* binary sequences even without any constraints on duplication length. This means that in a binary system, small duplication lengths are the major contributor to the diversity generated by tandem duplications. We observe this phenomenon in higher alphabets as well, as shown in Chapter 2, where a high generation capacity could be achieved even when the duplication lengths are bounded by 3 (capacity for a ternary tandem duplication string system is ≈ 0.88).

Uniqueness of Seed

We also investigated the *uniqueness* of *repeat-free* seed for every sequence that can be generated by tandem duplications. Example 1.3 below shows the generation of $ACGCACGCG$ from two different repeat-free seeds.

Example 1.3. $Seed\ 1 = ACG$

$$A\underline{CG} \rightarrow \underline{ACGC}G \rightarrow ACGCACGCG.$$

$Seed\ 2 = ACGCACG$

$$ACGCA\underline{CG} \rightarrow ACGCACGCG.$$

We found that if the duplication length is fixed or bounded by 3, the seed is unique, however in all other scenarios, there can be multiple seeds for the same sequence [20]. Algorithms for deciding the uniqueness of seed have been recently proposed in [21]. Our findings are described in detail in Chapter 4.

The theoretical findings and the insights developed from the aforementioned works turned out to be useful in the applications described next

1.2 Applications

Live-DNA Storage

DNA can be used as a medium for storing large amounts of data in compact form for a very long time. Due to the exponential drop in the synthesis and sequencing cost, DNA storage is becoming feasible and important. DNA storage can be done outside the organism as demonstrated in [22–24]. DNA storage outside living organism is prone to synthesis and sequencing errors. Lately, there have been several works

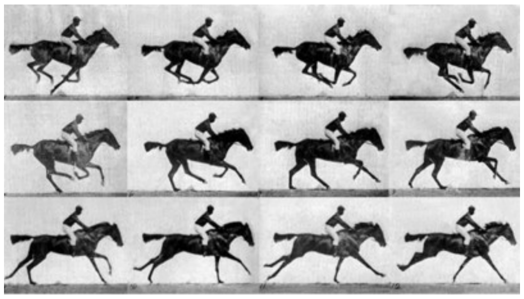


Figure 1.2: Shipman et al. Nature 2017, used CRISPR-Cas to store this gif inside a bacterial DNA.

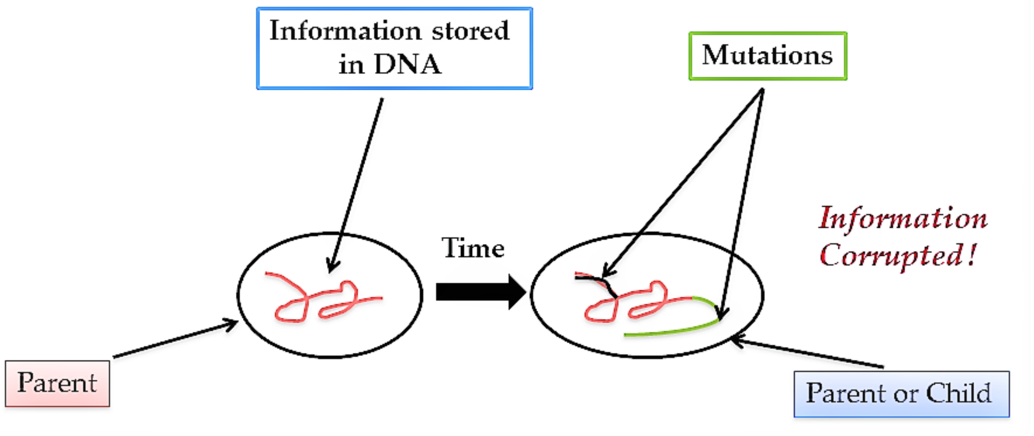


Figure 1.3: Information embedded in DNA can get corrupted by mutational errors over generations due to DNA replication that occurs during cell division.

that find capacity and error correcting codes for channels that model synthesis and sequencing errors [25–29]. Data storage can also be done *inside* the DNA of a living organism. Church and his group [30] used CRISPR technology to store an image and a gif inside the DNA of a bacterium recently. Data storage inside the living organism (henceforth live-DNA) has a multitude of applications. It can enable in-vivo synthetic-biology methods and algorithms that need "memory". It also allows watermarking genetically-modified organisms (GMOs) to verify authenticity and to track unauthorized use [31–33]. It can also be used to identify ancestral relationships between cells [34]. With information storage in live DNA, the information will be passed and can be recovered from next generations. This information inside living DNA is prone to errors depending on the physical medium and the processes involved in the synthesis and sequencing of DNA. As a result, in live-DNA storage, besides synthesis and sequencing errors, there will be errors which are governed by evolution events which comprise duplications, substitutions and indel errors. Figure 1.3 illustrates the setup for embedding information inside the DNA of a

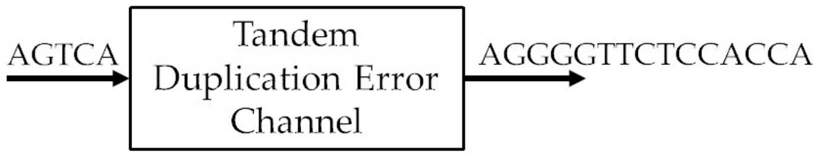


Figure 1.4: Tandem duplication error channel where any number of tandem duplication errors can occur corrupting the embedded information in the DNA.

living organism.

Using our generative model of string duplication systems in [16], in [20] we designed error correcting codes that can deal with *any* number of tandem duplication errors when the length of duplication was fixed and bounded respectively. Figure 1.4 shows a tandem duplication error channel where the input information can get corrupted by a number of tandem duplication errors. We found channels where doing decoding by removing repeats was optimal by showing that a given sequence can only be generated by a unique seed under those channels [20]. In fact for the uniform duplication channel and the bounded duplication channel with duplication length bounded by 2, our codes also achieved the full channel capacity. We extended these results by accommodating substitution errors along with tandem duplications and found lower and upper bounds on the sizes of spheres to be given possible codewords [35]. Here, we also dealt with the scenario of channel misinformation and found explicit sets of codeword intersection and defined *uncertainty* to measure the size of those sets [35]. We have described these codes and methods in detail in Chapters 4 and 5.

These works have been followed up on recently. Channel capacity and error correcting codes for uniform duplication channel with a *finite* number of duplication errors were studied in [36, 37]. Further, efficient encoding and decoding methods for squarefree words were proposed in [38]. Bounds on codes correcting tandem and palindromic duplication errors were derived in [39].

Inversion Symmetry in Genome

We also focused on properties observed in genomic data. One such property that is observed in most of the genomes is the 2nd Chargaff Rule, named after its discoverer Erwin Chargaff [40]. This rule is also known as Inversion Symmetry (IS) of the genome. The presence of IS means that in a *single* strand of DNA, the number of occurences of a k-mer is almost equal to the number of occurrences of its reverse

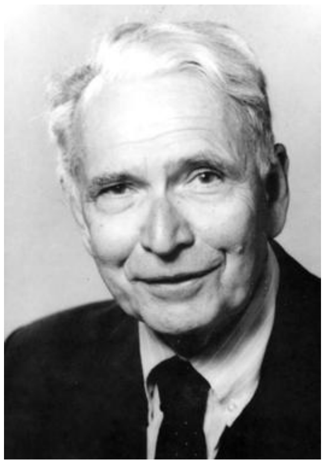
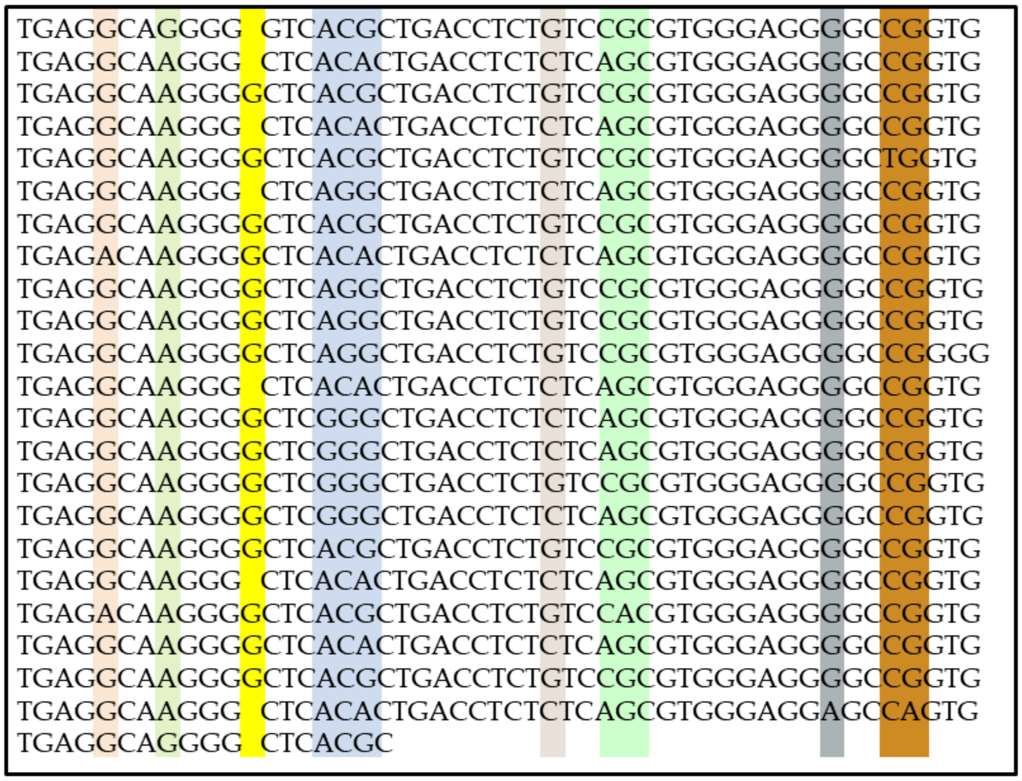


Figure 1.5: Erwin Chargaff, the scientist behind the discovery of the inversion symmetry in a single strand DNA, known as the 2nd Chargaff Rule. [Source: Wikipedia]

complement. For example, $\#A \approx \#T, \#C \approx \#G, \#AC \approx \#GT$ and so on. Inversion symmetry is observed to hold upto a certain k known as K-Limit which depends on the length of the genome [41]. For the human genome, the K-Limit is 10. There have been several works in the past that have verified inversion symmetry for different genomes [41–45]. Further, [41, 42, 46] showed that this symmetry only holds for reverse complement pairs. [43] also argued that IS may be due to whole genome or segmental inverse duplications.

The presence of IS makes it plausible that most of the species share common dynamics of evolution. We investigated mathematical models based on reversed tandem and interspersed duplications for sequence generation. Using probability tools and string duplication system insights, we show that a string duplication system with reverse complement tandem duplication as the operation provides a good approximation for IS as observed in human DNA [47]. We also find the lower bounds on the number of generations required to obtain the inversion symmetry given this generative model. These findings are discussed in detail in Chapter 6.

Tandem Repeat Regions: The insights gained above led us to investigate the tandem repeat regions that cover about 3% of the human genome. These regions have evolved by a sequence of tandem duplications due to replication slippage events [48, 49] and point mutations (single changes like substitutions, insertions and deletions in the DNA, e.g. $AC\underline{T}G \rightarrow AC\underline{A}G$). Alone, neither of these metrics provide insight into the genome's rate of change. When viewed together, however, one can learn the relative rates of these mutational events. For example, while point mutations



```
TGAGGCAGGGG GTCACGCTGACCTCTGTCCGCGTGGGAGGGGCCGGTG
TGAGGCAAGGG CTCACACTGACCTCTCTCAGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCACGCTGACCTCTGTCCGCGTGGGAGGGGCCGGTG
TGAGGCAAGGG CTCACACTGACCTCTCTCAGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCACGCTGACCTCTGTCCGCGTGGGAGGGGCTGGTG
TGAGGCAAGGG CTCAGGCTGACCTCTCTCAGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCACGCTGACCTCTGTCCGCGTGGGAGGGGCCGGTG
TGAGACAAGGGGCTCACACTGACCTCTCTCAGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCAGGCTGACCTCTGTCCGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCAGGCTGACCTCTGTCCGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCAGGCTGACCTCTGTCCGCGTGGGAGGGGCCGGGG
TGAGGCAAGGG CTCACACTGACCTCTCTCAGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCGGGCTGACCTCTCTCAGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCGGGCTGACCTCTCTCAGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCGGGCTGACCTCTGTCCGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCGGGCTGACCTCTCTCAGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCACGCTGACCTCTGTCCGCGTGGGAGGGGCCGGTG
TGAGGCAAGGG CTCACACTGACCTCTCTCAGCGTGGGAGGGGCCGGTG
TGAGACAAGGGGCTCACGCTGACCTCTGTCCACGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCACACTGACCTCTCTCAGCGTGGGAGGGGCCGGTG
TGAGGCAAGGGGCTCACGCTGACCTCTGTCCGCGTGGGAGGGGCCGGTG
TGAGGCAAGGG CTCACACTGACCTCTCTCAGCGTGGGAGGAGCCAGTG
TGAGGCAGGGG CTCACGC
```

Figure 1.6: A tandem repeat region in Chromosome 1: 136200-137288 in the reference human genome. The highlighted columns show the locations in the repeats with mismatches caused due to point mutations such as insertion, deletion and substitution.

are impossible to detect without reference to an initial genome, their occurrence in repeated regions is indicated by a change in the repeated sequence. Moreover, because the point mutation error is propagated in further repeats, we know exactly when the point mutation occurred relative to the tandem duplications, giving insights into the evolution history of the tandem repeat region. In a sense, tandem repeats are a nature-given *repetition error correcting code* where point mutation errors in copies store information about the history of the evolution of the region. Furthermore, the duplication rate in tandem repeat regions is very high due to replication slippage events [50], which allows point mutation errors to accumulate, strengthening the evolutionary signal. Hence, tandem repeat regions belong to those markers where we can *detect* and *measure* mutation activity.

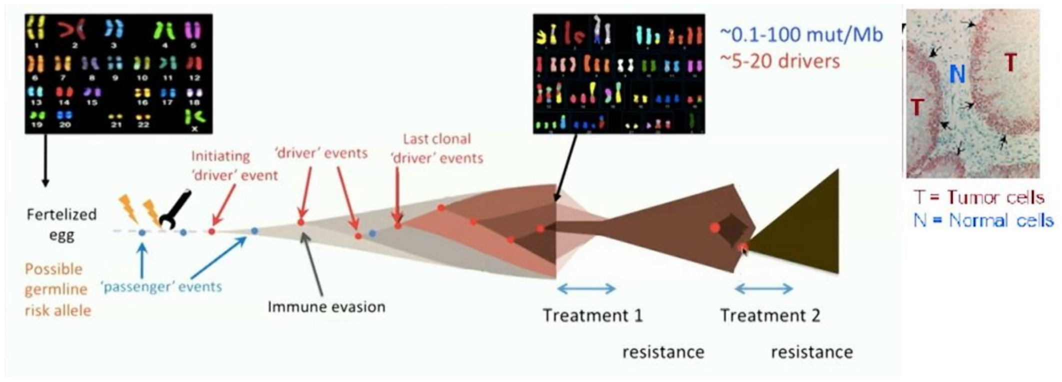


Figure 1.7: A normal cell becomes a tumor cell due to the accumulation of driver mutations. [Source: [51]]

Cancer Classification from healthy DNA

Cancer is the second leading cause of death in the world [52]. It is a mutation based disease caused by a mix of hereditary, environmental and stochastic factors [53] affecting the DNA in a single lifetime. Cancer is caused by the accumulation of driver mutations during cell division which make the cell cancerous (see Figure 1.7). Cancer studies in the past attempted to identify these driver mutations by comparing the genome of a cancerous cell with a non-cancerous cell [54]. We had a hypothesis that given an underlying evolution channel of the genome, if we can capture information about the rate of generation of these mutations (or we may call the *intrinsic mutation rate*) from a single DNA, it may lead us capture a signal about the propensity of the genome to incur these driver mutations. The mutation rates in tandem repeat regions are strong [50] and measurable [56]. We estimate the evolutionary history of short tandem repeat regions or microsatellites and aggregate it. We call this aggregated information the genome's *mutation profile*. The mutation profile carries information about the number of duplications and point mutations required in the evolution of each tandem repeat region in the DNA. Our hypothesis was that the healthy genome of each individual carries information about their *future* cancer risk, and capturing the signature of mutation activity can help us predict those risks. However, to verify this hypothesis, we required DNA samples from cancer patients before the onset of their cancer. Currently availing that data is not possible, we instead used DNA derived from blood or healthy tissue of cancer patients from The Cancer Genome Atlas (TCGA) [57] (See Figure 1.8). We estimated the mutation profiles for more than 5000 DNA samples on TCGA covering 14 different cancers including common cancers like lung, prostate, stomach, pancreas, skin, kidney, brain, etc. If we could classify different cancer-types based on the mutation profiles

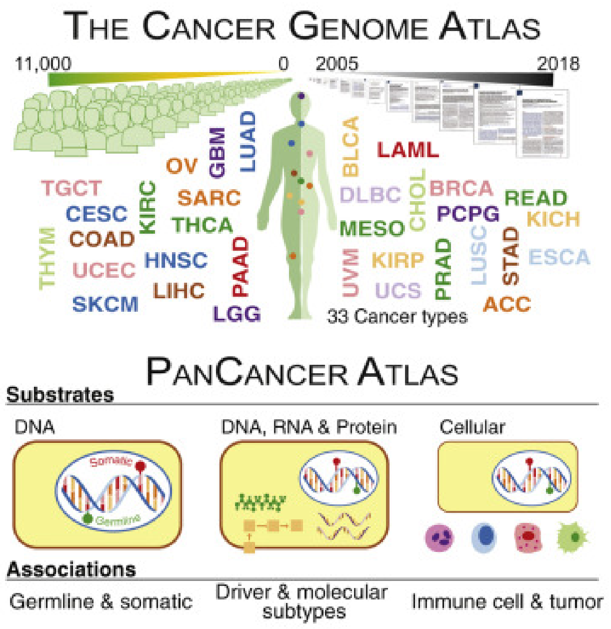


Figure 1.8: The Cancer Genome Atlas (TCGA) contains DNA and RNA information about individuals from 33 different cancer types. [Source: [55]]

of the healthy genome, it means that these mutation profiles carry a *cancer-type signal*. By dividing this data into a training and a test set, we built gradient boosting [58] based pairwise and multi classifiers that used mutation profiles as features to check if they carry any cancer-type signal [59, 60]. In fact, we found four clusters based on healthy DNA mutation profiles. Further, based on these classifiers, we also generated *cancer classification profiles* which measured the propensity of an individual to each cancer type [59, 60]. As the cancer-type signal detection is done by *only* using healthy genome, mutation profiles could be useful in predicting future cancer risk and early cancer detection. We describe our results in Chapter 7.

1.3 Organization

The rest of the thesis is organized as follows. In Chapter 2, we describe string duplication systems and the diversity that can be generated by bounded tandem duplication string systems using the measures of capacity and expressiveness. In Chapter 3, we analyze duplication distance to measure the number of evolutionary steps by tandem duplications. In Chapter 4, we find systems with unique and non-unique seeds and use these results to design duplication correcting codes applicable in live-DNA storage. Moreover, we consider the effect of channel mismatch on codeword decoding by defining uncertainty. In Chapter 5, we add substitution operation alongwith tandem duplications to answer questions regarding bounds on code size under these errors. In Chapter 6, we use string duplication systems to explain the inversion symmetry observed in the genome. In Chapter 7, we define mutation profiles which aggregate the evolutionary history of tandem repeat regions in human DNA. We use these mutation profiles to build classifiers that can classify different types of cancer based on the healthy DNA. These findings may have implications in early cancer detection and future cancer risk prediction. In Chapter 8, we conclude the thesis with the directions for future work.

Chapter 2

DIVERSITY OF TANDEM DUPLICATION STRING SYSTEMS

2.1 Introduction

Tandem Repeats are common in both prokaryote and eukaryote genomes. They are present in both coding and non-coding regions and are believed to be the cause of several genetic disorders. The effects of tandem repeats on several biological processes are understood by these disorders. They can result in generation of toxic or malfunctioning proteins, chromosome fragility, expansion diseases, silencing of genes, modulation of transcription and translation [61] and rapid morphological changes [62].

A process that leads to tandem repeats, e.g. through slipped-strand mispairing [7, 50], is called *tandem duplication*, which allows substrings to be duplicated next to their original position. For example, from the sequence $AGTCGTCGCT$, a tandem duplication of length 2 can give $AGTCGT\underline{CGCG}CT$, which, if followed by a duplication of length 3 may give $AGTCG\underline{TCGTCG}CGCT$. The prevalence of tandem repeats in the human genome [6] motivates us to study the capacity and expressiveness of string systems with tandem duplication, as defined below.

The model of a *string duplication system* consists of a *seed*, i.e., a starting string of finite length, a set of duplication rules that allows generating new strings from existing ones, and the set of all sequences that can be obtained by applying the duplication rules to the seed a finite number of times. The notion of *capacity*, introduced in [13] represents the average number of m-ary symbols per sequence symbol that are asymptotically required to encode a sequence in the string system, where m is the *alphabet* size (for DNA sequences the alphabet size is 4). The maximum value for capacity is 1. A duplication system is *fully expressive* if all strings with the alphabet appear as a substring of some string in the system. As we will show, if a system is not fully expressive, then its capacity is strictly less than 1.

Before presenting the notation, definitions, and the results more formally, in the rest of this section, we present two simple examples to illustrate the notions of expressiveness and capacity for tandem duplication string systems. Furthermore, we outline some useful tools as well as some of the results.

Example 2.1. Consider a string system on the binary alphabet $\Sigma = \{0, 1\}$ with 01 as the seed that allows tandem duplications of length up to 2. It is easy to check that the strings generated by this system start with 0 and end with 1. In fact, it can be proven that all binary strings of length n which start with 0 and end with 1 can be generated by this system. The proof is based on the fact that every such string can be written as $0^{r_1}1^{r_2}\cdots 0^{r_{v-1}}1^{r_v}$, where each $r_i \geqslant 1$ and v is even. A natural way to generate this string is to duplicate 01 $\frac{v}{2}$ times and then duplicate the 0s and 1s as needed via duplications of length 1.

Expressiveness: From the preceding paragraph, every binary sequence s can be generated as a substring in this system as $0s1$. For example, although 11010 cannot be generated by this system, it can be generated as a substring of 0110101 in the following way:

$$01 \to 0101 \to 010101 \to 0\underline{11010}1.$$

Hence this system is fully expressive.

Capacity: The number of length-n strings in this system is 2^{n-2}. Thus, encoding sequences of length n in this system requires $n-2$ bits. The capacity, or equivalently the asymptotic average number of bits (since the alphabet Σ is of size 2) per symbol, is thus equal to 1. This is not surprising as the system generates almost all binary sequences. □

Observing these facts for an alphabet of size 2, one can ask related questions on expressiveness and capacity for other alphabet sizes and duplication lengths. However, counting the number of length-n sequences for capacity calculation and characterizing fully expressive systems for larger alphabets are often not straightforward tasks. In this chapter, we study these questions and develop methods to answer them.

A useful tool in this study is the theory of finite automata. As a simple example note that the string system over the binary alphabet $\{0, 1\}$ in the preceding example can be represented by the finite automaton given in Figure 2.1. The regular expression for the language defined by the finite automaton is

$$R_{01} = (0^+1^+)^+, \tag{2.1}$$

which represents all binary strings that start with 0 and end with 1. For definitions of finite automata and regular expression, the reader can refer to Section 2.2.

One can use the Perron-Frobenius theory [14, 15] to count the number of sequences which can be generated by a finite automaton. This enables us to use finite automata

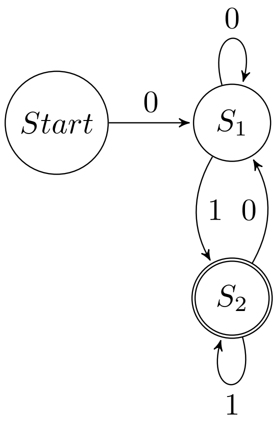


Figure 2.1: The finite automaton for the systems $S = (\{0, 1\}, 01, \mathcal{T}^{tan}_{\leqslant k})$, where $k \geqslant 2$, including the system of Example 2.1. Notation used here is described in detail in Section 2.2.

as a tool to calculate capacity for some string duplication systems with tandem repeats over larger alphabet.

In our results, we find that the exact capacity of the tandem duplication string system over the ternary alphabet $\{0, 1, 2\}$ with seed 012 and duplication length at most 3 equals $\log_3 \frac{3+\sqrt{5}}{2} \simeq 0.876036$. Moreover, we generalize this result by characterizing the capacity of tandem duplication string systems over an arbitrary alphabet and a seed with maximum duplication length of 3. Namely, we show that if the maximum duplication length is 3 and the seed contains abc as a substring, where a, b, and c are distinct symbols, then the capacity $\simeq 0.876036 \log_{|\Sigma|} 3$. If such a substring does not exist in the seed, then the capacity is given by $\log_{|\Sigma|} 2$, unless the seed is of the form a^m, in which case the capacity is 0. Some of these results are highlighted in Table 2.1.

Our next example presents a system that, unlike that of Example 2.1, is not fully expressive.

Example 2.2. Consider a tandem duplication string system over the ternary alphabet $\{0, 1, 2\}$ with seed 012 and maximum duplication length 3. This system is not fully expressive as it cannot generate 210, 102, or 021, even as a substring. We provide a simple proof.

Proof. Let $z = \alpha\gamma\beta$, where α, γ and β are strings over $\{0, 1, 2\}$ with $|\alpha|, |\beta| \geqslant 0$, and $1 \leqslant |\gamma| \leqslant 3$. Suppose z does not contain 210, 102 or 021 as a substring. We will now show that $z^* = \alpha\gamma\gamma\beta$ does not contain 210, 102 or 021 as a substring either. We have three cases:

- For $\gamma = a_1$, with $a_1 \in \{0, 1, 2\}$, the only possible new substrings generated in z^* which may not occur in z are the ones with suffix a_1a_1 or prefix a_1a_1.
- For $\gamma = a_1a_2$, with $a_1, a_2 \in \{0, 1, 2\}$, the only possible new substrings of length 3 generated in z^* are $a_1a_2a_1$ and $a_2a_1a_2$.
- For $\gamma = a_1a_2a_3$, with $a_1, a_2, a_3 \in \{0, 1, 2\}$, there is no substring of length 3 in z^* which does not occur in z.

Hence, if z does not contain 210, 102 or 021 as a substring, neither will z^*. Since the seed 012 does not contain 210, 102 or 021 as a substring, neither will any other string in the system. ▮

Therefore, this tandem duplication string system is not fully expressive. □

Based on the previous example, one may ask what happens if we start with a seed that contains one of the strings 210, 102, or 021, e.g., if we let the seed be 01210? Does the system become fully expressive? While this system can generate all strings of length 3 as substrings, the answer is still no as shown in Theorem 2.5: Regardless of the seed, a ternary system with maximum duplication length of 3 is not fully expressive. We show in Theorem 2.8, that a maximum duplication length of at least 4 is needed to arrive at a fully expressive ternary system.

While for alphabets of size 2 or 3, increasing the maximum length on duplications turns a system that is not fully expressive into one that is, for alphabets of size 4 or more, duplication systems are not fully expressive regardless of how large the bound on duplication length is. The main tool in constructing quaternary strings that do not appear independently or as substrings in these systems is Thue's result proving the existence of ternary *squarefree* sequences of any length. A string is called squarefree if it does not have a tandem repeat of any length (Note that unary and binary squarefree sequences of arbitrarily large length do not exist.). The existence of such sequences underlies the significant shift in the behavior of tandem duplication systems with regards to expressiveness as a function of alphabet size. Some of our results on expressiveness are summarized in Table 2.2.

As part of this chapter, we also study regular languages for tandem duplication string systems. In [63], it was shown that the tandem duplication string system is not regular if the maximum duplication length is 4 or more when the seed contains 3 consecutive distinct symbols as a substring. However for maximum duplication

Σ	s	k	Capacity
$\{0,1,2\}$	012	3	$\simeq 0.876036$
arbitrary	$xabcy$	3	$\simeq 0.876036 \log_{\lvert\Sigma\rvert} 3$

Table 2.1: Capacity values tandem duplication string systems $(\Sigma, s, \mathcal{T}_{\leqslant k}^{tan})$. Here $x, y \in \Sigma^*$, and $a, b, c \in \Sigma$ are distinct.

Σ	s	k	fully expressive
$\{0,1,2\}$	arbitrary	$\leqslant 3$	No
$\{0,1,2\}$	012	$\geqslant 4$	Yes
Size $\geqslant 4$	arbitrary	arbitrary	No

Table 2.2: Expressiveness of tandem duplication string systems $\left(\Sigma, s, \mathcal{T}_{\leqslant k}^{tan}\right)$.

length 3, this question remained open. In this chapter, we show in Theorem 2.9 that if the maximum duplication length is 3, a tandem duplication string system is regular irrespective of the seed and the alphabet size. Moreover, we characterize the exact capacity for all these systems.

In this chapter, we consider tandem duplication string systems, where we restrict the maximum size of the block being tandemly duplicated to a certain *finite* length. In the rest of the chapter, the term tandem duplication string system refers to string duplication systems with bounded duplication length.

The rest of the chapter is organized as follows. In Section 2.2, we present the preliminary definitions and notation. In Section 2.3, we derive our main results on capacity and expressiveness. In Section 2.4, we show that if the maximum duplication length is 3, then the tandem duplication string system is regular irrespective of the seed and alphabet size. Further, using the regularity of the systems, we extend our capacity results. We present our concluding remarks in Section 2.5.

2.2 Preliminaries

Let Σ be some finite alphabet. An n-string $x = x_1 x_2 \cdots x_n \in \Sigma^n$ is a finite sequence where $x_i \in \Sigma$ and $|x| = n$. The set of all finite strings over the alphabet Σ is denoted by Σ^*. For two strings $x \in \Sigma^n$ and $y \in \Sigma^m$, their concatenation is denoted by $xy \in \Sigma^{n+m}$. For a positive integer m and a string s, s^m denotes the concatenation of m copies of s. A string $v \in \Sigma^*$ is a substring of x if $x = uvw$, where $u, w \in \Sigma^*$.

A string system $S \subseteq \Sigma^*$ is represented as a tuple $S = (\Sigma, s, \mathcal{T})$, where $s \in \Sigma^*$ is a finite string called seed, which is used to initiate the duplication process, and $\mathcal{T}$ is a set of rules that allow generating new strings from existing ones [13]. In other words, the string system $S = (\Sigma, s, \mathcal{T})$ contains all strings that can be generated from s using rules from $\mathcal{T}$ a finite number of times.

A tandem duplication map $T_{i,k}$,

$$T_{i,k}(x) = \begin{cases} uvvw, & x = uvw, |u| = i, |v| = k, \\ x, & \text{else}, \end{cases}$$

creates and inserts a copy of the substring of length k which starts at position $i + 1$. We use $\mathcal{T}_k^{tan} : \Sigma^* \to \Sigma^*$ and $\mathcal{T}_{\leqslant k}^{tan}$ to denote the set of tandem duplications of length k, and tandem duplications of length at most k, respectively,

$$\mathcal{T}_k^{tan} = \left\{T_{i,k} : i \in \mathbb{N} \cup \{0\}\right\},$$
$$\mathcal{T}_{\leqslant k}^{tan} = \left\{T_{i,j} : i \in \mathbb{N} \cup \{0\}, j \in \mathbb{N},\ j \leqslant k\right\}.$$

With this notation, the system of Example 2.1 can be written as $(\{0, 1\}, 01, \mathcal{T}_{\leqslant 2}^{tan})$.

The *capacity* of the string system $S = (\Sigma, s, \mathcal{T})$ is defined as

$$\text{cap}(S) = \limsup_{n\to\infty} \frac{\log_{|\Sigma|} |S \cap \Sigma^n|}{n}. \tag{2.2}$$

Furthermore, it is *fully expressive* if for each $y \in \Sigma^*$, there exists a $z \in S$, such that y is a substring of z.

A useful tool in calculating capacity of tandem duplication string systems is deterministic finite automaton (DFA) which consists of:

- A finite set of states Z.
- Alphabet Σ.
- Transition Rule $\delta : Z \times \Sigma \to Z$.
- Start state $z_o \in Z$.
- A set of accept states Y.

There also exist non-determnisitc finite automata. In this chapter however, all the automata considered are deterministic. Henceforth, we will be using finite automaton to refer to a DFA. An example of a finite automaton is given in Figure 2.1. For this finite automaton we have,

- $Z = \{Start, S_1, S_2\}$.
- $\Sigma = \{0, 1\}$.
- $\delta(Start, 0) = S_1,\ \delta(S_1, 0) = S_1,\ \delta(S_1, 1) = S_2,\ \delta(S_2, 0) = S_1,\ \delta(S_2, 1) = S_2$.
- $z_o = Start$.
- $Y = \{S_2\}$.

The set of all possible strings that can be generated by a given DFA represent the language described by the finite automaton. This language L_R can be represented by a regular expression R. Formal definitions of regular expression can be found in [64]. For the purpose of this chapter, we define:

- $R = s^*$: represents the language L_R which consists of all strings with 0 or more concatanated copies of $s \in \Sigma^*$, i.e., $L_R = \{s^m : m \geqslant 0\}$.
- $R = s^+$: represents the set of all strings with 1 or more concatanated copies of $s \in \Sigma^*$, i.e., $L_R = \{s^m : m \geqslant 1\}$.
- $R = R_1R_2$: represents the language L_R formed by the concatenation of L_{R_1} and L_{R_2}, i.e. $L_R = \{s_1s_2 : s_1 \in L_{R_1}, s_2 \in L_{R_2}\}$.

2.3 Capacity and Expressiveness

In this section, we present our results on the capacity and expressiveness of tandem duplication system with bounded duplication length. The section is divided into two parts; the first part focuses on capacity and the second on expressiveness.

Capacity

Our first result is on the capacity of a tandem duplication string system over ternary alphabet.

Theorem 2.3. *For the tandem duplication string system $S = \left(\{0, 1, 2\}, 012, \mathcal{T}^{tan}_{\leqslant 3}\right)$, we have*

$$\text{cap}(S) = \log_3 \frac{3 + \sqrt{5}}{2} \simeq 0.876036.$$

Proof. We prove this theorem by showing that the finite automaton given in Figure 2.2 accepts precisely the strings in S, and then finding the capacity using the Perron-Frobenius theory [14, 15].

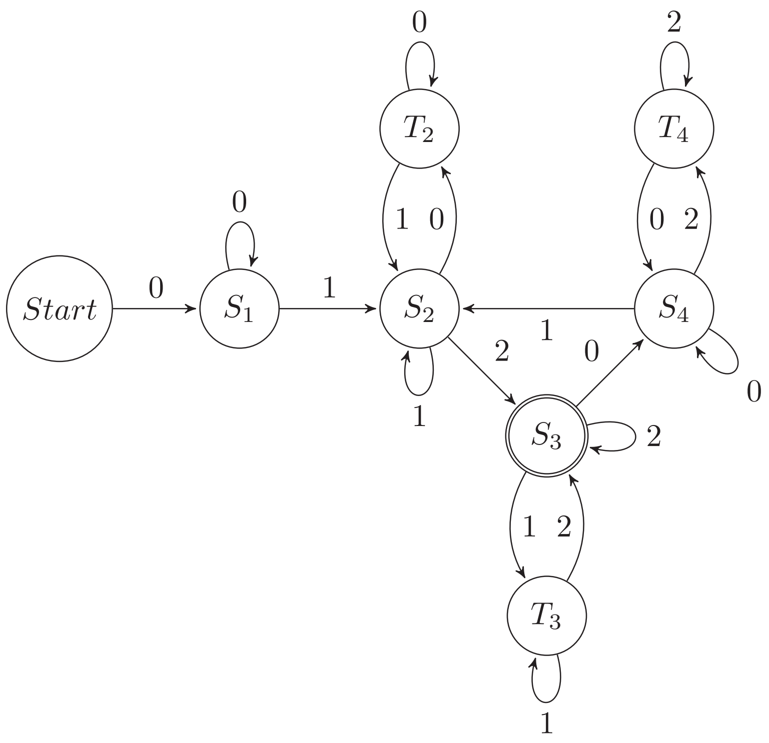


Figure 2.2: Finite automaton for $S = (\{0, 1, 2\}, 012, \mathcal{T}^{tan}_{\leqslant 3})$.

The regular expression R for the language defined by this finite automaton is given by (see [64] for details on how to find a regular expression given a finite automaton)

$$R = (0^+1^+)^+2^+(1^+2^+)^*[0^+(2^+0^+)^*1^+(0^+1^+)^*2^+(1^+2^+)^*]^*. \tag{2.3}$$

Let L_R be the language defined by the regular expression R (and by the finite automaton). We first show that $L_R \subseteq S$. The direct way of doing so is to start with 012 and generate all the sequences in L_R via duplication. For simplicity of presentation, however, we take the reverse route: We show that every sequence in L_R can be transformed to 012 by a sequence *deduplications*. A deduplication of length k is an operation that replaces a substring $\alpha\alpha$ by α if $|\alpha| = k$. For two regular expressions R_1 and R_2, we use $R_1 \xrightarrow{dd_{\leqslant k}} R_2$ to denote that *each* sequence in L_{R_1} can be transformed into *some* sequence in L_{R_2} via a sequence of deduplications of length at most k.

Note that $R = B_1 B_2{}^*$, where

$$B_1 = (0^+1^+)^+2^+(1^+2^+)^*,$$
$$B_2 = 0^+(2^+0^+)^*1^+(0^+1^+)^*2^+(1^+2^+)^*.$$

We have $B_1 \xrightarrow{dd_{\leqslant 3}} 012(12)^* \xrightarrow{dd_{\leqslant 3}} 012$, since $a^+ \xrightarrow{dd_{\leqslant 3}} a$ and $(ab)^+ \xrightarrow{dd_{\leqslant 3}} ab$ for all $a, b \in \Sigma$. Furthermore,

$$B_2 \xrightarrow{dd_{\leqslant 3}} 0(20)^*1(01)^*2(12)^* \xrightarrow{dd_{\leqslant 3}} 0(20)^*1(01)^*2 \xrightarrow{dd_{\leqslant 3}} 0(20)^*12 \xrightarrow{dd_{\leqslant 3}} \{02012, 012\}. \tag{2.4}$$

Note for example that $1(01)^*\underline{2}(12)^* \xrightarrow{dd_{\leqslant 3}} 1(01)^*2$ as the underlined 2 is always preceded by a 1.

We thus have $R = B_1B_2^* \xrightarrow{dd_{\leqslant 3}} \{01202012, 012012, 012\} \xrightarrow{dd_{\leqslant 3}} 012$, proving that $L_R \subseteq S$.

To complete the proof of $L_R = S$, we now show that $S \subseteq L_R$. In what follows, we say a finite automaton generates a sequence s, if there is a path with label s from $Start$ to an accepting state. If an automaton generates uvw, with $u, v, w \in \Sigma^*$, we may use v to refer both to the string v itself and to the part of the path that generates v. The meaning will be clear from the context.

We show $S \subseteq L_R$, by proving the following for the finite automaton in Figure 2.2:

i) It can generate 012.

ii) If the automaton can generate pqr, with $p, q, r \in \Sigma^*$ and $|q| \leqslant 3$, it can also generate pq^2r.

Condition i) holds trivially (see the path $Start - S_1 - S_2 - S_3$ in Figure 2.2). In order to prove ii), we define:

- *Path Label*: Given a path a in a finite automaton, the path label $l_a \in \Sigma^*$ is defined as the sequence obtained by concatenating the labels on the edges forming the path.
- *Path Length* is the number of edges of the path.
- *Duplicable Path*: Let q be a path that ends at state C. The path q is said to be *duplicable* if there exists path q' that **starts and ends at state** C such that the path labels of q and q' are the same.

Suppose a finite automaton can generate pqr. If q is duplicable, then pq^2r can also be generated by the finite automaton. As a result, to prove ii), it suffices to show that for each state C in Figure 2.2, all paths of length 1, 2 or 3 ending in C are duplicable.

Now, we show that all paths ending in $\{S_1, S_2, S_3, S_4, T_2, T_3, T_4, \}$ with length $\leqslant 3$ are duplicable. Note that there are no nontrivial paths ending in the $Start$ state.

Given a state C and $j \in \{1, 2, 3\}$, let P_j^C be the set of all length-j paths ending in C and let Q_j^C be the set of all length-j paths starting and ending in C. If

$$\bigcup_{a \in P_j^C} l_a = \bigcup_{a \in Q_j^C} l_a, \tag{2.5}$$

then all length-j paths ending in C are duplicable.

We prove that (2.5) holds for all states and all $j \in \{1, 2, 3\}$. This is done by computing $\mathcal{A}$, $\mathcal{A}^2$ and $\mathcal{A}^3$, where $\mathcal{A}_1$ is the (labeled) adjacency matrix of the finite automaton given in Figure 2.2. Here in computing the matrix products, symbols do not commute, e.g. $xy \neq yx$. The adjacency matrix $\mathcal{A}$ and its square $\mathcal{A}^2$, where x, y and z represent edges labeled by 0, 1, and 2, respectively, and where rows and columns correspond in order to $S_1, S_2, S_3, S_4, T_2, T_3, T_4$, are given by

$$\mathcal{A} = \begin{bmatrix} x & y & 0 & 0 & 0 & 0 & 0 \\ 0 & y & z & 0 & x & 0 & 0 \\ 0 & 0 & z & x & 0 & y & 0 \\ 0 & y & 0 & x & 0 & 0 & z \\ 0 & y & 0 & 0 & x & 0 & 0 \\ 0 & 0 & z & 0 & 0 & y & 0 \\ 0 & 0 & 0 & x & 0 & 0 & z \end{bmatrix},$$

$$\mathcal{A}^2 = \begin{bmatrix} x^2 & y^2+xy & yz & 0 & yx & 0 & 0 \\ 0 & y^2+xy & z^2+yz & zx & x^2+yx & zy & 0 \\ 0 & xy & z^2+yz & x^2+zx & 0 & y^2+zy & xz \\ 0 & y^2+xy & yz & x^2+zx & yx & 0 & z^2+xz \\ 0 & y^2+xy & yz & 0 & x^2+yx & 0 & 0 \\ 0 & 0 & z^2+yz & zx & 0 & y^2+zy & 0 \\ 0 & xy & 0 & x^2+zx & 0 & 0 & z^2+xz \end{bmatrix}.$$

Each entry in these matrices lists the paths of specific length from the state identified by its row to the state identified by its column. For example, the entry $(6, 3)$ of $\mathcal{A}^2$, which equals $z^2 + yz$, indicates that there are two paths of length 2 from T_3 to S_3 with labels $z^2 = 22$ and $yz = 12$.

To check (2.5), we need to verify that the nonzero terms in the non-diagonal elements of each column also appear in its diagonal element. For $\mathcal{A}$ and $\mathcal{A}^2$, this can be easily done by observing the matrices. For example, the entry $(3, 3)$ of $\mathcal{A}^2$ equals $z^2 + yz$ and contains all terms appearing in column 3 of $\mathcal{A}^2$, which are yz and $z^2 + yz$. We verified using a computer that $\mathcal{A}^3$ also satisfies the same condition. Hence, we have shown that all paths of length at most 3 ending in $\{S_1, S_2, S_3, S_4, T_2, T_3, T_4\}$ are duplicable.

This completes the proof of $S \subseteq L_R$.

Now that we have shown $S = L_R$, we use the Perron-Frobenius theory [14, 15] to count the number of sequences which can be generated via the finite automaton in Figure 2.2. The accepting state S_3 is reachable from every other state in the finite automaton, therefore we can compute the capacity by calculating the maximum absolute eigenvalue e^* of the (unlabeled) adjacency matrix B of the strongly connected component of the finite automaton (i.e. the subgraph induced by $S_2, S_3, S_4, T_2, T_3, T_4$).

$$B = \begin{bmatrix} 1&1&0&1&0&0 \\ 0&1&1&0&1&0 \\ 1&0&1&0&0&1 \\ 1&0&0&1&0&0 \\ 0&1&0&0&1&0 \\ 0&0&1&0&0&1 \end{bmatrix}. \tag{2.6}$$

The maximum absolute eigenvalue of B is $e^* = \frac{3+\sqrt{5}}{2} \simeq 2.618034$. By the Perron-Frobenius Theory, $\text{cap}(S) = \log_3 e^* \simeq 0.876036$. ∎

While the proof of the preceding theorem providing the exact capacity of the system under study is somewhat involved, it is easy to see why the capacity is strictly less than 1. One can observe from the regular expression for the finite automaton that it cannot generate a string that has 210, 021 or 102 as a substring, implying that the system is not fully expressive. As we will see in Lemma 2.6, such systems cannot have capacity 1. It is worth noting that the set of strings that avoid 210, 021, and 102 can be shown to have capacity $\simeq 0.914838$, which is slightly larger than the capacity of the system of the theorem.

Expressiveness

We now turn to study the expressiveness of tandem duplication systems with bounded duplication length. For completeness we start with binary systems, which is indeed the simplest case.

Lemma 2.4. *The system* $S = \left(\{0,1\}, s, \mathcal{T}^{tan}_{\leqslant 1}\right)$*, for any s is not fully expressive.*

Proof. The system cannot generate $(01)^m$ as a substring of any string in S for $2m > |s|$. ∎

As shown in Example 2.1, to obtain fully expressive binary systems, it suffices to increase the maximum duplication length to 2.

The next theorem is concerned with the expressiveness of $S = (\{0,1,2\}, s, \mathcal{T}^{tan}_{\leqslant 3})$. Larger alphabets and larger duplication lengths are considered in Theorems 2.7 and 2.8.

Theorem 2.5. *Consider* $S = (\{0,1,2\}, s, \mathcal{T}^{tan}_{\leqslant 3})$*, where s is any arbitrary starting string* $s \in \{0,1,2\}^*$*. Then, S is not fully expressive.*

Proof. A k*-irreducible* string is a string that does not have a tandem repeat $\alpha\alpha$, such that $|\alpha| \leqslant k$. For example, 01201, 01210, 02101, and 01210121 are 3-irreducible strings, while 01212, 021021 and 01112 are not 3-irreducible. To prove the theorem, we identify certain properties in new 3-irreducible strings that may appear after a duplication and then construct a 3-irreducible string that is neither a substring of s, nor it satisfies the properties that every new 3-irreducible substring must satisfy.

Consider a duplication event that transforms a sequence $z = uvw$ to $z^* = uvvw$, where $|v| \leqslant 3$. Let x be a 3-irreducible string of length at least 4 that is present in z^* but not in z. The string x must intersect with both copies of v in z^* or else it is also present in z. Furthermore, it cannot contains vv, since otherwise it would not be 3-irreducible. To determine the properties of x, we consider three case: $|v| = 1, 2, 3$. In what follows assume $a_1, a_2, a_3 \in \Sigma$.

First, suppose $|v| = 1$, say $v = a_1$. In this case, a string x with the aforementioned properties does not exist as all new substrings contain the square a_1a_1.

Second, assume $|v| = 2$, say $v = a_1a_2$. Then $z^* = ua_1a_2a_1a_2w$ and x either ends with $a_1a_2a_1$ or starts with $a_2a_1a_2$.

Third, suppose $|v| = 3$, say $v = a_1a_2a_3$. So $z^* = ua_1a_2a_3a_1a_2a_3w$. Recall that $|x| \geqslant 4$. The string x either ends with $a_1a_2a_3a_1$ or $a_2a_3a_1a_2$, or starts with $a_2a_3a_1a_2$ or $a_3a_1a_2a_3$.

So for any new 3-irreducible substring $x = x_1 \cdots x_j$, $x_i \in \Sigma$, $j \geqslant 4$, we have $x_1 = x_3$, $x_1 = x_4$, $x_j = x_{j-2}$, or $x_j = x_{j-3}$. Now consider the string $(0121)^\ell 0$, where $\ell > |s|$. This sequences is 3-irreducible but does not satisfy any of the 4 properties stated for x. Since it is not a substring of s and it cannot be generated as a new substring, it is not a substring of any $y \in S$. ∎

Next we consider the system $\left(\Sigma, s, \mathcal{T}^{tan}_{\leqslant k}\right)$, $|\Sigma| \geqslant 4$ in Theorem 2.7. The proof of the theorem, uses the following lemma, which states that the expressiveness of a system also has a bearing on its capacity.

Lemma 2.6. *If a string system S with alphabet Σ is not fully expressive, then* $\mathrm{cap}(S) < 1$.

Proof. Since S is not fully expressive, there exists a $z \in \Sigma^*$ that does not appear as a substring of any $y \in S$. Let $|z| = m$ and $\mu = n - m\lfloor \frac{n}{m} \rfloor$. We have

$$|S \cap \Sigma^n| \leqslant (|\Sigma|^m - 1)^{\lfloor \frac{n}{m} \rfloor} |\Sigma|^\mu.$$

Since m is finite, $\mathrm{cap}(S) < 1$. ▮

Theorem 2.7. *Consider* $S = \left(\Sigma, s, \mathcal{T}^{tan}_{\leqslant k}\right)$, *where* $|\Sigma| \geqslant 4$, *s is any arbitrary seed* $\in \Sigma^*$ *and k is some finite natural number. Then S is not fully expressive, which also implies* $\mathrm{cap}(S) < 1$.

Proof. Suppose $z = uvw \in S$, where $|v| \leqslant k$, and let $z^* = uvvw$ be the result of a duplication applied to z. Furthermore, suppose that $x = x_1 \cdots x_j$, where $x_i \in \Sigma$ and $j > k$, is a squarefree substring of z^* but not z. Similar to the proof of Theorem 2.5, x intersects both copies of v but does not contain both. As a result, either $x_1 = x_{1+i}$ or $x_j = x_{j-i}$, for some $2 \leqslant i \leqslant k$.

For definiteness assume Σ contains the symbols $\{0, 1, 2, 3\}$. The sequence $0t0$, where t is a squarefree sequence over the alphabet $\{1, 2, 3\}$ and $|t| > \max\{|s|, k\}$, is not a substring of s and cannot be generated as a substring since it does not satisfy the conditions stated for x above. Note that such a t exists since as shown by Thue [17], for an alphabet size $\geqslant 3$, there exists a squarefree string of any length. Hence S is not fully expressive. The second part of the theorem follows from Lemma 2.6. ▮

Theorem 2.8. *Consider* $S = (\{0, 1, 2\}, 012, \mathcal{T}^{tan}_{\leqslant 4})$. *Then S is a fully expressive string system.*

Proof. Let $S' = \left(\{0, 1, 2\}, 012, \mathcal{T}^{tan}_{\leqslant 3}\right)$. Clearly, $S' \subseteq S$. From the proof of Theorem 2.3, we know that the automaton of Figure 2.2 gives the same language as S'. By checking this automaton, we find that all strings of lengths 1, 2, and 3, except 021, 210, and 102, appear as a substring of some string in S' and, as a result, some string in S. To generate 021, 210, and 102 as substrings of some string in S, we proceed as follows:

$$012 \to 0\underline{1212} \to \underline{01\mathbf{210}121}2$$

$$012 \to \underline{012012} \to 01\underline{2020}12 \to 0\underline{12\mathbf{021}202}012$$

$$012 \to \underline{012012} \to 01\underline{2020}12 \to 012\underline{020\mathbf{102}01}2$$

where the repeats are underlined.

We have shown that all strings of length 3 appear in S as substrings. Now we show the same for every string $w = w_1w_2w_3w_4$ of length 4. To do so, we study 3 cases based on the structure of w:

I) First, suppose that w_4 is the same as w_1, w_2, or w_3. For generating such w as a substring, we first generate $w' = w_1w_2w_3$ as a substring of some string and then do a tandem duplication of w_3 if $w_4 = w_3$, of w_2w_3 if $w_4 = w_2$ and of $w_1w_2w_3$ if $w_4 = w_1$.

II) Suppose I) does not hold but $w_1 = w_2$ or $w_2 = w_3$. If the former holds, first generate $w_1w_3w_4$ and then duplicate w_1, and if the latter hold, generate $w_1w_2w_4$ and duplicate w_2.

III) If neither I) nor II) holds, then $w = 1210$, up to a relabling of the symbols. In this case, we first generate $w' = 0121$ and then do a tandem duplication of w' to get w. Note that w' is of type considered in I).

Until now, we have shown that all strings w of length at most 4 appear as a substring of some string in S. We use induction to complete the proof. Suppose all strings of length at most m appear as a substring of some string in S, where $m \geqslant 4$. We show that the same holds for strings of length $m + 1$.

Consider an arbitrary $w = a_1a_2 \cdots a_ma_{m+1}$. We now consider two cases:

i) If all three letters in the alphabet occur at least once in $a_{m-3}a_{m-2}a_{m-1}a_m$, then a_{m+1} equals a_{m-3}, a_{m-2}, a_{m-1}, or a_m, and w can be generated as a substring by a tandem duplication of some suffix of size $\leqslant 4$ of $w' = a_1a_2 \cdots a_m$. Note that by the induction hypothesis w' can be generated as a substring of some string.

ii) If at least one letter in the alphabet does not occur in $a_{m-3}a_{m-2}a_{m-1}a_m$, then $a_{m-3}a_{m-2}a_{m-1}a_m$ is a sequence over binary alphabet and so it has a tandem repeat. Therefore w can be generated as a substring by tandem duplication. Hence, we have proved the Theorem. ▮

Table 2.3 summarizes the result of this subsection. It can be observed from the table that a change of behavior in expressiveness occurs when the size of the alphabet increases to 4. If the size of the alphabet is 1, 2, or 3, for sufficiently large maximum duplication length, the systems are fully expressive. However, if the size of the alphabet is at least 4, then regardless of the maximum duplication length, the system is not fully expressive. This change is related to the fact that for alphabets of size 1 and 2, all squarefree strings are of finite length, but for alphabets of size 3 and larger, there are squarefree strings of any length. Specifically, in case ii) in the proof of Theorem 2.8, we used the fact that the binary string $a_{m-3}a_{m-2}a_{m-1}a_m$ has a tandem repeat. To adapt this proof for $|\Sigma| \geqslant 4$, we would need to show that the $(|\Sigma| - 1)$-ary string $a_{m-3}a_{m-2}a_{m-1}a_m$ has a tandem repeat. But this is not in general true, since there are squarefree strings over alphabets of size at least 3 per Thue's result [17]

Σ	s	k	fully expressive	Reason
$\{0\}$	0	$\geqslant 1$	Yes	Trivial
$\{0,1\}$	arbitrary	1	No	Lemma 2.4
$\{0,1\}$	01	$\geqslant 2$	Yes	Example 2.1
$\{0,1,2\}$	arbitrary	$\leqslant 3$	No	Theorem 2.5
$\{0,1,2\}$	012	$\geqslant 4$	Yes	Theorem 2.8
$\lvert\Sigma\rvert \geqslant 4$	arbitrary	arbitrary	No	Theorem 2.7

Table 2.3: Expressiveness of tandem duplication string systems $(\Sigma, s, \mathcal{T}^{tan}_{\leqslant k})$.

and indeed we showed in Theorem 2.7, again using Thue's result, that the system $\left(\Sigma, s, \mathcal{T}^{tan}_{\leqslant k}\right)$ is not fully expressive for $|\Sigma| \geqslant 4$ and any k.

2.4 Regular Languages for Tandem Duplication String Systems

Tandem duplication string systems that define regular languages are easier to study due to the fact that one can use tools from the Perron-Frobenius theory [14, 15] to calculate capacity. It was proved in [63] that for $|\Sigma| \geqslant 3$ and maximum duplication length $\geqslant 4$, the language defined by tandem duplication string systems is not regular, if the seed contains abc as a substring such that a, b and c are distinct. However, if the maximum duplication length is 3, this question was left unanswered. In Theorem 2.9, we show that the language resulting from a tandem duplication system with the maximum duplication length of 3 is regular regardless of the alphabet size and seed. Further, in Corollary 2.10 we characterize the exact capacity of such tandem duplication string systems.

Theorem 2.9. *Let $S = (\Sigma, s, \mathcal{T}^{tan}_{\leqslant 3})$, where Σ and s are arbitrary. The language defined by S is regular.*

Proof. We first assume that $s = a_1 \cdots a_m$, where a_i are distinct. The case in which a_i are not distinct is handled later.

For $3 \leqslant j \leqslant m$, let

$$\begin{aligned} R_{a_1\cdots a_j} = a_1^+ a_2^+ \left(a_1^+ a_2^+\right)^* a_3^+ \left(a_2^+ a_3^+\right)^* {B_{a_1a_2a_3}}^* \\ a_4^+ \left(a_3^+ a_4^+\right)^* {B_{a_2a_3a_4}}^* \\ \cdots \\ a_i^+ \left(a_{i-1}^+ a_i^+\right)^* {B_{a_{i-2}a_{i-1}a_i}}^* \\ \cdots \\ a_j^+ \left(a_{j-1}^+ a_j^+\right)^* {B_{a_{j-2}a_{j-1}a_j}}^*, \end{aligned}$$

Σ	s	k	Capacity	Fully Expressive
$\{0\}$	0^m for some $m \geqslant 1$	1	1	Yes
$\{0,1\}$	01	1	0	No
$\{0,1\}$	arbitrary but not a^m for some $a \in \{0,1\}$	$\geqslant 2$	1	Yes
$\lvert\Sigma\rvert \geqslant 3$	arbitrary but not a^m for some $a \in \Sigma$	2	$\log_{\lvert\Sigma\rvert} 2$	No
$\lvert\Sigma\rvert \geqslant 3$	$xabcy$ (x and $y \in \Sigma^*$, a, b and $c \in \Sigma$ and $a \neq b \neq c \neq a$)	3	$\log_{\lvert\Sigma\rvert} \frac{3+\sqrt{5}}{2}$	No
$\lvert\Sigma\rvert \geqslant 3$	No 3 consecutive symbols in the seed are all distinct and $s \neq a^m$ for $a \in \Sigma$	3	$\log_{\lvert\Sigma\rvert} 2$	No
$\{0,1,2\}$	012	$\geqslant 4$	?	Yes
$\lvert\Sigma\rvert \geqslant 4$	arbitrary	$\geqslant 4$	?	No

Table 2.4: Capacity and Expressiveness for different tandem duplication string systems $(\Sigma, s, \mathcal{T}^{tan}_{\leqslant k})$.

where, for $a, b, c \in \Sigma$,

$$B_{abc} = a^+(c^+a^+)^*b^+(a^+b^+)^*c^+(b^+c^+)^*.$$

We already know from Theorem 2.3 that $S = (\Sigma, s, \mathcal{T}^{tan}_{\leqslant 3})$ with $s = a_1 \cdots a_m$ is a regular language if $m = 3$. We show that for $m \geqslant 4$, S represents a regular language whose regular expression is given by $R_{a_1a_2\cdots a_m}$. Let L_R be the language defined by $R_{a_1a_2\cdots a_m}$. It suffices to show $L_R = S$.

We first show that $L_R \subseteq S$ by proving $R_{a_1a_2\cdots a_m} \xrightarrow{dd_{\leqslant 3}} s$. To do so, we show by induction that $R_{a_1a_2\cdots a_i} \xrightarrow{dd_{\leqslant 3}} a_1a_2\cdots a_i$. First note that this holds for $i = 3$, from the proof of Theorem 2.3. Assuming that it holds for i, to show that this also holds for $i+1$, where $i \geqslant 3$, we write

$$\begin{aligned}
R_{a_1a_2\cdots a_{i+1}} &= R_{a_1a_2\cdots a_i} a_{i+1}^+ \left(a_i^+ a_{i+1}^+\right)^* B_{a_{i-1}a_ia_{i+1}}{}^* \\
&\xrightarrow{dd_{\leqslant 3}} a_1a_2\cdots a_ia_{i+1}(a_ia_{i+1})^* B_{a_{i-1}a_ia_{i+1}}{}^* \\
&\xrightarrow{dd_{\leqslant 3}} a_1a_2\cdots a_ia_{i+1} B_{a_{i-1}a_ia_{i+1}}{}^* \\
&\xrightarrow{dd_{\leqslant 3}} \{a_1a_2\cdots a_ia_{i+1}(a_{i-1}a_ia_{i+1})^*, \\
&\qquad a_1a_2\cdots a_ia_{i+1}(a_{i-1}a_{i+1}a_{i-1}a_ia_{i+1})^*\} \\
&\xrightarrow{dd_{\leqslant 3}} a_1a_2\cdots a_ia_{i+1}.
\end{aligned}$$

Here we have used the fact that $cB_{abc} \xrightarrow{dd_{\leqslant 3}} cabc$ which follows from (2.4). Hence, $L_R \subseteq S$.

We now show that $S \subseteq L_R$. The finite automaton for L_R is given in Figure 2.3. Note that the seed s is in L_R. It thus suffices to show that if $x = pqr \in L_R$, then $y = pq^2r \in L_R$, where $p, q, r \in \Sigma^*$ and $|q| \leqslant 3$. We prove this by showing that any length-1, 2 or 3 path ending in any state of the finite automaton in Figure 2.3 is

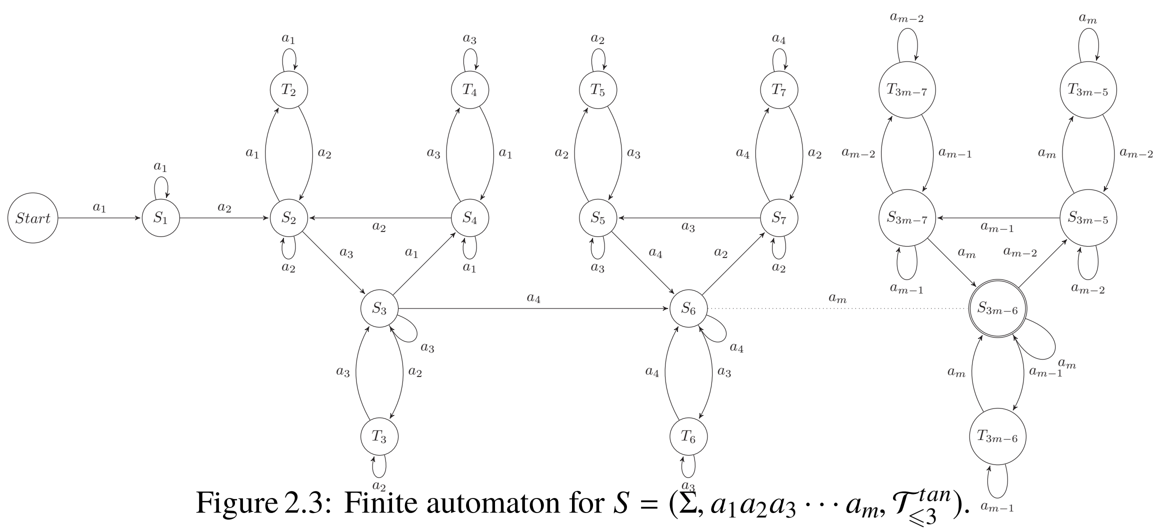


Figure 2.3: Finite automaton for $S = (\Sigma, a_1a_2a_3\cdots a_m, \mathcal{T}^{tan}_{\leqslant 3})$.

duplicable, or in other words (2.5) holds for all the states in Figure 2.3. The finite automaton in Figure 2.3 is a generalization of the one in Figure 2.2. Note that in Figure 2.3, the states $\{S_1, S_2, S_3, S_4, T_2, T_3, T_4\}$ are exactly the same as those in Figure 2.2. More precisely, there is no additional path ending in these states in Figure 2.3. So, from the proof of Theorem 1, (2.5) holds for these states.

Now we show for the newly added states, i.e. $\{S_i, T_i : i \geqslant 5\}$, (5) holds. Consider a set $Q_k = \{S_{3k-1}, S_{3k}, S_{3k+1},$
$T_{3k-1}, T_{3k}, T_{3k+1}\}$ for some $k \geqslant 2$. The labelled adjacency matrix $\mathcal{A}$ for the subgraph induced by these states is given by

$$\mathcal{A} = \begin{bmatrix} y & z & 0 & x & 0 & 0 \\ 0 & z & x & 0 & y & 0 \\ y & 0 & x & 0 & 0 & z \\ y & 0 & 0 & x & 0 & 0 \\ 0 & z & 0 & 0 & y & 0 \\ 0 & 0 & x & 0 & 0 & z \end{bmatrix},$$

where x is used as a label for a_k, y for a_{k+1} and z for a_{k+2}. $\mathcal{A}^2$ is given by

$$\mathcal{A}^2 = \begin{bmatrix} y^2+xy & z^2+yz & zx & x^2+yx & zy & 0 \\ xy & z^2+yz & x^2+zx & 0 & y^2+zy & xz \\ y^2+xy & yz & x^2+zx & yx & 0 & z^2+xz \\ y^2+xy & yz & 0 & x^2+yx & 0 & 0 \\ 0 & z^2+yz & zx & 0 & y^2+zy & 0 \\ xy & 0 & x^2+zx & 0 & 0 & z^2+xz \end{bmatrix},$$

The non-zero terms in the non diagonal entries of each column also appear in the diagonal entry of that column for $\mathcal{A}$ and $\mathcal{A}^2$. This can also be verified for $\mathcal{A}^3$. Hence, we have shown that any length-1, 2, 3 path starting in some state $C \in Q_k$ and ending in some state $D \in Q_k$ is duplicable for all $k \geqslant 2$.

We also need to show that any length-1, 2, 3 path that ends in Q_k but starts in a state that is not in Q_k is duplicable. Note that the states in Q_k are not reachable from states $\in Q_{k'}$ with $k' > k$, any possible path of length-1, 2 or 3 ending in some state $D \in Q_k$ and starting in a state $C \in Q_{k'}$ with $k' < k$, has to pass through state S_{3k-3}. Now, we enumerate the labels of all length-1, 2 and 3 paths ending in some state in Q_k but starting in some state $\in Q_{k'}$ with $k' < k$.

- Label of a path of length 1: a_{k+2} (ends in S_{3k}).
- Label of a path of length 2: i) ending in S_{3k}: $a_{k+1}a_{k+2}$, $a_{k+2}a_{k+2}$, ii) ending in S_{3k+1}: $a_{k+2}a_k$, iii) ending in T_{3k}: $a_{k+2}a_{k+1}$.
- Label of a path of length 3: i) ending in S_{3k-1}: $a_{k+2}a_ka_{k+1}$, ii) ending in S_{3k}: $a_{k+1}a_{k+1}a_{k+2}$, $a_ka_{k+1}a_{k+2}$, $a_{k+1}a_{k+2}a_{k+2}$, $a_{k+2}a_{k+2}a_{k+2}$, $a_{k+2}a_{k+1}a_{k+2}$, iii) ending in S_{3k+1}:$a_{k+1}a_{k+2}a_k$, $a_{k+2}a_ka_k$, $a_{k+2}a_{k+2}a_k$, iv) ending in T_{3k}: $a_{k+1}a_{k+2}a_{k+1}$, $a_{k+2}a_{k+1}a_{k+1}$, $a_{k+2}a_{k+2}a_{k+1}$, v) ending in T_{3k+1}: $a_{k+2}a_ka_{k+2}$.

All the path labels enumerated above are duplicable which can be verified by inspecting $\mathcal{A}$, $\mathcal{A}^2$, $\mathcal{A}^3$, for paths of length 1, 2 and 3 respectively. This completes the proof of $S \subseteq L_R$.

We have proved the statement of Theorem 2.9 assuming all a_i's in the seed s to be distinct. Now assume the symbols of s are not distinct. We color the symbols of s so that they become distinct and obtain the system $\tilde{S} = \left(\tilde{\Sigma}, \tilde{s}, \mathcal{T}^{tan}_{\leqslant 3}\right)$. Applying the preceding proof for distinct symbols to $\tilde{S}$, we find that $\tilde{S}$ is regular. Let $h : \tilde{\Sigma} \to \Sigma$ be a mapping that removes the colors. This mapping is called a morphism. By [65], we have that $S = h(\tilde{S})$ is also regular. ∎

An immediate corollary on the capacity of tandem duplication string system considered in Theorem 2.9 is stated next.

Corollary 2.10. *If for S in Theorem 2.9, s contains abc as a substring such that a, b, and c $\in \Sigma$ are distinct, then* $\mathrm{cap}(S) = \log_{|\Sigma|} \frac{3+\sqrt{5}}{2} \simeq 0.876036 \log_{|\Sigma|} 3$. *Otherwise, except for the seed s of the form a^m,* $\mathrm{cap}(S) = \log_{|\Sigma|} 2$. *If $s = a^m$,* $\mathrm{cap}(S) = 0$.

Proof. If abc occurs as a substring of the seed s such that a, b and $c \in \Sigma$ are distinct, then the adjacency matrix of the finite automaton for B_{abc} (strongly connected component of the finite automaton for $R_{a_1a_2\cdots a_m}$) has the maximum eigenvalue.

Therefore, the $\mathrm{cap}(S) = \log_{|\Sigma|}\frac{3+\sqrt{5}}{2} \simeq 0.876036\log_{|\Sigma|}3$ (see (2.6) in the proof of Theorem 2.3).

If no 3 consecutive symbols in the seed s are all distinct and $s \neq a^m$, then the maximum capacity component is a finite automaton only over 2 distinct symbols as in Figure 2.1. In other words, terms of the form $(a_i^+a_{i+1}^+)^*$ determine the capacity. Hence the capacity is $\log_{|\Sigma|}2$.

When seed $s = a^m$, there is exactly one sequence of any given length in the system. Hence $\mathrm{cap}(S) = 0$. ∎

The following examples illustrate the statement of Theorem 2.9 and an application of its proof method.

Example 2.11. The string system $S = (\{0,1,2,3\}, 0123, \mathcal{T}^{tan}_{\leqslant 3})$ is regular by Theorem 2.9 and the regular expression is given by

$$R_{0123} = 0^+1^+(0^+1^+)^*2^+(1^+2^+)^*{B_{012}}^*3^+(2^+3^+)^*{B_{123}}^*.$$

By Corollary 2.10, the capacity of this system $\simeq 0.876036\log_4 3 \simeq 0.694242$. □

Example 2.12. The string system $S = (\{0,1,2\}, 0112, \mathcal{T}^{tan}_{\leqslant 3})$ is regular by Theorem 2.9, and the regular expression is given by

$$R_{0112} = 0^+1^+(0^+1^+)^*1^+(1^+1^+)^*{B_{011}}^*2^+(1^+2^+)^*{B_{112}}^*.$$

By Corollary 2.10, the capacity of this system is given by $\log_3 2 \simeq 0.63093$. □

When a_i's are assumed to be distinct it can be verified from the regular expression $R_{a_1\cdots a_j}$ in the proof of Thereom 2.9 that the last occurence of a_i is before the first occurence of a_{i+3} for any $i = 1, 2, \cdots, j-3$ for all $z \in S$.

The following corollary follows for maximum duplication length 2 using the same idea as in Theorem 2.9

Corollary 2.13. *The capacity for $S = (\Sigma, a_1a_2\cdots a_m, \mathcal{T}^{tan}_{\leqslant 2})$ is given by* $\log_{|\Sigma|}2$, *except for the case in which seed $s = a^m$ for $a \in \Sigma$. In that case, the capacity is* 0.

Proof. The string system $S = (\Sigma, a_1a_2\cdots a_m, \mathcal{T}^{tan}_{\leqslant 2})$ is regular. This can be proved using the same method as used in the proof of Theorem 2.9. The regular expression $Q_{a_1a_2\cdots a_m}$ for $m \geqslant 2$ is given by

$$Q_{a_1a_2\cdots a_m} = a_1^+a_2^+(a_1^+a_2^+)^*a_3^+(a_2^+a_3^+)^*\cdots a_m^+(a_{m-1}^+a_m^+)^*.$$

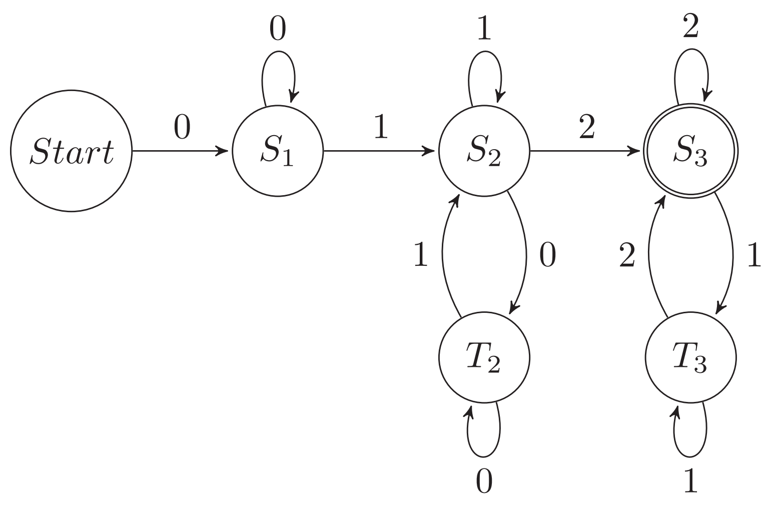


Figure 2.4: Finite automaton for $S = (\{0,1,2\}, 012, \mathcal{T}^{tan}_{\leqslant 2})$. The regular expression $R = 0^+1^+(0^+1^+)^*2^+(1^+2^+)^*$.

As in Proof of Corollary 2.10, the capacity is determined by term(s) of the form $(a_i^+ a_{i+1}^+)^*$, except for the case when seed $s = a^m$. Therefore, the capacity for language represented by $Q_{a_1a_2\cdots a_m}$ is $\log_{|\Sigma|} 2$, when $s \neq a^m$ and 0 when $s = a^m$. ∎

The finite automaton for a special case of Corollary 2.13 with $|\Sigma| = 3$ is given in Figure 2.4.

Table 2.4 lists the capacity and expressiveness results presented in this chapter and also the open question on capacity when $k \geqslant 4$. The expressiveness results follows from Table 2.3.

2.5 Conclusion

In this chapter, we showed that for tandem duplication string systems with bounded duplication length if the maximum duplication length is 3 or less, the language described by the string system is regular. Further, we computed exact capacities for these systems. Computing the capacities for bounded tandem duplication string systems with maximum duplication length greater than 3 remains an open problem.

Using Thue's result [17], we showed that a tandem duplication string system cannot be fully expressive if the alphabet size is $\geqslant 4$. However, for an alphabet of size 3 or less such systems can be fully expressive. Therefore, we have completely characterized fully expressive and non-fully expressive tandem duplication string systems with bounded duplication lengths.

We have studied questions related to the generation of a diverse set of sequences from a seed given bounded tandem duplication. One can also study the minimum number of steps required to generate a given sequence of length n from a squarefree seed and therefore define the notion of distance between a sequence and its seed given a tandem duplication rule. For the special case of binary sequences, we have studied this distance in the next chapter.

Chapter 3

DUPLICATION DISTANCE

3.1 Introduction

We focus on tandem duplication mutations and tandem repeats, and study the minimum number of tandem duplication events that can create a given sequence. More specifically, we define distance measures between pairs of sequences based on the number of exact or approximate tandem duplications that are needed to transform one sequence to the other. We then study the distances between sequences of length $n \in \mathbb{N}$ and their roots, i.e., the shortest sequences from which they can be obtained via these operations.

Formally, a (*tandem*) *repeat of length h* in a sequence is two identical adjacent blocks, each consisting of h consecutive elements. For example, the sequence $12\underline{134134}51$ contains the repeat 134134 of length 3. A repeat of length h may be created through a (*tandem*) *duplication of length h*, e.g., $1213451 \xrightarrow{d} 1213413451$, where $\xrightarrow{d}$ denotes a duplication operation. On the other hand, a repeat may be removed through a (*tandem*) *deduplication of length h*, i.e., by removing one of the two adjacent identical blocks, e.g., $1213413451 \xrightarrow{dd} 1213451$.

The *duplication/deduplication distance* between two sequences $\boldsymbol{x}$ and $\boldsymbol{y}$ is the smallest number of duplications and deduplications that can turn $\boldsymbol{x}$ into $\boldsymbol{y}$ (to denote sequences we use bold symbols). We set the distance to infinity if the task is not possible, for example, if $\boldsymbol{x} = 1$ and $\boldsymbol{y} = 0$.

For two sequences $\boldsymbol{x}$ and $\boldsymbol{y}$, if $\boldsymbol{y}$ can be obtained from $\boldsymbol{x}$ through duplications, we say that $\boldsymbol{x}$ is an *ancestor* of $\boldsymbol{y}$ and that $\boldsymbol{y}$ is a *descendant* of $\boldsymbol{x}$. An ancestor $\boldsymbol{x}$ of $\boldsymbol{y}$ is a *root* of $\boldsymbol{y}$, denoted $\boldsymbol{x} = \text{root}(\boldsymbol{y})$, if it is *square-free*, i.e., it does not contain a repeat. We define the *duplication distance* between two sequences as the minimum number of duplications required to convert the shorter sequence to the longer one.[1] This distance is finite if and only if one sequence is the ancestor of the other. This chapter is focused on finding bounds on the duplication distance of sequences to their roots. From an evolutionary point of view, the duplication distance between a sequence

[1]Note that using the term distance here is a slight abuse of notation as the duplication distance does not satisfy the triangle inequality.

and its root is of interest since it corresponds to a likely path through which a root may have evolved into a sequence present in the genome of an organism.

Our attention here is limited to binary sequences for the sake of simplicity, since for the binary alphabet, the root of every sequence is unique and belongs to the set $\{0, 1, 01, 10, 010, 101\}$. Specifically, the roots of 0^n and 1^n, $n \in \mathbb{N}$, are 0 and 1, respectively. For every other binary sequence $\boldsymbol{s}$ of length n, the root of $\boldsymbol{s}$ is the sequence in the set $\{01, 10, 010, 101\}$ that starts and ends with the same symbols as $\boldsymbol{s}$. For example, the root of $\boldsymbol{s} = 1001011$ is 101 since

$$101 \xrightarrow{d} \underline{1010}1 \xrightarrow{d} 1010\underline{11} \xrightarrow{d} 1\underline{00}1011 = \boldsymbol{s}.$$

A *run* in a sequence is a maximal substring consisting of one or more copies of a single symbol. Through duplication, we can generate every binary sequence from its root by first creating the correct number of runs of appropriate symbols. For example, since $\boldsymbol{s} = 1001011$ has 5 runs, the first being a run of the symbol 1, we first generate 10101 through duplication. It is not difficult to see that this is always possible. The next step is then to extend each run so that it has the appropriate length.

In the proofs in the chapter, it is often helpful to think of the distance to the root in terms of converting a sequence to its root via a sequence of deduplications, e.g. the sequence $\boldsymbol{s}$ above can be *deduplicated to* its root as

$$\boldsymbol{s} = 1\underline{00}1011 \xrightarrow{dd} 1010\underline{11} \xrightarrow{dd} \underline{1010}1 \xrightarrow{dd} 101 = \text{root}(\boldsymbol{s}).$$

We note that a celebrated result by Thue from 1906 [17] states that for alphabets of size $\geqslant 3$, there is an infinite square-free sequence. Thus, in contrast to the binary alphabet, the set of roots for such alphabets is infinite since each substring of Thue's sequence is square-free.

For a binary sequence $\boldsymbol{s}$, let $f(\boldsymbol{s})$ denote the duplication distance between $\boldsymbol{s}$ and its root and let $f(n)$ be the maximum of $f(\boldsymbol{s})$ for all sequences $\boldsymbol{s}$ of length n. Table 3.1, which was obtained through computer search, shows the values of $f(n)$ for $1 \leqslant n \leqslant 32$.

In this chapter, we provide bounds on $f(\boldsymbol{s})$ and on $f(n)$. We also consider a variation of the duplication distance, referred to as the *approximate-duplication distance*, where the duplication process is imprecise and so the inserted block is not necessarily an exact copy. Specifically, the *β-approximate-duplication distance*

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$f(n)$	0	1	2	2	3	4	4	5	6	6	7	7	8	8	9	9
n	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
$f(n)$	10	10	11	11	11	12	12	12	13	13	13	14	14	14	15	15

Table 3.1: $f(n)$ for $1 \leqslant n \leqslant 32$.

between two sequences $\boldsymbol{x}$ and $\boldsymbol{y}$ is the smallest number of duplications that can turn the shorter sequence into the longer one, where each duplication may produce a block that differs from the original in at most a β-fraction of positions. This distance between $\boldsymbol{s}$ and any of its roots is denoted by $f_\beta(\boldsymbol{s})$ and the maximum of $f_\beta(\boldsymbol{s})$ over all sequences $\boldsymbol{s}$ of length n is denoted by $f_\beta(n)$. We provide bounds on $f_\beta(n)$ and in particular show that there is a sharp transition in the behavior of f_β at $\beta = 1/2$.

Since each binary sequence has a unique root in the set $\{0, 1, 01, 10, 010, 101\}$, the set of sequences can be partitioned based on their roots. In the chapter, we also study the duplication distance to the root for sequences based on the part they belong to, that is, we consider $f_{\boldsymbol{\sigma}}(n)$ for $\boldsymbol{\sigma} \in \{0, 1, 01, 10, 010, 101\}$, where $f_{\boldsymbol{\sigma}}(n) = \max\{f(\boldsymbol{s}) : \text{root}(\boldsymbol{s}) = \sigma, |\boldsymbol{s}| = n\}$. The rest of the chapter is structured

as follows. In the next two subsections, we summarize the results of the chapter and describe the related work. Then, in Section 3.2, we prove the bounds on $f(n)$ and discuss some variants, as well as special classes of sequences. In Section 3.3, we discuss duplication distance for special class of sequence generating systems called Lindenmayer Systems. In Sections 3.4 and 3.5, we study the approximate-duplication distance to the root and the duplication distance for different roots, respectively. Finally, several open problems and possible future directions are presented in Section 4.7.

Results

In this subsection, we present the main results of the chapter. The proofs, unless they are very short, are postponed to later sections.

Suppose the root of $\boldsymbol{s}$ is $\boldsymbol{\sigma} \in \{0, 1, 01, 10, 010, 101\}$. It is easy to see that

$$\log \frac{|\boldsymbol{s}|}{|\boldsymbol{\sigma}|} \leqslant f(\boldsymbol{s}) \leqslant |\boldsymbol{s}|.$$

While the above lower bound is tight in the sense that there exist $\boldsymbol{\sigma}$ and $\boldsymbol{s}$ that satisfy it with equality, e.g., $\boldsymbol{s} = 0^{2^k}$ and $\boldsymbol{\sigma} = 0$, we show there is a positive constant c such that for most sequences of length n, the duplication distance to the root is bounded below by cn. We also improve the upper bound.

Theorem 3.1. *The limit* $\lim_{n\to\infty} f(n)/n$ *exists and*

$$0.045 \leqslant \lim_{n\to\infty} \frac{f(n)}{n} \leqslant \frac{2}{5}\,.$$

Furthermore, for sufficiently large n, $f(\boldsymbol{s}) \geqslant 0.045n$ *for all but an exponentially small fraction of sequences* $\boldsymbol{s}$ *of length n; and* $f(n) \leqslant 2n/5 + 15$.

Although the linear lower bound on the duplication distance to the root holds for almost all sequences, finding a specific family of sequences that satisfy it appears to be difficult. The next lemma and its corollary give the best known construction for a family with large distance to the root, namely, this family achieves distance $\Omega(n/\log n)$.

Lemma 3.2. *Consider a sequence* $\boldsymbol{s}$ *and a positive integer* $k \geqslant 4$*, and let* $K(\boldsymbol{s})$ *denote the number of distinct k-mers (sequences of length k) occurring in* $\boldsymbol{s}$*. We have*

$$f(\boldsymbol{s}) \geqslant \frac{K(\boldsymbol{s})}{k-1}\,.$$

Proof. For two sequences $\boldsymbol{x} = \boldsymbol{tuuv}$ and $\boldsymbol{y} = \boldsymbol{tuv}$, we have $K(\boldsymbol{y}) \geqslant K(\boldsymbol{x}) - (k-1)$, since the only case in which a k-mer occurs in $\boldsymbol{x}$ but not in $\boldsymbol{y}$ is when the only instance of that k-mer intersects both copies of $\boldsymbol{u}$ in $\boldsymbol{x}$. There are at most $k-1$ k-substrings of $\boldsymbol{x}$ that intersect both copies of $\boldsymbol{u}$. Finally, no root contains a k-mer for $k \geqslant 4$. ∎

A *binary De Bruijn sequence* [66] of order k is a binary sequence of length $n = 2^k$ that when viewed cyclically contains every possible binary sequence of length k as a substring exactly once. For example, 0011 and 00010111 are De Bruijn sequences of order 2 and order 3, respectively. A binary De Bruijn sequence of order k and length $n = 2^k$ has precisely $n - k + 1$ distinct k-mers. Hence, we have the following corollary.

Corollary 3.3. *For any binary De Bruijn sequence* $\boldsymbol{s}$ *of order k (which has length* $n = 2^k$*), we have*

$$f(\boldsymbol{s}) \geqslant \frac{n - \log_2 n}{\log_2 n}\,.$$

It is worth noting that using the same technique as the proof of $f(n) = \Omega(n)$ in Theorem 3.1, and the fact that there are at least $\frac{2^{n/2}}{n}$ De Bruijn sequences of length n when n is a power of two,[2] one can show that the largest duplication distance for De Bruijn sequences grows linearly in their length.

A question arising from observing that $f(n) = \Theta(n)$ is that how does allowing mismatches in the duplication process affect the distance to the root. In particular, for what values of β, is $f_\beta(n)$ linear in n and for what values is it logarithmic? The next theorem establishes that there is a sharp transition at $\beta = 1/2$.

Theorem 3.4. *If $\beta < 1/2$, then there exists a constant $c = c(\beta) > 0$ such that*

$$f_\beta(n) \geqslant cn.$$

Furthermore, if $\beta > 1/2$, for any constant $C > \left\lceil \frac{2\beta+1}{2\beta-1} \right\rceil^2$ and sufficiently large n,

$$f_\beta(n) \leqslant C \ln n.$$

Finally, we establish that the limit of $\frac{f(n)}{n}$ is the same if we consider only sequences with root 10 or only sequences with root 101.

Theorem 3.5. *The limits $\lim_n \frac{f_{10}(n)}{n}$ and $\lim_n \frac{f_{101}(n)}{n}$ exist and are equal to $\lim_n \frac{f(n)}{n}$.*

Related Work

Tandem duplications and repeats in sequences have been studied from a variety of points of view. One of the most relevant to this work is the study of estimating the tandem duplication history of a given sequence, i.e., a sequence of duplication events that may have generated the sequence, see e.g., [56, 67, 68]. While the aforementioned works study the problem from an algorithmic point of view, here we are focused on extremal distance values for binary sequences. Furthermore, [56, 68] have a more restrictive duplication model than that presented here. We also mention that the stochastic behavior of certain duplication systems has been studied in [69, 70].

3.2 Bounds on $f(n)$

Theorem 3.1. *The limit $\lim_{n\to\infty} f(n)/n$ exists and*

$$0.045 \leqslant \lim_{n\to\infty} \frac{f(n)}{n} \leqslant \frac{2}{5}\,.$$

[2]If De Bruijn seqences are defined cyclically as opposed to linearly, there are exactly $\frac{2^{n/2}}{n}$ De Bruijn sequences of length n

Furthermore, for sufficiently large n, $f(\boldsymbol{s}) \geqslant 0.045n$ for all but an exponentially small fraction of sequences $\boldsymbol{s}$ of length n; and $f(n) \leqslant 2n/5 + 15$.

The lower bound of Theorem 3.1 is proved with the help of Theorem 3.6, and its upper bound uses Theorem 3.9. These theorems are stated next.

Theorem 3.6. *For $0 < \alpha < 1$, consider the set of the $\lfloor 2^{n\alpha} \rfloor$ sequences of length n with the smallest duplication distance to the root and let F_α be the maximum duplication distance to the root for a sequence in this set. Then*

$$6n \sum_{f=1}^{F_\alpha} \binom{n+f}{f}\binom{2n+f}{f}\binom{2n+f+2}{f} 2^f \geqslant 2^{n\alpha} - 1. \tag{3.1}$$

Before stating the proof, we present some background, definitions, and a useful claim, as well as a simpler but weaker result.

Recall that if the sequence $\boldsymbol{s} = s_1 s_2 \cdots s_m$ contains a repeat, then omitting one of the two blocks of this repeat to obtain a new sequence is called a deduplication. We also refer to the resulting sequence $\boldsymbol{s}'$ as a deduplication of $\boldsymbol{s}$, and write $\boldsymbol{s} \xrightarrow{dd} \boldsymbol{s}'$. A *deduplication process* for a binary sequence $\boldsymbol{s}$ is a sequence of sequences $\boldsymbol{s} = \boldsymbol{s}_0 \xrightarrow{dd} \boldsymbol{s}_1 \xrightarrow{dd} \boldsymbol{s}_2 \xrightarrow{dd} \cdots \xrightarrow{dd} \boldsymbol{s}_f = \text{root}(\boldsymbol{s})$, where each $\boldsymbol{s}_{i+1}$ is a deduplication of $\boldsymbol{s}_i$ and the final sequence $\boldsymbol{s}_f$ is the (square-free) root of $\boldsymbol{s}$. The *length* of the deduplication process above is f, that is, the number of deduplications in it. A deduplication of $\boldsymbol{s}$ is an (i,h)-*step* if i is the starting position of (the first block) of a repeat of length h and one of the blocks of this repeat is omitted. For example, if $\boldsymbol{s} = 123\underline{134}13451$, a $(4,3)$-step produces $\boldsymbol{s}' = 12313451$. A deduplication process of length f of a sequence $\boldsymbol{s}$ can be described by a sequence of pairs $(i_t, h_t)_{t=1}^{f}$, where step number t is an (i_t, h_t)-step. It is not difficult to check that knowing the final sequence in the process, and knowing all the pairs (i_t, h_t) of deduplications in the process, in order, we can reconstruct the original sequence $\boldsymbol{s}$.

From the preceding discussion, each binary sequence $\boldsymbol{s}$ can be encoded as the pair $\left(\boldsymbol{\sigma}, (i_t, h_t)_{t=1}^{f(s)}\right)$, where $\boldsymbol{\sigma}$ is the root of $\boldsymbol{s}$ and $(i_t, h_t)_{t=1}^{f(s)}$ a deduplication process of $\boldsymbol{s}$. Since there are only 6 possibilities for $\boldsymbol{\sigma}$, and less than n^2 possibilities for each pair (i_t, h_t), if $F = f(n)$, then

$$6 \sum_{f=1}^{F} \left(n^2\right)^f \geqslant 2^n, \tag{3.2}$$

which implies that $F = f(n) = \Omega(n/\log n)$.

In the aforementioned encoding, several deduplication processes may map to the same sequence. We improve upon (3.2) by defining deduplication processes of a special form that remove some of the redundancy, and by doing so, we obtain (3.1), which will lead to the linear lower bound of Theorem 3.1.

Definition 3.7. A deduplication process $s = s_0 \xrightarrow{dd} s_1 \xrightarrow{dd} s_2 \xrightarrow{dd} \cdots \xrightarrow{dd} s_f = \text{root}(s)$ of a sequence s, in which the steps are $(i_1, h_1), (i_2, h_2), \ldots, (i_f, h_f)$, is *normal* if the following condition holds: For any $1 \leqslant t < f$, if $i_{t+1} < i_t$ then $i_{t+1} + 2h_{t+1} \geqslant i_t$.

The following claim shows that if we limit ourselves to normal deduplication processes, we can still encode every binary sequence with processes of the same length.

Claim 3.8. *For any deduplication process* $s = s_0 \xrightarrow{dd} s_1 \xrightarrow{dd} s_2 \xrightarrow{dd} \cdots \xrightarrow{dd} s_f = \text{root}(s)$ *of length* f *of a sequence* s*, there is a normal deduplication process* $s = s_0 \xrightarrow{dd} s'_1 \xrightarrow{dd} s'_2 \xrightarrow{dd} \cdots \xrightarrow{dd} s'_f = s_f$ *of the same length, with the same final sequence.*

Proof. Among all deduplication processes of length f starting with s and ending with s_f, consider the one minimizing the number of pairs (i_t, h_t), (i_q, h_q) with $1 \leqslant t < q \leqslant f$, and $i_q < i_t$. We claim that this process is normal. Indeed, otherwise there is some t, $1 \leqslant t < f$ so that $i_{t+1} < i_t$ and $i_{t+1} + 2h_{t+1} < i_t$. But in this case we can switch the steps (i_t, h_t) and (i_{t+1}, h_{t+1}), performing the step (i_{t+1}, h_{t+1}) just before (i_t, h_t). This will clearly leave all sequences $s_0, s_1, \ldots, s_f$, besides s_t, the same, and in particular $s_0 = s$ and $s_f = \text{root}(s)$ stay the same. This contradicts the minimality in the choice of the process, establishing the claim. ∎

We now turn to the proof of Theorem 3.6.

Proof of Theorem 3.6. Let U_α denote the set of $\lfloor 2^{n\alpha} \rfloor$ sequences that have the smallest duplication distances to their roots among binary sequences of length n and recall that $F_\alpha = \max\{f(s) : s \in U_\alpha\}$. By Claim 3.8, for each of the sequences s of U_α, there is a normal deduplication process $s = s_0 \xrightarrow{dd} s_1 \xrightarrow{dd} s_2 \xrightarrow{dd} \cdots \xrightarrow{dd} s_f$ of length $f \leqslant F_\alpha$. Let the steps of this process be $(i_1, h_1), (i_2, h_2), \ldots, (i_f, h_f)$. As before, it is clear that knowing the final sequence s_f and all the pairs (i_t, h_t), we can reconstruct s. There are 6 possibilities for s_f. As each step (i_t, h_t) reduces the length of the sequence by h_t, it follows that $\sum_{i=1}^{f} h_t < n$ and therefore there are at most $\binom{n+f}{f}$ possibilities for the sequence $(h_1, h_2, h_3, \ldots, h_f)$. In order to record the sequence $(i_1, i_2, \ldots, i_f)$ it suffices to record i_1 and all the differences $i_t - i_{t+1}$ for all $1 \leqslant t < n$.

There are less than n possibilities for i_1, and there are at most 2^f possibilities for deciding about the set of all indices t for which the difference $i_t - i_{t+1}$ is positive. As the process is normal, for each such positive difference, we know that $i_{t+1} + 2h_{t+1} \geqslant i_t$, that is $i_t - i_{t+1} \leqslant 2h_{t+1}$. It follows that the sum of all positive differences, $\sum_{t:i_t - i_{t+1}>0}(i_t - i_{t+1})$, is at most $2\sum_t h_t < 2n$, and hence the number of choices for these differences is at most $\binom{2n+f}{f}$.

Since $i_f \leqslant 3$, we have $i_1 - i_f \geqslant 1 - 3 = -2$. So

$$\sum_{t:i_t - i_{t+1}\leqslant 0}(i_t - i_{t+1}) = (i_1 - i_f) - \sum_{t:i_t - i_{t+1}>0}(i_t - i_{t+1}) > -2 - 2n.$$

Therefore, the number of choices for all non-positive differences $i_t - i_{t+1}$ is at most $\binom{2n+f+2}{f}$. Putting all of these together, and noting that $|U_\alpha| \geqslant 2^{n\alpha} - 1$, implies the assertion of Theorem 3.6. ∎

Since $\binom{p}{q} \leqslant 2^{pH(q/p)}$ for positive integers $0 < q < p$ [71, p. 309], Theorem 3.6 implies that

$$3\left(2 + \frac{F_\alpha}{n}\right)H\left(\frac{F_\alpha/n}{2 + F_\alpha/n}\right) + \frac{F_\alpha}{n} \geqslant \alpha + o(1),$$

where H is the binary entropy function, $H(x) = -x\log_2 x - (1-x)\log_2(1-x)$. The expression on the left side of the inequality is strictly increasing in $\frac{F_\alpha}{n}$, and it is less than 0.99 if we substitute $\frac{F_\alpha}{n}$ by 0.045. If we let $\alpha = 0.99$, it follows that for sufficiently large n, we have $\frac{F_\alpha}{n} \geqslant 0.045$, thereby establishing the lower bound in Theorem 3.1.

To prove the upper bound in Theorem 3.1, we prove the following theorem.

Theorem 3.9. *The limit* $\lim_{n\to\infty} f(n)/n$ *exists and for all n,* $f(n) \leqslant \frac{2}{5}n + 15$.

Proof. Note that for any positive integers n and m, $f(n+m) \leqslant f(n) + f(m) + 2$. Indeed, given a sequences of length $n + m$ we can deduplicate separately its first n bits and its last m bits, getting a concatenation of two square-free sequences (of total length at most 6). It then suffices to check that each such concatenation can be deduplicated to its root through at most 2 additional deduplication steps. Therefore, the function $g(n) = f(n) + 2$ is subadditive:

$$g(n+m) = f(n+m) + 2 \leqslant f(n) + f(m) + 4 = g(n) + g(m).$$

Now, by Fekete's Lemma [72], $g(n)/n$ tends to a limit (which is the infimum over n of $g(n)/n$), and it is clear that the limit of $f(n)/n$ is the same as that of $g(n)/n$. We term this limit the *binary duplicatoin constant*.

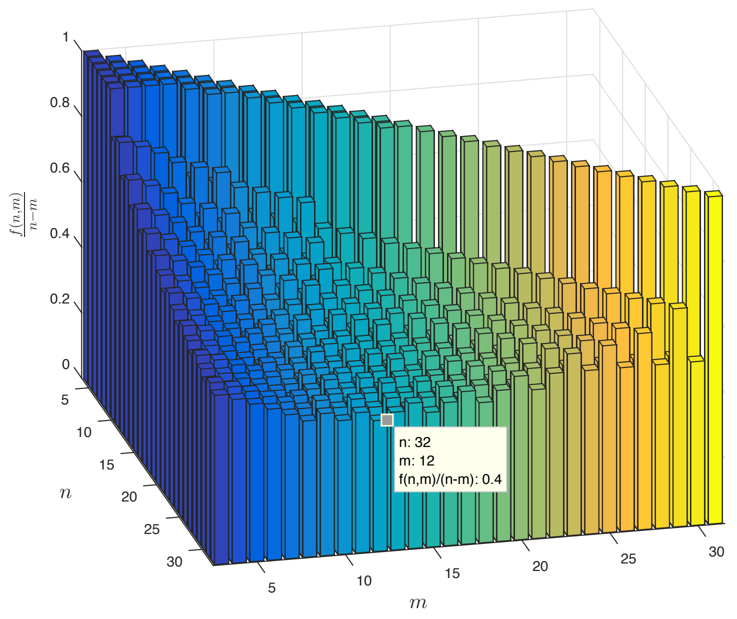


Figure 3.1: $\frac{f(n,m)}{n-m}$ for $3 \leqslant m < n \leqslant 32$.

This proof of the existence of $\lim_{n\to\infty} f(n)/n$ provides a simple way to derive an upper bound for the limit by computing $f(n)$ precisely for some small n. In particular, from Table 3.1, we find $\lim_{n\to\infty} f(n)/n \leqslant (f(32)+2)/32 = 17/32$. We can improve upon this result as follows.

For positive integers n, m, let $f(n, m)$ be the smallest number k such that every sequence of length n can be converted to a sequences of length at most m via k deduplication steps. A sequence of length n can be converted to its root by first repeatedly converting its a-substrings to substrings of length at most b via $f(a, b)$ deduplication steps. Thus for integers $a > b > 0$, we have

$$f(n) \leqslant \frac{f(a,b)}{a-b} n + \max_{i<a} f(i) \tag{3.3}$$

With the help of a computer we find the values of $f(n, m)$ for $3 \leqslant m < n \leqslant 32$. An illustration is given in Figure 3.1. In particular we have $\frac{f(32,12)}{20} = \frac{8}{20} = \frac{2}{5}$ from Figure 3.1 and $\max_{i<32} f(i) = 15$ from Table 3.1, implying $f(n) \leqslant \frac{2}{5}n + 15$. ∎

Weaker upper bounds on $f(n)$ can be obtained without resorting to computation in the following ways. First, to deduplicate a sequence to its root, we first can deduplicate each block of t consecutive identical bits to a single bit by $\lceil \log_2 t \rceil$ deduplications and then finish in less than $\log_2 n$ additional steps. This shows that for large n , $f(n) \leqslant \frac{2}{3}n + o(n)$ (the extremal case for this argument is the one in which each block is of size 3). Second, it is known that every binary sequence of length at least 19 contains a repeat of length at least 2 [73], implying that $f(n) \leqslant \frac{1}{2}n + o(n)$.

Parallel duplication One can also define the parallel duplication distance to the root by allowing non-overlapping duplications to occur simultanously, with $f'(n)$ being the maxmimum parallel duplication distance to the root of a sequence of length n. Similar to the normal duplication distance it is helpful to think in terms of deduplications. Since each parallel deduplication step decreases the length of a sequence by at most a factor of 2, $f'(n) > \log_2 n - 2$ (and in fact $f'(s) \geqslant \log_2 n - 2$ for every sequence of length n.) It is not difficult to see that $f'(n) < 2\log_2 n$ by first deduplicating, in parallel, all blocks of identical elements in the sequence to blocks of size 1, and then by deduplicating the resulting alternating sequence to its root.

Partial deduplication The definition of $f(n, m)$ gives rise to the following question: For a fixed $0 < \alpha \leqslant 1$, what is $\lim_n \frac{f(n,\lfloor \alpha n \rfloor)}{1-\alpha}$, if it exists? At first glance, one may expect $\lim_n \frac{f(n,\lfloor \alpha n \rfloor)}{1-\alpha}$ to be decreasing in α since if α is large, one may think it is easier to find enough long repeats to reduce the length of the sequence quickly by a factor of $1 - \alpha$. However, we show that $\lim_n \frac{f(n,\lfloor \alpha n \rfloor)}{n(1-\alpha)} = \lim_n \frac{f(n)}{n}$.

Let $\gamma = \lim_n \frac{f(n)}{n}$. For $\epsilon > 0$, there exists k such that for all $n > k$, $f(n) \leqslant (\gamma + \epsilon)n$. Thus

$$f(n, \lfloor \alpha n \rfloor) \leqslant f(n - \lfloor \alpha n \rfloor + 3) \leqslant (\gamma + \epsilon)((1 - \alpha)n + 4). \tag{3.4}$$

On the other hand, let $\delta = \liminf_n \frac{f(n,\lfloor \alpha n \rfloor)}{(1-\alpha)n}$. For $\epsilon > 0$, there exists k such $f(k, \lfloor \alpha k \rfloor) \leqslant (\delta + \epsilon)(1 - \alpha)k$. Hence,

$$f(n) \leqslant \frac{f(k, \lfloor \alpha k \rfloor)}{k - \lfloor \alpha k \rfloor} n + k \leqslant (\delta + \epsilon)n + k. \tag{3.5}$$

The result follows by dividing (3.4) by $(1 - \alpha)n$ and taking a $\limsup_n$ and by dividing (3.5) by n and taking a $\lim_n$.

3.3 Duplication Distance for L-systems

L-systems, or Lindenmayer systems are sequence rewriting systems developed by Lindemayer in 1968 [74]. He used them in the context of biology to model the growth process of plant development. He introduced context-free as well as context-sensitive L-systems. Here we will discuss distance to the root for sequences arising in context-free L-systems, also known as 0L-systems. A 0L-system comprises three components:

- Alphabet (Σ): An alphabet of symbols used to construct sequences.
- Axiom sequence or initiator (ω): The starting sequence from which a 0L-system is constructed.
- Production rule (h): A rule that constructs new sequences by expanding each symbol in a given sequence into a sequence of symbols. The production rule is represented by the function $h : \Sigma^* \to \Sigma^*$, which for any two sequences $\boldsymbol{a}$ and $\boldsymbol{b} \in \Sigma^*$ satisfies
$$h(\boldsymbol{a}\boldsymbol{b}) = h(\boldsymbol{a})h(\boldsymbol{b})$$
where $h(\boldsymbol{a})h(\boldsymbol{b})$ represents the concatenation of $h(\boldsymbol{a})$ and $h(\boldsymbol{b})$. The production rule h can be deterministic or stochastic. Here we consider only deterministic rules. Such 0L-systems with deterministic h are denoted as D0L-systems [75].

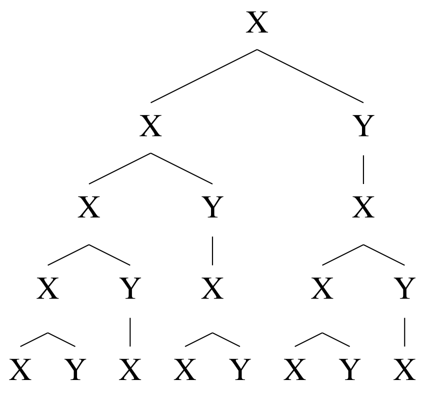
Example 3.10 (*Fibonacci words*)**.** Consider $\Sigma = \{X, Y\}$, $\omega = X$, and

$$h(X) = XY, \quad h(Y) = X.$$

For this D0L-system, the first 5 sequences are as follows:

$$\begin{aligned} h^0(\omega) &= X \\ h^1(\omega) &= XY \\ h^2(\omega) &= XYX \\ h^3(\omega) &= XYXXY \\ h^4(\omega) &= XYXXYXYX \\ h^5(\omega) &= XYXXYXYXXYXXY \end{aligned}$$

This can also be represented by the following tree:



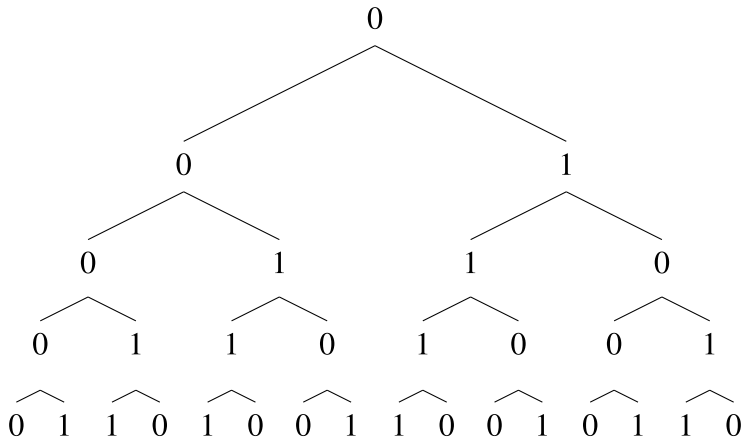
These sequences are called Fibonacci words as they satisfy

$$h^n(\omega) = h^{n-1}(\omega)h^{n-2}(\omega)\ \forall\ n \geqslant 2.$$

Example 3.11 (*Thue-Morse Sequence*)**.** Let $\Sigma = \{0, 1\}$, $\omega = 0$, and

$$h(0) = 01, \quad h(1) = 10.$$

For this D0L-system the tree of sequence generation is given below:



The sequence generated by this D0L-system are called Thue-Morse sequences. Alternatively, the Thue-Morse sequences can be defined recursively by starting with $\boldsymbol{t}_0 = 0$ and forming $\boldsymbol{t}_{i+1}$ by concatenating $\boldsymbol{t}_i$ and its complement $\overline{\boldsymbol{t}_i}$.

We show that binary D0L-systems, which have production rules of the form $h(0) = \boldsymbol{u}$ and $h(1) = \boldsymbol{v}$, with $\boldsymbol{u}, \boldsymbol{v} \in \{0, 1\}^*$ have a logarithmic distance to their roots.

Lemma 3.12. *For any binary D0L-system with initiator ω and production rule h, we have*

$$f(h^r(\omega)) = \Theta\big(\log_2|h^r(\omega)|\big), \qquad \text{as } r \to \infty.$$

Proof. For any sequence $\boldsymbol{t}$, since $f(\boldsymbol{t}) \geqslant \log_2 |\boldsymbol{t}|$, we have $f(h^r(\omega)) \geqslant \log_2|h^r(\omega)|$. It remains to show that $f(h^r(\omega)) = O\big(\log_2|h^r(\omega)|\big)$. We start by proving the following claim.

Claim. *For any binary D0L-system with initiator ω and production rule h, we have*

$$f(h^r(\omega)) \leqslant f\Big(h^{r-1}(\omega)\Big) + c \leqslant f(\omega) + rc, \tag{3.6}$$

where $c = \max_{z \in \{0,1,01,10,010,101\}} f(h(z))$.

To prove the claim, let $\boldsymbol{x} = h^{r-1}(\boldsymbol{\omega})$ and $\boldsymbol{y} = h^{r}(\boldsymbol{\omega})$ and consider the sequence of deduplications that turns x into its root $\boldsymbol{z} \in \{0, 1, 01, 10, 010, 101\}$. We can deduplicate $\boldsymbol{y}$ in a similar manner to $h(\boldsymbol{z})$: For each step in the deduplication process of $\boldsymbol{x}$ that deduplicates a substring $a_1 \cdots a_k a_1 \cdots a_k$ to $a_1 \cdots a_k$, we deduplicate $h(a_1) \cdots h(a_k) h(a_1) \cdots h(a_k)$ to $h(a_1) \cdots h(a_k)$ in the deduplication process of $\boldsymbol{y}$, resulting eventually in $h(\boldsymbol{z})$. This completes the proof of the claim.

We now turn to proving $f(h^r(\boldsymbol{\omega})) = O\big(\log_2 |h^r(\boldsymbol{\omega})|\big)$. If $|h^r(\boldsymbol{\omega})| = O(1)$, then $f(h^r(\boldsymbol{\omega})) = O(1)$ as well, and there is nothing to prove. If $|h^r(\boldsymbol{\omega})| = 2^{\Omega(r)}$, then $r = O\big(\log_2 |h^r(\boldsymbol{\omega})|\big)$ and the desired result follows from (3.6). The last case that we need to consider is when $|h^r(\boldsymbol{\omega})| \to \infty$ but $|h^r(\boldsymbol{\omega})| = 2^{o(r)}$. Without loss of generality, assume $|h(1)| \geqslant |h(0)|$. Then the condition $|h^r(\boldsymbol{\omega})| = 2^{o(r)}$ can be shown to occur only if the initiator $\boldsymbol{\omega}$ contains at least one occurrence of 1, $h(0) = 0$, and $h(1)$ has exactly one occurrence of 1 and one or more 0s. In this case, the number of 1s in $h^r(\boldsymbol{\omega})$ is constant and again $f(h^r(\boldsymbol{\omega})) = O\big(\log_2 |h^r(\boldsymbol{\omega})|\big)$. ∎

The previous lemma shows that the duplication distances to the root for both of Fibonacci words and Thue-Morse sequences are logarithmic in sequence length. This is particularly interesting in the case of the Thue-Morse sequence. Despite the fact that the Thue-Morse sequence grows by taking the complement, it contains enough repeats to allow a logarithmic distance. Note also that the Thue-Morse sequence is used to generate ternary square-free sequences.

In the next lemma, we give better bounds than those that can be obtained from Lemma 3.12 or (3.6) for Thue-Morse and Fibonacci sequences.

Lemma 3.13. *Let $\boldsymbol{t}_r$ and $\boldsymbol{u}_r$ denote the rth Thue-Morse and Fibonacci words, respectively. For $r \geqslant 2$, we have*

$$f(\boldsymbol{t}_r) \leqslant 2r,$$
$$f(\boldsymbol{u}_r) \leqslant r.$$

Proof. We first prove the upper bound for $\boldsymbol{t}_r$. For $r \geqslant 3$, we have

$$\begin{aligned}
f(\boldsymbol{t}_r) &= f\left(\boldsymbol{t}_{r-1}\overline{\boldsymbol{t}}_{r-1}\right)\\
&= f\left(\boldsymbol{t}_{r-2}\overline{\boldsymbol{t}}_{r-2}\overline{\boldsymbol{t}}_{r-2}\boldsymbol{t}_{r-2}\right)\\
&\leqslant 1 + f\left(\boldsymbol{t}_{r-2}\overline{\boldsymbol{t}}_{r-2}\boldsymbol{t}_{r-2}\right)\\
&= 1 + f\left(\boldsymbol{t}_{r-3}\overline{\boldsymbol{t}}_{r-3}\overline{\boldsymbol{t}}_{r-3}\boldsymbol{t}_{r-3}\boldsymbol{t}_{r-3}\overline{\boldsymbol{t}}_{r-3}\right)\\
&\leqslant 3 + f\left(\boldsymbol{t}_{r-3}\overline{\boldsymbol{t}}_{r-3}\boldsymbol{t}_{r-3}\overline{\boldsymbol{t}}_{r-3}\right)\\
&\leqslant 4 + f\left(\boldsymbol{t}_{r-3}\overline{\boldsymbol{t}}_{r-3}\right)\\
&= 4 + f(\boldsymbol{t}_{r-2}).
\end{aligned}$$

If $r \geqslant 3$ is even, then $f(\boldsymbol{t}_r) \leqslant 4\frac{r-2}{2} + f(\boldsymbol{t}_2) = 2(r-2)+1 = 2r-3$; and if $r \geqslant 3$ is odd, then $f(\boldsymbol{t}_r) \leqslant 4\frac{r-1}{2} + f(\boldsymbol{t}_1) = 2(r-1)$. This completes the proof of the first claim.

We now turn to $f(\boldsymbol{u}_r)$. The rth Fibonacci word can be obtained via the following recursion: $\boldsymbol{u}_r = \boldsymbol{u}_{r-1}\boldsymbol{u}_{r-2}$ for $r \geqslant 2$ and $\boldsymbol{u}_0 = 0$, $\boldsymbol{u}_1 = 01$. If $r \geqslant 5$, then

$$\begin{aligned}
\boldsymbol{u}_r &= \boldsymbol{u}_{r-1}\boldsymbol{u}_{r-2}\\
&= \boldsymbol{u}_{r-2}\boldsymbol{u}_{r-3}\boldsymbol{u}_{r-3}\boldsymbol{u}_{r-4}\\
&= \boldsymbol{u}_{r-2}\boldsymbol{u}_{r-3}\boldsymbol{u}_{r-4}\boldsymbol{u}_{r-5}\boldsymbol{u}_{r-4}\\
&= \boldsymbol{u}_{r-2}^2\boldsymbol{u}_{r-5}\boldsymbol{u}_{r-4}.
\end{aligned}$$

Hence, $f(\boldsymbol{u}_r) \leqslant 1+f(\boldsymbol{u}_{r-2}\boldsymbol{u}_{r-5}\boldsymbol{u}_{r-4})$. Noting that $\boldsymbol{u}_{r-2}\boldsymbol{u}_{r-5}\boldsymbol{u}_{r-4} = \boldsymbol{u}_{r-3}\boldsymbol{u}_{r-4}\boldsymbol{u}_{r-5}\boldsymbol{u}_{r-4} = \boldsymbol{u}_{r-3}^2\boldsymbol{u}_{r-4}$, we write

$$\begin{aligned}
f(\boldsymbol{u}_r) &\leqslant 1 + f(\boldsymbol{u}_{r-2}\boldsymbol{u}_{r-5}\boldsymbol{u}_{r-4})\\
&= 1 + f\left(\boldsymbol{u}_{r-3}^2\boldsymbol{u}_{r-4}\right)\\
&\leqslant 2 + f(\boldsymbol{u}_{r-3}\boldsymbol{u}_{r-4})\\
&= 2 + f(\boldsymbol{u}_{r-2}).
\end{aligned}$$

Now, if $r \geqslant 5$ is even, then $f(\boldsymbol{u}_r) \leqslant (r-4) + f(\boldsymbol{u}_4) \leqslant r-2$ since $f(\boldsymbol{u}_4) = f(01001010) \leqslant 2$; and if $r \geqslant 5$ is odd, then $f(\boldsymbol{u}_r) \leqslant (r-3) + f(\boldsymbol{u}_3) \leqslant r-1$ as $f(\boldsymbol{u}_3) = f(01001) \leqslant 2$. ∎

3.4 Approximate-duplication distance

Recall that $f_\beta(n)$ is the least k such that every sequence of length n can be converted to a square-free sequence in k approxmiate deduplication steps, with at most a β

fraction of mismatches in each step. In this section, we provide bounds on $f_\beta(n)$ for $\beta < 1/2$ and $\beta > 1/2$. We first however present some useful definitions.

For $0 \leqslant \beta < 1$, a *β-repeat of length h* in a binary sequence consists of two consecutive blocks in the sequence, each of length h, such that the Hamming distance between them is at most βh. If $\boldsymbol{u}\boldsymbol{v}\boldsymbol{v}'\boldsymbol{w}$ is a binary sequence, and $\boldsymbol{v}\boldsymbol{v}'$ is a β-repeat, then a *β-deduplication* produces $\boldsymbol{u}\boldsymbol{v}\boldsymbol{w}$ or $\boldsymbol{u}\boldsymbol{v}'\boldsymbol{w}$. Note that in this case the set of roots of $\boldsymbol{s}$ is not necessarily unique, but the length of any root is at most 3, even if $\beta = 0$.

The next theorem establishes a sharp phase transition in the behavior of $f_\beta(n)$ at $\beta = 1/2$. Its proof relies on Theorem 3.14, which guarantees the existence of β-repeats under certain conditions. In what follows, for an integer m, we use $[m]$ to denote $\{1, \ldots, m\}$.

Theorem 3.4. *If $\beta < 1/2$, then there exists a constant $c = c(\beta) > 0$ such that*

$$f_\beta(n) \geqslant cn.$$

Furthermore, if $\beta > 1/2$, for any constant $C > \left\lceil \frac{2\beta+1}{2\beta-1} \right\rceil^2$ and sufficiently large n,

$$f_\beta(n) \leqslant C \ln n.$$

Proof. The proof for $\beta < 1/2$ is similar to the proof of the lower bound in Theorem 3.1. In this case however, to make the deduplication process reversible, for every deduplication we need to record whether it is of the form $\boldsymbol{u}\boldsymbol{v}\boldsymbol{v}'\boldsymbol{w} \xrightarrow{dd} \boldsymbol{u}\boldsymbol{v}\boldsymbol{w}$ or of the form $\boldsymbol{u}\boldsymbol{v}'\boldsymbol{v}\boldsymbol{w} \xrightarrow{dd} \boldsymbol{u}\boldsymbol{v}\boldsymbol{w}$, and we must also encode the sequence $\boldsymbol{v}'$. In the tth deduplication step, we have $|\boldsymbol{v}| = |\boldsymbol{v}'| = h_t$. Since $\boldsymbol{v}'$ is in the Hamming sphere of radius βh_t around $\boldsymbol{v}$, there are at most $2^{h_t H(\beta)}$ options for $\boldsymbol{v}'$ [76, Lemma 4.7]. Thus

$$6n \sum_{f=1}^{F_\beta} \binom{n+f}{f} \binom{2n+f}{f} \binom{2n+f+2}{f} 2^{nH(\beta)} 2^{2f} \geqslant 2^n,$$

where $F_\beta = f_\beta(n)$ and we have used $\sum_t h_t \leqslant n$. The desired result then follows since $H(\beta) < 1$.

Suppose $\beta > 1/2$. Let $K = \left\lceil \frac{2\beta+1}{2\beta-1} \right\rceil^2$ and $\epsilon = C - K$. Note that $\epsilon > 0$. By appropriately choosing C_1, we can have $f_\beta(i) \leqslant \left(K + \frac{\epsilon}{2}\right) \ln i + C_1$ for all $i < M$, where M is sufficiently large and in particular $M > K$. Assuming that this holds also for all $i < n$, where $n \geqslant M$, we show that it holds for $i = n$. From Theorem 3.14, every binary sequence $\boldsymbol{s}$ of length n has a β-repeat of length $\ell\lfloor n/K \rfloor$ for some $\ell \in \left[\sqrt{K}\right]$, implying

$$\begin{aligned} f_\beta(\boldsymbol{s}) &\leqslant f_\beta\left(n - \ell\left\lfloor\frac{n}{K}\right\rfloor\right) + 1 \\ &\leqslant \left(K + \frac{\epsilon}{2}\right)\ln\left(n - \left\lfloor\frac{n}{K}\right\rfloor\right) + 1 + C_1 \\ &\leqslant \left(K + \frac{\epsilon}{2}\right)\ln n - \frac{(K + \frac{\epsilon}{2})(n-K)}{Kn} + 1 + C_1 \\ &\leqslant \left(K + \frac{\epsilon}{2}\right)\ln n + C_1 \\ &\leqslant C\ln n, \end{aligned}$$

where the last two steps hold for sufficiently large n. Hence, $f_\beta(n) \leqslant C\ln n$. ∎

Theorem 3.14. *If $\beta > \frac{1}{2}$, then for any integer $k \geqslant \frac{2\beta+1}{2\beta-1}$, any binary sequence of length n contains a β-repeat of length $\ell\lfloor n/k^2 \rfloor$ for some $\ell \in [k]$.*

Proof. Let k be a positive integer to be determined later and put $K = k^2$. Furthermore, let $\boldsymbol{s}' = \boldsymbol{s}_1 \cdots \boldsymbol{s}_K$ be a partition of the first KB symbols of $\boldsymbol{s}$ into blocks of length $B = \left\lfloor \frac{n}{K} \right\rfloor$. We now consider as a code [71] the $k+1$ binary vectors

$$\boldsymbol{t}_i = \boldsymbol{s}_i \cdots \boldsymbol{s}_{i+K-k-1}, \qquad (1 \leqslant i \leqslant k+1),$$

each of length $m = (K-k)B$. By Plotkin's bound [71, p. 41], the minimum Hamming distance of this code is at most $\left(\frac{1}{2} + \frac{1}{2k}\right)m$. Thus there exist $\boldsymbol{t}_i$ and $\boldsymbol{t}_j$ with $i < j$ with Hamming distance at most $\left(\frac{1}{2} + \frac{1}{2k}\right)m$.

Put $h = (j-i)B$ and let $m' = h\lfloor m/h \rfloor$ be the largest integer which is at most m and is divisible by h. Let $\boldsymbol{t}'_i$ and $\boldsymbol{t}'_j$ consist of the first m' bits of $\boldsymbol{t}_i$ and $\boldsymbol{t}_j$, respectively. The Hamming distance between $\boldsymbol{t}'_i$ and $\boldsymbol{t}'_j$ is clearly still at most $\left(\frac{1}{2} + \frac{1}{2k}\right)m$. But $\left(\frac{1}{2} + \frac{1}{2k}\right)m \leqslant \left(\frac{1}{2} + \frac{1}{k-1}\right)m'$ since

$$\left(\frac{1}{2} + \frac{1}{2k}\right)m = \left(\frac{1}{2} + \frac{1}{2k}\right)\frac{m}{m'}m' \overset{(*)}{\leqslant} \left(\frac{1}{2} + \frac{1}{2k}\right)\frac{k}{k-1}m' = \left(\frac{1}{2} + \frac{1}{k-1}\right)m',$$

where $(*)$ can be proved as follows. By the definition of m', we have $m - m' < h$. Additionally, $h \leqslant kB$ since $1 \leqslant i < j \leqslant k+1$. So,

$$\frac{m - m'}{B} < k,$$

which since B divides m, m', implies $\frac{m-m'}{B} \leqslant k-1$ and, in turn, $m' \geqslant m - (k-1)B = (k-1)^2 B$. Hence $\frac{m}{m'} \leqslant \frac{k(k-1)B}{(k-1)^2 B} = \frac{k}{k-1}$.

Split $\boldsymbol{t}'_i$ and $\boldsymbol{t}'_j$ into blocks of length h each: $\boldsymbol{t}'_i = z_1 z_2 \cdots z_p$, $\boldsymbol{t}'_j = z_2 z_3 \cdots z_p z_{p+1}$, where $p = m'/h$. The Hamming distance between $\boldsymbol{t}'_i$ and $\boldsymbol{t}'_j$ is the sum of the Hamming distances between z_q and z_{q+1} as q ranges from 1 to p. Thus, by averaging, there exists an index r so that the Hamming distance between z_r and z_{r+1} is at most $\left(\frac{1}{2} + \frac{1}{k-1}\right)h$. Putting $k \geqslant \frac{2\beta+1}{2\beta-1}$ so that $\frac{1}{2} + \frac{1}{k-1} \leqslant \beta$ ensures that $z_r z_{r+1}$ is β-repeat of length $h = (j-i)B = (j-i)\lfloor n/K \rfloor$. ∎

Let a β_h-repeat be a repeat of length h with at most $h\beta_h$ mismatches, i.e., the two blocks are at Hamming distance at most $h\beta_h$. In the preceding theorems and their proofs, in principal, we do not need the maximum number of permitted mismatches to be a linear function of the length of the repeat, so we can apply the same techniques to β_h-repeats with nonlinear relationships:

Theorem 3.15. *Let $\beta_h^a = \frac{1}{2} + \frac{1}{h^a}$, where $0 < a < 1$ is a constant, and let $f_a(n)$ be the smallest number f such that any binary sequence of length n can be deduplicated to a root in f steps by deduplicating β_h^a-repeats. There exist positive constants c_2, c_3 such that*

$$f_a(n) \leqslant c_2 n^{2a/(1+a)} + c_3. \tag{3.7}$$

Proof. By making appropriate changes to the proof of Theorem 3.14, one can show that for $k = \lceil 2n^{a/(1+a)} \rceil$, every binary sequence of sufficiently long length n contains a β_h^a-repeat of length $h = \ell \lfloor n/k^2 \rfloor$, for some $\ell \in [k]$. To do so, we need to prove $\left(\frac{1}{2} + \frac{1}{k-1}\right)h \leqslant \beta_h^a h$ for all h of the form $h = \ell\lfloor n/k^2 \rfloor$, $\ell \in [k]$. This holds since with the aforementioned value of k,

$$\beta^a_{\ell\lfloor n/k^2 \rfloor} = \frac{1}{2} + \frac{1}{\left(\ell\lfloor n/k^2 \rfloor\right)^a} \geqslant \frac{1}{2} + \frac{1}{\left(k\lfloor n/k^2 \rfloor\right)^a} \geqslant \frac{1}{2} + \frac{1}{k-1},$$

for all $\ell \in [k]$ and sufficiently large n.

We can now prove (3.7) by induction. Clearly, for any M, there exist constants c_2, c_3 such that $f_a(i) \leqslant c_2 i^{2a/(1+a)} + c_3$ for all $i \leqslant M$. Choose M to be sufficiently large as to satisfy the requirements of the rest of the proof. Fix $n > M$ and assume that $f_a(i) \leqslant c_2 i^{2a/(1+a)} + c_3$ for all $i < n$. Since in every sequence of length n, there exists

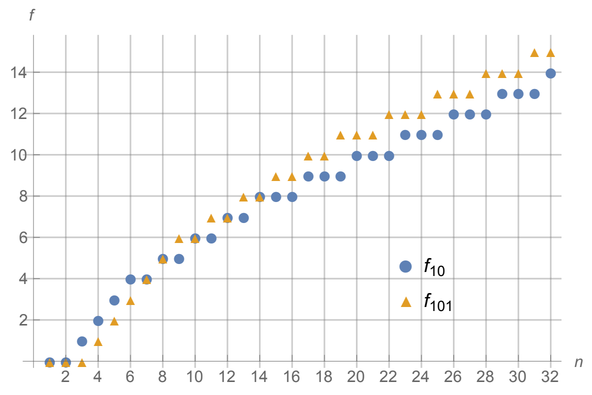


Figure 3.2: $f_{10}(n)$ and $f_{101}(n)$ for $1 \leqslant n \leqslant 32$.

a β_h^a-repeat with $h = \ell\lfloor n/k^2 \rfloor$, for some $\ell \in [k]$ and $k = \lceil 2n^{a/(1+a)} \rceil$, it holds that

$$\begin{aligned} f_a(n) &\leqslant 1 + c_2\Big(n - \ell\lfloor n/k^2 \rfloor\Big)^{2a/(1+a)} + c_3 \\ &\leqslant 1 + c_2\left(n - \frac{1}{5}n^{\frac{1-a}{1+a}}\right)^{2a/(1+a)} + c_3 \\ &= 1 + c_2 n^{2a/(1+a)}\left(1 - \frac{1}{5}n^{-\frac{2a}{1+a}}\right)^{2a/(1+a)} + c_3 \\ &\leqslant 1 + c_2 n^{2a/(1+a)}\left(1 - \frac{2a}{5(1+a)}n^{-\frac{2a}{1+a}}\right) + c_3 \\ &= c_2 n^{2a/(1+a)} + \left(1 - \frac{2ac_2}{5(1+a)}\right) + c_3 \\ &\leqslant c_2 n^{2a/(1+a)} + c_3, \end{aligned}$$

where the inequalities hold for sufficiently large n. The third inequality follows from Bernoulli's inequality and the the last one follows from the fact that we can choose c_2 to be arbitrarily large. ∎

3.5 Duplication distances for different roots

In this section, we study f_σ for $\sigma \in \{0, 1, 01, 10, 010, 101\}$. It is easy to see that $f_0(n) = f_1(n) = \lceil \log_2 n \rceil$. Clearly $f_{10} = f_{01}$ and $f_{101} = f_{010}$. So we limit our attention to roots $\sigma = 10$ and $\sigma = 101$. Plots for $f_{10}(n)$ and $f_{101}(n)$, obtained through computer search, are given in Figure 3.2.

Theorem 3.5. *The limits* $\lim_n \frac{f_{10}(n)}{n}$ *and* $\lim_n \frac{f_{101}(n)}{n}$ *exist and are equal to* $\lim_n \frac{f(n)}{n}$.

Proof. The general approach in this proof is similar to that of the proof of Fekete's lemma in [72]. We prove the theorem for $\lim_n \frac{f_{10}(n)}{n}$. The proof for $\frac{f_{101}(n)}{n}$ is similar.

Let $\gamma = \liminf_n \frac{f_{10}(n)}{n}$ and let $k \geqslant 3$ be such that $f_{10}(k) + 5 + 2\log_2 k \leqslant k(\gamma + \epsilon)$ for $\epsilon > 0$. Let $\boldsymbol{s}$ be a sequence of length n. Starting from the beginning of $\boldsymbol{s}$, partition it into substrings that are the shortest possible while having length at least k and different symbols at the beginning and the end (so that their root is either 10 or 01). Name these substrings $\boldsymbol{s}_1, \dots, \boldsymbol{s}_{m+1}$, where $|\boldsymbol{s}_i| \geqslant k$ for $i \leqslant m$ and $1 \leqslant |\boldsymbol{s}_{m+1}| \leqslant k$. Let $s_{i,j}$ denote the jth element of $\boldsymbol{s}_i$. We deduplicate $\boldsymbol{s}$ to its root by first deduplicating its substrings $\boldsymbol{s}_i$ to their roots.

For each substring $\boldsymbol{s}_i$ of the partition, except the last one, we consider the following cases and deduplicate $\boldsymbol{s}_i$ as indicated, where without loss of generality we assume $\boldsymbol{s}_i$ starts with 1 and ends with 0:

- $|\boldsymbol{s}_i| = k$: Deduplicate this substring to 10 in $f_{10}(k)$ steps.
- $|\boldsymbol{s}_i| > k$ and $s_{i,k-1} = 1$: In this case, $\boldsymbol{s}_i = 1\boldsymbol{x}11{,}1^*0$, where $\boldsymbol{x} \in \{0,1\}^{k-3}$, for clarity a comma is placed after the kth element of $\boldsymbol{s}_i$, and a^* denotes that the symbol a appears 0 or more times. We reduce the length of the last run of 1s in $\boldsymbol{s}_i$ by $|\boldsymbol{s}_i| - k$ in $\lceil \log_2(|\boldsymbol{s}_i| - k + 1) \rceil$ deduplication steps to obtain $1\boldsymbol{x}10$. Then deduplicate the result to 10 in $f_{10}(k)$ steps.
- $|\boldsymbol{s}_i| > k$ and $s_{i,k-1} = 0$: In this case, $\boldsymbol{s}_i = 1\boldsymbol{x}01{,}1^*0$, where $\boldsymbol{x} \in \{0,1\}^{k-3}$ and where a comma is placed after the kth element of $\boldsymbol{s}_i$. We reduce the length of the last run of 1s in $\boldsymbol{s}_i$ by $|\boldsymbol{s}_i| - k - 1$ in $\lceil \log_2(|\boldsymbol{s}_i| - k) \rceil$ deduplication steps to obtain $\hat{\boldsymbol{s}}_i = 1\boldsymbol{x}01, 0$ and note that $\hat{\boldsymbol{s}}_i$ has length $k+1$ and ends with 010. Now either $\hat{\boldsymbol{s}}_i$ has a run of length at least 2 or not. If it does, we reduce the length of this run by 1 to obtain a sequence of length k, which we then convert to 10 in $f_{10}(k)$ deduplication steps. If not, then $\hat{\boldsymbol{s}}_i$ is an alternating sequence of the form $101010 \cdots 10$ which can be deduplicated to 10 in no more than $\lceil \log_2 \frac{k+1}{2} \rceil$ steps.

The resulting sequences has length at most $2m + k$ and can be deduplicated to its root in at most as many steps. We thus have

$$\begin{aligned} f(n) &\leqslant m f_{10}(k) + \sum_{i=1}^{m} \lceil \log_2(|\boldsymbol{s}_i| - k + 1) \rceil + m \left\lceil \log_2 \frac{k+1}{2} \right\rceil + 3m + k \\ &\leqslant m f_{10}(k) + \sum_{i=1}^{m} \log_2 |\boldsymbol{s}_i| + m \log_2 k + 5m + k \\ &\leqslant \frac{n}{k} f_{10}(k) + \frac{2n}{k} \log_2 k + 5\frac{n}{k} + k, \end{aligned}$$

where for the last step we have used the fact that

$$\sum_{i=1}^{m} \log_2 |\boldsymbol{s}_i| \leqslant m \log_2(n/m) \leqslant \frac{n}{k} \log_2 k$$

which holds since $\sum_{i=1}^{m} |\boldsymbol{s}_i| \leqslant n$, $\frac{d}{dm} m \log_2 \frac{n}{m} > 0$ and $m \leqslant \frac{n}{k}$. It follows that

$$\frac{f(n)}{n} \leqslant \frac{f_{10}(k)}{k} + \frac{2\log_2 k}{k} + \frac{5}{k} + \frac{k}{n} \leqslant \gamma + \epsilon + \frac{k}{n} .$$

Taking lim of both sides and noting that $\epsilon > 0$ is arbitrary proves that $\lim_n \frac{f(n)}{n} \leqslant \liminf_n \frac{f_{10}(n)}{n}$. On the other hand, it is clear that $\limsup_n \frac{f_{10}(n)}{n} \leqslant \lim_n \frac{f(n)}{n}$. Hence, $\lim_n \frac{f(n)}{n} = \lim_n \frac{f_{10}(n)}{n}$. Similar arguments hold for $f_{101}(n)$. ∎

3.6 Open Problems

We now describe some of the open problems related to extremal values of duplication distance to the root. First, the binary duplication constant, $\lim_n \frac{f(n)}{n}$ is unknown. It is also interesting to find bounds tighter than the one given in Theorem 3.1, namely $0.045 \leqslant \lim \frac{f(n)}{n} \leqslant 0.4$. Furthermore, although the lower bound $f(\boldsymbol{s}) \geqslant 0.045n$ is valid for all but an exponentially small fraction of sequences of length n, we have not been able to find an explicit family of sequences whose distance is linear in n. A related problem to identifying sequences with large duplication distance is improving bounds on $f(\boldsymbol{s})$ that depend on the structure of $\boldsymbol{s}$, such as the bound given in Lemma 3.2, relating $f(\boldsymbol{s})$ to the number of distinct k-mers of $\boldsymbol{s}$.

While we showed in our study of approximate duplication that at $\beta = 1/2$, $f_\beta(n)$ transitions from a linear dependence on n to a logarithmic one, the behavior at $\beta = 1/2$ is not known. Furthermore, we can alter the setting by decoupling duplications and substitutions, i.e., we generate the sequence through exact duplications and substitutions, possibly with limitations on the number of substitutions. We can then study the same problems as the ones we have in this chapter as well as new problems, e.g., the minimum number substitutions required to generate the sequence via a logarithmic number of duplication steps.

Chapter 4

UNIQUENESS OF SEED AND ERROR CORRECTION

4.1 Introduction

Data storage in the DNA of living organisms (henceforth *live DNA*) has a multitude of applications. It can enable in-vivo synthetic-biology methods and algorithms that need "memory," e.g., to store information about their state or record changes in the environment. Embedding data in live DNA also allows watermarking genetically-modified organisms (GMOs) to verify authenticity and to track unauthorized use [77–79], as well as labeling organisms in biological studies [80]. DNA watermarking can also be used to tag infectious agents used in research laboratories to identify sources of potential malicious use or accidental release [81]. Furthermore, live DNA can serve as a protected medium for storing large amounts of data in a compact format for long periods of time [80, 82]. An additional advantage of using DNA as a medium is that data can be disguised as part of the organisms' original DNA, thus providing a layer of secrecy [83].

While the host organism provides a level of protection to the data-carrying DNA molecules as well as a method for replication, the integrity of the stored information suffers from mutations such as tandem duplications, point mutations, insertions, and deletions. Furthermore, since each DNA replication may introduce new mutations, the number of such deleterious events increases with the number of generations. As a result, to ensure decodability of the stored information, the coding/decoding scheme must be capable of a level of error correction. Motivated by this problem, we study designing codes that can correct errors arising from tandem duplications. In addition to improving the reliability of data storage in live DNA, studying such codes may help to acquire a better understanding of how DNA stores and protects biological information in nature.

Different approaches to the problem of error-control for data stored in live DNA have been proposed in the literature. In the work of Arita and Ohashi [77], each group of five bits of information is followed by one parity bit for error detection. Heider and Barnekow [78] use the extended $[8, 4, 4]$ binary Hamming code or repetition coding to protect the data. Yachie et al. [84] propose to enhance reliability by inserting multiple copies of the data into multiple regions of the genome of the host organism.

Finally, Haughton and Balado [85] present an encoding method satisfying certain biological constraints, which is studied in a substitution-mutation model. None of the aforementioned encodings, with the possible exception of repetition coding, are designed to combat tandem duplications, which is the focus of this chapter. While repetition coding can correct duplication errors, it is not an efficient method because of its high redundancy.

It should also be noted that error control for storage in live DNA is inherently different from that in DNA that is stored outside of a living organism (see [86] for an overview), since the latter is not concerned with errors arising during organic DNA replication.

We also note that tandem duplication, as well as other duplication mechanisms, were studied in the context of information theory [13, 16, 87]. However, these works used duplications as a generative process, and attempted to measure its capacity and diversity. In contrast, we consider duplications as a noise source, and design error-correcting codes to combat it.

We will first consider the tandem-duplication channel with duplications of a fixed length k. For example with $k = 3$, after a tandem duplication, the sequence $ACAGT$ may become $ACAG\underline{CAG}T$, which may then become $ACA\underline{ACA}GCAGT$ where the copy is underlined. In our analysis, we provide a mapping in which tandem duplications of length k are equivalent to insertion of k zeros. Using this mapping, we demonstrate the strong connection between codes that correct duplications of a fixed length and Run-Length Limited (RLL) systems. We present constructions for codes that can correct an unbounded number of tandem duplications of a fixed length and show that our construction is optimal, i.e., of the largest size. A similar idea was used in [88], where codes were constructed for duplication-error correction with the number of tandem duplications restricted to a given size r and a duplication length of 1 only. In this chapter, we generalize their result by constructing optimal (i.e., maximum size) error-correcting codes for arbitrary duplication length k and with no restriction on the number of tandem duplications.

We then turn our attention to codes that correct t tandem duplications (as opposed to an unbounded number of duplications), and show that these codes are closely related to constant-weight codes in the ℓ_1 metric.

We also consider codes for correcting duplications of bounded length. Here, our focus will be on duplication errors of length at most 2 or 3, for which we will present a construction that corrects any number of such errors. In the case of duplication length at most 2 the codes we present are optimal. Another problem we also study here is the mismatch between code design and channel characteristics for tandem-duplication channels with respect to the maximum length of the duplication errors. In this case, we quantify the uncertainty resulting from this mismatch, i.e., the maximum number of possible inputs for one output, for channels in which the maximum duplication length is bounded by 3.

Finally, when a sequence has been corrupted by a tandem-duplication channel, the challenge arises in finding the *root* sequences from which the corrupted sequence could be generated. A root sequence does not contain any tandem-duplicated subsequences. For example, for the sequence $ACGT\underline{GT}$, with $GTGT$ as a tandem-duplication error, a root sequence would be $ACGT$ since $ACGTGT$ can be generated from $ACGT$ via a tandem duplication of length 2 on GT. But there can be sequences that have more than one root. For example, the sequence $ACGCACGCG$ can be generated from ACG through a tandem duplication of CG first, followed by a tandem duplication of $ACGC$. Alternatively, it can also be generated from $ACGCACG$ by doing a tandem duplication of the suffix CG. Hence, $ACGCACGCG$ has two roots. However, if we restrict the length of duplication to 2 in the previous example, then $ACGCACGCG$ has only one root i.e., $ACGCACG$. This means that the number of roots that a sequence can have depends on the set of duplication lengths that are allowed, and the size of the alphabet. We provide in Section 4.6 a complete classification of the parameters required for the unique-root property. This unique-root property of the fixed length, 2-bounded and 3-bounded tandem-duplication channels allows us to construct error-correcting codes for them.

The chapter is organized as follows. The preliminaries and notation are described in Section 4.2. In Sections 4.3 and 4.4 we present the results concerning duplications of a fixed length k and duplications of length at most k, respectively. Our results concerning the uncertainty resulting from the mismatch between the code and the channel are given in Section 4.5. In Section 4.6, we fully characterize tandem-duplication channels which have a unique root. We conclude with some open questions in Section 4.7.

4.2 Preliminaries

We let Σ denote some finite alphabet, and Σ^* denote the set of all finite strings (words) over Σ. The unique empty word is denoted by ϵ. The set of finite non-empty words is denoted by $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$. Given two words $x, y \in \Sigma^*$, their concatenation is denoted by xy, and x^t denotes the concatenation of t copies of x, where t is some positive integer. By convention, $x^0 = \epsilon$. The length of a string $x \in \Sigma^*$ is denoted by $|x|$. We normally index the letters of a word starting with 1, i.e., $x = x_1 x_2 \dots x_n$, with $x_i \in \Sigma$. With this notation, the t-prefix and t-suffix of x are defined by

$$\begin{aligned} \mathrm{Pref}_t(x) &= x_1 x_2 \dots x_t, \\ \mathrm{Suff}_t(x) &= x_{n-t+1} x_{n-t+2} \dots x_n. \end{aligned}$$

Given a string $x \in \Sigma^*$, a *tandem duplication of length k* is a process by which a contiguous substring of x of length k is copied next to itself. More precisely, we define the tandem-duplication rules, $T_{i,k} : \Sigma^* \to \Sigma^*$, as

$$T_{i,k}(x) = \begin{cases} uvvw & \text{if } x = uvw,\ |u| = i,\ |v| = k \\ x & \text{otherwise.} \end{cases}$$

We note that the "otherwise" case describes a degenerate case when $|x| < k + i$, and therefore x cannot be decomposed into a prefix u of length i, an inner part v of length k, and some suffix w. Two specific sets of duplication rules are of interest to us throughout the chapter.

$$\begin{aligned} \mathcal{T}_k &= \left\{ T_{i,k} \,\middle|\, i \geqslant 0 \right\}, \\ \mathcal{T}_{\leqslant k} &= \left\{ T_{i,k'} \,\middle|\, i \geqslant 0, 1 \leqslant k' \leqslant k \right\}. \end{aligned}$$

Given $x, y \in \Sigma^*$, if there exist i and k such that

$$y = T_{i,k}(x),$$

non-degenerately[1],then we say y is a direct descendant of x, and denote it by

$$x \underset{k}{\Longrightarrow} y.$$

If a sequence of t non-degenerate tandem duplications of length k is employed to reach y from x we say y is a t-descendant of x and denote it by

$$x \overset{t}{\underset{k}{\Longrightarrow}} y.$$

[1]Here, and throughout the chapter, non-degenerately refers to tandem duplications that avoid the "otherwise" case in the definition of $T_{i,k}$.

More precisely, we require the existence of t non-negative integers $i_1, i_2, \dots, i_t$, with $0 \leqslant i_j \leqslant |x| + k(j-2)$, such that

$$y = T_{i_t,k}(T_{i_{t-1},k}(\dots T_{i_1,k}(x)\dots)).$$

Finally, if there exists a finite sequence of tandem duplications of length k transforming x into y, we say y is a descendant of x and denote it by

$$x \overset{*}{\underset{k}{\Longrightarrow}} y.$$

We note that x is its own descendant via an empty sequence of tandem duplications.

Example 4.1. Let $\Sigma = \{0, 1, 2, 3\}$ and $x = 02123$. Since, $T_{1,2}(x) = 0212123$ and $T_{0,2}(0212123) = 020212123$, the following hold

$$02123 \underset{2}{\Longrightarrow} 0212123, \qquad\qquad 02123 \overset{2}{\underset{2}{\Longrightarrow}} 020212123,$$

where in both expressions, the relation could be replaced with $\overset{*}{\underset{2}{\Longrightarrow}}$.

We define the *descendant cone* of x as

$$D_k^*(x) = \left\{ y \in \Sigma^* \;\middle|\; x \overset{*}{\underset{k}{\Longrightarrow}} y \right\}.$$

In a similar fashion we define the *t-descendant cone* $D_k^t(x)$ by replacing $\overset{*}{\underset{k}{\Longrightarrow}}$ with $\overset{t}{\underset{k}{\Longrightarrow}}$ in the definition of $D_k^*(x)$.

The set of definitions given thus far was focused on tandem-duplication rules of substrings of length exactly k, i.e., for rules from $\mathcal{T}_k$. These definitions as well as others in this section are extended in the natural way for tandem-duplication rules of length up to k, i.e., $\mathcal{T}_{\leqslant k}$. We denote these extensions by replacing the k subscript with the $\leqslant k$ subscript. Thus, we also have $D_{\leqslant k}^*(x)$ and $D_{\leqslant k}^t(x)$.

Example 4.2. Consider $\Sigma = \{0, 1\}$ and $x = 01$. It is not difficult to see that

$$\begin{aligned} D_1^2(x) &= \{0001, 0011, 0111\}, \\ D_1^*(x) &= \left\{0^i 1^j \;\middle|\; i, j \in \mathbb{N}\right\}, \\ D_2^*(x) &= \left\{(01)^i \;\middle|\; i \in \mathbb{N}\right\}, \\ D_{\leqslant 2}^*(x) &= \{0s1 \mid s \in \Sigma^*\}. \end{aligned}$$

Using the notation D_k^*, we restate the definition of the *tandem string-duplication system* given in [13]. Given a finite alphabet Σ, a seed string $s \in \Sigma^*$, the tandem string-duplication system is given by

$$S_k = S(\Sigma, s, \mathcal{T}_k) = D_k^*(s),$$

i.e., it is the set of all the descendants of s under tandem duplication of length k.

The process of tandem duplication can be naturally reversed. Given a string $y \in \Sigma^*$, for any positive integer, $t > 0$, we define the *t-ancestor cone* as

$$D_k^{-t}(y) = \left\{ x \in \Sigma^* \;\middle|\; x \overset{t}{\underset{k}{\Longrightarrow}} y \right\},$$

or in other words, the set of all words for which y is a t-descendant.

Yet another way of viewing the t-ancestor cone is by defining the *tandem-deduplication rules*, $T_{i,k}^{-1} : \Sigma^* \to \Sigma^*$, as

$$T_{i,k}^{-1}(y) = \begin{cases} uvw & \text{if } y = uvvw,\ |u| = i,\ |v| = k \\ \epsilon & \text{otherwise,} \end{cases}$$

where we recall ϵ denotes the empty word. This operation takes an adjacently-repeated substring of length k, and removes one of its copies. Thus, a string x is in the t-ancestor cone of y (where we assume $x, y \neq \epsilon$ to avoid trivialities) iff there is a sequence of t non-degenerate deduplication operations transforming y into x, i.e., there exist t non-negative integers $i_1, i_2, \ldots, i_t$, such that, non-degenerately,

$$x = T_{i_t,k}^{-1}(T_{i_{t-1},k}^{-1}(\ldots T_{i_1,k}^{-1}(y)\ldots)).$$

In a similar fashion we define the *ancestor cone* of y as

$$D_k^{-*}(y) = \left\{ x \in \Sigma^* \;\middle|\; x \overset{*}{\underset{k}{\Longrightarrow}} y \right\}.$$

By flipping the direction of the derivation arrow, we let $\Longleftarrow$ denote deduplication. Thus, if y may be deduplicated to obtain x in a single step, we write

$$y \underset{k}{\Longleftarrow} x.$$

For multiple steps we add $*$ in superscript.

Example 4.3. We have

$$0212123 \underset{2}{\Longleftarrow} 02123, \qquad 020212123 \overset{2}{\underset{2}{\Longleftarrow}} 02123,$$

and

$$D_2^{-*}(020212123) = \{020212123, 0212123, 0202123, 02123\}.$$

A word $y \in \Sigma^*$ is said to be *irreducible* (with respect to duplications of length k) if there is nothing to deduplicate in it, i.e., y is its only ancestor, meaning

$$D_k^{-*}(y) = \{y\}.$$

The set of irreducible words is denoted by Irr_k. We will find it useful to denote the set of irreducible words of length n by

$$\mathrm{Irr}_k(n) = \mathrm{Irr}_k \cap \Sigma^n.$$

The ancestors of $y \in \Sigma^*$ that cannot be further deduplicated, are called the *roots* of y, and are denoted by

$$R_k(y) = D_k^{-*}(y) \cap \mathrm{Irr}_k .$$

Note that since the aforementioned definitions extend to tandem-duplication rules of length up to k, we also have $S_{\leqslant k}$, $D_{\leqslant k}^{-t}(y)$, $D_{\leqslant k}^{-*}(y)$, $\mathrm{Irr}_{\leqslant k}$, $\mathrm{Irr}_{\leqslant k}(n)$, and $R_{\leqslant k}(y)$. In some previous works (e.g., [89]), S_k is called the *uniform-bounded-duplication system*, whereas $S_{\leqslant k}$ is called the *bounded-duplication system*.

Example 4.4. For the binary alphabet $\Sigma = \{0, 1\}$,

$$\mathrm{Irr}_{\leqslant 2} = \{0, 1, 01, 10, 010, 101\},$$

and for any alphabet that contains $\{0, 1, 2, 3\}$,

$$\begin{aligned} R_2(020212123) &= \{02123\}, \\ R_{\leqslant 4}(012101212) &= \{012, 0121012\}. \end{aligned}$$

Inspired by the DNA-storage scenario, we now define error-correcting codes for tandem string-duplication systems.

Definition 4.5. An $(n, M; t)_k$ code C for the k-tandem-duplication channel is a subset $C \subseteq \Sigma^n$ of size $|C| = M$, such that for each $x, y \in C$, $x \neq y$,

$$D_k^t(x) \cap D_k^t(y) = \emptyset.$$

Here t stands for either a non-negative integer, or $*$. In the former case we say the code can correct t errors, whereas in the latter case we say the code can correct all errors. In a similar fashion, we define an $(n, M; t)_{\leqslant k}$ by replacing all "k" subscripts by "$\leqslant k$".

Assume the size of the finite alphabet is $|\Sigma| = q$. We then denote the size of the largest $(n, M; t)_k$ code over Σ by $A_q(n; t)_k$. The capacity of the channel is then defined as

$$\mathsf{cap}_q(t)_k = \limsup_{n\to\infty} \frac{1}{n} \log_q A_q(n; t)_k.$$

Analogous definitions are obtained by replacing k with $\leqslant k$ or by replacing t with $*$.

In certain places we shall point out connections between string-duplication systems and formal languages. These connections have not gone unnoticed, and appear in chapters such as [89], which we describe in the appropriate context. A *language*, L, is nothing but a set of words, $L \subseteq \Sigma^*$, where Σ is some finite alphabet. A type of language we shall encounter frequently is a *regular language*, which is exactly a set of words that may be recognized by a finite automaton. Intuitively, such an automaton is defined by a finite set of states, and a finite set of transitions between states, each transition labeled by a symbol from Σ. Additionally, a single state is assigned the role of a starting state, and a subset of the states is assigned the role of accepting states. A word is recognized, if it is the result of concatenating the symbols of transitions describing a path from the starting state to some accepting state. Regular languages may also be described by regular expressions. The interested reader is referred to [90].

4.3 k-Tandem-Duplication Codes

In this section we consider tandem string-duplication systems where the substring being duplicated is of a constant length k. Such systems were studied in the context of formal languages [89] (also called *uniform-bounded-duplication systems*), and also in the context of coding and information theory [13].

In [89] it was shown that for any finite alphabet Σ and any word $x \in \Sigma^*$, under k-tandem duplication, x has a unique root, i.e.,

$$|R_k(x)| = 1.$$

Additionally, finding the unique root may be done efficiently, even by a greedy algorithm which searches for occurrences of ww as substrings of x, with $|w| = k$, removing one copy of w, and repeating the process. This was later extended in [91], where it was shown that the roots of a regular language also form a regular language. In what follows we give an alternative elementary proof to the uniqueness of the root. This proof will enable us to easily construct codes for k-tandem-duplication systems, as well as to state bounds on their parameters. The proof technique may be seen as an extension of the string-derivative technique used in [88], which was applied only for $k = 1$ over a binary alphabet.

The system S_k was also studied in [13] from a coding and information-theoretic perspective. In particular, it was proved in [13] that the capacity of S_k is 0. This fact will turn out to be extremely beneficial when devising error-correcting codes for k-tandem-duplication systems.

Throughout this section, without loss of generality, we assume $\Sigma = \mathbb{Z}_q$. We also use $\mathbb{Z}_q^*$ to denote the set of all finite strings of $\mathbb{Z}_q$ (not to be confused with the non-zero elements of $\mathbb{Z}_q$), and $\mathbb{Z}_q^{\geqslant k}$ to denote the set of all finite strings over $\mathbb{Z}_q$ of length k or more.

We shall require the following mapping, $\phi_k : \mathbb{Z}_q^{\geqslant k} \to \mathbb{Z}_q^k \times \mathbb{Z}_q^*$. The mapping is defined by,

$$\phi_k(x) = (\mathrm{Pref}_k(x), \mathrm{Suff}_{|x|-k}(x) - \mathrm{Pref}_{|x|-k}(x)),$$

where subtraction is performed entry-wise over $\mathbb{Z}_q$. We easily observe that ϕ_k is a bijection between $\mathbb{Z}_q^n$ and $\mathbb{Z}_q^k \times \mathbb{Z}_q^{n-k}$ by noting that we can recover x from $\phi_k(x)$ in the following manner: first set $x_i = \phi_k(x)_i$, for all $1 \leqslant i \leqslant k$, and for $i = k+1, k+2, \dots$, set $x_i = x_{i-k} + \phi_k(x)_i$, where $\phi_k(x)_i$ denotes the ith symbol of $\phi_k(x)$. Thus, ϕ_k^{-1} is well defined.

Another mapping we define is one that injects k consecutive zeros into a string. More precisely, we define $\zeta_{i,k} : \mathbb{Z}_q^k \times \mathbb{Z}_q^* \to \mathbb{Z}_q^k \times \mathbb{Z}_q^*$, where

$$\zeta_{i,k}(x, y) = \begin{cases} (x, u0^k w) & \text{if } y = uw,\ |u| = i \\ (x, y) & \text{otherwise.} \end{cases}$$

The following lemma will form the basis for the proofs to follow.

Lemma 4.6. *The following diagram commutes:*

$$\begin{tikzcd} \mathbb{Z}_q^{\geqslant k} \arrow[r, "T_{i,k}"] \arrow[d, "\phi_k"] & \mathbb{Z}_q^{\geqslant k} \arrow[d, "\phi_k"] \\ \mathbb{Z}_q^k \times \mathbb{Z}_q^* \arrow[r, "\zeta_{i,k}"] & \mathbb{Z}_q^k \times \mathbb{Z}_q^* \end{tikzcd}$$

i.e., for every string $x \in \mathbb{Z}_q^{\geqslant k}$,

$$\phi_k(T_{i,k}(x)) = \zeta_{i,k}(\phi_k(x)).$$

Before presenting the proof, we provide an example for the diagram of the lemma.

Example 4.7. Assume $\Sigma = \mathbb{Z}_4$. Starting with 02123 and letting $i = 1$ and $k = 2$ leads to

$$\begin{tikzcd} 02123 \arrow[r, "T_{1,2}"] \arrow[d, "\phi_2"] & 02\underline{12}123 \arrow[d, "\phi_2"] \\ (02, 102) \arrow[r, "\zeta_{1,2}"] & (02, 1\underline{00}02) \end{tikzcd}$$

where the inserted elements are underlined.

Proof. Let $x \in \mathbb{Z}_q^{\geqslant k}$ be some string, $x = x_1x_2 \dots x_n$. Additionally, let $\phi_k(x) = (y, z)$ with $y = y_1 \dots y_k$, and $z = z_1 \dots z_{n-k}$. We first consider the degenerate case, where $i \geqslant n - k + 1$. In that case, $T_{i,k}(x) = x$, and then by definition $\zeta_{i,k}(y, z) = (y, z)$ since z does not have a prefix of length at least $n - k + 1$. Thus, for $i \geqslant n - k + 1$ we indeed have

$$\phi_k(T_{i,k}(x)) = \phi_k(x) = (y, z) = \zeta_{i,k}(y, z) = \zeta_{i,k}(\phi_k(x)).$$

We are left with the case of $0 \leqslant i \leqslant n - k$. We now write

$$T_{i,k}(x) = x_1x_2 \dots x_{i+k}x_{i+1}x_{i+2} \dots x_n.$$

Thus, if we denote $\phi_k(T_{i,k}(x)) = (y, z)$, then

$$\begin{aligned} y &= x_1 \dots x_k = \mathrm{Pref}_k(x), \\ z &= x_{k+1} - x_1, \dots, x_{k+i} - x_i, 0^k, \\ &\quad x_{k+i+1} - x_{i+1}, \dots, x_n - x_{n-k}. \end{aligned}$$

This is exactly an insertion of 0^k after i symbols in the second part of $\phi_k(x)$. It therefore follows that

$$\phi_k(T_{i,k}(x)) = (y, z) = \zeta_{i,k}(\phi_k(x)),$$

as claimed. ∎

Recalling that ϕ_k is a bijection between $\mathbb{Z}_q^n$ and $\mathbb{Z}_q^k \times \mathbb{Z}_q^{n-k}$, together with Lemma 5.1 gives us the following corollary.

Corollary 4.8. *For any $x \in \mathbb{Z}_q^{\geqslant k}$, and for any sequence of non-negative integers $i_1, \dots, i_t$,*

$$T_{i_t,k}(\dots T_{i_1,k}(x) \dots) = \phi_k^{-1}(\zeta_{i_t,k}(\dots \zeta_{i_1,k}(\phi_k(x)) \dots)).$$

Example 4.9. Continuing Example 4.7 and using the notation of Corollary 4.8, let $x = 02123$, $k = t = 2$, $i_1 = 1$, and $i_2 = 0$. Then

$$\begin{aligned}
&T_{0,2}(T_{1,2}(02123)) \\
&= T_{0,2}(0212123) \\
&= 020212123 \\
&= \phi_k^{-1}((02, 0010002)) \\
&= \phi_k^{-1}(\zeta_{0,2}((02, 10002))) \\
&= \phi_k^{-1}(\zeta_{0,2}(\zeta_{1,2}((02, 102)))) \\
&= \phi_k^{-1}(\zeta_{0,2}(\zeta_{1,2}(\phi_k(02123)))).
\end{aligned}$$

Corollary 4.8 paves the way to working in the ϕ_k-transform domain. In this domain, a tandem-duplication operation of length k translates into an insertion of a block of k consecutive zeros. Conversely, a tandem-deduplication operation of length k becomes a removal of a block of k consecutive zeros.

The uniqueness of the root, proved in [89], now comes for free. In the ϕ_k-transform domain, given $(x, y) \in \mathbb{Z}_q^k \times \mathbb{Z}_q^*$, as long as y contains a substring of k consecutive zeros, we may perform another deduplication. The process stops at the unique outcome in which the length of every run of zeros in y is reduced modulo k.

This last observation motivates us to define the following operation on a string in $\mathbb{Z}_q^*$. We define $\mu_k : \mathbb{Z}_q^* \to \mathbb{Z}_q^*$ which reduces the lengths of runs of zeros modulo k in the following way. Consider a string $x \in \mathbb{Z}_q^*$, where

$$x = 0^{m_0} w_1 0^{m_1} w_2 \dots w_t 0^{m_t},$$

where m_i are non-negative integers, and $w_1, \dots, w_t \in \mathbb{Z}_q \setminus \{0\}$, i.e., $w_1, \dots, w_t$ are single non-zero symbols. We then define

$$\mu_k(x) = 0^{m_0 \bmod k} w_1 0^{m_1 \bmod k} w_2 \dots w_t 0^{m_t \bmod k}.$$

For example, for $z = 0010002$,

$$\mu_2(z) = 102.$$

Additionally, we define

$$\sigma_k(x) = \left(\left\lfloor \frac{m_0}{k} \right\rfloor, \left\lfloor \frac{m_1}{k} \right\rfloor, \dots, \left\lfloor \frac{m_t}{k} \right\rfloor \right) \in (\mathbb{N} \cup \{0\})^*$$

and call $\sigma(x)$ the *zero signature* of x. For z given above,

$$\sigma_2(z) = (1, 1, 0).$$

We note that $\mu_k(x)$ and $\sigma(x)$ together uniquely determine x.

We also observe some simple properties. First, the Hamming weight of a vector, denoted wt_H, counts the number of non-zero elements in a vector. By definition we have for every $x \in \mathbb{Z}_q^n$,

$$\mathrm{wt}_H(x) = \mathrm{wt}_H(\mu_k(x)).$$

Additionally, the length of the vector $\sigma_k(x)$, denoted $|\sigma_k(x)|$, is given by

$$|\sigma_k(x)| = \mathrm{wt}_H(x) + 1 = \mathrm{wt}_H(\mu_k(x)) + 1. \tag{4.1}$$

Note that for $z = 0010002$ as above, we have

$$|\sigma_2(z)| = 3 = \mathrm{wt}_H(z) + 1 = \mathrm{wt}_H(102) + 1.$$

Thus, our previous discussion implies the following corollary.

Corollary 4.10. *For any string $x \in \mathbb{Z}_q^{\geqslant k}$,*

$$R_k(x) = \left\{ \phi_k^{-1}(y, \mu_k(z)) \,\middle|\, \phi_k(x) = (y, z) \right\}.$$

We recall the definition of the $(0, k-1)$-RLL system over $\mathbb{Z}_q$ (for example, see [92, 93]). It is defined as the set of all finite strings over $\mathbb{Z}_q$ that do not contain k consecutive zeros. We denote this set as $C_{\mathrm{RLL}_q(0,k-1)}$. In our notation,

$$C_{\mathrm{RLL}_q(0,k-1)} = \left\{ x \in \mathbb{Z}_q^* \,\middle|\, \sigma_k(x) \in 0^* \right\}.$$

By convention, $C_{\mathrm{RLL}_q(0,k-1)} \cap \mathbb{Z}_q^0 = \{\epsilon\}$. The following is another immediate corollary.

Corollary 4.11. *For all $n \geqslant k$,*

$$\mathrm{Irr}_k(n) = \left\{ \phi_k^{-1}(y, z) \,\middle|\, y \in \mathbb{Z}_q^k, z \in C_{\mathrm{RLL}_q(0,k-1)} \cap \mathbb{Z}_q^{n-k} \right\}.$$

Proof. The proof is immediate since x is irreducible iff no deduplication action may be applied to it. This happens iff for $\phi_k(x) = (y, z)$, z does not contain k consecutive zeros, i.e., $z \in C_{\mathrm{RLL}_q(0,k-1)} \cap \mathbb{Z}_q^{n-k}$. ∎

Given two strings, $x, x' \in \mathbb{Z}_q^{\geqslant k}$, we say x and x' are k-congruent, denoted $x \sim_k x'$, if $R_k(x) = R_k(x')$. It is easily seen that $\sim_k$ is an equivalence relation.

Corollary 4.12. *Let $x, x' \in \mathbb{Z}_q^*$ be two strings, and denote $\phi_k(x) = (y, z)$ and $\phi_k(x') = (y', z')$. Then $x \sim_k x'$ iff $y = y'$ and $\mu_k(z) = \mu_k(z')$.*

Proof. This is immediate when using Corollary 4.10 to express the roots of x and x'. ▮

Example 4.13. For instance, 02123, 0212323, 0212123, and 020212123 are all 2-congruent, since they have the unique root 02123. In the ϕ_2-transform domain, for each sequence x in the preceding list, if we let $\phi_2(x) = (y, z)$, then $y = 02$ and $\mu_2(z) = 102$.

The following lemma appeared in [89, Proposition 2]. We restate it and give an alternative proof.

Lemma 4.14. *For all $x, x' \in \mathbb{Z}_q^{\geqslant k}$, we have*

$$D_k^*(x) \cap D_k^*(x') \neq \emptyset$$

if and only if $x \sim_k x'$.

Proof. In the first direction, assume $x \nsim_k x'$. By the uniqueness of the root, let us denote $R_k(x) = \{u\}$ and $R_k(x') = \{u'\}$, with $u \neq u'$. If there exists $w \in D_k^*(x) \cap D_k^*(x')$, then w is a descendant of both u and u', therefore $u, u' \in R_k(w)$, which is a contradiction. Hence, no such w exists, i.e., $D_k^*(x) \cap D_k^*(x') = \emptyset$.

In the other direction, assume $x \sim_k x'$. We construct a word $w \in D_k^*(x) \cap D_k^*(x')$. Denote $\phi_k(x) = (y, z)$ and $\phi_k(x') = (y', z')$. By Corollary 4.12 we have

$$y = y',$$
$$\mu_k(z) = \mu_k(z').$$

Let us then denote

$$z = 0^{m_0} v_1 0^{m_1} v_2 \dots v_t 0^{m_t},$$
$$z' = 0^{m'_0} v_1 0^{m'_1} v_2 \dots v_t 0^{m'_t},$$

with v_i a non-zero symbol, and

$$m_i \equiv m'_i \pmod{k},$$

for all i. We now define

$$z'' = 0^{\max(m_0,m_0')} v_1 0^{\max(m_1,m_1')} v_2 \dots v_t 0^{\max(m_t,m_t')}.$$

Since z'' differs from z and z' by insertion of blocks of k consecutive zeros, it follows that

$$w = \phi_k^{-1}(y, z'') \in D_k^*(x) \cap D_k^*(x'),$$

which completes the proof. ▮

We now turn to constructing error-correcting codes. The first construction is for a code capable of correcting any number of tandem duplications of length k.

Construction 4.15. *Fix $\Sigma = \mathbb{Z}_q$ and $k \geqslant 1$. For any $n \geqslant k$ we construct*

$$C = \bigcup_{i=0}^{\lfloor n/k \rfloor - 1} \left\{ \phi_k^{-1}(y, z0^{ki}) \;\middle|\; \phi_k^{-1}(y, z) \in \mathrm{Irr}_k(n - ik) \right\}.$$

Theorem 4.16. *The code C from Construction 4.15 is an $(n, M; *)_k$ code, with*

$$M = \sum_{i=0}^{\lfloor n/k \rfloor - 1} q^k M_{\mathrm{RLL}_q(0,k-1)}(n - (i+1)k).$$

Here $M_{\mathrm{RLL}_q(0,k-1)}(m)$ denotes the number of strings of length m which are $(0, k-1)$-RLL over $\mathbb{Z}_q$, i.e.,

$$M_{\mathrm{RLL}_q(0,k-1)}(m) = \left| C_{\mathrm{RLL}_q(0,k-1)} \cap \mathbb{Z}_q^m \right|.$$

Proof. The size of the code is immediate, by Corollary 4.11. Additionally, the roots of distinct codewords are distinct as well, since we constructed the code from irreducible words with blocks of k consecutive zeros appended to their end. Thus, by Lemma 4.14, the descendant cones of distinct codewords are disjoint. ▮

We can say more about the size of the code we constructed.

Theorem 4.17. *The code C from Construction 4.15 is optimal, i.e., it has the largest cardinality of any $(n; *)_k$ code.*

Proof. By Lemma 4.14, any two distinct codewords of an $(n; *)_k$ code must belong to different equivalence classes of $\sim_k$. The code C of Construction 4.15 contains exactly one codeword from each equivalence class of $\sim_k$, and thus, it is optimal. ▮

The code C from Construction 4.15 also allows a simple decoding procedure, whose correctness follows from Corollary 4.10. Assume a word $x' \in \mathbb{Z}_q^{\geqslant k}$ is received, and let $\phi_k(x') = (y', z')$. The decoded word is simply

$$\tilde{x} = \phi_k^{-1}(y', \mu_k(z')0^{n-k-|\mu_k(z')|}), \tag{4.2}$$

where n is the length of the code C. In other words, the decoding procedure recovers the unique root of the received x', and in the ϕ_k-transform domain, pads it with enough zeros.

Example 4.18. Let $n = 4$, $q = 2$, and $k = 1$. By inspection, the code C of Construction 4.15 can be shown to equal

$$C = \left\{\underline{0}000, \underline{01}11, \underline{010}0, \underline{0101}, \underline{1}111, \underline{10}00, \underline{101}1, \underline{1010}\right\},$$

where in each codeword the k-irreducible part is underlined. As an example of decoding, both 01100 and 01000 decode to 0100. Specifically for the former case, $x' = 01100$, we have $\phi_k(x') = (y', z') = (0, 1010)$. So $\mu_k(z') = 11$ and

$$\tilde{x} = \phi_k^{-1}(0, 110) = 0100.$$

Encoding may be done using any of the many various ways for encoding RLL-constrained systems. The reader is referred to [92, 93] for further reading. After encoding the RLL-constrained string z, a string $y \in \mathbb{Z}_q^k$ is added, and ϕ_k^{-1} employed, to obtain a codeword.

Finally, the asymptotic rate of the code family may also be obtained, thus, giving the capacity of the channel.

Corollary 4.19. *For all $q \geqslant 2$ and $k \geqslant 1$,*

$$\mathsf{cap}_q(*)_k = \mathsf{cap}(\mathrm{RLL}_q(0, k-1)),$$

where $\mathsf{cap}(\mathrm{RLL}_q(0, k-1))$ is the capacity of the q-ary $(0, k-1)$-RLL constrained system.

Proof. We use C_n to denote the code from Construction 4.15, where the subscript n is used to denote the length of the code. It is easy to see that for $n \geqslant k$,

$$q^k M_{\mathrm{RLL}_q(0,k-1)}(n-k) \leqslant |C_n| \leqslant nq^k M_{\mathrm{RLL}_q(0,k-1)}(n-k).$$

Then by standard techniques [92] for constrained coding,

$$\lim_{n\to\infty} \frac{1}{n} \log_2 |C_n| = \mathsf{cap}(\mathrm{RLL}_q(0, k-1)).$$

▮

It is well known (see e.g. [92]) that

$$\mathsf{cap}(\mathrm{RLL}_q(0, k-1)) = \log_2 \lambda(A_q(k-1)),$$

where $\lambda(A_q(k-1))$ is the largest eigenvalue of the $k \times k$ matrix $A_q(k-1)$ defined as

$$A_q(k-1) = \begin{pmatrix} q-1 & 1 & & & \\ q-1 & & 1 & & \\ \vdots & & & \ddots & \\ q-1 & & & & 1 \\ q-1 & & & & \end{pmatrix}. \tag{4.3}$$

As a side note, we comment that an asymptotic (in k) expression for the capacity may be given by

$$\mathsf{cap}(\mathrm{RLL}_q(0, k)) = \log_2 q - \frac{(q-1)\log_2 e}{q^{k+2}}(1 + o(1)). \tag{4.4}$$

This expression agrees with the expression for the binary case $q = 2$ mentioned in [94] without proof or reference. For completeness, we give a short proof of this claim below:

Proof. We need to estimate the largest eigenvalue of $A_q(k)$ from (4.3), i.e., to estimate the largest root λ of its characteristic polynomial

$$\chi_{A_q(k)}(x) = \frac{x^{k+2} - qx^{k+1} + q - 1}{x - 1}.$$

Since this largest root is strictly greater than 1, we can alternatively find the largest root of the polynomial

$$f(x) = x^{k+2} - qx^{k+1} + q - 1.$$

We shall require the following simple bounds. Taking the first term in the Taylor expansion of e^x, and the error term, we have for all $x > 0$,

$$e^x = 1 + xe^{x'},$$

for some $x' \in [0, x]$. Since $x > 0$ and e^x is increasing, we have

$$e^x = 1 + xe^{x'} \leqslant 1 + xe^x,$$

or alternatively,

$$1 - e^x \geqslant -xe^x. \tag{4.5}$$

Similarly, taking the first two terms of the Taylor expansion, for all $x > 0$, we get the well-known bound

$$e^x > 1 + x. \tag{4.6}$$

We return to the main proof. In the first direction, let us first examine what happens when we set

$$x = qe^{-\frac{q-1}{q^{k+2}}}.$$

Then

$$\begin{aligned} f(x) &= q^{k+2}e^{-\frac{q-1}{q^{k+2}}(k+2)} - q^{k+2}e^{-\frac{q-1}{q^{k+2}}(k+1)} + q - 1 \\ &= q^{k+2}e^{-\frac{q-1}{q^{k+2}}(k+2)}\left(1 - e^{\frac{q-1}{q^{k+2}}}\right) + q - 1 \\ &\overset{(a)}{\geqslant} (q-1)\left(1 - e^{-\frac{q-1}{q^{k+2}}(k+1)}\right) \\ &> 0, \end{aligned}$$

where (a) follows by an application of (4.5).

In the other direction, we examine the value of $f(x)$ when we set

$$x = qe^{-\frac{q-1}{q^{k+2}}\alpha},$$

where α is a constant depending on q and k. To specify α we recall $W(z)$, $z \geqslant -\frac{1}{e}$, denotes the Lambert W-function, defined by

$$W(z)e^{W(z)} = z.$$

We define

$$\alpha = \frac{W\left(-\frac{q-1}{q^{k+2}}(k+2)\right)}{-\frac{q-1}{q^{k+2}}(k+2)} = e^{-W\left(-\frac{q-1}{q^{k+2}}(k+2)\right)}.$$

Except for $k = 1$ and $q = 2$, for all other values of the parameters we have

$$-\frac{q-1}{q^{k+2}}(k+2) \geqslant -\frac{1}{e},$$

rendering the use of the W function valid. We also note that for these parameters we have $\alpha \geqslant 1$.

Let us calculate $f(x)$,

$$\begin{aligned}
f(x) &= q^{k+2} e^{-\frac{q-1}{q^{k+2}}(k+2)\alpha} - q^{k+2} e^{-\frac{q-1}{q^{k+2}}(k+1)\alpha} + q - 1 \\
&= q^{k+2} e^{-\frac{q-1}{q^{k+2}}(k+2)\alpha}\left(1 - e^{\frac{q-1}{q^{k+2}}\alpha}\right) + q - 1 \\
&\overset{\text{(a)}}{<} (q-1)\left(1 - \alpha e^{-\frac{q-1}{q^{k+2}}(k+2)\alpha}\right) \\
&\overset{\text{(b)}}{=} (q-1)(1-1) = 0,
\end{aligned}$$

where (a) follows by an application of (4.6), and (b) follows by substituting the value of α.

In summary, $f(x)$ is easily seen to be decreasing in the range $[1, (k+1)q/(k+2)]$, and increasing in the range $[(k+1)q/(k+2), \infty)$, and therefore, its unique largest root λ is in the range

$$q e^{-\frac{q-1}{q^{k+2}}\alpha} \leqslant \lambda \leqslant q e^{-\frac{q-1}{q^{k+2}}}.$$

It is easy to verify that $\alpha = 1 + o(1)$, where $o(1)$ denotes a function decaying to 0 as $k \longrightarrow \infty$. Hence,

$$\lambda = q e^{-\frac{q-1}{q^{k+2}}(1+o(1))},$$

and therefore

$$\begin{aligned}
\mathsf{cap}(\mathrm{RLL}_q(0,k)) &= \log_2 \lambda \\
&= \log_2 q - \frac{(q-1)\log_2 e}{q^{k+2}}(1 + o(1)).
\end{aligned}$$

∎

Having considered $(n, M; *)_k$ codes, we now turn to study $(n, M; t)_k$ codes for $t \in \mathbb{N} \cup \{0\}$. We note that $\mathbb{Z}_q^n$ is an optimal $(n, q^n; 0)_k$ code. Additionally, any $(n, M; *)_k$ code is trivially also an $(n, M; t)_k$ code, though not necessarily optimal.

We know by Lemma 4.14 that the descendant cones of two words overlap if and only if they are k-congruent. Thus, the strategy for constructing $(n, M; *)_k$ codes was to pick single representatives of the equivalence classes of $\sim_k$ as codewords. However, the overlap that is guaranteed by Lemma 4.14 may require a large number of duplication operations. If we are interested in a small enough value of t, then an $(n, M; t)_k$ code may contain several codewords from the same equivalence class. This observation will be formalized in the following, by introducing a metric on k-congruent words, and applying this metric to pick k-congruent codewords.

Fix a length $n \geqslant 1$, and let $x, x' \in \mathbb{Z}_q^n$, $x \sim_k x'$, be two k-congruent words of length n. We define the distance between x and x' as

$$d_k(x, x') = \min\left\{t \geqslant 0 \,\middle|\, D_k^t(x) \cap D_k^t(x') \neq \emptyset\right\}.$$

Since x and x' are k-congruent, Lemma 4.14 ensures that d_k is well defined.

Lemma 4.20. *Let $x, x' \in \mathbb{Z}_q^n$, $x \sim_k x'$, be two k-congruent strings. Denote $\phi_k(x) = (y, z)$ and $\phi_k(x') = (y, z')$. Additionally, let*

$$\sigma_k(z) = (s_0, s_1, \dots, s_r),$$
$$\sigma_k(z') = (s'_0, s'_1, \dots, s'_r).$$

Then

$$d_k(x, x') = \frac{1}{2}\sum_{i=0}^{r}\left|s_i - s'_i\right| = \frac{1}{2} d_{\ell_1}(\sigma_k(z), \sigma_k(z')),$$

where d_{ℓ_1} stands for the ℓ_1-distance function.

Proof. Let x and x' be two strings as required. By Corollary 4.12 we indeed have $y = y'$, and $\mu_k(z) = \mu_k(z')$. In particular, the length of the vectors of the zero signatures of z and z' are the same,

$$|\sigma_k(z)| = |\sigma_k(z')| = r + 1.$$

We now observe that the action of a k-tandem duplication on x corresponds to the addition of a standard unit vector e_i (an all-zero vector except for the ith coordinate which equals 1) to $\sigma_k(z)$.

Let $\tilde{x}$ denote a vector that is a descendant both of x and x', and that requires the least number of k-tandem duplications to reach from x and x'. If we denote $\phi_k(\tilde{x}) = (\tilde{y}, \tilde{z})$, then we have

$$\tilde{y} = y = y',$$
$$\mu_k(\tilde{z}) = \mu_k(z) = \mu_k(z'),$$
$$\sigma_k(\tilde{z}) = (\max(s_0, s'_0), \dots, \max(s_r, s'_r)).$$

Thus,

$$\begin{aligned} d_k(x,x') &= \sum_{i=0}^{r}(\max(s_i, s_i') - s_i) \\ &= \sum_{i=0}^{r}(\max(s_i, s_i') - s_i') \\ &= \frac{1}{2}\sum_{i=0}^{r}\left|s_i - s_i'\right| = \frac{1}{2}d_{\ell_1}(\sigma_k(z), \sigma_k(z')). \end{aligned}$$

∎

From Lemma 4.20 we also deduce that d_k is a metric over any set of k-congruent words of length n.

The following theorem shows that a code is $(n;t)_k$ if and only if the zero signatures of the z-part of k-congruent codewords in the ϕ_k-transform domain, form a constant-weight code in the ℓ_1-metric with distance at least $2(t+1)$. We recall that the ℓ_1-metric weight of a vector $s = s_1s_2\dots s_n \in \mathbb{Z}^n$ is defined as the ℓ_1-distance to the zero vector, i.e.,

$$\mathrm{wt}_{\ell_1}(s) = \sum_{i=1}^{n}|s_i|.$$

Theorem 4.21. *Let $C \subseteq \mathbb{Z}_q^n$, $n \geqslant k$, be a subset of size M. Then C is an $(n,M;t)_k$ code if and only if for each $y \in \mathbb{Z}_q^k$, $z \in \mathbb{Z}_q^{n-k}$, the following sets*

$$C(y,z) = \left\{\sigma_k(z') \,\middle|\, z' \in \mathbb{Z}_q^{n-k}, \mu_k(z) = \mu_k(z'), \phi_k^{-1}(y,z') \in C\right\}$$

are constant-weight $(n(y,z), M(y,z), 2(t+1))$ codes in the ℓ_1-metric, with constant weight

$$\mathrm{wt}_{\ell_1}(\sigma(z)) = \frac{n-k-|\mu_k(z)|}{k},$$

and length

$$n(y,z) = \mathrm{wt}_H(z) + 1 = \mathrm{wt}_H(\mu_k(z)) + 1,$$

where wt_H denotes the Hamming weight.

Proof. In the first direction, let C be an $(n,M;t)_k$ code. Fix y and z, and consider the set $C(y,z)$. Assume to the contrary that there exist distinct $\sigma_k(z'), \sigma_k(z'') \in C(y,z)$, $z', z'' \in \mathbb{Z}_q^{n-k}$, such that $d_{\ell_1}(\sigma_k(z'), \sigma_k(z'')) \leqslant 2t$ (note that d_{ℓ_1} between two vectors of the same weight is even).

The length of the code, $n(y, z)$, is obvious given (4.1). We note that $\sigma_k(z') \neq \sigma_k(z'')$ implies $z' \neq z''$. By definition, we have

$$\mu_k(z) = \mu_k(z') = \mu_k(z'').$$

Thus,

$$\begin{aligned}\mathrm{wt}_{\ell_1}(\sigma(z)) &= \mathrm{wt}_{\ell_1}(\sigma(z')) = \mathrm{wt}_{\ell_1}(\sigma(z'')) \\ &= \frac{n - k - |\mu_k(z)|}{k},\end{aligned}$$

where $|\mu_k(z)|$ denotes the length of the vector $\mu_k(z)$. Additionally, the two codewords

$$c' = \phi_k^{-1}(y, z') \in C \qquad \text{and} \qquad c'' = \phi_k^{-1}(y, z'') \in C$$

are k-congruent and distinct. By Lemma 4.20,

$$d_k(c', c'') = \frac{1}{2} d_{\ell_1}(\sigma_k(z'), \sigma_k(z'')) \leqslant t. \tag{4.7}$$

However, that contradicts the code parameters since we have (4.7) imply $D_k^t(c') \cap D_k^t(c'') \neq \emptyset$, whereas in an $(n, M; t)_k$ code, the t-descendant cones of distinct codewords have an empty intersection.

In the other direction, assume that for every choice of y and z, the corresponding $C(y, z)$ is a constant-weight code with minimum ℓ_1-distance of $2(t + 1)$. Assume to the contrary C is not an $(n, M; t)_k$ code. Therefore, there exist two distinct codewords, $c', c'' \in C$ such that $d_k(c', c'') \leqslant t$.

By Lemma 4.14 we conclude that c' and c'' are k-congruent. Thus, there exist $y \in \mathbb{Z}_q^k$ and $z \in \mathbb{Z}_q^{n-k}$ (z is not necessarily unique) such that,

$$\begin{aligned}\phi_k(c') &= (y, z') \\ \phi_k(c'') &= (y, z'') \\ \mu_k(z) &= \mu_k(z') = \mu_k(z'').\end{aligned}$$

We can now use Lemma 4.20 and obtain

$$d_{\ell_1}(\sigma_k(z'), \sigma_k(z'')) = 2 d_k(c', c'') \leqslant 2t,$$

which contradicts the minimal distance of $C(y, z)$. ∎

With the insight given by Theorem 4.21 we now give a construction for $(n, M; t)_k$ codes.

Construction 4.22. *Fix* $\Sigma = \mathbb{Z}_q$, $k \geqslant 1$, $n \geqslant k$, *and* $t \geqslant 0$. *Furthermore, for all*

$$1 \leqslant m \leqslant n - k + 1,$$
$$0 \leqslant w \leqslant \left\lfloor \frac{n-k}{k} \right\rfloor,$$

fix ℓ_1*-metric codes over* $\mathbb{Z}_q$*, denoted* $C_1(m, w)$*, which are of length* m*, constant* ℓ_1*-weight* w*, and minimum* ℓ_1*-distance* $2(t+1)$*. We construct*

$$C = \left\{ \phi_k^{-1}(y, z) \;\middle|\; y \in \mathbb{Z}_q^k, z \in \mathbb{Z}_q^{n-k}, \right.$$
$$\left. \sigma_k(z) \in C_1\left(\mathrm{wt}_H(\mu_k(z)) + 1, \frac{n-k-|\mu_k(z)|}{k} \right) \right\}.$$

Corollary 4.23. *The code* C *from Construction 4.22 is an* $(n, M; t)_k$ *code.*

Proof. Let $c, c' \in C$ be two k-congruent codewords, i.e., $\phi_k(c) = (y, z)$, $\phi_k(c') = (y, z')$, and $\mu_k(z) = \mu_k(z')$. It follows, by construction, that $\sigma_k(z)$ and $\sigma_k(z')$ belong to the same ℓ_1-metric code with minimum ℓ_1-distance at least $2(t+1)$. By Theorem 4.21, C is an $(n, M; t)_k$ code. ∎

Due to Theorem 4.21, a choice of optimal ℓ_1-metric codes in Construction 4.22 will result in optimal $(n, M; t)_k$ codes. We are unfortunately unaware of explicit construction for such codes. However, we may deduce such a construction from codes for the similar Lee metric (e.g., [95]), while applying a standard averaging argument for inferring the existence of a constant-weight code. We leave the construction of such codes for a future work.

4.4 $\leqslant k$-Tandem-Duplication Codes

In this section, we consider error-correcting codes that correct duplications of length at most k, which correspond to $S_{\leqslant k}$. In particular, we present constructions for codes that can correct any number of duplications of length $\leqslant 3$ as well as a lower bound on the capacity of the corresponding channel. In the case of duplications of length $\leqslant 2$ we give optimal codes, and obtain the exact capacity of the channel.

It is worth noting that the systems $S_{\leqslant k}$ were studied in the context of formal languages [89] and also in the context of coding and information theory [16]. In [89], it was shown that $S_{\leqslant k}$, with $k \geqslant 4$, is not a regular language for alphabet size $|\Sigma| \geqslant 3$. However, it was proved in [16] that $S_{\leqslant 3}$ is indeed a regular language irrespective of the seed string and the alphabet size.

In this chapter, we will show that strings that can be generated by bounded tandem string-duplication systems with maximum duplication length 3 have a unique duplication root, a fact that will be useful for our code construction. Theorem 4.26 formalizes this statement. To simplify our description, we use the term *square* to denote a sequence of the form $\alpha^2 = \alpha\alpha$, where $\alpha \in \Sigma^*$. We begin with the following definition.

Definition 4.24. Let two squares $y_1 = \alpha\alpha \in \Sigma^+$ and $y_2 = \beta\beta \in \Sigma^+$ appear as substrings of some string $u \in \Sigma^*$, i.e.,

$$u = x_1 y_1 z_1 = x_2 y_2 z_2,$$

with $|x_1| = i$, $|x_2| = j$. We say y_1 and y_2 are *overlapping squares in u* if the following conditions both hold:

1. $i \leqslant j \leqslant i + 2|\alpha| - 1$ or $j \leqslant i \leqslant j + 2|\beta| - 1$.
2. If $i = j$, then $\alpha \neq \beta$.

Example 4.25. Consider the sequence u,

$$u = 0\,1\,\overbrace{\underbrace{2\,3\,2\,3}_{\beta_1\beta_1}\underbrace{4\,5\,2\,4\,5\,2}_{\beta_2\beta_2}3\,2\,3\,4\,5\,\rlap{$\underbrace{}_{\beta_3\beta_3}$}2\,4\,5}^{\alpha\alpha}6\,2\,4\,5\,6\,7,$$

where $\alpha\alpha$ and $\beta_i\beta_i$ for each $i \in \{1, 2, 3\}$ are overlapping squares.

The following theorem shows that every word has a unique root under tandem deduplication of length up to 3.

Theorem 4.26. *For any $z \in \Sigma^*$ we have $\left|R_{\leqslant 3}(z)\right| = 1$.*

Proof. Fix some $z \in \Sigma^*$, and assume z has exactly m distinct roots, $R_{\leqslant 3}(z) = \{y_1, y_2, \ldots, y_m\}$. Let us assume to the contrary that $m \geqslant 2$.

Let us follow a deduplication sequence starting at $x_0 = z$. At each step, we deduplicate $x_i \underset{\leqslant 3}{\Longleftarrow} x_{i+1}$, and we must have $\left|R_{\leqslant 3}(x_i)\right| \geqslant \left|R_{\leqslant 3}(x_{i+1})\right|$. At each step, out of the possible immediate ancestors of x_i, we choose x_{i+1} to be one with $\left|R_{\leqslant 3}(x_{i+1})\right| \geqslant 2$ if possible. Since the end-point of a deduplication process is an irreducible sequence, we must reach a sequence x in the deduplication process with the following properties:

1. $z \overset{*}{\underset{\leqslant 3}{\Longleftarrow}} x$
2. $\left|R_{\leqslant 3}(x)\right| \geqslant 2$
3. For each $x' \in \Sigma^*$ such that $x \underset{\leqslant 3}{\Longleftarrow} x'$, $\left|R_{\leqslant 3}(x')\right| = 1$.
4. There exist $v, w \in \Sigma^*$ such that $x \underset{\leqslant 3}{\Longleftarrow} v$ and $x \underset{\leqslant 3}{\Longleftarrow} w$ with $\left|R_{\leqslant 3}(v)\right| = \left|R_{\leqslant 3}(w)\right| = 1$.
5. $R_{\leqslant 3}(v) = \{y_i\} \neq \{y_j\} = R_{\leqslant 3}(w)$.

Intuitively, in the deduplication process starting from z, we reach a sequence x with more than one root, but any following single deduplication moves us into a single descendant cone of one of the roots of z. We note that all ancestors of v must have a single root y_i, and all ancestors of w must have a single root y_j.

Thus, x must contain a square $u_v u_v$ whose deduplication results in v, and a square $u_w u_w$ whose deduplication results in w. We contend that the squares $u_v u_v$ and $u_w u_w$ overlap. Otherwise, if $u_v u_v$ and $u_w u_w$ do not overlap in x, we may deduplicate them in any order to obtain the same result. Hence, there exists $t \in \Sigma^*$ such that $v \underset{\leqslant 3}{\Longleftarrow} t$ and $w \underset{\leqslant 3}{\Longleftarrow} t$. But then, since t is an ancestor both of v and w,

$$\{y_i\} = R_{\leqslant 3}(v) = R_{\leqslant 3}(t) = R_{\leqslant 3}(w) = \{y_j\},$$

a contradiction.

We now know that $u_v u_v$ and $u_w u_w$ must overlap. We also note $|u_v|, |u_w| \leqslant 3$. Let $a, b, c \in \Sigma$ be three distinct symbols. If the alphabet is smaller, then some of the cases below may be ignored, and the proof remains the same. We use brute force to enumerate all the overlapping squares, and the results are given in Table 4.1. In the table, each string describes the shortest subsequence that contains the overlapping squares. The enumeration is complete, up to a permutation of the alphabet symbols. For example, if $u_v = abc$ and $u_w = cbc$, the corresponding string appears in the table as $abcabcbccbc$ since we have,

$$\overbrace{a\,b\,c}^{u_v}\,\overbrace{a\,b\,c}^{u_v}\!\!\!\!\!\!\!\;\;\underbrace{\,b\,c}_{u_w}\,\underbrace{c\,b\,c}_{u_w}$$

$\lvert u_v\rvert$	$\lvert u_w\rvert$	Overlapping squares u_v and u_w
1	1	aaa
1	2	$aaaa, aaaaa, aabab, ababb$
1	3	$aaaaaa, aaaaaaa, aaabaab, aabaab, aabaaba, aabaabb, aabbabb, aabcabc, abaaba, abaabaa, abbabb, abbabbb, abcabcc$
2	2	$aaaaa, aaaaaa, aaaaaaa, aaaabab, ababa, ababab, abababa, ababbbb, ababcbc$
2	3	$aaaaaa, aaaaaaa, aaaaaaaa, aaaaaaaaa, aaaaaabab, aaaaabaab, aaaabaab, aaaabaaba, aaaabbabb, aaaabcabc$ $aabaabab, aabaababa, aabaabbbb, aabaabcbc, abaabaaaa, abaabab, abaababa, abaababab, abaabacac, ababaaba$ $ababaabaa, abababbab, ababacbac, ababbab, ababbabb, ababbabba, ababbbbbb, ababbcbbc, ababcabc, ababcabca$ $ababcbbcb, ababccbcc, ababcdbcd, abbabbaba, abbabbbb, abbabbbbb, abbabbcbc, abcabcaca, abcabcbc, abcabcbcb$ $abcabcccc, abcabcdcd$
3	3	$aaaaaaa, aaaaaaaa, aaaaaaaaa, aaaaaaaaaa, aaaaaaaaaaa, aaaaaaabaab, aaaaaabaab, aaaaaabaaba, aaaaaabbabb, aaaaaabcabc$ $aabaaba, aabaabaa, aabaabaab, aabaabaaba, aabaabaabaa, aabaababbab, aabaabacbac, aabaabbabb, aabaabbabba, aabaabbbbbb$ $aabaabbcbbc, aabaabcabc, aabaabcabca, aabaabcbbcb, aabaabccbcc, aabaabcdbcd, abaabaa, abaabaaaaaa, abaabaab, abaabaaba$ $abaabaabaa, abaabaabaab, abaabaacaac, abaababaaba, abaababbab, abaababbabb, abaababcabc, abaabacaaca, abaabacbac, abaabacbacb$ $abaabaccacc, abaabacdacd, abbabba, abbabbaabaa, abbabbab, abbabbabb, abbabbabba, abbabbabbab, abbabbacbac, abbabbbabba$ $abbabbbbbb, abbabbbbbbb, abbabbbcbbc, abbabbcabca, abbabbcbbc, abbabbcbbcb, abbabbccbcc, abbabbcdbcd, abcabca, abcabcaacaa$ $abcabcab, abcabcabc, abcabcabca, abcabcabcab, abcabcaccac, abcabcadcad, abcabcbacba, abcabcbbcb, abcabcbbcbb, abcabcbccbc$ $abcabcbdcbd, abcabccacca, abcabccbcc, abcabccbccb, abcabcccccc, abcabccdccd, abcabcdacda, abcabcdbcd, abcabcdbcdb, abcabcdccdc$ $abcabcddcdd, abcabcdecde$

Table 4.1: A list of all overlapping squares of length at most 3 (up to a permutation of the alphabet symbols)

It is tedious, yet easy, to check that each of the cases in Table 4.1 has a unique root if deduplication of maximum length 3 is allowed[2]. In the above example, indeed, the only possible root is abc,

$$\underline{abcabc}bccbc \underset{\leqslant 3}{\Longleftarrow} abcbccbc \overset{*}{\underset{\leqslant 3}{\Longleftarrow}} abc,$$

$$abcab\underline{cbccbc} \underset{\leqslant 3}{\Longleftarrow} abcabcbc \overset{*}{\underset{\leqslant 3}{\Longleftarrow}} abc.$$

Let $x = \alpha\beta\gamma \in \Sigma^*$, where β covers exactly the overlapping squares, and is one of the above listed cases. Then, by deduplication of $u_v u_v$ from β in x, we get v, and by deduplication of $u_w u_w$ from β in x, we get w. However, since β has a unique root, we may deduplicate v and w to the same word $t = \alpha\beta'\gamma \in \Sigma^*$, where $R(\beta) = \{\beta'\}$, i.e., β' is the unique root of β. Thus, t is an ancestor of both v and w. Again,

$$\{y_i\} = R_{\leqslant 3}(v) = R_{\leqslant 3}(t) = R_{\leqslant 3}(w) = \{y_j\},$$

which is a contradiction. ∎

Corollary 4.27. *For any $z \in \Sigma^*$ we have $\left|R_{\leqslant k}(z)\right| = 1$ for $k = 1, 2$.*

In a similar fashion to the previous section, we define the following relation. We say $x, x' \in \Sigma^*$ are $\leqslant 3$-congruent, denoted $x \sim_{\leqslant 3} x'$, if $R_{\leqslant 3}(x) = R_{\leqslant 3}(x')$. Clearly $\sim_{\leqslant 3}$ is an equivalence relation. Having shown any sequence has a unique root when duplicating up to length 3, we obtain the following corollary.

Corollary 4.28. *For any two words $x, x' \in \Sigma^*$, if*

$$D^*_{\leqslant 3}(x) \cap D^*_{\leqslant 3}(x') \neq \emptyset$$

then $x \sim_{\leqslant 3} x'$.

[2]We used a computer to verify the list of overlapping squares and the fact that they each have a unique root.

We note that unlike Lemma 4.14, we do not have $x \sim_{\leqslant 3} x'$ necessarily imply that their descendant cones intersect. Here is a simple example illustrating this case. Fix $q = 3$, and let $x = 012012$ and $x' = 001122$. We note that $x \sim_{\leqslant 3} x'$, since

$$R_{\leqslant 3}(x) = R_{\leqslant 3}(x') = \{012\}.$$

However, $D^*_{\leqslant 3}(x) \cap D^*_{\leqslant 3}(x') = \emptyset$ since all the descendants of x have a 0 to the right of a 2, whereas none of the descendants of x' do.

We are missing a simple operator which is required to define an error-correcting code. For any sequence $x \in \Sigma^+$, we define its k-suffix-extension to be

$$\xi_k(x) = x(\mathrm{Suff}_1(x))^k,$$

i.e., the sequence x with its last symbol repeated an extra k times.

Construction 4.29. *Let Σ be some finite alphabet. We construct the code*

$$C = \bigcup_{i=1}^{n} \left\{ \xi_{n-i}(x) \mid x \in \mathrm{Irr}_{\leqslant 3}(i) \right\}.$$

Theorem 4.30. *The code C from Construction 4.29 is an $(n, M; *)_{\leqslant 3}$ code, where*

$$M = \sum_{i=1}^{n} \left| \mathrm{Irr}_{\leqslant 3}(i) \right|.$$

Proof. The parameters of the code are obvious. Since the last letter duplication induced by the suffix extension may be deduplicated, we clearly have exactly one codeword from each equivalence class of $\sim_{\leqslant 3}$. By Corollary 4.28, the descendant cones of the codewords do not intersect and the code can indeed correct all errors. ∎

Example 4.31. Let $n = 4$ and $\Sigma = \{0, 1, 2\}$. The code $C_{\leqslant 3}$ obtained by Construction 4.29 is given by

$$C_{\leqslant 3} = \bigcup_{\{a,b,c\}=\Sigma} \left\{ \underline{a}aaa, \underline{ab}bb, \underline{aba}a, \underline{abc}c, \underline{abac}, \underline{abca}, \underline{abcb} \right\},$$

where in each codeword, the irreducible part is underlined, and the union is taken over all possible assignments of a, b, and c, to distinct symbols of Σ. Thus $\left|C_{\leqslant 3}\right| = 7 \cdot 3! = 42$.

For the remainder of the section we denote by $\mathrm{Irr}_{q;\leqslant 3}$ the set of irreducible words with respect to $\underset{\leqslant 3}{\Longleftarrow}$ over $\mathbb{Z}_q$, in order to make explicit the dependence on the size of the alphabet. We also assume $q \geqslant 3$, since $q = 2$ is a trivial case with

$$\mathrm{Irr}_{2;\leqslant 3} = \{0, 1, 01, 10, 010, 101\}. \tag{4.8}$$

We observe that $\mathrm{Irr}_{q;\leqslant 3}$ is a regular language. Indeed, it is defined by a finite set of subsequences we would like to avoid. This set is exactly

$$\mathcal{F}_q = \left\{ uu \in \mathbb{Z}_q^* \,\middle|\, 1 \leqslant |u| \leqslant 3 \right\}.$$

We can easily construct a finite directed graph with labeled edges such that paths in the graph generate exactly $\mathrm{Irr}_{q;\leqslant 3}$. This graph is obtained by taking the De Bruijn graph $\mathcal{G}_q = (\mathcal{V}_q, \mathcal{E}_q)$ of order 5 over $\mathbb{Z}_q$, i.e., $\mathcal{V}_q = \mathbb{Z}_q^5$, and edges of the form $(a_1, a_2, a_3, a_4, a_5) \longrightarrow (a_2, a_3, a_4, a_5, a_6)$, for all $a_i \in \mathbb{Z}_q$ (for more on De Bruijn graphs the reader is referred to [96, Chapter 8]). Thus, each edge is labeled with a word $w = (a_1, a_2, a_3, a_4, a_5, a_6) \in \mathbb{Z}_q^6$. We then remove all edges labeled by words $\alpha\beta\gamma \in \mathbb{Z}_q^6$ such that $\beta \in \mathcal{F}_q$. We call the resulting graph $\mathcal{G}'_q$. It is easy to verify that each path in $\mathcal{G}'_q$ generates a sequence of sliding windows of length 6. Reducing each window to its first letter we get exactly $\mathrm{Irr}_{q;\leqslant 3}$. An example showing $\mathcal{G}'_3$ is given in Figure 4.1. Finally, using known techniques [92], we can calculate $\mathsf{cap}(\mathrm{Irr}_{q;\leqslant 3})$.

Corollary 4.32. *For all $q \geqslant 3$,*

$$\mathsf{cap}_q(*)_{\leqslant 3} \geqslant \mathsf{cap}(\mathrm{Irr}_{q;\leqslant 3}).$$

Proof. Let M_n denote the size of the length n code over $\mathbb{Z}_q$ from Construction 4.29. By definition, $A_q(n; *)_{\leqslant 3} \geqslant M_n$. We note that trivially

$$M_n = \sum_{i=1}^{n} \left|\mathrm{Irr}_{q;\leqslant 3}(i)\right| \geqslant \left|\mathrm{Irr}_{q;\leqslant 3}(n)\right|.$$

Plugging this into the definition of the capacity gives us the desired claim. ∎

Example 4.33. Using the constrained system presented in Figure 4.1 that generates $\mathrm{Irr}_{3;\leqslant 3}$, we can calculate

$$\mathsf{cap}_3(*)_{\leqslant 3} \geqslant 0.347934.$$

Stronger statements may be given when the duplication length is upper bounded by 2 instead of 3.

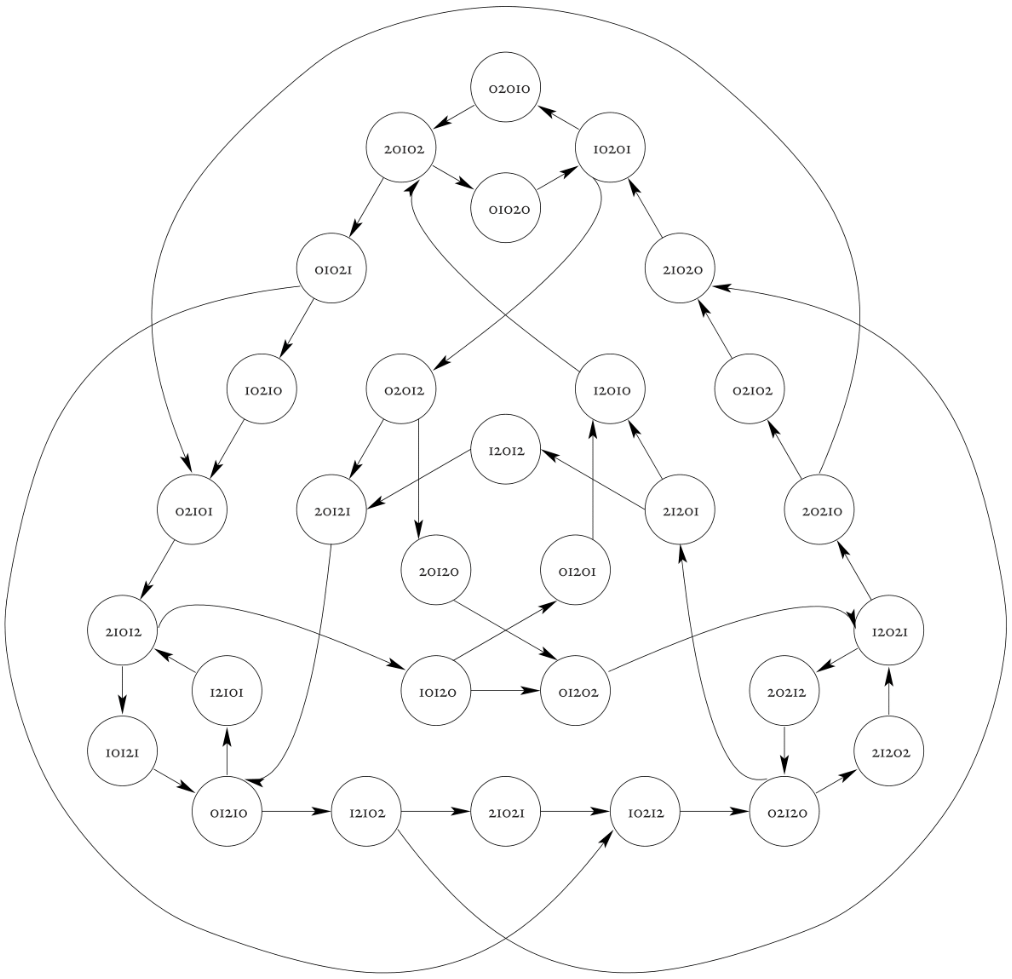


Figure 4.1: The graph $\mathcal{G}'_3$ producing the set of ternary irreducible words $\mathrm{Irr}_{3;\leqslant 3}$. Vertices without edges were removed as well.

Lemma 4.34. *For all $x, x' \in \Sigma^*$, we have*

$$D^*_{\leqslant 2}(x) \cap D^*_{\leqslant 2}(x') \neq \emptyset$$

if and only if $x \sim_{\leqslant 2} x'$.

Proof. In the first direction, assume $x \nsim_{\leqslant 2} x'$. By the uniqueness of the root from Corollary 4.27, let us denote $R_{\leqslant 2}(x) = \{u\}$ and $R_{\leqslant 2}(x') = \{u'\}$, with $u \neq u'$. If there exists $w \in D^*_{\leqslant 2}(x) \cap D^*_{\leqslant 2}(x')$, then w is a descendant of both u and u', therefore u and $u' \in R_{\leqslant 2}(w)$, which is a contradiction. Hence, no such w exists, i.e., $D^*_{\leqslant 2}(x) \cap D^*_{\leqslant 2}(x') = \emptyset$.

In the other direction, assume $x \sim_{\leqslant 2} x'$. We construct a word $w \in D^*_{\leqslant 2}(x) \cap D^*_{\leqslant 2}(x')$. Let $R_{\leqslant 2}(x) = R_{\leqslant 2}(x') = \{v\}$, and denote $v = a_1 a_2 \ldots a_m$, where $a_i \in \Sigma$. Consider the tandem-duplication string system $S_{\leqslant 2} = (\Sigma, v, \mathcal{T}_{\leqslant 2})$. Using [16], the regular expression for the language generated by $S_{\leqslant 2}$ is given by

$$a_1^+ a_2^+ (a_1^+ a_2^+)^* a_3^+ (a_2^+ a_3^+)^* \dots a_m^+ (a_{m-1}^+ a_m^+)^*.$$

Since $x, x' \in S$, we have

$$x = \prod_{i=1}^{\alpha_1} (a_1^{p_{1i}} a_2^{q_{1i}})\, a_3^{q_{21}} \prod_{i=2}^{\alpha_2} (a_2^{p_{2i}} a_3^{q_{2i}}) \\ \dots a_m^{q_{(m-1)1}} \prod_{i=2}^{\alpha_{m-1}} (a_{m-1}^{p_{(m-1)i}} a_m^{q_{(m-1)i}}),$$

and

$$x' = \prod_{i=1}^{\beta_1} (a_1^{e_{1i}} a_2^{f_{1i}})\, a_3^{f_{21}} \prod_{i=2}^{\beta_2} (a_2^{e_{2i}} a_3^{f_{2i}}) \\ \dots a_m^{f_{(m-1)1}} \prod_{i=2}^{\beta_{m-1}} (a_{m-1}^{e_{(m-1)i}} a_m^{f_{(m-1)i}}),$$

where $\prod$ represents concatenation and $p_{ji}, q_{ji}, e_{ji}, f_{ji}, \alpha_j, \beta_j \geqslant 1$. Now, it is easy to observe that we can obtain

$$w = \prod_{i=1}^{\gamma_1} (a_1^{g_1} a_2^{h_1})\, a_3^{h_2} \prod_{i=2}^{\gamma_2} (a_2^{g_2} a_3^{h_2}) \dots a_m^{h_{m-1}} \prod_{i=2}^{\gamma_{m-1}} (a_{m-1}^{g_{m-1}} a_m^{h_{m-1}})$$

by doing tandem duplication of length up to 2 on x and x', and choosing $\gamma_j = \max\{\alpha_j, \beta_j\}$, $g_j = \max_i\{p_{ji}, e_{ji}\}$, and $h_j = \max_i\{q_{ji}, f_{ji}\}$. Note, p_{ji} and q_{ji} are assumed to be 0 for $i > \alpha_j$ and e_{ji} and f_{ji} are assumed to be 0 for $i > \beta_j$. Thus, $w \in D_{\leqslant 2}(x) \cap D_{\leqslant 2}(x')$. ▮

Construction 4.35. *Let Σ be some finite alphabet. The constructed code is*

$$C = \bigcup_{i=1}^{n} \{\xi_{n-i}(x) \mid x \in \mathrm{Irr}_{\leqslant 2}(i)\}.$$

Theorem 4.36. *The code C from Construction 4.35 is an optimal $(n, M; *)_{\leqslant 2}$ code, where*

$$M = \sum_{i=1}^{n} |\mathrm{Irr}_{\leqslant 2}(i)|.$$

Proof. The correctness of the parameters follows the same reasoning as the proof of Theorem 4.30. By Lemma 4.34, any two distinct codewords of an $(n; *)_{\leqslant 2}$ code must belong to different equivalence classes of $\sim_{\leqslant 2}$. The code C of Construction 4.35 contains exactly one codeword from each equivalence class of $\sim_{\leqslant 2}$, and thus, it is optimal. ▮

Corollary 4.37. *For all $q \geqslant 3$,*

$$\mathsf{cap}_q(*)_{\leqslant 2} = \mathsf{cap}(\mathrm{Irr}_{q;\leqslant 2}).$$

Proof. Let M_n denote the size of the length n code over $\mathbb{Z}_q$ from Construction 4.35. By definition, $A_q(n;*)_{\leqslant 2} \geqslant M_n$. We note that trivially

$$M_n = \sum_{i=1}^{n} \left|\mathrm{Irr}_{q;\leqslant 2}(i)\right| \geqslant \left|\mathrm{Irr}_{q;\leqslant 2}(n)\right|.$$

Additionally, $\left|\mathrm{Irr}_{q;\leqslant 2}\right|(n)$ is monotone increasing in n since any irreducible length-n word x may be extended to an irreducible word of length $n+1$ by adding a letter that is not one of the last two letters appearing in x. Thus,

$$M_n = \sum_{i=1}^{n} \left|\mathrm{Irr}_{q;\leqslant 2}(i)\right| \leqslant n\left|\mathrm{Irr}_{q;\leqslant 2}(n)\right|.$$

Plugging this into the definition of the capacity gives us the desired claim. ∎

4.5 Uncertainty in Coding for $S_{\leqslant k}$

Suppose we have an error-correcting code designed specifically to correct tandem duplications from $\mathcal{T}_{\leqslant k}$. However, if we incorrectly estimate the value of k, we may end up transmitting our codewords over a channel that tandem-duplicates using rules from $\mathcal{T}_{\leqslant k'}$, $k' \neq k$. This mismatch between the design parameter and the actual channel parameter may cause mis-decoding. We quantify the number of possible mis-decodings as the channel-code uncertainty, which we study in this section.

In particular, the following code was constructed for correcting any number of tandem-duplication errors of length $\leqslant k$, for $k = 2$ and $k = 3$,

$$C_{\leqslant k}(n) \triangleq \bigcup_{i=1}^{n} \left\{\xi_{n-i}(x) \mid x \in \mathrm{Irr}_{\leqslant k}(i)\right\}.$$

where for any sequence $x = x_0 \dots x_{n-1} \in \Sigma^+$, $x_i \in \Sigma$, its ℓ-suffix-extension

$$\xi_\ell(x) \triangleq x x_{n-1}^{\ell},$$

i.e., the sequence x with its last symbol repeated an extra ℓ times.

Motivated by the error-correcting code defined above, in this section we consider the problem of sending codewords of the form

$$C_U(n) \triangleq \bigcup_{i=1}^{n} \{\xi_{n-i}(x) \mid x \in \mathrm{Irr}_U(i)\} \tag{4.9}$$

through a $\mathcal{T}_{\leqslant 3}$ channel. Here $U \subseteq \mathbb{N}$ is a finite set of positive integers, and Irr_U denotes the set of strings not containing a tandem repeat of length appearing in U. We quantify this problem by defining *uncertainty*, which measures the size of the maximum subset $M_U(n) \subseteq C_U(n)$, such that $D^*_{\leqslant 3}(c) = y$ for every $c \in M_U(n)$ and some $y \in \Sigma^*$. Mathematically,

$$\mathrm{Unc}_U(n) \triangleq \max_{y\in\Sigma^*} \left|\left\{x \in C_U(n) \mid y \in D^*_{\leqslant 3}(x)\right\}\right|. \tag{4.10}$$

We first recall some results from the previous section which we summarize in the following lemma.

Lemma 4.38. *For any $y \in \Sigma^*$ there exists a unique $x \in \mathrm{Irr}_{\leqslant 3}$ such that $y \in D^*_{\leqslant 3}(x)$. Additionally, for $y_1, y_2 \in \Sigma^*$, we have $D^*_{\leqslant 3}(y_1) \cap D^*_{\leqslant 3}(y_2) \neq \emptyset$ if and only if there exists $x \in \mathrm{Irr}_{\leqslant 3}$ such that $y_1, y_2 \in D^*_{\leqslant 3}(x)$.*

We also make the following trivial observation. For all $k, k' \in \mathbb{N}$, $k \leqslant k'$, we have $\mathrm{Irr}_{\leqslant k'} \subseteq \mathrm{Irr}_{\leqslant k}$, and therefore $C_{\leqslant k'}(n) \subseteq C_{\leqslant k}(n)$.

The next lemma provides a characterization of the uncertain codewords, and gives an expression for the uncertainty.

Lemma 4.39. *For all $n \in \mathbb{N}$, and finite $U \subseteq \mathbb{N}$,*

$$\mathrm{Unc}_U(n) = \max_{s\in\mathrm{Irr}_{\leqslant 3}} \left|D^*_{\leqslant 3}(s) \cap C_U(n)\right|.$$

Proof. By (4.10) we are looking for elements of $C_U(n)$ that share some descendant when using $\mathcal{T}_{\leqslant 3}$. By Lemma 4.38 they must also share a unique irreducible ancestor (under $\mathcal{T}_{\leqslant 3}$). ∎

Before proceeding we need another simple lemma.

Lemma 4.40. *Let $U \subseteq \mathbb{N}$ be finite, with $\{1,2,3\} \cap U \in \{\emptyset, \{1\}, \{1,2\}, \{1,2,3\}\}$. Then for any $s \in \Sigma^*$,*

$$D^*_{\leqslant 3}(s) \cap \mathrm{Irr}_U = D^*_{\{1,2,3\}\setminus U}(s) \cap \mathrm{Irr}_U\,.$$

Proof. The right-hand side of the equation is trivially contained in the left-hand side. Thus, we only need to prove any $x \in D^*_{\leqslant 3}(s) \cap \mathrm{Irr}_U$ may be derived from s using tandem-duplication operations with lengths from $\{1,2,3\} \setminus U$. This, again, is trivial when $\{1,2,3\} \cap U \in \{\emptyset, \{1,2,3\}\}$.

We now observe that once a tandem duplication of length 1 is employed, all future descendants contain a substring aa, $a \in \Sigma$. Thus, if $\{1,2,3\} \cap U = \{1\}$, and $x \in D^*_{\leqslant 3}(s) \cap \mathrm{Irr}_U$, i.e., x does not contain a substring aa, $a \in \Sigma$, then deriving x from s using $\mathcal{T}_{\leqslant 3}$ does not require a tandem duplication of length 1. Similarly, if a tandem duplication of length 2 is employed, a substring of the form $abab$, $a, b \in \Sigma$ is created. The only way to eliminate it is by tandem duplication of length 1, which in itself cannot be later eliminated. Thus, the case of $\{1,2,3\} \cap U = \{1,2\}$ is handled as well. ▮

This brings us to the following corollary, which provides an upper bound on the uncertainty.

Corollary 4.41. *Let $U \subseteq \mathbb{N}$ be finite, with $\{1,2,3\} \cap U \in \{\emptyset, \{1\}, \{1,2\}, \{1,2,3\}\}$. Then for any $s \in \Sigma^*$,*

$$\mathrm{Unc}_U(n) \leqslant \max_{s \in \mathrm{Irr}_{\leqslant 3}} \left| D^*_{\{1,2,3\} \setminus U}(s) \cap \Sigma^{\leqslant n} \right|. \tag{4.11}$$

Proof. We have,

$$\begin{aligned}
\mathrm{Unc}_U(n) &\overset{(a)}{=} \max_{s \in \mathrm{Irr}_{\leqslant 3}} \left| D^*_{\leqslant 3}(s) \cap C_U(n) \right| \\
&\overset{(b)}{\leqslant} \max_{s \in \mathrm{Irr}_{\leqslant 3}} \sum_{i=1}^{n} \left| D^*_{\leqslant 3}(s) \cap \mathrm{Irr}_U(i) \right| \\
&\overset{(c)}{=} \max_{s \in \mathrm{Irr}_{\leqslant 3}} \sum_{i=1}^{n} \left| D^*_{\{1,2,3\} \setminus U}(s) \cap \mathrm{Irr}_U(i) \right| \\
&\leqslant \max_{s \in \mathrm{Irr}_{\leqslant 3}} \left| D^*_{\{1,2,3\} \setminus U}(s) \cap \Sigma^{\leqslant n} \right|,
\end{aligned}$$

where (a) is by Lemma 4.39, (b) is by (4.9), and (c) is by Lemma 4.40. ▮

We note that for large values of n, the right-hand side of (4.11) is dominated by a term of the form $|\Sigma|^{cn}$, where $c = \mathsf{cap}(S^0_{\{1,2,3\} \setminus U}(s))$, which by [13, 16], is known in some cases.

We now turn to another issue of importance. A mismatch between a code designed for the $\mathcal{T}_{\leqslant k}$-channel and an actual $\mathcal{T}_{\leqslant k'}$-channel, $k < k'$, may be avoided if we can determine the value of k'. As a first step, we set out to find substrings that may never occur in $D^*_{\leqslant k}(s)$, but may appear as substrings of $D^*_{\leqslant k+1}(s)$. An occurrence of such substrings in a channel's output can be used as a marker indicating it is a $\mathcal{T}_{\leqslant k'}$-channel, with $k' \geqslant k+1$.

A closely related work is [16], where the expressiveness of bounded tandem duplication string systems was characterized. The relevant results of [16] are summarized in Table 2.2.

From Table 2.2, we observe the following:

- For $|\Sigma| = 2$, $S_{\leqslant k}$ is fully expressive for any $k > 1$ and was shown to generate any binary string from some seed with $k > 1$ in [16]. Hence there exists no binary string which cannot be generated by $S_{\leqslant k}(s)$ system but can be generated by $S_{\leqslant k+1}(s)$ system for $k > 1$.
- For $|\Sigma| = 3$, $S_{\leqslant k}$ is fully expressive for $k > 3$, but it is not known whether an $S_{\leqslant k+1}$ system can generate anything new over $S_{\leqslant k}$ for $k > 3$. Since, the system is fully expressive this problem is intuitively harder compared to higher alphabets.
- For $|\Sigma| > 3$, since we do not have full expressiveness for any k, finding examples which can be generated by $S_{\leqslant k+1}(s)$ but not by $S_{\leqslant k}(s)$ is intuitively easier.

To summarize, for $|\Sigma| > 3$, we should gain in expressiveness by increasing k. The next two lemmas prove this statement. We recall that a string is called *squarefree* if it does not contain a substring of the form ww, with $w \in \Sigma^+$.

Lemma 4.42. *Let $\Sigma = \mathbb{Z}_q$, $q \geqslant 4$, and $k > 0$. If $z \in (\mathbb{Z}_q \setminus \{0\})^k$ is squarefree, and $s \in \mathbb{Z}_q^*$ does not contain $w \triangleq 0z0$ as a substring, then there is no $y \in D^*_{\leqslant k}(s)$ which contains w as a substring.*

Proof. Let $x \in D^*_{\leqslant k}(s)$ and assume $v = v_1 v_2 \dots v_\ell$ is a substring of x, where $\ell > k$. If $v \in \mathrm{Irr}_{\leqslant k}$, then by [16], either $v_1 = v_{1+i}$ for some $2 \leqslant i \leqslant k$, or $v_\ell = v_{\ell - j}$ for some $2 \leqslant j \leqslant k$. We note that $w \triangleq 0z0$ does not satisfy these requirements. On the other hand, $w \in \mathrm{Irr}_{\leqslant k}$, and the claim follows. ∎

Lemma 4.43. *Let $\Sigma = \mathbb{Z}_q$, $q \geqslant 4$, and $k > 0$. If $z \in (\mathbb{Z}_q \setminus \{0\})^k$ is squarefree, and $s \in \mathbb{Z}_q^*$ contains $w' \triangleq 0z$ as a substring, then there exists $y \in D^*_{\leqslant k+1}(s)$ which contains $w \triangleq 0z0$ as a substring.*

Proof. Simply duplicate the w' part (of length $k+1$) to obtain a string with w as a substring. ∎

4.6 Duplication Roots

In Section 4.3, we stated that if the duplication length is uniform (i.e., a constant k), then every sequence has a unique root. Further in Section 4.4, we proved in Theorem 4.26 that if the duplication length is bounded by 3 (i.e. $\leqslant 3$), then again every sequence will have a unique root. The full characterization of all cases that have a unique root is stated in Theorem 4.50. Before moving to Theorem 4.50, we present an example and some lemmas required to prove the theorem.

Example 4.44. Let $U = \{2, 3, 4\}$ be a set of duplication lengths and $\Sigma = \{1, 2, 3\}$. Consider

$$z = 1\,2\,3\,\overbrace{2\,1\,2\,3\,2\,3\,2}^{\alpha\alpha}\ldots$$

$$z = \overbrace{1\,2\,3\,2\,1\,2\,3\,2}^{\alpha\alpha}\underbrace{\,3}_{}$$

The sequence z has two tandem repeats $\alpha\alpha$ and $\beta\beta$ with $|\alpha| = 4$ and $|\beta| = 2$. If we deduplicate $\alpha\alpha$ first from z , we get

$$123212323 \underset{4}{\Longleftarrow} 12323 \underset{2}{\Longleftarrow} 123.$$

However, if we deduplicate $\beta\beta$ first from z we get

$$123212323 \underset{2}{\Longleftarrow} 1232123.$$

Both 123 and 1232123 are irreducible and thus roots of z.

Theorem 4.50 generalizes the statement presented in the example above to any set of duplication lengths. We naturally extend all previous notation to allow duplication and deduplication of several lengths by replacing the usual k subscript with a set U, where $U \subseteq \mathbb{N}$. For example, $R_U(z)$ denotes the set of roots obtained via a sequence of deduplications with lengths from U, starting with the string z. The property we would like to study is formally defined next.

Definition 4.45. Let $\Sigma \neq \emptyset$ be an alphabet, and $U \subseteq \mathbb{N}$, $U \neq \emptyset$, a set of tandem-duplication lengths. We say (Σ, U) is a *unique-root pair*, iff for all $z \in \Sigma^*$ we have $|R_U(z)| = 1$. Otherwise, we call (Σ, U) a *non-unique-root pair*.

We observe that the actual identity of the letters in the alphabet is immaterial, and only the size of Σ matters. Additionally, simple monotonicity is evident: If (Σ, U) is a unique-root pair, then so is (Σ', U), for all $\Sigma' \subseteq \Sigma$. Similarly, if (Σ, U) is a non-unique-root pair, then so is (Σ', U), for all $\Sigma \subseteq \Sigma'$.

The following sequence of lemmas will provide the basis for a full classification of unique-root pairs.

Lemma 4.46. *Let $\Sigma = \{a\}$ be an alphabet with only a single letter. Let $U \subseteq \mathbb{N}$, and denote $k = \min(U)$. Then (Σ, U) is a unique-root pair if and only if $k|m$ for all $m \in U$.*

Proof. If $k|m$ for all $m \in U$, then any sequence a^n, $n \in \mathbb{N}$ has a unique root

$$a^n \underset{U}{\overset{*}{\Longleftarrow}} a^{k+(n \bmod k)},$$

where in the expression above $n \bmod k$ denotes the unique integer from $\{1, 2, \dots, k\}$ with the same residue modulo k as n.

In the other direction, if there exists $m \in U$ such that $k \nmid m$, let us consider the sequence a^{k+2m}. By first deduplicating a length m sequence, and then via as many deduplications of length k as needed, we obtain

$$a^{k+2m} \underset{U}{\Longleftarrow} a^{k+m} \underset{U}{\overset{*}{\Longleftarrow}} a^{k+(m \bmod k)} = x.$$

However, by only deduplicating length k sequences, we also get

$$a^{k+2m} \underset{U}{\overset{*}{\Longleftarrow}} a^{k+(2m \bmod k)} = y.$$

Both x and y are irreducible since $1 \leqslant |x|, |y| \leqslant k$. However, since $m \not\equiv 0 \pmod{k}$, we have

$$m \not\equiv 2m \pmod{k},$$

and therefore $x \neq y$, and a^{k+2m} has two distinct roots. ∎

Lemma 4.47. *Let Σ be an alphabet, $|\Sigma| \geqslant 2$, $km > 1$, and $U = \{k, k+m\} \cup V$, where $V \subseteq \mathbb{N} \setminus \{1, 2, \dots, k+m\}$. Then (Σ, U) is a non-unique-root pair.*

Proof. By Lemma 4.46 and monotonicity, if $k \nmid m$, then (Σ, U) is already a non-unique-root pair, and we are done. Thus, for the rest of the proof we assume $m = \ell k$, for some $\ell \in \mathbb{N}$.

Let $a, b \in \Sigma$ be two distinct letters, and let $v_1 v_2 \dots v_{k+m} \in \Sigma^{k+m}$ be a sequence defined as follows:

$$v_i = \begin{cases} a & i < k+m \text{ and } \lceil i/k \rceil \text{ is odd,} \\ b & i < k+m \text{ and } \lceil i/k \rceil \text{ is even,} \\ v_m & i = k+m. \end{cases}$$

Consider now the sequence

$$z = v_1 v_2 \dots v_{k+m} v_1 v_2 \dots v_{k+m} v_{m+1} \dots v_{k+m-1}.$$

We can write z as

$$\begin{aligned} z &= (v_1 v_2 \dots v_{k+m-1} v_m)^2 v_{m+1} \dots v_{k+m-1} \\ &= v_1 v_2 \dots v_{k+m-1} v_m v_1 v_2 \dots v_{m-1} (v_m v_{m+1} \dots v_{k+m-1})^2. \end{aligned}$$

As is evident, there are two squares in z, one of which is of length $2k + 2m$ and the other is of length $2k$. Deduplicating the square of length $2k + 2m$ in z first gives

$$\begin{aligned} z &\underset{U}{\Longleftarrow} v_1 v_2 \dots v_{k+m-1} v_m v_{m+1} \dots v_{k+m-1} \\ &\underset{U}{\Longleftarrow} v_1 v_2 \dots v_{k+m-1} = y. \end{aligned}$$

Deduplicating the square of length $2k$ first gives

$$z \underset{U}{\Longleftarrow} v_1 v_2 \dots v_{k+m-1} v_m v_1 v_2 \dots v_{k+m-1} = x.$$

We note that $|x| = 2k + 2m - 1$ and $|y| = k + m - 1$. Thus, if further deduplications are possible, they must be deduplications of length k, since both x and y are too short to allow deduplications of other allowed lengths from U. We observe that y is certainly irreducible, since it is made up of alternating blocks of a's and b's of length k. However, it is conceivable that x may be further deduplicated to obtain y.

We recall $m = \ell k$. Depending on the parity of ℓ, we have two cases. If ℓ is even, we can write explicitly

$$\begin{aligned} y &= (a^k b^k)^{\ell/2} a^{k-1}, \\ x &= (a^k b^k)^{\ell/2} a^{k-1} b (a^k b^k)^{\ell/2} a^{k-1}. \end{aligned}$$

The sequence x may be further deduplicated, by noting the square $ba^{k-1}ba^{k-1}$, to obtain

$$x \underset{U}{\Longleftarrow} (a^k b^k)^{\ell} a^{k-1} = x'.$$

We easily observe that x' is irreducible, and $x' \neq y$ since their lengths differ, $|y| = (\ell + 1)k - 1$, $|x| = (2\ell + 1)k - 1$, and $\ell \geqslant 1$.

If ℓ is odd, we explicitly write

$$\begin{aligned} y &= (a^k b^k)^{(\ell-1)/2} a^k b^{k-1}, \\ x &= (a^k b^k)^{(\ell-1)/2} a^k b^{k-1} a (a^k b^k)^{(\ell-1)/2} a^k b^{k-1}. \end{aligned}$$

We recall our requirement that $km > 1$, which translates to $k \geqslant 1$, $\ell \geqslant 1$ and odd, but not $k = \ell = 1$. If $k \neq 1$ and $\ell \neq 1$, we easily see that x is irreducible, $x \neq y$. If $\ell = 1$ and $k \neq 1$, we have $x = a^k b^{k-1} a^{k+1} b^{k-1}$ which is again irreducible, and $x \neq y$. The final case is $k = 1$ and $\ell \neq 1$, in which

$$x = (ab)^{(\ell-1)/2} a^2 (ab)^{(\ell-1)/2} a \overset{*}{\underset{U}{\Longleftarrow}} (ab)^{\ell-1} a = x'$$

by twice deduplicating the square a^2. However, $y = (ab)^{(\ell-1)/2} a$, and $y \neq x$ since $|y| = 1 + (\ell - 1)/2$ and $|x| = \ell$, while $\ell \geqslant 3$. ∎

Lemma 4.48. *For any alphabet Σ, $|\Sigma| \geqslant 3$, and for any $V \subseteq \mathbb{N} \setminus \{1, 2, 3\}$, $V \neq \emptyset$, if $U = \{1, 2\} \cup V$, then (Σ, U) is a non-unique-root pair.*

Proof. Let $a, b, c \in \Sigma$ be distinct symbols, and let $m = \min(V)$. Consider the sequence

$$z = ab^{m-3}caab^{m-3}ca.$$

We now have the following two distinct roots,

$$z \underset{U}{\Longleftarrow} ab^{m-3}ca \overset{*}{\underset{U}{\Longleftarrow}} abca,$$
$$z \underset{U}{\Longleftarrow} ab^{m-3}cab^{m-3}ca \overset{*}{\underset{U}{\Longleftarrow}} abcabca.$$

∎

Lemma 4.49. *For any alphabet Σ, $|\Sigma| \geqslant 3$, and for any $V \subseteq \mathbb{N} \setminus \{1, 2, 3\}$, $V \neq \emptyset$, if $U = \{1, 2, 3\} \cup V$, then (Σ, U) is a non-unique-root pair.*

Proof. Let $a, b, c \in \Sigma$ be 3 distinct symbols. Consider the sequence

$$z = ab^{m-3}cbab^{m-3}cbc,$$

where $m = \min(V)$. We now have the following two distinct roots,

$$z \underset{U}{\Longleftarrow} ab^{m-3}cbc \overset{*}{\underset{U}{\Longleftarrow}} abcbc \underset{U}{\Longleftarrow} abc,$$
$$z \underset{U}{\Longleftarrow} ab^{m-3}cbab^{m-3}c \overset{*}{\underset{U}{\Longleftarrow}} abcbabc.$$

∎

We are now in a position to provide a full classification of unique-root pairs.

Theorem 4.50. *Let $\Sigma \neq \emptyset$ be an alphabet, and $U \subseteq \mathbb{N}$, $U \neq \emptyset$, a set of tandem-duplication lengths. Denote $k = \min(U)$. Then (Σ, U) is a unique-root pair if and only if it matches one of the following cases:*

$\lvert\Sigma\rvert = 1$	$U \subseteq k\mathbb{N}$
$2*\lvert\Sigma\rvert = 2$	$U = \{k\}$
	$U \supseteq \{1, 2\}$
$3*\lvert\Sigma\rvert \geqslant 3$	$U = \{k\}$
	$U = \{1, 2\}$
	$U = \{1, 2, 3\}$

Proof. The case of $|\Sigma| = 1$ is given by Lemma 4.46. The case of $|U| = 1$ was proved in [89], and with an alternative proof, in Section 4.3. The case of $|\Sigma| = 2$ and $\{1, 2\} \nsubseteq U$, was proved in Lemma 4.47. It is also folklore that having $|\Sigma| = 2$ and $\{1, 2\} \subseteq U$ gives a unique-root pair, since we can always deduplicate runs of symbols to single letters, and then deduplicate pairs, to obtain one of only six possible roots: a, b, ab, ba, aba, bab. The choice of root depends only on the first letter of the word, its last letter, and when they're the same, on the existence of a different letter inside. No deduplication actions change those, regardless of the length of the deduplication.

When $|\Sigma| \geqslant 3$, the unique-root property for $U = \{1, 2\}$ and $U = \{1, 2, 3\}$ was established in Corollary 4.27 and Theorem 4.26, respectively. The non-unique-root property for the other cases was proved in Lemma 4.47, Lemma 4.48, and Lemma 4.49. ∎

4.7 Conclusion

We provided error-correcting codes, and in some cases, exact information-theoretic capacity, for the tandem-duplication channel. These codes mostly rely on unique-root pairs of alphabets and duplication lengths, which we also investigated.

An additional source of errors we considered in this chapter is due to a mismatch between the channel parameters and the error-correcting code we employ. We focused on the the bounded tandem-duplication system $S_{\leqslant k}(s)$, and studied the effects of sending codewords from a code designed for $S_{\leqslant k}(s)$ through a channel that uses $S_{\leqslant k+1}(s)$.

Several interesting questions remain open. In particular, we do not know yet how to construct general $(n, M; *)_U$ over Σ, especially when we do not necessarily have unique roots. We also mention the interesting combinatorial problem of counting the number of distinct roots of a string, and finding strings of a given length with as many roots as possible.

Chapter 5

STRING DUPLICATION SYSTEMS WITH SUBSTITUTIONS

5.1 Introduction

Error-correcting codes for errors caused by tandem duplications were studied in the previous chapter. In particular, an optimal family of codes for correcting errors due to tandem duplications of a fixed length and any number of errors was presented. We also studied codes for correcting tandem duplications of length up to a given constant k, where we primarily focused on the cases of $k = 2, 3$.

While the capacity and expressiveness of tandem-duplication systems have been studied in [13] and Chapter 2, these mutations do not occur in isolation and other types of mutations, such as point mutations, are typically also present. In this chapter, we study the capacity of noisy tandem duplication where both tandem duplications and point mutations occur. We show in particular that if the number of point mutations is small compared to the number of duplications with a fixed length, the capacity is 0. However, a linear number of point mutations results in a nonzero capacity.

The rest of the chapter is organized as follows. Section 5.2 presents the required definitions and notations. In Section 5.3, we study the capacity of noisy tandem-duplication systems and related problems. We present conclusions and open problems in Section 5.4.

5.2 Preliminaries

Given a string $x \in \Sigma^*$, a *tandem duplication of length* k is a process by which a contiguous substring of x of length k is copied next to itself. More precisely, we define the tandem-duplication rules, $T_{i,k} : \Sigma^* \to \Sigma^*$, for all $k \in \mathbb{N}$, $i \in \mathbb{N}_0$, as

$$T_{i,k}(x) \triangleq \begin{cases} uvvw & \text{if } x = uvw,\ |u| = i,\ |v| = k \\ x & \text{otherwise.} \end{cases}$$

We note that the "otherwise" case applies exactly when $|x| < k + i$, and therefore x cannot be decomposed into a prefix u of length i, an inner part v of length k, and some suffix w. A specific set of duplication rules that would be of interest to us throughout the chapter is

$$\mathcal{T}_k \triangleq \left\{ T_{i,k} \,\middle|\, i \geqslant 0 \right\}.$$

Another operation of interest is that of point mutation. This operation overwrites a symbol in a string with another symbol (perhaps the same, in which case, no change happens). More precisely, we define the point-mutation rules, $P_{i,a} : \Sigma^* \to \Sigma^*$, for all $i \in \mathbb{N}_0$, $a \in \Sigma$, as

$$P_{i,a}(x) \triangleq \begin{cases} uaw & \text{if } x = ubw, |u| = i, b \in \Sigma, \\ x & \text{otherwise,} \end{cases}$$

and then define

$$\mathcal{P} \triangleq \left\{ P_{i,a} \mid i \geqslant 0, a \in \Sigma \right\}.$$

Given $x, y \in \Sigma^*$, if there exist $f \in \mathcal{T}_k \cup \mathcal{P}$ such that $y = f(x)$, then we say y is a direct descendant of x. If a sequence of $t+p$ operations $f_1, \dots, f_{t+p} \in \mathcal{T}_k \cup \mathcal{P}$, exactly p of which are point mutations from $\mathcal{P}$ and the rest are t k-tandem duplications from $\mathcal{T}_k$, such that $y = f_{t+p}(\dots(f_1(x))\dots)$, then we say y is an $(t+p)$-descendant of x and denote it by $x \underset{k}{\overset{t,p}{\Longrightarrow}} y$. Finally, if there exists a finite sequence of transformations from $\mathcal{T}_k \cup \mathcal{P}$, transforming x into y, we say y is a descendant of x and denote it by $x \underset{k}{\overset{*,*}{\Longrightarrow}} y$. We note that x is its own descendant via an empty sequence of transformations.

We define the *descendant cone* of $x \in \Sigma^*$ as

$$D_k^{*,*}(x) \triangleq \left\{ y \in \Sigma^* \;\middle|\; x \underset{k}{\overset{*,*}{\Longrightarrow}} y \right\}.$$

In a similar fashion we define the *(t,p)-descendant cone* $D_k^{t,p}(x)$ by replacing $\underset{k}{\overset{*,*}{\Longrightarrow}}$ with $\underset{k}{\overset{t,p}{\Longrightarrow}}$ in the definition of $D_k^{*,*}(x)$. Additionally, whenever $p = 0$, i.e., no point mutations are involved, we use the simpler notation of $\underset{k}{\overset{*}{\Longrightarrow}}$, $\underset{k}{\overset{t}{\Longrightarrow}}$, D_k^*, and D_k^t.

We are now ready to define the *noisy tandem-duplication system*, denoted $S_k^p(s)$ over the alphabet Σ, for all tandem-duplication length $k \in \mathbb{N}$, amount of point-mutation $p : \mathbb{N} \to \mathbb{N}_0$, and initial seed string $s \in \Sigma^*$,

$$S_k^p(s) \triangleq \bigcup_{t \geqslant 0} D_k^{t,p(t)}(s),$$

i.e., it is the set of all the descendants of s obtained by using t transformations, of which $p(t)$ are point mutations. Using this notation, the tandem-duplication system studied in [13, 20] is simply S_k^0.

An important figure of merit associated with any string system $S \subseteq \Sigma^*$ is its capacity, which intuitively measures the average information content in a symbol from a string in S. Assuming $|\Sigma| = q$, the *capacity* of $S \subseteq \Sigma^*$ is defined by

$$\mathsf{cap}(S) \triangleq \limsup_{n\to\infty} \frac{1}{n} \log_q |S \cap \Sigma^n|.$$

Another property of interest to us is expressiveness. We say a string-duplication system with a fixed seed $s \in \Sigma^*$ is *fully expressive* if for every $v \in \Sigma^*$ there exist $u, w \in \Sigma^*$ such that

$$s \overset{*,*}{\underset{k}{\Longrightarrow}} uvw.$$

Namely, every given sequences v appears as a subsequence of some sequence derived from s.

Some strings cannot be derived from other strings. Formally, a string $s \in \Sigma^*$ is said to be *irreducible* with respect to $\mathcal{T}_k$ if there is no $s' \in \Sigma^*$, $s' \neq s$, such that $s' \overset{*}{\underset{k}{\Longrightarrow}} s$. We emphasize the fact that no point mutations are considered when defining irreducible strings. The set of all irreducible strings is denoted by Irr_k, and the set of all irreducible strings of length n is denoted by $\mathrm{Irr}_k(n)$.

Finally, we also consider the set of tandem-duplication rules,

$$\mathcal{T}_{\leqslant k} \triangleq \left\{ T_{i,k'} \,\middle|\, i \geqslant 0, k \geqslant k' \right\}.$$

Replacing $\mathcal{T}_k$ by $\mathcal{T}_{\leqslant k}$ in all the definitions above gives us another string-duplication system of interest. All notation remains the same except k is replaced by $\leqslant k$ whenever appropriate.

5.3 Capacity and Expressiveness in S_k^p

The study of k-tandem duplication as a source of noise, using only duplication rules from $\mathcal{T}_k$, was described in [13, 20]. In this section we consider a mix of k-tandem duplication together with point mutation, as a model for channel noise. In particular, we are interested in the capacity of error patterns, and the expressiveness of such a system.

Let us assume throughout this section that $\Sigma = \mathbb{Z}_q$. This does not constrain us in any way, and provides a structure we can use. An important tool in analyzing k-tandem-duplication systems is the transform domain defined by ϕ_k, which was described in [20]. We define the mapping in a slightly different form, which keeps its essence but simplifies notation. Define $\phi_k : \mathbb{Z}_q^{\geqslant k} \to \mathbb{Z}_q^{\geqslant k}$ by,

$$\phi_k(x) \triangleq x0^k - 0^k x,$$

where subtraction is performed entry-wise over $\mathbb{Z}_q$. We comment that to obtain the original definition of ϕ_k of [20] we delete the last k symbols and separate the sequence into its k-prefix and $|x| - k$ suffix.

Another mapping defined in [20] injects k consecutive zeros into a string. We adjust the definition of $\zeta_{i,k}$ to match the change in ϕ_k. We therefore define $\zeta_{i,k} : \mathbb{Z}_q^{\geqslant k} \to \mathbb{Z}_q^{\geqslant k}$ by

$$\zeta_{i,k}(uv) \triangleq u0^k v,$$

where $u \in \Sigma^i$, $v \in \Sigma^*$.

The following lemma was proved in [20].

Lemma 5.1. *For every string $x \in \mathbb{Z}_q^{\geqslant k}$, $0 \leqslant i \leqslant |x|$,*

$$\phi_k(T_{i,k}(x)) = \zeta_{i,k}(\phi_k(x)).$$

Intuitively, tandem-duplication operations of length k in the original domain appear as injections of 0^k in the transform domain. Thus, during many of the proofs it will be easier for us to consider strings in the transform domain, and only at the end use the reverse transform to obtain the result in the original domain.

One easily observes that the ϕ_k transform is linear, i.e., for all $x, x' \in \mathbb{Z}_q^{\geqslant k}$, $|x| = |x'|$,

$$\phi_k(x + x') = \phi_k(x) + \phi_k(x').$$

The same also holds for $\zeta_{i,k}$.

It was shown in [13] that $\mathsf{cap}(S_k^0) = 0$, regardless of the size of the alphabet Σ, and the starting string $s \in \Sigma^*$. We now show this changes when noisy duplication is present.

Theorem 5.2. *For any finite alphabet $\Sigma = \mathbb{Z}_q$, a seed string $s \in \Sigma^{\geqslant k}$, a tandem-duplication length $k \in \mathbb{N}$, and amount of point mutations $p : \mathbb{N} \to \mathbb{N}_0$, if $p(t) = o(t)$, then*

$$\mathsf{cap}(S_k^p(s)) = 0.$$

Proof. Let e_i denote a vector of all zeros except for the ith position which is 1, and whose length is implicit and understood from the context. The basis for the proof is the observation that in the transform domain, is a sequence with two non-zero elements: a 1 in the ith position, and a -1 in the $(k + i)$th position. Additionally, the elements of $\phi_k(e_i)$ sum to 0 over $\mathbb{Z}_q$.

Linearity of the transform ϕ_k guarantees that a single symbol change in the ith position of a sequence x, becomes

$$\phi_k(x + a \cdot e_i) = \phi_k(x) + a \cdot \phi_k(e_i),$$

for any $a \in \mathbb{Z}_q$.

Assume $s \overset{t,p(t)}{\underset{k}{\Longrightarrow}} x$, where we have

$$n \triangleq |x| = |s| + tk.$$

Further assume that in the derivation of x from s, the jth k-tandem duplication used is $T_{i_j,k}$. Additionally, the jth point mutation affects the ℓ_j coordinate by increasing it by a_j. Moreover, the jth point mutation occurs before the m_jth tandem duplications occurred. It now follows that

$$\begin{aligned}\phi_k(x) = \zeta_{i_t,k}(\zeta_{i_{t-1},k}(\dots\zeta_{i_1,k}(\phi_k(s))\dots)) \\ + \sum_{j=1}^{p(t)} a_j \cdot \zeta_{i_t,k}(\zeta_{i_{t-1},k}(\dots\zeta_{m_j,k}(\phi_k(e_{\ell_j}))\dots)).\end{aligned}$$

Hence, each derivation may be decomposed as a sum of a “noiseless” k-tandem-duplication process

$$\zeta_{i_t,k}(\zeta_{i_{t-1},k}(\dots\zeta_{i_1,k}(\phi_k(s))\dots))$$

and a “noise” component due to point mutations

$$\sum_{j=1}^{p(t)} a_j \cdot \zeta_{i_t,k}(\zeta_{i_{t-1},k}(\dots\zeta_{m_j,k}(\phi_k(e_{\ell_j}))\dots)).$$

The latter is a vector in the transform domain of length $n + k$ that has at most $2p(t)$ non-zero entries. The number of such noise vectors is upper bounded by

$$\Phi^{n+k}_{2p(t)} \triangleq \sum_{j=0}^{2p(t)} \binom{n+k}{j}(q-1)^j.$$

which is the size of a ball in the Hamming metric over $\mathbb{Z}_q^{n+k}$ and radius $2p(t)$.

We now have

$$\left|D_k^{t,p}(s) \cap \mathbb{Z}_q^n\right| \leqslant \left|D_k^{t,0}(s) \cap \mathbb{Z}_q^n\right| \cdot \Phi^{n+k}_{2p(t)}. \tag{5.1}$$

By [13, Th. 12],

$$\left|D_k^{t,0}(s) \cap \mathbb{Z}_q^n\right| \leqslant \binom{|s| - k + t}{|s| - 1} = q^{n \cdot o(1)}.$$

Additionally, when $p(t) = o(t)$ we have (see [71])

$$\Phi^{n+k}_{2p(t)} = q^{n \cdot o(1)}.$$

Combining these together we obtain

$$\mathsf{cap}(S^p_k(s)) = 0.$$

∎

Theorem 5.3. *For any finite alphabet $\Sigma = \mathbb{Z}_q$, a seed string $s \in \Sigma^{\geqslant k}$, a tandem-duplication length $k \in \mathbb{N}$, and amount of point mutations $p : \mathbb{N} \longrightarrow \mathbb{N}_0$, if $p(t) = \alpha t k$, $\alpha \geqslant 0$, then*

$$\mathsf{cap}(S^p_k(s)) \leqslant \begin{cases} H_q(2\alpha) & 0 \leqslant \alpha \leqslant \frac{1}{2}\left(1 - \frac{1}{q}\right), \\ 1 & \alpha > \frac{1}{2}\left(1 - \frac{1}{q}\right), \end{cases}$$

and

$$\mathsf{cap}(S^p_k(s)) \geqslant \begin{cases} H_q(\alpha) & 0 \leqslant \alpha \leqslant 1 - \frac{1}{q}, \\ 1 & \alpha > 1 - \frac{1}{q}, \end{cases}$$

where $H_q(\alpha)$ denotes the q-ary entropy function,

$$H_q(\alpha) \triangleq \alpha \log_q(q-1) - \alpha \log_q \alpha - (1-\alpha)\log_q(1-\alpha).$$

Proof. For the upper bound we use (5.1) again. Let $s \underset{k}{\overset{t,p(t)}{\Longrightarrow}} x$, and $n \triangleq |x| = |s| + tk$. However now, when $p(t) = \alpha t k$, we have (see [71])

$$\Phi^{n+k}_{2p(t)} = \begin{cases} q^{nH_q(2\alpha)(1+o(1))} & 0 \leqslant \alpha \leqslant \frac{1}{2}\left(1 - \frac{1}{q}\right), \\ q^{n(1+o(1))} & \alpha > \frac{1}{2}\left(1 - \frac{1}{q}\right). \end{cases}$$

For the lower bound, fix a single noiseless derivation, $s \underset{k}{\overset{t,0}{\Longrightarrow}} x$, denoting $n \triangleq |x| = |s| + tk$. In the original domain, the number of (distinct) sequences obtainable from x by changing at most $p(t)$ positions is exactly $\Phi^n_{p(t)}$. Thus,

$$\left|D^{t,p(t)}_k(s)\right| \geqslant \Phi^n_{p(t)} = \begin{cases} q^{nH_q(\alpha)(1+o(1))} & 0 \leqslant \alpha \leqslant 1 - \frac{1}{q}, \\ q^{n(1+o(1))} & \alpha > 1 - \frac{1}{q}. \end{cases}$$

∎

The lower bound of Theorem 5.3 may be improved by carefully constructing more strings. This is shown in the following.

Lemma 5.4. *For any finite alphabet $\Sigma = \mathbb{Z}_q$, a seed string $s \in \mathbb{Z}_q^{\geqslant k}$, a tandem-duplication length $k \in \mathbb{N}$, amount of point mutations $p :\in \mathbb{N} \to \mathbb{N}_0$, and any real constant $\beta \in [1, k]$, if $p(t) = \alpha t k$, $\alpha \in [0, \frac{\beta}{2k}]$, then*

$$\begin{aligned}\mathsf{cap}(S_k^p(s)) \geqslant{} & \frac{1}{k} H_2\left(\frac{2\alpha k}{\beta}\right)\log_q 2 + \frac{2\alpha}{\beta}\log_q 2 \\ & + \frac{\alpha}{\beta} H_2(\beta - \lfloor\beta\rfloor)\log_q 2 \\ & + \frac{\alpha}{\beta}\log_q\left(\binom{k}{\lfloor\beta\rfloor}(q-1)^{\lfloor\beta\rfloor} - 1\right) \\ & + \frac{\alpha}{\beta}(\beta - \lfloor\beta\rfloor)\log_q \frac{\binom{k}{\lfloor\beta\rfloor+1}(q-1)^{\lfloor\beta\rfloor+1} - 1}{\binom{k}{\lfloor\beta\rfloor}(q-1)^{\lfloor\beta\rfloor} - 1}.\end{aligned}$$

Proof. Our proof strategy relies on our ability to generate many distinct strings from the seed string s. We shall only use the last k symbols of s, and therefore we may assume w.l.o.g. that $|s| = k$. The choice of symbols of s will be immaterial. We shall additionally apply only tandem duplication operations $T_{i,k}$ where $k|i$. Thus, we may think of the seed string, as well as any intermediary string as a string comprised of a concatenation of blocks of length k, where each tandem-duplication operates on these blocks only. After t tandem-duplication operations we shall obtain a string of length $(t+1)k$, i.e., a string of $t+1$ blocks.

In addition, we have a budget of $\alpha t k$ point mutations. Whenever we use point mutations, we shall do so immediately after a block is tandem duplicated, mutating only symbols in the newly created block.

We encode each derivation sequence using a string made up of block identifiers of the form X_i, and delimiters from the set $\{(,), .\}$ (a left and right parentheses and a dot), with exactly one delimiter between adjacent block identifiers. Thus,

$$X_1.X_2(X_3(X_4)X_5.X_6)X_7 \tag{5.2}$$

is such a possible string. The string should have balanced parentheses, i.e., in every prefix of the string the number of left parentheses is at least the number of right parentheses. The encoding matches the derivation sequence using the following rules:

1. $X_i.X_{i+1}$ means the block X_{i+1} was tandem duplicated from X_i without any point mutations.

$$\begin{tikzcd}
X_1 \arrow[r] & X_2 \arrow[rrrrr] \arrow[rd, dotted] & & & & & X_7 \\
& & X_3 \arrow[rr] \arrow[rd, dotted] & & X_5 \arrow[r] & X_6 & \\
& & & X_4 & & &
\end{tikzcd}$$

Figure 5.1: The derivation process of (5.2). Solid arrows represent tandem duplication. Dashed arrows represent tandem duplication with at least one point mutation. The process proceeds from top to bottom, and left to right.

2. $X_i(X_{i+1}$ means the block X_{i+1} was tandem duplicated from X_i with at least one point mutation in X_{i+1}.

3. $X_{i_1}(\xi)X_{i_2}$, where ξ is a balanced string, means that X_{i_2} was tandem duplicated from X_{i_1} without any point mutations.

4. Whenever we have $X_{i_1}(\xi X_{i_2}(X_{i_3}$, where ξ is a balanced string, then $X_{i_1} \neq X_{i_3}$.

The derivation proceeds from the outer parentheses nesting level to the innermost, and within each nesting level, from left to right. Thus, the string of (5.2) uniquely describes a derivation depicted in Figure 5.1. The requirements also imply $X_1 = X_2 = X_7$, $X_3 \neq X_2$, $X_4 \neq X_3$, $X_4 \neq X_2$, and $X_3 = X_5 = X_6$.

We first note that two derivations with the same encoding string but different point mutations obviously create distinct final sequences. Moreover, we contend that two derivations with distinct encoding strings result in distinct final sequences. To show the latter we prove by simple induction that given the values of the blocks X_i, we can uniquely find the missing delimiters. The induction is on the index i. The base case is obvious. Assume we correctly found the missing delimiters given $X_1 \dots X_i$ and we are now given X_{i+1}. If $X_{i+1} = X_i$, then necessarily we have $X_i.X_{i+1}$. Otherwise $X_{i+1} \neq X_i$ and we distinguish between two cases depending on the largest $j < i$ such that $X_j(\xi X_i$ where ξ is a balanced string. If no such j exists, i.e., X_i is at the outermost nesting level of parentheses, then we must have $X_i(X_{i+1}$. If $X_{i+1} = X_j$, then by the third and fourth requirements we must have $X_i)X_{i+1}$. Otherwise, we have $X_i(X_{i+1}$.

At this point we introduce the parameter β. We shall consider only derivations whose encoding strings contain exactly $\frac{\alpha t k}{\beta}$ pairs of parentheses. Since each left parenthesis implies at least one point mutation, we shall distribute our budget of $\alpha t k$ point mutations as evenly as possible between the left parentheses. Thus, in

$\frac{\alpha t k}{\beta}(1-\beta+\lfloor\beta\rfloor)$ of the left parentheses we shall use $\lfloor\beta\rfloor$ point mutations, and in the remaining $\frac{\alpha t k}{\beta}(\beta-\lfloor\beta\rfloor)$ left parentheses we shall use $\lfloor\beta\rfloor+1$ point mutations. Indeed, the two sum to the total budget,

$$\frac{\alpha t k}{\beta}(1-\beta+\lfloor\beta\rfloor)\lfloor\beta\rfloor+\frac{\alpha t k}{\beta}(\beta-\lfloor\beta\rfloor)(\lfloor\beta\rfloor+1)=\alpha t k.$$

We are now ready to count the number of sequences resulting from the process that was described above. We have an encoding string with tk delimiters, of which $2\alpha t k/\beta$ positions hold $\alpha t k/\beta$ pairs of parentheses. We have

$$\binom{t}{\frac{2\alpha t k}{\beta}}=2^{tH_2(2\alpha k/\beta)(1+o(1))}$$

ways of choosing these positions. These $2\alpha t k/\beta$ positions are to be filled with a balanced sequence of parentheses, which may be done in

$$C_{\alpha t k/\beta}\triangleq\frac{1}{\frac{\alpha t k}{\beta}+1}\binom{\frac{2\alpha t k}{\beta}}{\frac{\alpha t k}{\beta}}=2^{\frac{2\alpha t k}{\beta}(1+o(1))},$$

where C_i denotes the ith Catalan number. Next, of the $\alpha t k/\beta$ left parentheses we need to choose $\alpha t k/\beta\cdot(\beta-\lfloor\beta\rfloor)$ positions to have $\lfloor\beta\rfloor+1$ point mutations each. This may be done in

$$\binom{\frac{\alpha t k}{\beta}}{\frac{\alpha t k}{\beta}(\beta-\lfloor\beta\rfloor)}=2^{\frac{\alpha t k}{\beta}H_2(\beta-\lfloor\beta\rfloor)(1+o(1))}$$

distinct ways. Now, given a single tandem duplication with $\lfloor\beta\rfloor$ point mutations, we have

$$\binom{k}{\lfloor\beta\rfloor}(q-1)^{\lfloor\beta\rfloor}-1$$

ways to choose the point mutations (choosing $\lfloor\beta\rfloor$ positions from the k positions in the block, and choosing for each point mutation a mutated value different from the current one). We note that we subtract one since we would like to eliminate an option that contradicts the fourth requirement. A similar expression is derived for blocks with $\lfloor\beta\rfloor+1$ point mutations.

Combining all of the expressions above and substituting them in the expression for the capacity we get the desired lower bound. ∎

An example showing the improved lower bounds is given in Figure 5.2.

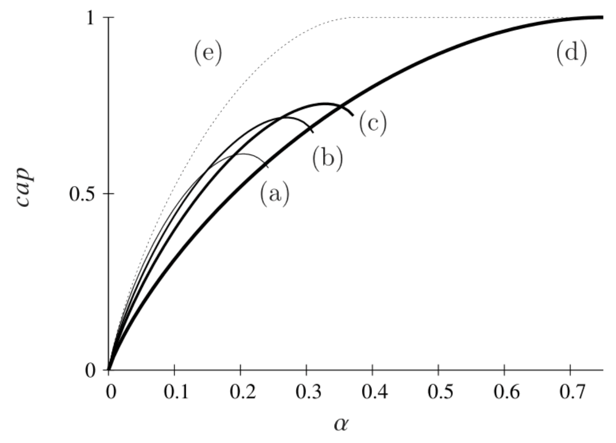


Figure 5.2: Bounds on $\mathsf{cap}(S_k^p(s))$ with $\Sigma = \mathbb{Z}_4$, $k = 2$, and $p(t) = \alpha kt$. The improved lower bound of Lemma 5.4 with (a) $\beta = 1$, (b) $\beta = 1.25$, (c) $\beta = 1.5$, as well as (d) the lower bound of Theorem 5.3 and (e) the upper bound of Theorem 5.3.

Theorem 5.5. *For any finite alphabet $\Sigma = \mathbb{Z}_q$, a seed string $s \in \Sigma^*$, a tandem-duplication length $k \in \mathbb{N}$, and amount of point mutations $p : \mathbb{N} \longrightarrow \mathbb{N}_0$, the system $S_k^p(s)$ is fully expressive if and only if $p(t) = \omega(1)$.*

Proof. We are given that $p(t) = \omega(1)$, i.e., we have an unbounded budget of point mutations. Then, for any $v \in \mathbb{Z}_q^*$, there exists $t \in \mathbb{N}$ such that $\min(p(t), |s|+tk) \geqslant |v|$. Consider now the following derivation: first employ t tandem duplications $T_{0,k}$ starting from s, resulting in a string x whose length satisfies $|x| \geqslant |v|$. Now apply $|v| \leqslant p(t)$ point mutations to the first $|v|$ positions of x, so that they are mutated into v. Thus, $s \overset{t,p(t)}{\underset{k}{\Longrightarrow}} vw$, for some $w \in \mathbb{Z}_q^*$, and the system is fully expressive.

In the other direction, suppose $p(t) = O(1)$. Then for any $s \overset{t,p(t)}{\underset{k}{\Longrightarrow}} v$ we have that $\phi_k(v)$ contains at most $2(|s| + p(t)) < M$ non-zero elements, where M is some constant. In particular, we have no descendant of s whose ϕ_k-transform has at least M non-zero elements. ∎

We point out some interesting observations. First we note that $S_k^p(s)$ is the first natural string-duplication system that is fully expressive, yet without full capacity (cf. [13, 16]). Second, note that as long as $p(t) = o(t)$ and $p(t) = \omega(1)$, we have a natural fully expressive system with capacity 0.

5.4 Discussion

The capacity of the tandem-duplication system without point mutations, $\mathsf{cap}(S_k^0(s))$, was proven to be 0 in [13]. This capacity determines the exponential growth rate of descendant cones, when using only tandem-duplication operations. When building an error-correcting code to protect against tandem duplications, these descendant cones take on the role of error spheres, and an error-correcting code is therefore a packing of these spheres. Even though their capacity (without point mutations) is 0, the channel capacity (determined by the size of the optimal code) is not full, as was shown in Chapter 4, Section 4.3.

As shown in this chapter, in the presence of point mutations, the growth rate of the tandem-duplication descendant cones is positive, $\mathsf{cap}(S_k^p(s))$, as long as the fraction of point mutations does not vanish. Thus, we may expect the channel capacity in the model with point mutations and tandem duplications, to drop, and perhaps vanish.

Chapter 6

INVERSION SYMMETRY BY TANDEM DUPLICATIONS

Erwin Chargaff in 1950 made an experimental observation that the count of A is equal to the count of T and the count of C is equal to the count of G in DNA [97, 98]. This observation played a crucial role in realizing the base pair grouping in DNA as discovered by Watson and Crick [99] in their double helix structure.

A similar symmetry was observed when in a long enough *single* DNA strand [40], the count of A is *almost* equal to the count to the count of T and the count of C is *almost* equal to the count of G. This symmetry was termed as 2nd Chargaff rule. This rule was verified globally for all eukaryotic chromosomes [100] as well as archael and bacterial chromosomes. However it does not hold in mitochondria, plasmids, single stranded DNA and RNA viruses.

Not only does the 2nd Chargaff rule hold for mononucleotides, but it also holds for k-mers (substrings of length k) upto length 7-8 for bacterial genomes and length 10 in human genome. There have been several chapters in the past that verified this symmetry for different values of k for more than 700 different species [42, 43, 45, 101, 102]. Given a genome of length n, the k-limit or the value of k upto which the 2nd Chargaff rule holds was empirically observed to be about $0.7 \ln n$ [41]. For human genome, the k-limit value that results from this approximation is 10. The 2nd Chargaff rule is termed as inversion symmetry (IS) in [41].

However, not being derived from any compelling principle, the existence of 2nd Chargaff rule (henceforth inversion symmetry (IS)) still remains a mystery. The presence of IS makes it plausible that most of the species share common dynamics of evolution. In [41, 43], the authors also showed that this symmetry only holds for reverse complement pairs and not for complement or any random pair of k-mers. [43] also argued that IS may be due to whole genome or segmental inverse duplications. Duplication based sequence generating models have been analysed in the past from a combinatorial [9–12, 16, 63] and probabilistic [69, 103] perspective. However, none of these duplication models analysed *reversed complement* tandem duplications. In this chapter, we investigate a mathematical model for sequence generation that is based on reverse complement tandem duplications. We show that the sequences generated by this model satisfy IS after sufficiently many generations and find estimates for the number of generations required to achieve IS for different duplication lengths.

The reverse complement of a sequence $s = s_1 s_2 \cdots s_m$ is given by $s^* = s_m^c s_{m-1}^c \cdots s_2^c s_1^c$, where s_i^c denotes the complement of symbol s_i. DNA consists of 4 nucleotides or symbols A, C, G and T, where $A^c = T$, $T^c = A$, $G^c = C$ and $C^c = G$.

Example 6.1. The reverse complement of $s = GTCCAGGT$ is given by $s^* = ACCTGGAC$.

In our model, we start from a *seed* string v and iteratively perform reverse complement tandem duplications at random positions inside v. The following example illustrates reverse complement tandem duplications:

Example 6.2. Consider a seed $v = AGTTGGCA$, an instance of generating new strings by reverse complement tandem duplication process on v is

$$Generation\ 1\ :\ v = AG\mathbf{TTG}GCA \longrightarrow v' = AG\mathbf{TTG}\underline{CAA}GCA.$$

$$Generation\ 2\ :\ v' = \mathbf{AG}TTGCAAGCA \longrightarrow v'' = \mathbf{AG}\underline{CT}TTGCAAGCA.$$

In generation 1, we choose a 3-length substring of v highlighted in bold and replicate its reverse complement in tandem highlighted by an underline to give v'. In generation 2, we choose a 2-length substring of v' and replicate its reverse complement to give v''. In generation 1 and generation 2 the replication or duplication length is 3 and 2 respectively.

In this chapter, we show that the reverse complement tandem duplication string system, described above in Example 6.2, generates strings that satisfy the 2nd Chargaff Rule or Inversion symmetry (IS) after a certain number of generations. The number of generations that are needed to attain inversion symmetry are dependent on the length of substrings that are replicated in reverse complement manner. For example, a single generation with a reverse complement tandem duplication of the entire seed is enough to satisfy the 2nd Chargaff rule (see Lemma 6.4). A quantity R_X^k to measure IS is defined in [46], which is based on averaging the absolute difference between the frequency of a k-mer and its reverse complement, and has been used extensively in the literature in the past to experimentally verify the 2nd Chargaff rule for different genomes. In Figure 6, we show that the value of R_X^k computed on sequences generated by our reverse complement tandem duplication model for a suitable choice of duplication length is in consistence with the value observed in ChrX, Chr14, Chr17, Chr21 in human genome.

In Section 6.1, we provide insights as to why IS arises as a result of reverse complement tandem duplications. In Section 6.2, we formally describe our model and explain the boundary/edge effects that arise in the reverse complement tandem duplication model. We further derive upper bounds on IS disruption that is caused by the boundary effects. In Section 6.3, we analyse our model and calculate the number of generations needed to create IS for some choices of duplication lengths. In Section 6.4, we show consistence in the R_X^k values for the sequences obtained by our model to those that are observed in different chromosomes in human genome for k-mer lengths $\leqslant 10$. In Section 6.5 we conclude the chapter, providing directions for future work.

6.1 Motivation for the Model

For any sequence Y, appending its reverse complement Y^* to itself can easily be shown to attain IS for all k-mers upto length $2|YY^*|$ (see Lemma 6.4).

Definition 6.3. The complement of $a \in \{A, C, G, T\}$ is denoted by a^c, where $A^c = T, G^c = C, C^c = G, T^c = A$. The reverse complement of $Z \in \{A, C, G, T\}^m$ is denoted by Z^*, i.e., if

$$Z = Z_1 Z_2 \cdots Z_m, \text{ then}$$
$$Z^* = Z_m^c Z_{m-1}^c \cdots Z_2^c Z_1^c.$$

Let u be any k-mer in Z. Let $N_Z(u)$ be the number of occurrences of u in Z, and note that

$$N_Z(u) = N_{Z^*}(u^*). \tag{6.1}$$

In the following lemma, let $Z \triangleq YY^*$ for some $Y \in \{A, C, G, T\}^n$.

Lemma 6.4. *For any k-mer u with $|u| \leqslant 2n$ in Z, $N_Z(u) = N_Z(u^*)$.*

Proof. For any k-mer u in Z,

$$N_Z(u) = N_Y(u) + N_{Y^*}(u) + B(u),$$
$$N_Z(u^*) = N_Y(u^*) + N_{Y^*}(u^*) + B(u^*),$$

$B(u)$ and $B(u^*)$ denote the number of times u and u^* occur at the boundary of Y and Y^* in Z, respectively. Note that from Eq. (6.1), $N_Y(u) = N_{Y^*}(u^*)$ and $N_{Y^*}(u) = N_Y(u^*)$, therefore in order to show $N_Z(u) = N_Z(u^*)$, we need to show

$$B(u) = B(u^*). \tag{6.2}$$

In order to show (6.2), we show for every occurrence of u on the boundary, there also exists an occurrence of u^*. Let

$$u = Y_{\max\{n-l+1,1\}} \cdots Y_n {Y_n}^c \cdots Y^c_{\max\{1,n-m+1\}}$$

where $l > 0$, $m > 0$ and $\min\{l, n\} + \min\{m, n\} = k$, then

$$u^* = Y_{\max\{1,n-m+1\}} \cdots Y_n {Y_n}^c \cdots Y^c_{\max\{1,n-l+1\}}.$$

∎

It is easy to check that inversion symmetry is not guaranteed in the same way as described in Lemma 6.4, if $Z = YY^c$ or $Z = YY$.

Lemma 6.4 readily implies that the special case of a reverse complement tandem duplication of length n induces IS within 1 generation. Hence, this hints that IS, which is prevalent in many genomes, might be the result of such duplications. Since a reverse complement tandem duplication of length n is unlikely, a natural question to study in this regard is how short can reverse complement tandem duplications be in order to attain IS within a reasonable number of generations. Various aspects of this question are studied in the remainder of this chapter.

6.2 Boundary effect

For a sequence X and an integer k, the quantity

$$R_X^k = \frac{\frac{1}{2}\sum_{s\in\{A,C,G,T\}^k} |N_X(s) - N_X(s^*)|}{|X| - k + 1} \tag{6.3}$$

was defined in [46] as a means to estimate IS in X. It was also shown in [46] that R_X^k is monotone w.r.t k, i.e. $R_X^k \leqslant R_X^{k+1}$.

Now consider our reverse complement tandem duplication model. Let $v = xyz$, where $|y| = d$ and $|x|, |z| \geqslant 0$. Replicating y in a reversed complement tandem fashion results in $v_{new} = xyy^*z$. Let $y = y_1 y_2 \cdots y_d$, and $u = y_l y_{l+1} \cdots y_{l+k-1}$ be a k- length substring of y. In Lemma 6.4, we found that for every u in yy^*, $N_{yy^*}(u) = N_{yy^*}(u^*)$. In addition to that, due to the presence of z in v and v_{new}, we observe the following boundary effects:

1. Boundary Effect 1: Any k-mer that appears on the boundary of y and z in v can get lost in the creation of v_{new}, i.e., k-mers of the form $y_{d-i+1} \cdots z_j$, where $i + j = k$ and $i, j \geqslant 1$ may not exist in v_{new}. For example,
   Example 6.5. Let $v = AGACA$ with $y = GA$ and $z = CA$, $v_{new} = AGATCCA$, the 2-mer AC exists at the boundary of y and z in v but is lost in v_{new}.

2. Boundary Effect 2: At the boundary of y^* and z in v_{new}, new k-mers are created which may not occur in v and are also not locally balanced in yy^*, i.e. k-mers of the form $y_i^c y_{i-1}^c \cdots z_1 \cdots z_j$, where $i + j = k$, and $i, j \geqslant 1$. For instance in Example 6.5, CC is a new 2-mer that is created at the boundary of y^* and z in v_{new}. Note that TC and AT are the other newly created 2-mers in v_{new} but they lie entirely in $yy^* = GATC$. They are locally balanced in yy^* by their reverse complements GA and AT respectively.

Definition 6.6. We define a recursive process of generating strings by reverse complement tandem duplication as follows:

- Seed: $v_0 = x_0 y_0 z_0$
- Replication operation $(\mathcal{T}_{R_c})$: $\mathcal{T}_{R_c}(v_i) = v_{i+1} = x_i y_i y_i^* z_i, |x_i|, |z_i| \geqslant 0$ and $|y_i| = d_i,\, d_i > 0, \forall i \geqslant 0$.
- $\mathcal{T}_{R_c}^{m}(v_j) = \mathcal{T}_{R_c}^{m-1}(v_{j+1})\ \forall m > 0$ and $\mathcal{T}_{R_c}^{0}(v_j) = v_j$.

Note that y_i is chosen uniformly at random, and let $X = \mathcal{T}_{R_c}^{g}(v_0)$, where g is the number of generations that we wish to study. Let $X = \mathcal{T}_{R_c}^{g}(v_0)$.

In order to do a cleaner analysis of R_X^k for X generated by reverse complement tandem duplication system $\mathcal{T}_{R_c}$ above, we wish to upper bound the Boundary effect 1 and Boundary effect 2. We do so by computing $\Delta_{(k,g)}$, which measures the worst case impact of Boundary effect 1 and Boundary effect 2 on R_X^k after g generations.

In each operation $\mathcal{T}_{R_c}$, we lose $k-1$ k-mers due to Boundary Effect 1 and gain $k-1$ k-mers due to Boundary Effect 2. Therefore, in each generation the numerator of R_X^k has a worst case change of $2(k-1)$. Hence, after g operations the worst case effect on the numerator of R_X^k due to boundary effects is $2(k-1)g$. Also note that $|X| = |v_0| + \sum_{i=0}^{g-1} d_i$, and hence $\Delta_{(k,g)}$ is given by

$$\Delta_{(k,g)} = \frac{2(k-1)g}{|v_0| + \sum_{i=0}^{g-1} d_i - k + 1}. \tag{6.4}$$

Therefore after g generations, R_X^k is given by $|R_X^k - Q_X^k| \leqslant \Delta_{(k,g)}$, where Q_X^k is the approximation of R_X^k by ignoring the boundary effects 1 and 2 after g generations.

It is notable here that in $\mathcal{T}_{R_c}$ (Definition 6.6), if any string y_i of length d_i is duplicated to create $y_i y_i^*$, then by Lemma 6.4, it balances the occurence of any k-mer for all $k \leqslant 2d_i$ in $y_i y_i^*$. If every symbol in v_0 is chosen at some stage in the generation process as a substring of y_i, then once all the symbols in v_0 have been chosen, Q_X^k will be 0 for all $k \leqslant 2 \min \{d_i\}_{i=1}^{g}$. In the next section, we find for a given $\epsilon > 0$, number of generations g that are needed to obtain $Q_X^k \leqslant \epsilon$ for different choices of duplication lengths; and consequently, the number of generations g which is required to obtain $R_X^k \leqslant \Delta_{(k,g)} + \epsilon$ for a given k. Intuitively one can expect lesser number of generations for a higher value of duplication length. We define these ideas more formally in the following section.

6.3 Results and Discussions

Balanced and Unbalanced Segments

Definition 6.7. Consider the string replication system $\mathcal{T}_{R_c}$ given in Definition 6.6. A symbol $a \in v_i$ is called *balanced* if it belonged to some y_j or y_j^* for $j < i$, and otherwise it is *unbalanced*.

Note that all symbols in $Z = YY^*$ given in Lemma 6.4 are balanced.

Definition 6.8. Let u be a substring of v_i. Let a and b be the symbols preceeding and succeeding u in v_i respectively. u is called a balanced segment of v_i if all the symbols in u are balanced and a, b are unbalanced.

Let u be a substring of v_i. Let a and b be the symbols preceeding and succeeding u in v_i respectively. u is called an unbalanced segment of v_i if all the symbols in u are unbalanced and a, b are balanced.

Note that in the case where u is a prefix or suffix of v_i, we ignore symbol a and b in Definition 6.8 accordingly. Also note that all the symbols in v_0 are unbalanced, hence v_0 is an unbalanced segment. We will now investigate the generation of balanced segments in v_i for $i \geqslant 1$ for the string replication system $\mathcal{T}_{R_c}$ described in Definition 6.6.

Generation of Balanced segments

In every operation $\mathcal{T}_{R_c}(v_i)$ for $i \geqslant 0$, either a new balanced segment is added or some previously existing balanced segment(s) is modified. The operation $\mathcal{T}_{R_c}(v_i)$

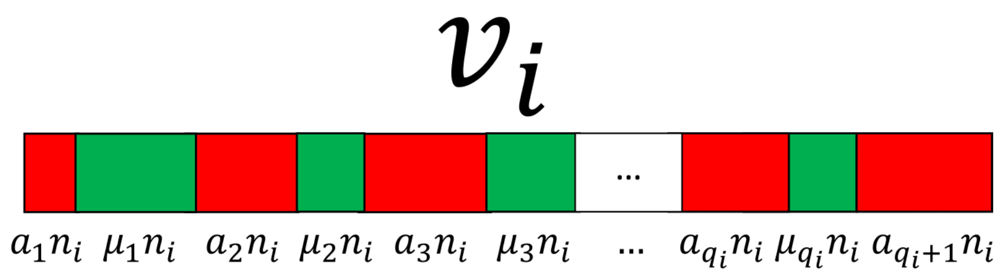

v_i
...
a_1n_i μ_1n_i a_2n_i μ_2n_i a_3n_i μ_3n_i ... $a_{q_i}n_i$ $\mu_{q_i}n_i$ $a_{q_i+1}n_i$

Figure 6.1: Red and green segments represent unbalanced and balanced segments in v_i respectively. Each $a_jn_i(1 \leqslant j \leqslant q_i + 1)$ represents a unbalanced segment. Each $\mu_jn_i(1 \leqslant j \leqslant q_i)$ represents a balanced segment. Note $q_i \leqslant i$.

uniformly and randomly chooses a substring y_i of length d_i in v_i and replicates it to give $v_{i+1} = x_iy_iy_i^*z_i$. The addition and modification of balanced segments is described below.

1. *Addition:* If all the symbols in y_i are unbalanced and the symbols before and after y_i are both unbalanced in v_i, then $y_iy_i^*$ is *added* as a new balanced segment in v_{i+1}, thereby increasing the number of balanced segments in v_{i+1} by 1.

2. *Modification:* If some of the symbols in y_i are balanced or y_i is preceeded/succeeded by a balanced symbol in v_i, then $y_iy_i^*$ *modifies* previously created balanced segment(s), thereby not increasing the count of balanced segments in v_{i+1}.

Addition and modification operations are described in Figure 2 and Figure 3 respectively. It is clear from the description of *addition* and *modification* above, that v_i has at most i balanced segments.

Figure 1 shows *balanced* (green) and *unbalanced* (red) segments in v_i. Here μ_j and a_j represent the fraction of v_i covered by the j-th balanced and unbalanced segments respectively. Therefore,

$$n_i = \sum_{j=1}^{q_i} \mu_jn_i + \sum_{j=1}^{q_i+1} a_jn_i \tag{6.5}$$

where q_i is the number of balanced segments in v_i. Let N_i denote the total length of

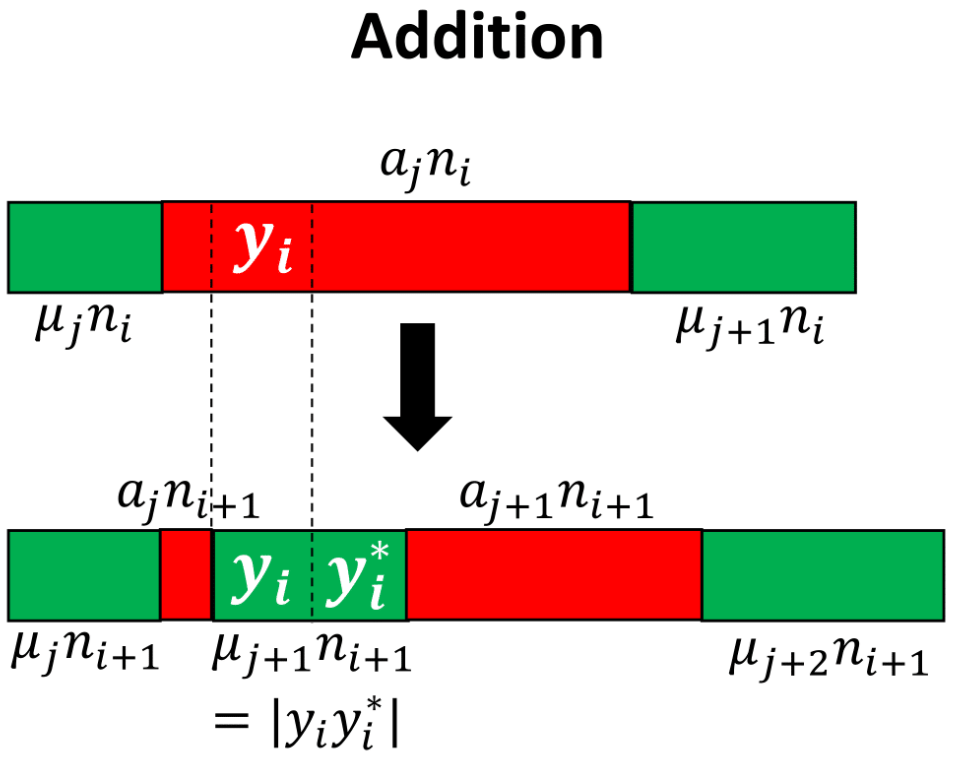


Figure 6.2: Addition Operation: All the symbols of y_i are unbalanced in v_i and the symbol before and after y_i are also unbalanced in v_i. $y_i y_i^*$ is added as a new balanced segment in v_{i+1}. Note that the count of balanced segments is 1 more than the count of balanced segments in v_i.

balanced segments in v_i, i.e. $N_i = \sum_{j=1}^{q_i} \mu_j n_i$. We also note that

$$n_i - N_i = \sum_{j=1}^{q_i+1} a_j n_i. \tag{6.6}$$

Let X_i and Y_i denote the total length of unbalanced and balanced segments in y_i, respectively, and note that N_i, X_i and Y_i are random variables that satisfy

$$\begin{aligned} X_i + Y_i &= |y_i| = d_i \\ N_{i+1} &= N_i + 2X_i + Y_i \\ &= N_i + X_i + d_i. \end{aligned}$$

In turn, this readily implies that

$$E[N_{i+1}|N_i, a_1, a_2, \cdots, a_{q_i+1}] \quad = \quad N_i \ + \ d_i \ + \ E[X_i|N_i, a_1, a_2, \cdots a_{q_i+1}]. \tag{6.7}$$

We compute $E[X_i|N_i, a_1, a_2, \cdots a_{q_i+1}]$ for $d_i = d$ for all i and some $d > 0$.

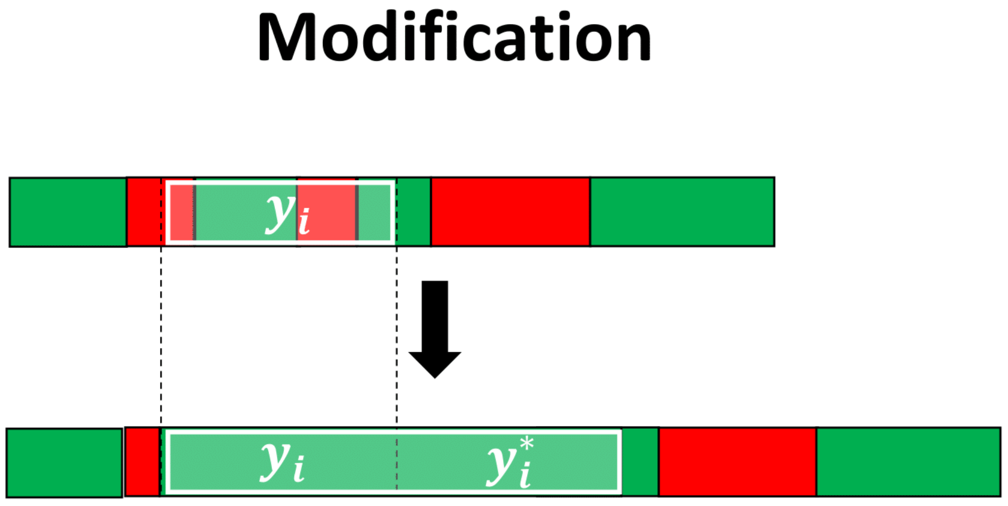


Figure 6.3: Modification Operation: Some symbols of y_i are balanced in v_i. Hence $y_i y_i^*$ in v_{i+1} is a modification of the previously created balanced segments in v_i. Note that the count of balanced segments in v_{i+1} has not increased from v_i. In fact in this particular instance, it has decreased.

Lemma 6.9. *Let $d_i = d$. Then $\forall\ i \geqslant 0$, the length of each balanced segment in v_i is at least $2d$.*

Proof. Observe that for any i, any *newly added* balanced segment in v_i, i.e., one that was generated by the most recent application of $\mathcal{T}_{R_c}$, is of length at least $2d$. ∎

We derive $E[X_i|N_i, a_1, a_2, \cdots a_{q_i+1}]$ by using

$$E[X_i|N_i, a_1, a_2, \cdots a_{q_i+1}] = \Sigma_{l=1}^{d} P(X_i \geqslant l|N_i, a_1, a_2, \cdots a_{q_i+1}). \quad (6.8)$$

Lemma 6.10. *For $d_i = d\ \forall i$, y_i can overlap with at most 2 balanced segments in v_i.*

Proof. From Lemma 6.9, the length of each balanced segment in v_i is at least $2d$. We have the following 4 cases:

- Case 1: y_i does not overlap with any balanced segment in v_i.
- Case 2: either some prefix of y_i overlaps with a suffix of a balanced segment j for some j in v_i or some suffix of y_i overlaps with a prefix of a balanced segment j for some j in v_i but not both.

- Case 3: y_i is a substring of some balanced segment j.
- Case 4: some prefix of y_i overlaps with a suffix of a balanced segment j and some suffix of y_i overlaps with a prefix of next balanced segment after j in v_i for some j.

In case 2 and 3 above, y_i overlaps with 1 balanced segment and in case 4 y_i overlaps with 2 balanced segments. ∎

We derive $E[X_i|N_i, a_1, a_2, \cdots a_{q_i+1}]$ by using

$$E[X_i|N_i, a_1, a_2, \cdots a_{q_i+1}] \quad = \quad \Sigma_{l=1}^{d} P(X_i \quad \geqslant \quad l|N_i, a_1, a_2, \cdots a_{q_i+1}). \quad (6.9)$$

Using Lemma 6.10,

$$P(X_i \geqslant l|N_i, a_1, a_2, \cdots, a_{q_i+1}) = \sum_{j=2}^{q_i} I(a_j n_i \geqslant l)\frac{a_j n_i + d - 2l + 1}{n_i - d + 1} + I(a_1 n_i \geqslant l)\frac{a_1 n_i - l + 1}{n_i - d + 1} + I(a_{q_i+1} n_i \geqslant l)\frac{a_{q_i+1} n_i - l + 1}{n_i - d + 1} \quad (6.10)$$

$I(.)$ represents the indicator function.
Solving (6.9) and (6.10) and using (6.6) gives

$$E[X_i|N_i, a_1, a_2, \cdots, a_{q_i+1}] = \frac{(n_i - N_i - a_1 n_i - a_{q_i+1} n_i)d}{n_i - d + 1} + \sum_{l=1}^{\min(a_1 n_i, d)} \frac{a_1 n_i - l + 1}{n_i - d + 1} + \sum_{l=1}^{\min(a_{q_i+1} n_i, d)} \frac{a_{q_i+1} n_i - l + 1}{n_i - d + 1}.$$

To do numerical simulations for the recursive Eq. (6.7), we can omit a_1 and a_{q_i+1} by approximating $E[X_i|N_i, a_1, a_2, \cdots, a_{q_i+1}]$. We do this by stitching the end of v_i with its start, thereby making v_i circular. Let X_i' denote the number of unbalanced symbols chosen in this circular version of v_i. In turn, $P(X_i' \geqslant l|N_i, a_1, a_2, \cdots, a_{q_i+1})$ is given by

$$P(X_i' \geqslant l|N_i, a_1, a_2, \cdots, a_{q_i+1}) = \sum_{j=2}^{q_i} I(a_j n_i \geqslant l)\frac{a_j n_i + d - 2l + 1}{n_i} + I((a_1 + a_{q_i+1})n_i \geqslant l)\frac{(a_1 + a_{q_i+1})n_i + d - 2l + 1}{n_i} \quad (6.11)$$

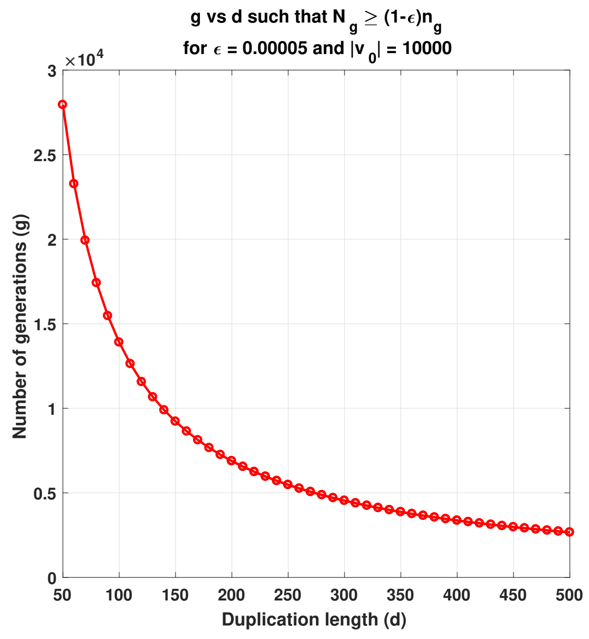


Figure 6.4: Number of generations (g) needed such that $\frac{N_g}{n_g} = (1-\epsilon)$ for $\epsilon = 0.00005$, $|v_0| = 10000$ and $d_i = d \ \forall i \geqslant 0$.

Solving (6.9) and (6.11) and using (6.6) gives

$$E[X_i'|N_i, a_1, a_2, \cdots, a_{q_i+1}] = \frac{(n_i - N_i)d}{n_i} = E[X_i'|N_i]$$

We can now use $E[X_i'|N_i, a_1, a_2, \cdots, a_{q_i+1}]$ as an approximation for $E[X_i|N_i, a_1, a_2, \cdots, a_{q_i+1}]$. Using (6.7) we get,

$$E[N_{i+1}|N_i] \approx N_i + d + \frac{(n_i - N_i)d}{n_i}. \tag{6.12}$$

Figure 4 is obtained by using $E[N_i|N_{i-1}]$ as approximation for N_i and $N_0 = 0$ in Eq. (6.12). It shows the number of generations g needed such that $\frac{N_g}{n_g} = 1 - \epsilon$ for $\epsilon = 0.00005$, $|v_0| = 10000$ for different values of d.

When $d_i = d \ \forall i$, $\Delta_{(k,g)}$ using Eq. (6.4) is given by

$$\Delta_{(k,g)} = \frac{2(k-1)g}{|v| + gd - k + 1}$$

We also see that

$$\lim_{g\to\infty} \Delta_{(k,g)} = \frac{2(k-1)}{d}. \tag{6.13}$$

Further since $\Delta_{(k,g)}$ is an increasing function in g for $|v| > k - 1$, we have for a given k and d that $\Delta_{(k,g)} \leqslant \frac{2(k-1)}{d}$, when $|v| > k - 1$. Figure 5 shows the variation of $\Delta_{(10,g)}$ with g for different values of d at $|v| = 10000$. We see from Figure 5 that

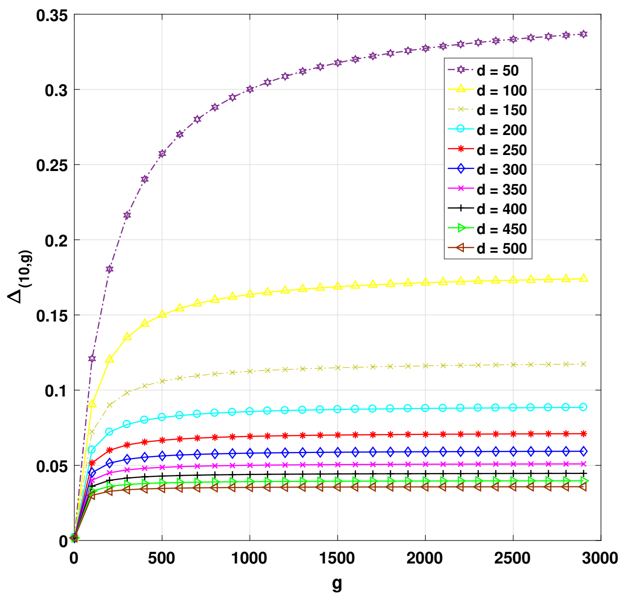


Figure 6.5: Variation of $\Delta_{(10,g)}$ vs g for d = 50, 100, 150, 200, 250, 300, 350, 400, 450, 500. and $|v| = 10000$. We can see that a larger value of d diminishes the boundary effects.

$\Delta_{(10,g)} \approx 0.09$ for $d = 200$ for $g > 1000$ generations. Therefore $|R_X^{10} - Q_X^{10}| \leqslant 0.09$ for $d = 200$ after 1000 generations. For $d = 200$, Figure 4 implies that after approximately 7000 generations we have that at least $(1 - 0.00005)$ fraction of the sequence is balanced. Further, since $R_X^{10} \leqslant Q_X^{10} + \Delta_{(10,g)}$, Figure 5 implies that $R_X^{10} \leqslant 0.09005$. Similar bounds on R_X^{10} can be derived for other values of d using Figures 2 and 3. Note that the choice of $\epsilon = 0.00005$ for generating the plot in Figure 4 is arbitrary here and similar plots can be obtained for other values of ϵ by using (6.12). Moreover, similar bounds can be obtained for other values of k by calculating $\Delta_{(k,g)}$.

Unbalanced Shorter Sequences

Inversion symmetry however is not observed in the shorter segments of the genome [41]. For example, if a segment of length 5000 is selected from our genome, it will not possess inversion symmetry upto $k = 10$. Our generative model based on reverse complement tandem duplication is also in consensus with this experimental observation. More precisely, as more and more duplication happens, the k-mers that became balanced in the creation of yy^* will pull apart in future generations if duplication happens somewhere inside yy^*. This distance between a k-mer and its reverse complement arises due to extra duplications that happen in the segments that lie in between them. As a result, the whole sequence remains balanced however the shorter segments inside it become unbalanced. This is illustrated using the following example:

Example 6.11. Consider

$$v = GTCCG\underline{AGCACTGA}AGTCA.$$

Let y denote the underlined substring of v. u is obtained by duplicating y in v in reversed complement tandem. Therefore

$$u = GTCCG\underline{AGCACTGA}\mathbf{TCAGTGCT}AGTCA.$$

y^* is denoted by the bold portion in u. Let us now focus on the 2-mer CA and its reverse complement TG in y and y^* respectively. We note that $|y| = 8$. Below we highlight this 2-mer and its reverse complement in yy^* in u.

$$u = GTCCGAG\mathbf{CA}CTGATCAG\mathbf{TG}CTAGTCA.$$

Now if we further duplicate the underlined portion in

$$u = GTCCGAG\mathbf{CA}\underline{CTGATCAG}\mathbf{TG}CTAGTCA$$

to get

$$u' = GTCCGAG\mathbf{CA}\underline{CTGATCAGCTGATCAG}\mathbf{TG}CTAGTCA$$

We observe that in u CA and TG were apart by 8 symbols in u and by 16 symbols in u'. More such duplications in between CA and TG in the future generations will pull them further apart. Note that the duplication length chosen here cannot be more than 8 as the distance between CA and TG is 8.

We analyze the phenomenon explained in Example 6.11 above by defining Δ_0 as the initial distance between a k-mer and its reverse complement when they are created, and Δ_i as their distance after i generations. The expected behavior of Δ_i can be modeled by the equation below:

$$E[\Delta_{i+1}|\Delta_i] = \Delta_i(1 + \frac{d}{n_i - d + 1}). \qquad (6.14)$$

For $n_i \gg d$, $E[\Delta_{i+1}|\Delta_i] \approx \Delta_i(1 + \frac{d}{n_i})$. Note $n_{i+1} = n_i + d$. Therefore by approximating Δ_i with $E[\Delta_i|\Delta_{i-1}]$, we have

$$E[\Delta_m|\Delta_0] \approx \Delta_0(1 + \frac{md}{n_0}). \qquad (6.15)$$

Note here $d \leqslant \Delta_0 \leqslant 2d - k - 1$.

Example 6.12. For $k = 10$, $n_0 = 10000$ and $d = 200$. Using (6.15) above, we have $E[\Delta_m|\Delta_0] \approx \Delta_0(1 + 0.02m)$. Now using $200 \leqslant \Delta_0 \leqslant 389$, we have $200 + 4m \leqslant E[\Delta_m|\Delta_0] \leqslant 389 + 7.78m$. We see that $200 + 4m = 5000$, for $m = 1200$, which implies that after 1200 generations $E[\Delta_{1200}|\Delta_0] \geqslant 5000$. In Figure 4, we see that for $d = 200$, $\frac{N_g}{n_g} \geqslant 0.99995$ only after about 7000 generations which implies that the balanced k-mers would have been pulled further apart and will not be localized in smaller blocks of length 5000 inside the sequence.

6.4 Experimental Findings

In Figure 6, we compare the experimentally observed value of R_X^k for different sequences X and $1 \leqslant k \leqslant 10$. The sequences chosen are chromosomes $X, 14, 17, 21$ (shown by solid lines) in the human genome (Hg38), sequences generated by tandem reverse complement duplication system discussed in the chapter for $d = 20, 50, 200, 500$ (shown by dashed lines). We observe that tandem at $d = 200$ is well in consistence with ChrX and Chr14. We also observe that $d = 50$ tandem is in consistence with Chr17 and Chr21 for $k = 9, 10$. We have further added 2 more plots (shown by dotted lines) that model reverse complement duplications done in an interspersed manner. In interspersed duplication, unlike tandem the chosen substring is replicated at any location and not necessarily next to the original string. An example illustrating interspersed duplication is given below

Example 6.13. Consider a seed $v = AGTTGGCA$, an instance of generating new strings by reverse complement interspersed duplication process on v is

$$Generation\ 1 : v = AG\mathbf{TTG}GCA \rightarrow v' = AG\mathbf{TTG}GC\underline{CAA}A.$$

$$Generation\ 2 : v' = \mathbf{AG}TTGGCCAAA \rightarrow v'' = \mathbf{AG}TTG\underline{CT}GCCAAA.$$

In generation 1, we choose a 3-length substring of v highlighted in bold and replicate its reverse complement in an interspersed manner highlighted by an underline to give v'. In generation 2, we choose a 2-length substring of v' and replicate its reverse complement to give v''. In generation 1 and generation 2 the replication or duplication length is 3 and 2 respectively.

In the plot in Figure 6, we have included two plots where the sequences are generated by interspersed duplication and the duplication length is 50 and 500. The site where the reverse complement duplicate is placed is chosen uniformly and randomly in the interspersed model. We see that at $d = 500$, interspersed duplication is in

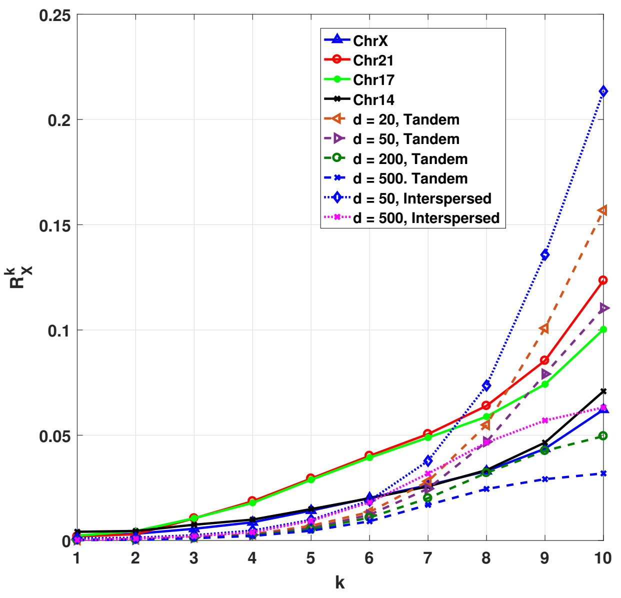


Figure 6.6: Comparison of R_X^k value calculated experimentally in chromsomes X, 14, 17 and 21 (solid lines) with those obtained by reverse complement tandem (dashed lines) and interspersed (dotted lines) duplications for different duplication lengths d.

consistence with values observed in ChrX and Chr14 for $k \leqslant 7$, whereas at $d = 50$ there seem to be no consistence with any of the suggested chromosomes.

These plots suggest that reverse complement duplications can potentially be playing a key role in the evolution of genome. We believe that the inconsistencies with chromosome data that are seen for some $k's$ can be attributed to point mutations which was not taken into account.

6.5 Conclusion

We showed that the reverse complement tandem duplication model generates sequences satisfying the $2nd$ Chargaff Rule. Moreover, even when the length of duplication is chosen to be a constant d which is very small as compared to the sequence length, this symmetry can be obtained using our model. Further, we provided estimates on the number of generations needed to create this symmetry given a choice of duplication length(s). In our analysis, we found an upper bound given in (6.4) on the disruption caused by boundary effects.

We see that the error due to boundary effect given in Eq. 6.13 for $d = 50$ and $k = 10$ is 0.36. However, we note from Figure 6, the R_X^{10} value obtained experimentally when we generate sequences from our model for $d = 50$ is 0.11. This means that the theoretical bound on R_X^k given by $R_X^k \leqslant Q_X^k + \Delta_{(k,g)}$ obtained in this chapter is loose for lower values of duplication length d ($d < 150$). For such lower values of d, a finer boundary effect analysis is needed and is deferred to future work. Another interesting question is how does the value of d affect the probability distribution of k-mers observed using this model of sequence generation and comparing it with the k-mer distribution observed in real DNA.

Chapter 7

CANCER CLASSIFICATION FROM HEALTHY DNA

7.1 Introduction

The human genome has evolved over time by an interplay of mutational events. An enhanced understanding of the genome's evolution has numerous direct and practical applications in improving healthcare, discerning ancestry, materializing DNA storage and designing synthetic biology devices for computation. Traditionally, the genome has been viewed as a *time-independent source* of information, and hence much of the genomic research has been focused on discovering variants that cause a certain phenotype. Linkage studies have discovered genes for *Mendelian* diseases such as Cystic Fibrosis [104], Huntington disease [105], Fragile-X syndrome [106] and many others [2] by investigating genetic variants across families. For more complex diseases, Genome Wide Association Studies (GWAS) [107] can be used to discover large amounts of risk factors working in conjunction. The broader scope of GWAS has led to the discovery of several new genes and pathways [3], but many diseases still remain unexplained. Instead of searching for disease-causing variants, we view the genome as a *time-dependent signal*, searching for indicators for how the genome is mutating over time. This gives rise to the following question - What are the possible ways to measure the evolution of mutations? Put differently, how can we quantify the accumulation of mutations in the genome of an individual?

Our approach for extracting time-dependent information about a person's mutation history is to focus on the tandem repeat regions of this person's *healthy* genome. We have studied two types of mutations in the genome: tandem duplications and point mutations. Tandem duplications involve the consecutive repetition of a subsequence (e.g. $T\underline{CAT}G \rightarrow T\underline{CATCAT}G$). Point mutations, which include substitutions, insertions, and deletions, are single changes in the DNA (e.g. $AC\underline{T}G \rightarrow AC\underline{A}G$). When these two processes occur in the same location, point mutations can propagate through tandem duplications, leaving a change in the repeated sequence (see Figure 7.1a and Methods section). This allows us to construct a likely history of tandem duplications and point mutations. Slippage events can cause regions with many tandem duplications [50], which are a convenient locations to observe this interaction between mutation processes. In a sense, these *tandem repeats regions* are a nature given *repetition error-detecting code* [108], where the point mutation errors in the copies store information about the history of the evolution of these regions. These repeat regions effectively characterize an *evolution channel*, which can shed light on the accumulation of mutations in the genome.

7.2 Cancer Genomics

Cancer is currently the second leading cause of death worldwide [52]. Cancer is caused by an intricate mixture of complex factors whose inter-relations are not well understood. While the roles of environmental and hereditary factors are well accepted, recent studies suggest that two-thirds of the mutations in human cancers are caused by replication errors [53].

Most GWAS studies on cancer risk have focused on differences between healthy (i.e., normal) and tumor DNA samples, namely in Single Nucleotide Polymorphisms (SNPs) and Copy Number Variations (CNVs). These studies have discovered tumor suppressor genes like BRCA1, BRCA2, TP53 and oncogenes like HER2 and RAs family [54]. Previous work has also shown that tumor genomes have significantly more genes with repeat instabilities, linking microsatellite instability to colorectal [109] and other cancers [109–114]. Another recent approach identified 21 signatures for mutational processes in human cancer using healthy genome based on 96 substitution classifications that were defined by 6 single base substitution classes and the sequence context left and right of the mutated base [115].

Unlike previous works, we aim to study cancer risk factors while using a *healthy genome*, without using the genome of the tumor itself, which opens the door to cancer prediction and risk assessment. To do this, we analyzed tandem repeat region data in different cancer types from The Cancer Genome Atlas (TCGA) [57]. We estimated the number of point mutations (m) and tandem duplications (d) in each tandem repeat region by predicting the evolutionary history of those regions [56, 116] (see Figure 7.1a). We used the aggregate of this evolution information to form what we call the *mutation profile* of the genome. We then used a gradient boosting algorithm to learn the association between these mutation profiles and the probability of developing specific cancers [58]. The association between the mutation profile and the cancer-type signifies the presence of a cancer-type “signal” in the mutation profiles of the *healthy* genome, which could be useful for future cancer prediction and early cancer detection.

7.3 Results

We hypothesized that different genetic mutation processes accounted for varying risks of developing cancer, and that these processes would leave detectable signals in an individual's profile. Rigorous verification of this hypothesis would require DNA samples from cancer patients before the onset of their cancer. Such a dataset is not currently available, but blood derived DNA is accessible on The Cancer Genome Atlas (TCGA) [57], and closely resembles the DNA of cancer patients before they developed the disease. From TCGA, we gathered 3874 *unamplified* blood derived WXS samples which spanned 12 cancers: TCGA-GBM, LUAD, LUSC, PRAD, PAAD, STAD, HNSC, BLCA, KIRC, LGG, SKCM, THCA (Table 7.1A (Column 2), Supplementary Files 1-12). We used microsatellites (tandem repeats with pattern lengths $\leqslant$ 10 bp) with at most 100 repeats to obtain mutation profiles (see Methods, Figure 7.1).

Pairwise Cancer Classifiers - Using only Blood Derived Normal Samples

Here we use 3843 unamplified blood-derived normal samples spanning 11 cancers for our analysis (see Table 7.1A, Supplementary Files 1-12). We did not use the blood derived samples from TCGA-KIRC in this analysis as we only had 31 samples for KIRC which was not enough to construct a reliable classifier. We verified the existence of cancer-type signals within the mutation profiles of blood-derived normal samples by training cancer classifiers using xgboost [58] and testing their accuracy on separate *validation-set* data (see Methods, Code/Software, Figure 7.2).

As can be seen in Figure 7.2, mutation profiles of blood-derived normal DNA of GBM patients shows strongly distinctive signals from the rest of the tested cancers with classification accuracies ranging in between 75% for HNSC to as high as 93% for SKCM and PAAD. A similar observation is made for both SKCM and PAAD as they are distinguishable from all of the other cancers with more than 71% accuracy. For other cancers - STAD, BLCA, LGG, PRAD, LUAD, THCA, LUSC, HNSC, the distinguishing signal is much weaker for many cancers. For example, LGG when compared against PRAD, BLCA, LUAD, LUSC gives pairwise accuracies of 59%, 64%, 59% and 58% respectively. Cancers with risk factors that emit different mutation profiles are easier to distinguish, resulting in more accurate classifiers. Hence, accuracy gives a notion of distance on the scale of 50% (close, indistinguishable) to 100% (far, different). The order of the cancers in the display minimizes the distances between neighboring cancers using the travelling salesman problem (TSP) [117], giving a likely low dimensional projection of the features being learned by the classifiers. The observed accuracies and specificity/sensitivity observed in Figure 7.2 confirm the presence of *cancer-type* signal in the blood-derived normal DNA.

The clustering of cancers in Figure 7.2 led us to define four cancer classes: Class 1 = [GBM], Class 2 = [SKCM], Class 3 = [PAAD] and Class 4 = [LUAD, LUSC, PRAD, STAD, HNSC, BLCA, LGG, THCA]. The seriation matrices in Figure 7.3 represent the binary classifier accuracies and sensitivity/specificity for these different classes.

Cancer Classification Profiles

To assess a patient's propensity of developing a class of cancers, we trained a multiclassifier for the four cancer classes using gradient boosting. This classifier uses a mutation profile to predict the relative probability of each class of cancer. Figure 7.4 shows the mean and standard deviation of these probabilities when tested on patients from each cancer class. Class 1, 2, and 3 all give large probabilities for their respective classes. Classes 2 and 3 give weaker signals because they are closer to Class 4 than Class 1 is (see Figure 7.3). Individuals in Class 4 in the test set have similar scores for Classes 2, 3 and 4, showing that Class 4's signal is not very distinct from Classes 2 and 3. This can again be attributed to the closeness of Class 4 to both Class 2 and Class 3 in the seriation diagram in Figure 7.3. Figure 7.9 gives the classification profile for Class 4 individuals when training only on Classes 1, 2 and 3 individuals. Again, Class 4 seems to imitate Classes 2 and 3, but the high standard deviation in Class 1 probability suggests Class 4 cancer patients can also have a high probability for Class 1 cancers.

Effect of Adding NAT samples on classifiers

Recent studies have shown positive associations of Solid Tissue Normal (Normal Adjacent to Tumor (NAT)) samples on TCGA with the tumor DNA of cancer patients [118, 119]. We added 687 unamplified NAT samples as mentioned in Table 7.1A (Column 3) in our analysis to check if their presence is useful in discovering a stronger cancer-type signal. More precisely, we combined the 3874 blood-derived and 687 NAT samples to construct the pairwise classifiers. Here, we also covered TCGA-KIRC as now we had 210 (179 NAT and 31 blood-derived) samples that were enough to build reliable classifiers. We didn't observe any significant improvement in cancer signal detection by adding NAT samples over only using blood-derived normal samples and found the same cancer classes that we discovered previously (see Figure 7.5). Further, we found that TCGA-KIRC belonged to the same class as TCGA-GBM showing strongly distinctive signal from the other 10 cancers (see Figure 7.5, Figure 7.10, Figure 7.11).

Analysis of Amplified Samples

Amplification techniques have been shown to bias tandem repeat information [120], especially in TCGA data [121]. To control for this, we separately analyzed amplified samples. Figure 7.6 shows the seriation diagrams for accuracy and sensitivity/specificity of pairwise classifiers built using 525 samples *amplified* by MDA technology. Because of the limited data, this test only covered TCGA-GBM (brain), TCGA-OV (ovary) and TCGA-LAML (leukemia) and both the normal DNA types, i.e. blood-derived and solid tissue normal (NAT) were used (Table 7.1B, Supplementary Files 13-15). The high accuracy and sensitivity/specificity values in these diagrams suggest a strong cancer-type signal in the mutation profiles of the healthy DNA. Further, we also generated the classification profiles using these amplified samples for individuals with brain, ovary and leukemia cancer. Figure 7.7 shows the mean and standard deviation of the predicted cancer probabilities for these three populations. The highest probability cancers correspond with the cancers that the patients were diagnosed with, affirming that healthy DNA contains a cancer-type signal.

Genome analysis for the results presented in Figures 7.2-7.8 and Figures 7.9-7.18 was done using samtools [122] (see Methods, Code/Software) and the pipeline presented in Figure 7.1b. We also verified these results for unamplified samples by using another genome analysis tool for short tandem repeats (STR)- hipSTR [123] that only detects tandem repeats with pattern lengths at most 6 (see Figure 7.19).

Driver Genes

Studies in the past have identified driver genes like TP53, BRCA-1, BRCA-2, etc. We considered 723 such genes that are listed in Supplementary File 16 obtained from Cancer gene census - COSMIC [124, 125].

To test whether these regions provided special information, we filtered our mutation profiles to only use tandem repeats that overlapped with driver gene regions. We conducted this experiment for the 4561 unamplified samples and 525 amplified samples separately. Figure 7.8 shows a comparison the classifiers trained on these filtered mutation profiles and mutation profiles which contain all the features *except* those in the filtered profiles. Darker cells in this figure correspond to large differences in the accuracy of the classifiers, indicating that these signals exist outside of driver gene regions. We notice these darker cells especially for TCGA-PAAD (pancreas) and TCGA-SKCM (skin). We also see noticeable differences when TCGA-OV (ovary) is compared against TCGA-GBM (brain) and TCGA-LKCM (leukemia). The driver-gene classifiers always performed worse than the classifiers trained on the rest of the genome, indicating that the signal exists both inside and outside driver gene regions.

7.4 Discussion

Early Cancer prediction

We have shown that the mutation profiles of the blood-derived normal genomes of cancer patients contain a cancer-type signal (Figures 7.2 - 7.4). It is reasonable to assume that the mutation profiles of cancer-free patients may also contain these signals, and we can use our classifier to quantify their presence. The cancer classification profiles given by this classifier could be used to screen individuals for those who may benefit from more comprehensive and expensive cancer detection tests.

Accumulation of Mutations

Searching for information-containing features within 3 billion nucleotides is a formidable task. This has traditionally been simplified by comparing individuals to extract variants, which compresses the genome into a smaller set of features to analyze. These differences, known as SNPs and CNVs, are central to both Mendelian studies [2] and GWAS [3, 54, 107].

This form of genome compression loses crucial information about how the genome is *changing* by only considering differences in the genome's *current state*. Every individual's genome passes through a distinct evolution channel that is controlled by hereditary, environmental and stochastic factors. These evolution channels differ among the population and can give rise to different risks of disease, but we cannot easily identify these differences from the single-generation SNP and CNV analysis used in GWAS. Mendelian studies may provide insight into inter-generational processes, but do so at the cost of requiring inter-generational data, which severely limits the scope of a feature search. Even with additional data, Mendelian studies still lack the ability to detect differences in mutation processes that occur throughout one's lifetime.

Mutation profiles are generated without any comparative analysis, reducing the data-demand. Instead, the tandem repeat regions in a *single* genome provide a window into its history, capturing information about the individual's evolution channel. This ability to reconstruct a genome's history from repeat regions is lost when studies only view differences between individuals. The use of mutation profiles expands our access to time-dependent traits which may be essential to understanding developed diseases like cancer.

Sequencing Technology Limitations

TCGA samples are obtained from Illumina platforms with a coverage depth 30-40X. The read lengths used are short ranging between 100-500 bp. This poses a problem in the detection of longer tandem repeats [126–128]. In our analysis, we only used repeats with pattern lengths $\leqslant$ 10 bp and number of copies not greater than 100.

7.5 Methods

WXS data

We used exome data from "blood derived normal" and "solid tissue normal" samples in the TCGA [57] database, details about which are provided in the Supplementary Files 1-15. The BAM file for each sample was aligned against hg38. All the autosomes from each sample were recovered using samtools [122].

Algorithms

Our algorithms are partitioned to Part A and Part B (see Fig. 7.1b). Part A is only performed once, where Part B is performed whenever cancer prediction is required. In Part A, a dataset of healthy DNA is first processed by the Benson [116] and Tang *et al*. [56] algorithms to deduce the mutation profiles. Then, these vectors are aligned by a dynamic programming algorithm to resolve missing regions. Finally, the aligned vectors are fed into a training algorithm to produce a classifier. In Part B, this classifier is applied over any individual's genome, to assess the overall probability to contract any of the cancer in question.

Tandem Repeat Detection and Duplication History Estimation

Tandem duplications are consecutively repeated patterns caused by replication slippage events [48, 49], in which a pattern is duplicated next to the original. For example, the following shows two tandem duplications of length 4, where the duplicated part is highlighted in bold. The underlined segment is the *microsatellite* or *repeat region*.

$$\texttt{ATGAC}\mathbf{\texttt{GTGA}}\texttt{GT} \Rightarrow \texttt{ATGAC}\underline{\texttt{GTGA}\mathbf{\texttt{GTGA}}}\texttt{GT} \Rightarrow \texttt{ATGAC}\underline{\texttt{GTGAGTGAGTGA}}\texttt{GT}. \tag{7.1}$$

The *pattern* of a region is the short strand which repeats itself. The *copy number d* of a repeat region indicates the number of times that the pattern is repeated. For example, the pattern of the underlined repeat region in the right hand side of (7.1) is GTGA, and its copy number is 3.

Microsatellites are usually accompanied by various types of errors: substitutions (replacement of one nucleotide by another), deletions (omission of a nucleotide), and insertions (addition of a nucleotide). The total number of substitutions, deletions, and insertions in a repeat region is called the *error number m*. For example, the following shows the contamination of (7.1) by 1 substitution, 1 deletion, and 1 insertion.

$$\begin{aligned}\texttt{ATGAC}\underline{\texttt{GTGAGTGAGTGA}}\texttt{GT} &\Rightarrow \texttt{ATGAC}\underline{\texttt{GT}\mathbf{\texttt{T}}\texttt{AGTGAGTGA}}\texttt{GT}\\ &\Rightarrow \texttt{ATGAC}\underline{\texttt{GTTAGGAGTGA}}\texttt{GT}\\ &\Rightarrow \texttt{ATGAC}\underline{\texttt{GTTAGGAGTGA}\mathbf{\texttt{G}}}\texttt{GT}. \qquad (7.2)\end{aligned}$$

Clearly, the copy number of (7.2) is 3 and its error number is 3, and hence its mutation index is $(m, d) = (3, 3)$. In the first step of Part A we use the Benson Tandem Repeat Finder to detect repeats with consensus pattern size at most 10 and copy number at most 100. These size limitations mean we only consider regions smaller than 1000 nucleotides. The single block version of the duplication history estimation algorithm given in Tang *et al.* [56] was then applied to each tandem repeat region to obtain the respective *mutation index* = (m, d). The aggregation of these (m, d) values gives a vector twice the size of the number of repeat regions, which we call an individual's *mutation profile*. Since, TCGA data is WXS, we only calculated a unique *mutation profile* of an individual's exome.

Alignment

Following the completion of the Benson and Tang *et al.* algorithms, it was sometimes the case that certain repeat regions appeared in some patients and did not appear in others. In addition, minor differences were observed in the patterns of identical repeat regions in different individuals. As a result, a technical difficulty arose in handling the input to the learning algorithm. Consider the following two patients, in which the repeat regions are underlined.

$$\text{Patient 1: } \underline{\texttt{AAAAAAA}}\texttt{CGATCGAGTTCAGTATTGC}\underline{\texttt{CGCGAGCG}} \overset{\substack{\text{Benson}\\ \text{Tang } \textit{et al.}}}{\Longrightarrow} (\texttt{A} : (0,7),\ \texttt{CG} : (1,4))$$

$$\text{Patient 2: } \underline{\texttt{AAAAAAAA}}\texttt{CGA}\underline{\texttt{CGTACGTACGTA}}\texttt{TTGC}\underline{\texttt{CGCGCG}} \overset{\substack{\text{Benson}\\ \text{Tang } \textit{et al.}}}{\Longrightarrow} (\texttt{A} : (0,8),\ \texttt{CGTA} : (0,3),\ \texttt{CG} : (0,3))$$

The success of machine learning depend on the detection of patterns in specific *positions* of feature vector, so entries which correspond to the same repeat region must also be placed in the same position for all inputs. This is clearly not the case in the above example, in which the second entries of the vectors correspond to different repeat regions.

This issue is resolved by using a dynamic programming alignment algorithm. In this algorithm, a similarity score is computed recursively for each possible alignment, and the alignment which leads to the best possible score is chosen. Each possible alignment is defined as the sum of normalized *edit-distances*[1] between the patterns of all respective pairs. Further, the distance between any pattern and a "missing pattern", denoted by '–' below, is defined as 0.4. Namely, two patterns whose respective normalized edit distance is less than 0.4 were considered to be equal for the sake of the alignment. For example, the vectors above are aligned in the following way.

$$\begin{array}{lcl}
(\texttt{A} : (0,7),\ \texttt{CG} : (1,4)) & \overset{\text{Alignment}}{\Longrightarrow} & (\texttt{A} : (0,7),\ \ - \ \ , \texttt{CG} : (1,4)) \\
(\texttt{A} : (0,8),\ \texttt{CGTA} : (0,3),\ \texttt{CG} : (1,3)) & \overset{\text{Alignment}}{\Longrightarrow} & (\texttt{A} : (0,8),\ \texttt{CGTA} : (0,3), \texttt{CG} : (1,3))
\end{array} \tag{7.3}$$

The score for the alignment (7.3) is $d_e(\texttt{A}, \texttt{A}) + d_e(-, \texttt{CGTA}) + d_e(\texttt{CG}, \texttt{CG}) = 0 + 0.4 + 0 = 0.4$, where d_e denotes edit distance. For comparison, the alternative alignment

$$\begin{array}{lcl}
(\texttt{A} : (0,7),\ \texttt{CG} : (1,4)) & \overset{\text{Alignment}}{\Longrightarrow} & (\texttt{A} : (0,7),\ \texttt{CG} : (1,4) \ \ , \ \ - \ \ ) \\
(\texttt{A} : (0,8),\ \texttt{CGTA} : (0,3),\ \texttt{CG} : (1,3)) & \overset{\text{Alignment}}{\Longrightarrow} & (\texttt{A} : (0,8),\ \texttt{CGTA} : (0,3), \texttt{CG} : (1,3))
\end{array} \tag{7.4}$$

[1] That is, the minimal number of insertions, deletions, and substitutions that are required to transform one pattern to the other, divided by the average length of the sequences.

has score of $d_e(\mathtt{A}, \mathtt{A}) + d_e(\mathtt{CG}, \mathtt{CGTA}) + d_e(-, \mathtt{CG}) = 0 + 2/3 + 0.4 \approx 1.06$, and hence (7.3) is preferred over (7.4).

The mutation profile of each individual was aligned against the mutation profile of the reference genome (hg38) by using the method that is mentioned above. The repeat regions that were missing in the reference genome were omitted from these aligned mutation profiles. Further, given the aligned mutation profiles, every '–' is replaced by $(0, 0)$. This gave aligned mutation profiles of the same size that can now be used as features for the learning part described next.

Machine Learning

The aligned mutation profiles were used as features for the learning algorithm. Machine learning classifiers for distinguishing cancers were obtained using two approaches:

Pairwise Classifiers

We trained a binary classifier for *every pair* of types of cancer, generating $\binom{12}{2} = 66$ pairwise classifiers for unamplified samples and $\binom{3}{2} = 3$ pairwise classifiers for amplified samples. The accuracy in either of those classifiers is used as a measure for the "uniqueness" of the mutation profiles that cause a certain type of cancer, and can additionally be seen as a distance measure between different types of cancer. We used xgboost [58] algorithm at default parameters with max-depth = 2, and performed 4-fold validation to build each of these pairwise classifiers.

Multiclassifier

This was built using xgboost 'multi:softprob' parameter with max-depth = 2 and predicted the probability of all the cancers simultaneously. Again 4-fold cross validation was performed to avoid over-fitting.

Code/Software

The code and necessary documentation for the pipeline used is available at `http://paradise.caltech.edu/~sidjain/Codes.tar.gz`.

Data Availability

The BAM files for WXS samples of cancer patients used in the study were obtain from The Cancer Genome Atlas (TCGA) [57]. These files have controlled access and cannot be availed publicly. However, request to access TCGA controlled data can be made via dbGap [129] (accession code: phs000178.v1.p1). The file names for the analyzed samples are given in Supplementary Files 1-15.

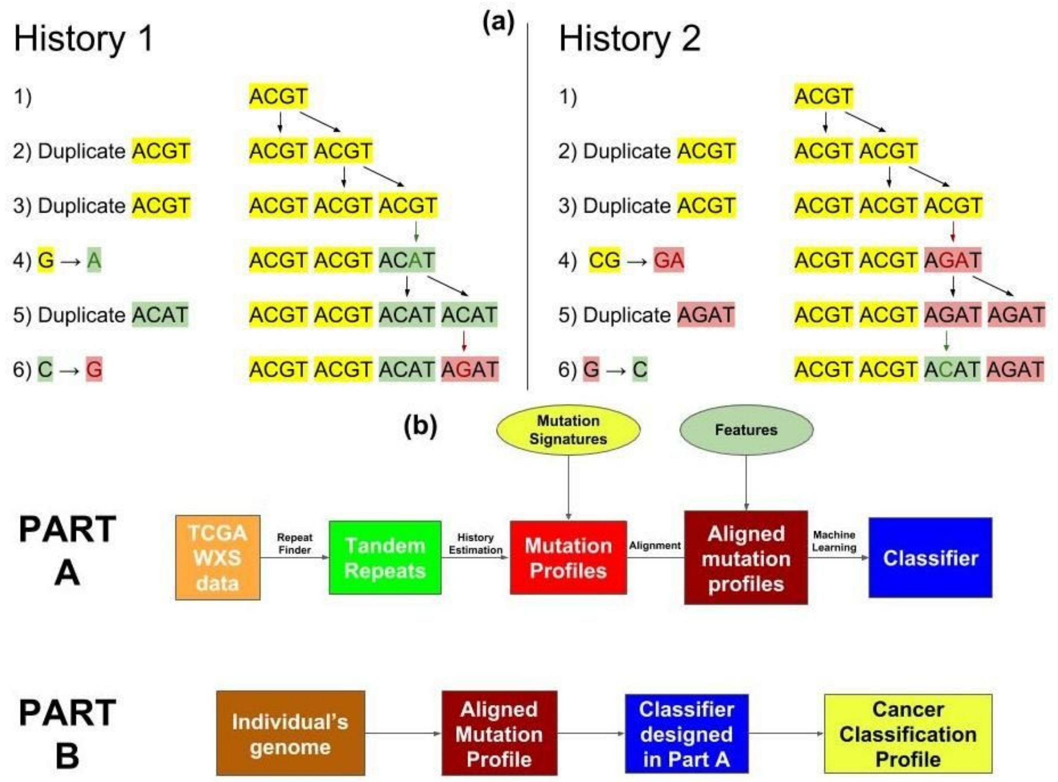

(a)
History 1
1) ACGT
2) Duplicate ACGT — ACGT ACGT
3) Duplicate ACGT — ACGT ACGT ACGT
4) G → A — ACGT ACGT ACAT
5) Duplicate ACAT — ACGT ACGT ACAT ACAT
6) C → G — ACGT ACGT ACAT AGAT
History 2
1) ACGT
2) Duplicate ACGT — ACGT ACGT
3) Duplicate ACGT — ACGT ACGT ACGT
4) CG → GA — ACGT ACGT AGAT
5) Duplicate AGAT — ACGT ACGT AGAT AGAT
6) G → C — ACGT ACGT ACAT AGAT
(b)
Mutation Signatures
Features
PART A
TCGA WXS data
Repeat Finder
Tandem Repeats
History Estimation
Mutation Profiles
Alignment
Aligned mutation profiles
Machine Learning
Classifier
PART B
Individual's genome
Aligned Mutation Profile
Classifier designed in Part A
Cancer Classification Profile

Figure 7.1: **(a)** Two different evolution histories for the tandem repeat region *ACGTACGTACATAGAT* with pattern length 4 and repeat region length 16. In History 1, only 2 point mutations were needed (marked with green and red respectively). In History 2, 3 point mutations were needed: 1 marked with green and 2 marked with red. In our approach, we would consider History 1 to be more likely as it involves lesser number of point mutations. Therefore, for this tandem repeat region we have that $m = 2$ and $d = 4$. **(b)** The workflow of our algorithm. In Part A a classifier is trained based on the mutation profiles generated from the healthy DNA of cancerous individuals. This part in only performed once per training set. In Part B, the resulting classifier is applied over a given genome to assess an individual's inclination of developing different cancers.

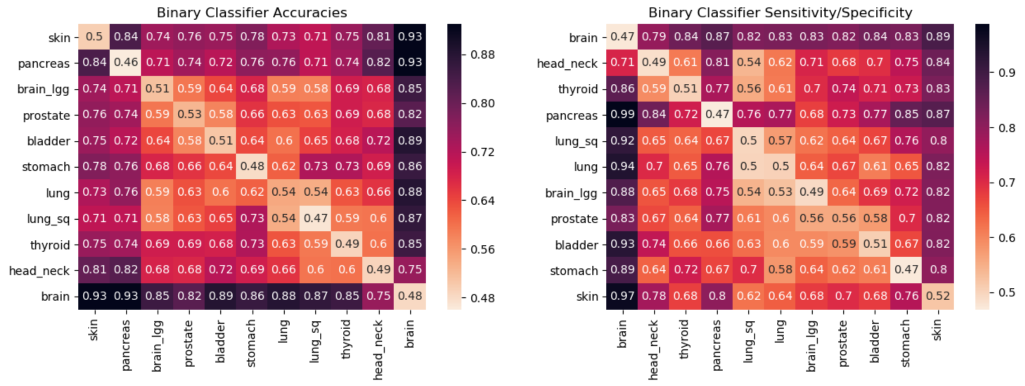


Figure 7.2: The matrix on the left contains the pairwise validation accuracies of classifiers built by using 3843 blood-derived normal DNA samples covering 11 different cancer types. These cancer types are TCGA-SKCM (skin), PAAD (pancreas), STAD (stomach), BLCA (bladder), PRAD (prostate), LGG (brain_lgg), LUAD (lung), THCA (thyroid), LUSC (lung_sq), HNSC (head_neck), GBM (brain). Each cell in the accuracy seriation matrix represents the average validation accuracy of the binary pairwise classifiers. Each pairwise classifier between cancer X and cancer Y (X≠Y) was constructed using 4-fold cross-validation with patients of each cancer type. These accuracies can be interpreted as distances. The darker the cell, farther are the cancers being compared. The darker rows corresponding to brain, skin and pancreas are indicative of the presence of cancer-type signal in the blood-derived normal (healthy) DNA of cancer patients. The diagonal entries in the seriation matrix represent the accuracies when half of the patients of cancer X were labeled 0 and half of the patients of the same cancer X were labeled 1. As one can expect, the average test accuracy for such classifier should be around 50%. The value in the cell corresponding to the row "pancreas" and the column "prostate" signifies that an average of 74% of the people were correctly classified in each validation pass. The matrix on the right, contains the sensitivity/specificity values. Each cell in the sensitivity/specificity seriation matrix represents the sensitivity value when the row cancer is considered positive and the column cancer is considered negative. It can also be regarded as specificity when the row cancer is considered negative and the column cancer is considered positive. Sensitivity is defined as $TP/(TP + FN)$ and specificity is defined as $TN/(TN + FP)$, where TP = True Positive, FP = False Positive, TN = True Negative, FN = False Negative. A value of .77 in the row "prostate" and the column "pancreas" means that 77% of the prostate patients in the test set were truly classified as prostate type (sensitivity when prostate is considered positive). A value of .73 in the row "pancreas" and the column "prostate" means that 73% of the pancreas patients in the test set were truly classified as pancreas type (specificity when prostate is considered positive). The seriation ordering is obtained by solving TSP (i.e., the Travelling Salesman Problem) exhaustively [117].

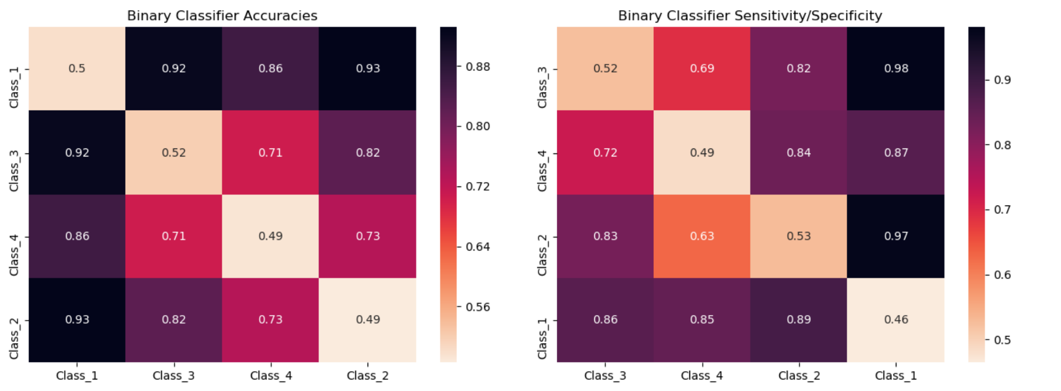


Figure 7.3: These seriation matrices show 4-fold validation accuracy and sensitivity/specificity for the four main clusters of cancers in Figure 7.2 generated using 3843 blood-derived normal samples. Class 1 = (brain), Class 2 = (skin), Class 3 = (pancreas), Class 4 = (stomach, bladder, prostate, brain_lgg, lung, thyroid, lung_sq, head_neck).

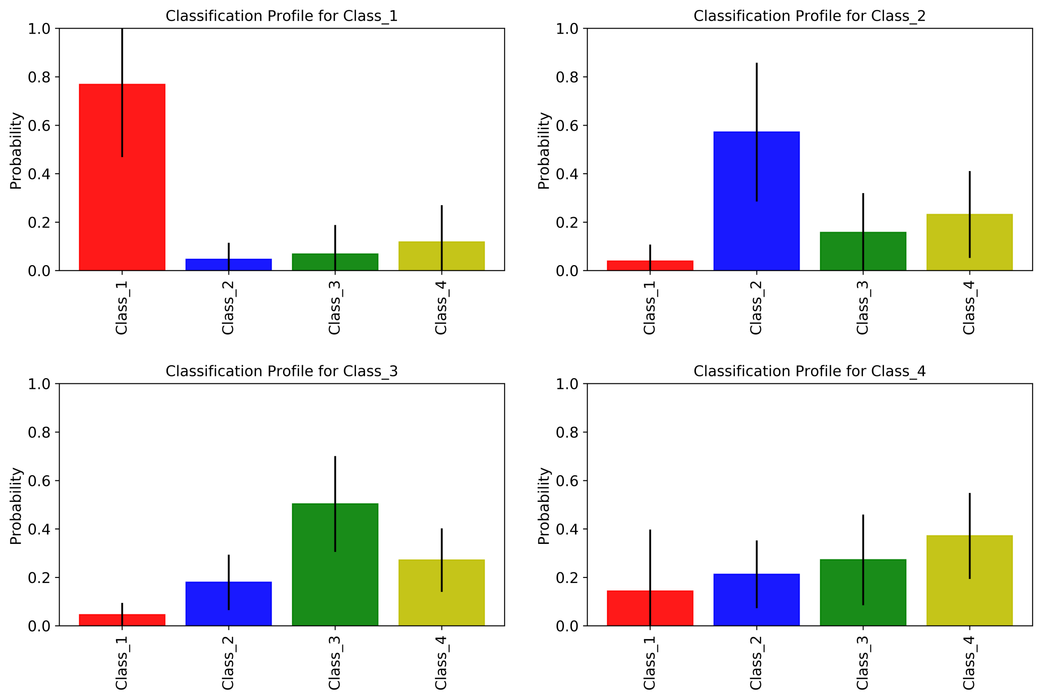


Figure 7.4: Mean and standard deviations for the cancer classification profiles of individuals in Class 1, Class 2, Class 3 and Class 4 (viewing left to right). To generate these profiles, we a trained multiclassifier on all four classes of cancers using gradient boosting. We then used this multiclassifier to obtain cancer classification profiles for a different set of individuals reported the average results for each cancer class. Class 1 individuals show a high probability for Class 1 cancers. Class 2 and Class 3 individuals also show a higher probability for their respective classes, but with a slightly weaker signal. Class 4 individuals show similar probabilities for Class 2, Class 3 and Class 4.

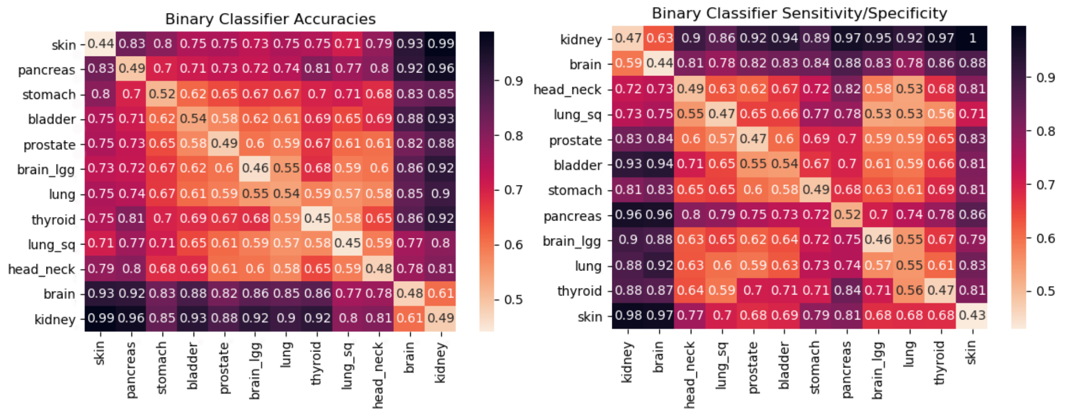


Figure 7.5: The matrix on the left contains the pairwise validation accuracies of classifiers built by using 4561 unamplified samples covering both blood-derived and solid tissue normal DNA of 12 different cancer types. These cancer types are TCGA-SKCM (skin), PAAD (pancreas), STAD (stomach), BLCA (bladder), PRAD (prostate), LGG (brain_lgg), LUAD (lung), THCA (thyroid), LUSC (lung_sq), HNSC (head_neck), GBM (brain), KIRC (kidney).

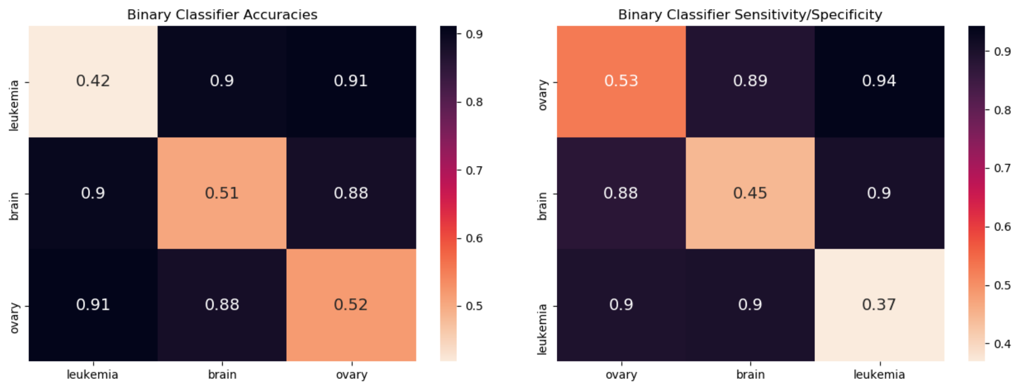


Figure 7.6: Classifier accuracies and sensitivity/specificity when training and testing is done using amplified samples only. The cancers covered here are TCGA-GBM (brain), TCGA-OV (ovary) and TCGA-LAML (leukemia). There is a strong signal distinguishing all these cancer types. We speculate that TCGA-KIRC would behave similarly to TCGA-GBM, since these cancers are similar in Figure 7.5.

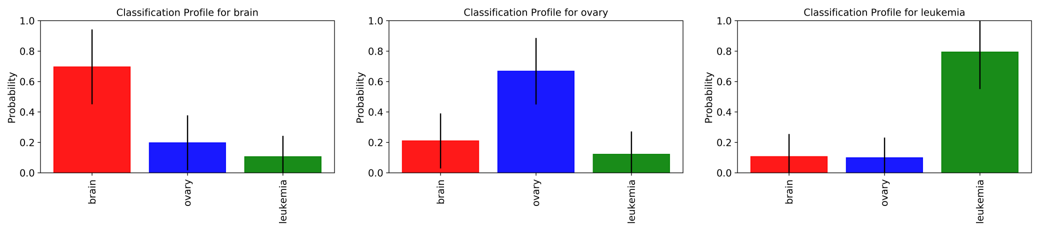


Figure 7.7: Mean and standard deviation for the cancer classification profiles of individuals belonging to TCGA-GBM (brain), TCGA-OV (ovary) and TCGA-LAML (leukemia). These classification profiles are obtained by building a multiclassifier using gradient boosting. The multiclassifier is obtained by training *only* on amplified healthy samples of brain, ovary and leukemia cancers on TCGA. The testing is done on a separate set of amplified samples for these cancers. It can be seen that brain, ovary and leukemia cancer patients in the test set are showing a higher probability for the respective cancer using their healthy DNA mutation profile.

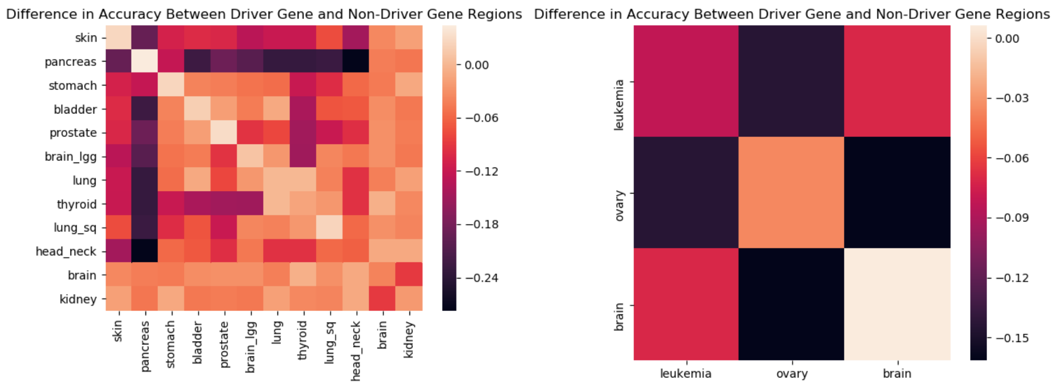


Figure 7.8: A comparison of classifiers trained on mutation profiles restricted to known driver-gene regions and those restricted to non-driver-gene regions. Darker cells represent a larger difference in the testing accuracies of the classifier, indicated by the scale. This experiment was performed on unamplified (left) and amplified (right) samples separately. The uneven coloring suggests that some cancer-type signals exist more primarily in driver-gene regions than others.

A: Unamplified Samples		
Cancer	Blood Derived Normal	Solid Tissue Normal
SKCM	344	0
PAAD	153	31
STAD	396	49
BLCA	393	20
PRAD	440	56
LGG	513	0
LUAD	411	102
THCA	432	68
LUSC	316	180
HNSC	190	0
GBM	255	2
KIRC	31	179

B: Amplified Samples		
Cancer	Blood Derived Normal	Solid Tissue Normal
GBM	171	0
LAML	0	135
OV	160	59

Table 7.1: **(A)** Number of unamplified healthy samples used for each cancer type in the study showing the number of blood derived normal and solid tissue normal samples. In total, the number of blood derived healthy samples are 3874 and the tissue derived healthy samples are 687. The sample information is provided in Supplementary Files 1-12. **(B)** Number of amplified healthy samples used for each cancer type in the study showing the number of blood derived normal and solid tissue normal samples. In total, the number of blood derived healthy samples are 331 and the tissue derived healthy samples are 194. The sample information is provided in Supplementary Files 13-15.

Supplementary Information

We replicated our experiments with only error number m and only copy number d values in the mutation profiles. These experiments gave results similar to the complete profiles, suggesting that both m and d contain cancer-type signals. Figure 7.13 shows the associated pairwise accuracies and sensitivity/specificity for cancers with 3843 blood-derived normal unamplified samples when only d was used in the mutation profile and Figure 7.14 shows the associated pairwise accuracies and sensitivity/specificity when only m was used. Figures 7.15 and 7.16 show the plots using all the 4561 unamplified samples. Figures 7.17 and 7.18 show the plots for the 525 amplified samples.

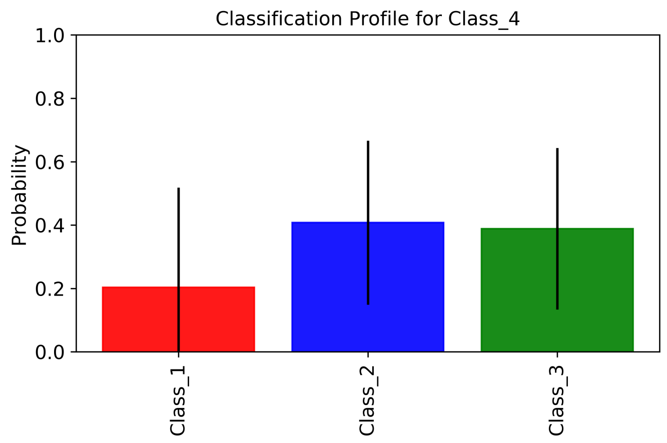


Figure 7.9: Cancer classification profile for Class 4 individuals when the multiclassifier was trained only using Class 1, Class 2 and Class 3 samples mentioned in Figure 7.3. This shows a stronger association of Class 4 with Class 2 and Class 3 than Class 1. However the high standard deviation for Class 1 also means that some individuals in Class 4 have a stronger association with Class 1 than Class 2 and Class 3.

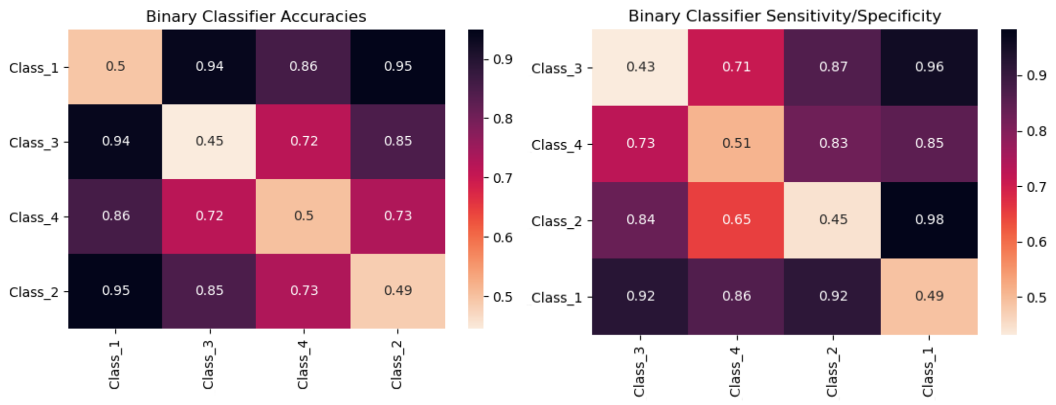


Figure 7.10: These seriation matrices show 4-fold validation accuracy and sensitivity/specificity for the four main clusters of cancers in Figure 7.5 where all the 4561 unamplified samples were used. Class 1 = (brain, kidney), Class 2 = (skin), Class 3 = (pancreas), Class 4 = (stomach, bladder, prostate, brain_lgg, lung, thyroid, lung_sq, head_neck).

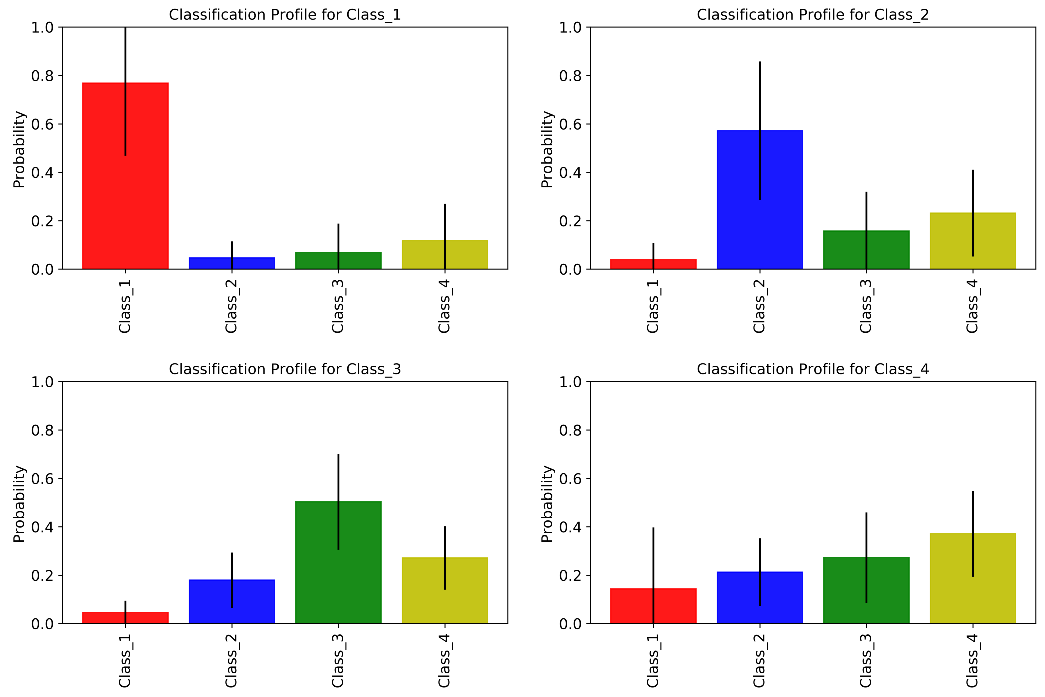


Figure 7.11: Mean and standard deviations for the cancer classification profiles of individuals in Class 1, Class 2, Class 3 and Class 4 (viewing left to right). To generate these profiles, we a trained multi-classifier on all four classes of cancers using gradient boosting. We then used this multi-classifier to obtain cancer classification profiles for a different set of individuals reported the average results for each cancer class. Class 1 individuals show a high probability for Class 1 cancers. Class 2 and Class 3 individuals also show a higher probability for their respective classes, but with a slightly weaker signal. Class 4 individuals show similar probabilities for Class 2, Class 3 and Class 4.

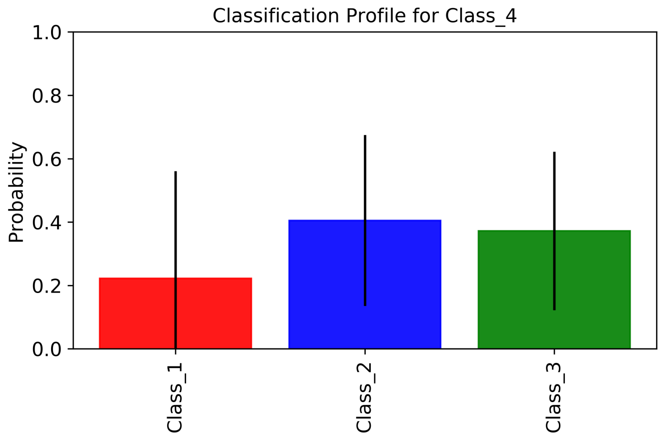


Figure 7.12: Cancer classification profile for Class 4 individuals when the multi-classifier was trained only using Class 1, Class 2 and Class 3 samples shown in Figure 7.10. This shows a stronger association of Class 4 with Class 2 and Class 3 than Class 1. However the high standard deviation for Class 1 also means that some individuals in Class 4 have a stronger association with Class 1 than Class 2 and Class 3.

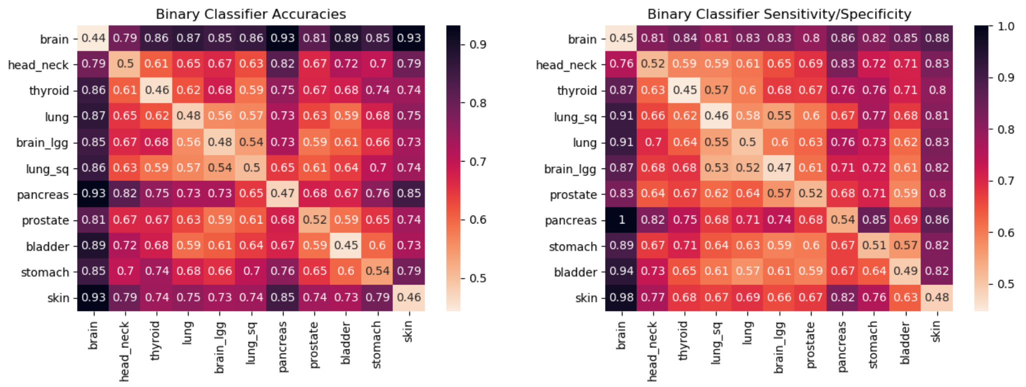


Figure 7.13: Accuracies and sensitivity/specificity for the pairwise classifiers when only the copy number (d) information was used in the mutation profile for the 3843 WXS blood-derived unamplified samples mentioned in Table 7.1A. Notice that like Figure 7.2, we again find strong distinguishing signals for brain, pancreas and skin cancers as can be seen by the darker rows corresponding to these cancers.

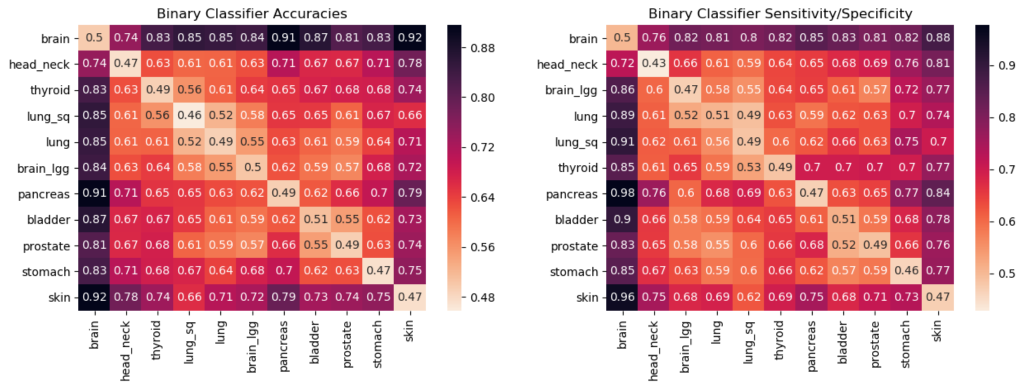


Figure 7.14: Accuracies and sensitivity/specificity for the pairwise classifiers when only the error number (m) information was used in the mutation profile for the 3843 WXS blood-derived unamplified samples mentioned in Table 7.1A. Notice that like Figure 7.2, we again find strong distinguishing signals for brain and skin cancers as can be seen by the darker rows corresponding to these cancers. However unlike Figure 7.2, the signal for pancreas cancer is not as strong.

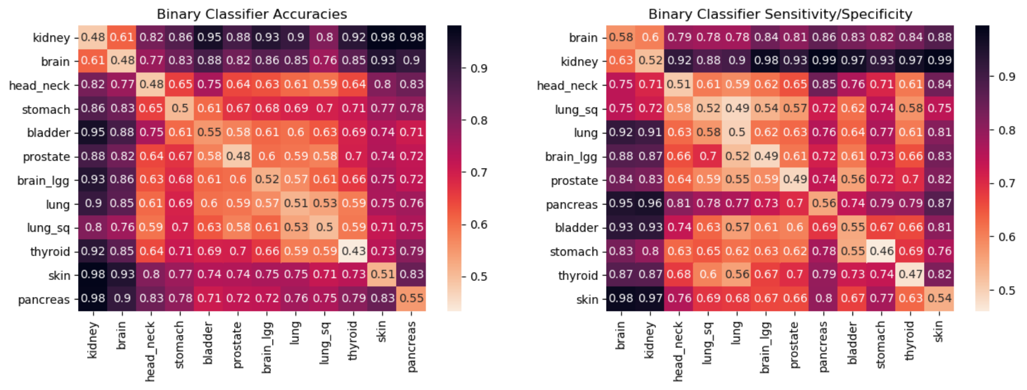


Figure 7.15: Accuracies and sensitivity/specificity for the pairwise classifiers when only the copy number (d) information was used in the mutation profile for the 4561 WXS unamplified samples that comprised both blood-derived and solid tissue normal type DNA as mentioned in Table 7.1A. Notice that like Figure 7.5, we again find strong distinguishing signals for (brain,kidney), pancreas and skin cancers as can be seen by the darker rows corresponding to these cancers.

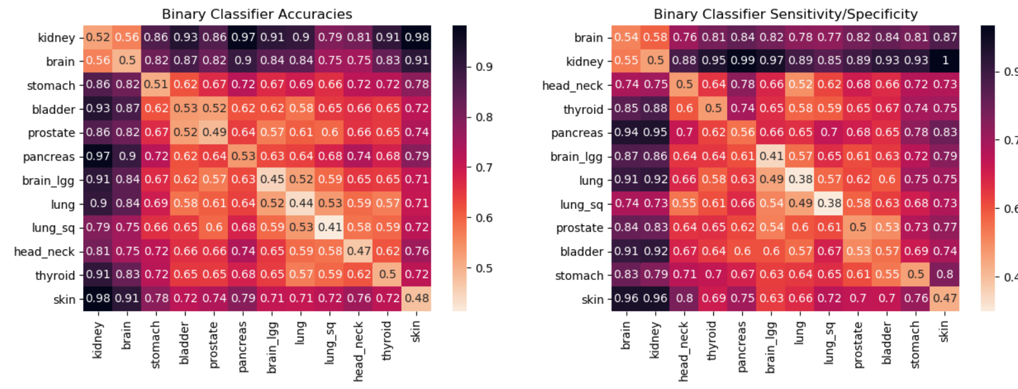


Figure 7.16: Accuracies and sensitivity/specificity for the pairwise classifiers when only the error number (m) information was used in the mutation profile for the 4561 WXS unamplified samples that comprised both blood-derived and solid tissue normal type DNA as mentioned in Table 7.1A. Notice that like Figure 7.5, we again find strong distinguishing signals for (brain,kidney) and skin cancers as can be seen by the darker rows corresponding to these cancers. However, unlike Figure 7.5, the signal for pancreas cancer is not as strong.

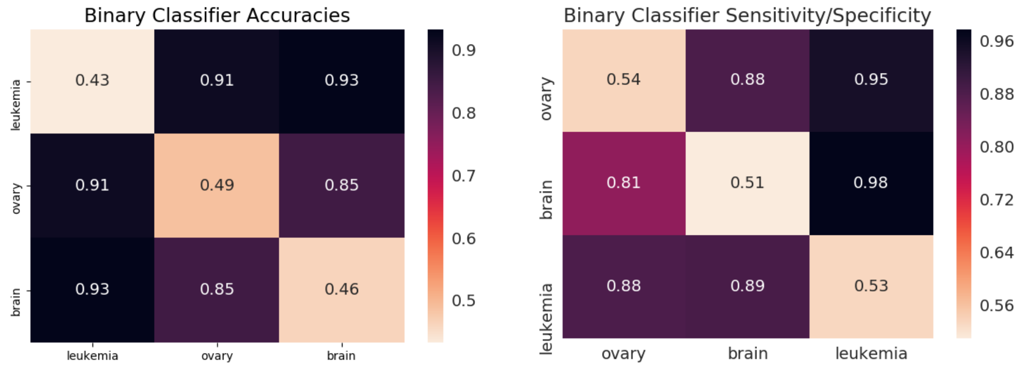


Figure 7.17: Accuracies and sensitivity/specificity for the pairwise classifiers when only the copy number(d) information was used in the mutation profile for the 525 WXS amplified samples mentioned in Table 7.1B. Notice that like Figure 7.6, we again find strong distinguishing signals for all the three cancers.

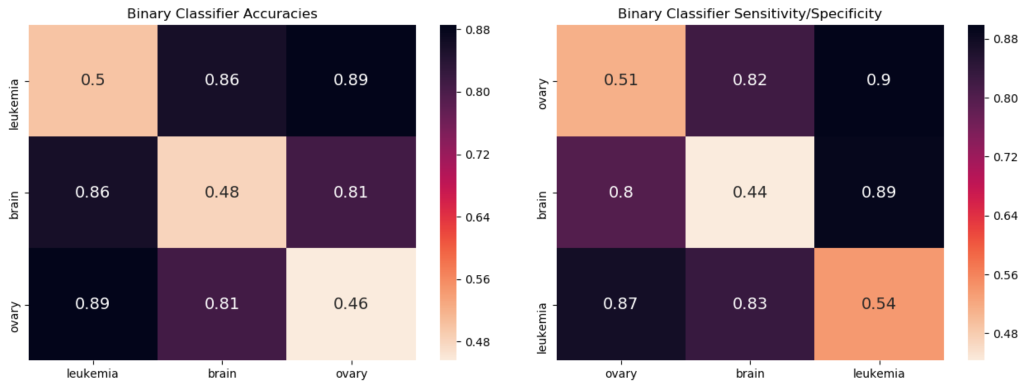


Figure 7.18: Accuracies and sensitivity/specificity for the pairwise classifiers when only the error number (m) information was used in the mutation profile for the 525 WXS amplified samples mentioned in Table 7.1B. Notice that like Figure 7.6, we again find strong distinguishing signals for all the three cancers.

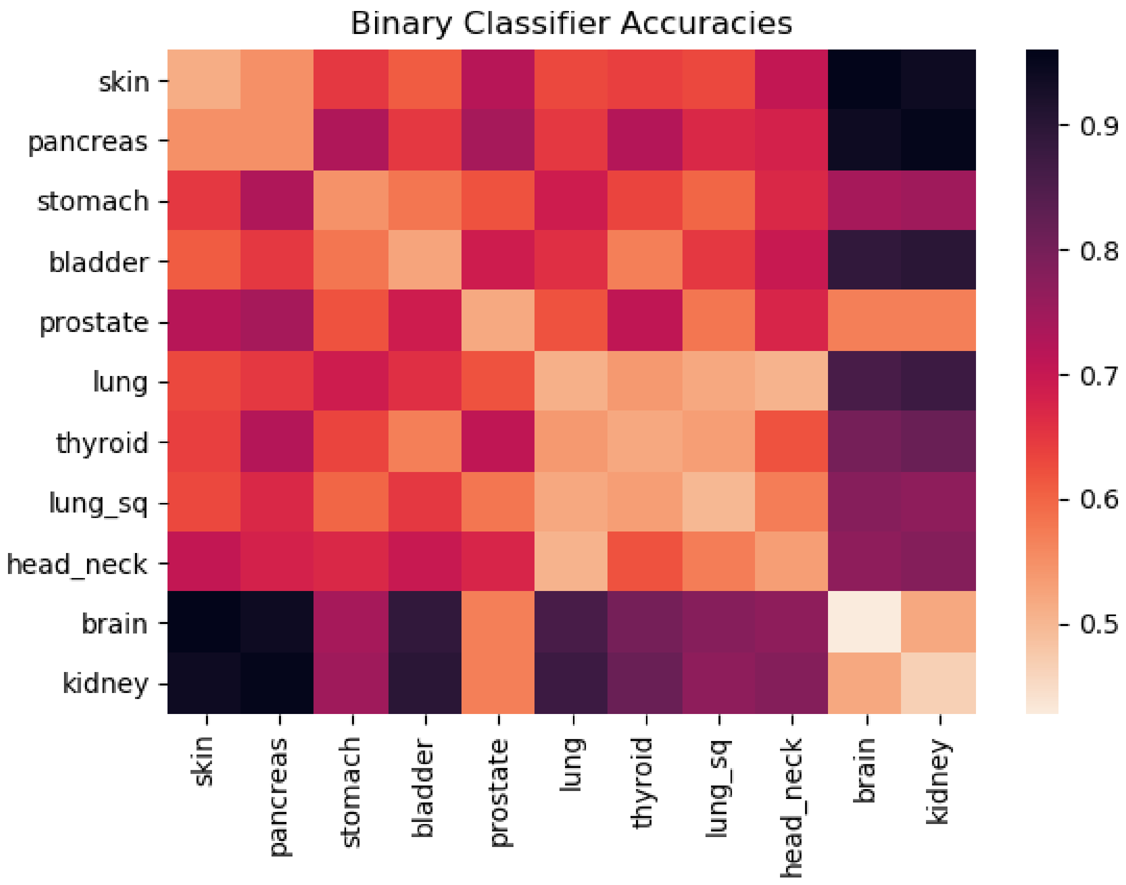


Figure 7.19: Accuracies for the pairwise classifiers when hipSTR was used for STR detection. hipSTR detects short tandem repeats with pattern lengths 6 or less. We used **–min-reads** 1 **–def-stutter-model** setting. Even though hipSTR requires further trimming of mutation profiles to tandem repeats with pattern lengths $\leqslant 6$, we were still able to detect cancer-specific signals. The differences from Figure 7.5 in Figure 7.19 (for example: prostate and brain, prostate and kidney, skin and pancreas) can be attributed to the fact that there are tandem repeat regions with pattern lengths greater than 6 that contained critical cancer-specific information. Further, we only used STRs that were commonly detected by hipSTR in all the samples being analyzed. Therefore, the STRs that were not detected in some samples were not used in this analysis.

Chapter 8

FUTURE WORK

We are proposing a new microscope to view the genome. This microscope looks for regions in the genome where mutation history can be inferred directly and uses that information to measure the accumulation of mutations in the genome over its evolutionary history. Within the context of machine learning, our approach is akin to feature extraction for each genome independently. We had some initial success by using tandem repeat regions in detecting the cancer signal from the healthy cell. Moving forward, the following directions can be taken:

8.1 Repeat regions and disease prediction

- **Biological Significance:** We found association of cancer with tandem repeat regions of the healthy genome. Though correlation doesn't mean causation, we can investigate if the important tandem repeat regions or features picked by the classifiers built using mutation profiles are of any biological significance. More precisely, i) Is there a correspondence between the already known driver genes and those regions? ii) Are there some new driver mutations to which these important features correspond? One possible way for accomplishing i) would be to investigate the driver gene information published in [124, 130, 131]. For ii) collaboration with cancer experts to investigate the role of those regions in understanding the cause of a specific cancer. This may lead to developing potential therapies if new cancer pathways are discovered.

- **Other mutation based diseases:** Our approach intuitively is measuring the accumulation of mutations. Therefore, it can potentially be applied to predict risks for any mutation based disease, such as Alzheimer's and utoimmune diseases [132, 133]. Recently, National Institute on Aging Genetics of Alzheimer's Disease Data Storage Site (NIAGADS) [134] released a dataset availing 4789 whole genome sequences for Alzheimer's patients.

- **Strength of the signal in different tissues/cells:** Another avenue we can take here is to investigate the intensity of this accumulation of mutations in DNA derived from different cells/tissues of an individual, to see how far in the *disease trajectory* DNA derived from each of those cells/tissues has reached.

A stronger signal in a certain tissue/cell compared to others might indicate the locations of the origins of the disease of interest. This would require DNA from different tissues/cells of an individual.

Nonetheless, this microscope is still far from being perfect and it can be improved by finding other markers of *decodable* evolution history in the genome. Other such markers could be *interspersed repeats* which we describe next.

8.2 Interspersed Repeats

Interspersed or transposon driven repeats cover about 48% of the human genome [6]. The method adopted would be to first detect these repeats and then design algorithms for history estimation. There are a few software tools available for interspersed repeat detection [135]. The ideas in phylogeny literature [136] and stochastic approximation [137] can be used to come up with generation models to estimate history of these regions.

Interspersed repeats like LINE, LTR, and DNA transposon that cover about 70% of interspersed elements in the DNA are long enough to pose several challenges in their accurate detection [126].
Challenges in detection of long tandem and interspersed repeats include: read lengths and coverage depth of the sequencers are the two determining factors that govern accurate detection of tandem and interspersed repeats in the human genome [126–128]. Long tandem repeats and most of the interspersed elements in the DNA are much longer than the Illumina read length (100-500 bp) [126] and this poses a problem in accurate detection as the same read can align to multiple locations [126, 127, 138]. There is a new tool [139] that was released recently which promises to solve this problem for longer tandem repeats. Further, there have been recent advances in the 3rd generation sequencing platforms like pacbio, oxford nanopore [140] that have much larger read lengths (of the order of 10,000's or more) and hybrid sequencers like 10Xgenomics [140]. It is hopeful that with these 3rd generation sequencers, the problem of accurately detecting long repeats will be resolved in the near future.

At the same time, one can also develop theoretical insights about different mutations to search for more ways to infer the evolutionary signal and for this mathematical modeling of evolutionary mutations can be studied which we describe next.

8.3 Mathematical Models of Evolution

Live DNA Storage: Live DNA storage was recently realized using CRISPR [30]. In live-DNA storage, information is embedded in the DNA of a cell and this cell replicates over time creating several copies of the same information. Due to substitution, indel and duplication mutations, these copies are not the same. One immediate application here is to design error correcting schemes so that the stored information can be uniquely recovered. This can be related to the reconstruction problem introduced by Levenshtein [141] and models used in phylogeny estimation [136]. Another application would be to use these erroneous copies and the initial information to characterize or infer the evolution channel. Both these applications are also related, as a better characterization of the evolution channel will also result in a more realistic error correction scheme. These days CRISPR technology can be used to design experiments to characterize the DNA storage channel. The data collected from these experiments can be used to make an inference about the evolution channel, which can then be incorporated into the channel model for the design of error correcting codes for live DNA storage. There are several questions that can be of interest here:

- Given a finite number of outputs but an unknown number of cell divisions or generations, what are the possible channels for which there are error correcting codes with high rate. In [20], we constructed high rate codes for bounded and uniform tandem duplication channels with one channel output and an unknown number of generations.
- How does partial information about the number of generations affect the code rate and construction for the above scenario? By partial information, it is meant that for some outputs, the number of generations is known, and for some, it is unknown.
- How robust is the code rate and construction to channel mismatch? We touched upon the question of channel mismatch in [35] for bounded tandem duplication string systems and characterized it by the measure of uncertainty.

Use Properties of Genome to understand evolution: The idea here is to use some mathematical properties of a genome sequence to come up with a sequence generation model that explains those properties to get insights about evolution limits. As an example, *single* strand DNA follows the 2nd Chargaff Rule [41], according to which upto a certain k, the frequency of a k-mer is almost the same as its reverse complement. The evolutionary reason for this property is still unknown. Further, this property is observed in many species which makes a case for them sharing a similar evolution dynamics [41]. We came up with a reverse complement tandem duplication model that could generate sequences which satisfied the 2nd Chargaff rule in a similar way as the real DNA [47]. There are several theoretical questions here which can be pursued: How does the seed and duplication length affect the timing of achieving this property and the frequency of k-mers observed? What is the effect of point mutations on the evolution process and how does it affect the timing? In a similar way, we can also look for other peculiar sequence properties of DNA and design mathematical models to simulate them.

BIBLIOGRAPHY

[1] C. E. Shannon. “A Mathematical Theory of Communication”. In: *The Bell System Technical Journal* 27.3 (July 1948), pp. 379–423. ISSN: 0005-8580. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[2] E. S. Lander. “Initial impact of the sequencing of the human genome”. In: *Nature* 470 (2011), pp. 187–197.

[3] P. M. Visscher, N. R. Wray, Q. Zhang, P. Sklar, M. I. McCarthy, M. A. Brown, and J. Yang. “10 years of GWAS Discovery: Biology, Function and Translation”. In: *Am. J. Hum. Genet.* 101.1 (2017), pp. 5–22.

[4] T. A. Manolio, F. S. Collins, N. J. Cox, D. B. Goldstein, L. A. Hindorff, D. J. Hunter, M. I. McCarthy, E. M. Ramos, L. R. Cardon, and A. Chakravarti. “Finding the missing heritability of complex diseases”. In: *Nature* 461 (2009), pp. 747–753.

[5] C. McIntosh and S. D. Wilton. “Polyglutamine ataxias : From Clinical and Molecular Features to Current Therapeutic Strategies”. In: 2017.

[6] E. S Lander, L. M Linton, B. Birren, C. Nusbaum, M. C Zody, Jennifer Baldwin, Keri Devon, Ken Dewar, M. Doyle, William FitzHugh, and others. “Initial Sequencing and Analysis of the Human Genome”. In: *Nature* 409.6822 (2001), pp. 860–921.

[7] N. I. Mundy and A. J Helbig. “Origin and Evolution of Tandem Repeats in the Mitochondrial DNA Control Region of Shrikes (Lanius Spp.)” In: *Journal of Molecular Evolution* 59.2 (2004), pp. 250–257.

[8] S. Ohno. *Evolution by Gene Duplication*. Springer-Verlag, 1970.

[9] F. Farnoud Hassanzadeh, M. Schwartz, and J. Bruck. “The Capacity of String-Duplication Systems”. In: *IEEE Transactions on Information Theory* 62.2 (2016), pp. 811–824. ISSN: 0018-9448. DOI: 10.1109/TIT.2015.2505735.

[10] J. Dassow, V. Mitrana, and Gheorghe Paun. “On the Regularity of Duplication Closure”. In: *Bulletin of the EATCS* 69 (1999), pp. 133–136.

[11] J. Dassow, V. Mitrana, and A. Salomaa. “Operations and language generating devices suggested by the genome evolution”. In: *Theoretical Computer Science* 270.1 (2002), pp. 701 –738. ISSN: 0304-3975. DOI: https://doi.org/10.1016/S0304-3975(01)00096-2. URL: http://www.sciencedirect.com/science/article/pii/S0304397501000962.

[12] P. Leupold, V. Mitrana, and J. M. Sempere. “Formal Languages Arising from Gene Repeated Duplication”. en. In: *Aspects of Molecular Computing*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2003, pp. 297–308. ISBN: 978-3-540-20781-8 978-3-540-24635-0. DOI: 10.1007/978-3-540-24635-0_22.

[13] F. Farnoud, M. Schwartz, and J. Bruck. “The capacity of string-duplication systems”. In: 62.2 (Feb. 2016), pp. 811–824.

[14] K. A. Schouhamer Immink and P. H. Siegel. “Codes for Mass Data Storage Systems (Second)”. In:

[15] D. Lind and B. H. M. *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, 1985.

[16] S. Jain, F. Farnoud Hassanzadeh, and J. Bruck. “Capacity and Expressiveness of Genomic Tandem Duplication”. In: *IEEE Trans. Information Theory* 63.10 (Oct. 2017). DOI: 10.1109/TIT.2017.2728079.

[17] A. Thue. “Über Unendliche Zeichenreihen”. In: *Norske Vid. Selsk. Skr. I. Mat. Nat. Kl., Christiana* (1906).

[18] N. Alon, J. Bruck, F. Farnoud Hassanzadeh, and S. Jain. “Duplication Distance to the Root for Binary Sequences”. In: *IEEE Transactions on Information Theory* 63.12 (Dec. 2017), pp. 7793–7803. ISSN: 0018-9448. DOI: 10.1109/TIT.2017.2730864.

[19] A. Lindenmayer. “Mathematical Models for Cellular Interactions in Development”. In: *Theoretical Biology* 18 (1968), pp. 300–315.

[20] S. Jain, F. Farnoud Hassanzadeh, M. Schwartz, and J. Bruck. “Duplication-Correcting Codes for Data Storage in the DNA of Living Organisms”. In: *IEEE Transactions on Information Theory* 63.8 (Aug. 2017), pp. 4996–5010. ISSN: 0018-9448. DOI: 10.1109/TIT.2017.2688361.

[21] Y. Meng Chee, J. Chrisnata, H. Mao Kiah, and Tuan Thanh Nguyen. “Deciding the Confusability of Words under Tandem Repeats”. In: *arXiv e-prints*, arXiv:1707.03956 (2017), arXiv:1707.03956. arXiv: 1707.03956 [math.CO].

[22] G. M. Church, Yuan Gao, and Sriram Kosuri. “Next-Generation Digital Information Storage in DNA”. en. In: *Science* 337.6102 (Sept. 2012), pp. 1628–1628. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1226355.

[23] N. Goldman, P. Bertone, S. Chen, C. Dessimoz, E. M. LeProust, B. Sipos, and E. Birney. “Towards Practical, High-Capacity, Low-Maintenance Information Storage in Synthesized DNA”. en. In: *Nature* 494.7435 (Feb. 2013), pp. 77–80. ISSN: 0028-0836. DOI: 10.1038/nature11875.

[24] R. N. Grass, R. Heckel, M. Puddu, D. Paunescu, and W. J. Stark. "Robust Chemical Preservation of Digital Information on DNA in Silica with Error-Correcting Codes". en. In: *Angewandte Chemie International Edition* 54.8 (Feb. 2015), pp. 2552–2555. ISSN: 1521-3773. DOI: `10.1002/anie.201411378`.

[25] S. M. H. T. Yazdi, R. Gabrys, and O. Milenkovic. "Portable and Error-Free DNA-Based Data Storage". In: *Scientific Reports* 7.1 (2017), p. 5011. ISSN: 2045-2322. DOI: `10.1038/s41598-017-05188-1`. URL: `https://doi.org/10.1038/s41598-017-05188-1`.

[26] R. Gabrys, H. M. Kiah, and O. Milenkovic. "Asymmetric Lee Distance Codes for DNA-Based Storage". In: *IEEE Transactions on Information Theory* 63.8 (2017), pp. 4982–4995. ISSN: 0018-9448. DOI: `10.1109/TIT.2017.2700847`.

[27] C. Schoeny, A. Wachter-Zeh, R. Gabrys, and E. Yaakobi. "Codes Correcting a Burst of Deletions or Insertions". In: *IEEE Transactions on Information Theory* 63.4 (2017), pp. 1971–1985. ISSN: 0018-9448. DOI: `10.1109/TIT.2017.2661747`.

[28] R. Gabrys, E. Yaakobi, and O. Milenkovic. "Codes in the damerau distance for DNA storage". In: *2016 IEEE International Symposium on Information Theory (ISIT)*. 2016, pp. 2644–2648. DOI: `10.1109/ISIT.2016.7541778`.

[29] R. Heckel, I. Shomorony, K. Ramchandran, and D. N. C. Tse. "Fundamental limits of DNA storage systems". In: *2017 IEEE International Symposium on Information Theory (ISIT)*. 2017, pp. 3130–3134. DOI: `10.1109/ISIT.2017.8007106`.

[30] S. L. Shipman, J. Nivala, J. D. Macklis, and G. M. Church. "CRISPR–Cas Encoding of a Digital Movie into the Genomes of a Population of Living Bacteria". en. In: *Nature* 547.7663 (July 2017), pp. 345–349. ISSN: 0028-0836. DOI: `10.1038/nature23017`.

[31] M. Arita and Y. Ohashi. "Secret Signatures inside Genomic DNA". eng. In: *Biotechnology Progress* 20.5 (2004 Sep-Oct), pp. 1605–1607. ISSN: 8756-7938. DOI: `10.1021/bp049917i`.

[32] D. Heider and A. Barnekow. "DNA Watermarks: A Proof of Concept". In: *BMC Molecular Biology* 9 (2008), p. 40. ISSN: 1471-2199. DOI: `10.1186/1471-2199-9-40`.

[33] M. Liss, D. Daubert, K. Brunner, K. Kliche, U. Hammes, A. Leiherer, and R. Wagner. "Embedding Permanent Watermarks in Synthetic Genes". In: *PLoS ONE* 7.8 (Aug. 2012), e42465. DOI: `10.1371/journal.pone.0042465`.

[34] K. L. Frieda, J. M. Linton, S. Hormoz, J. Choi, K-H. K. Chow, Z. S. Singer, M. W. Budde, M. B. Elowitz, and L. Cai. "Synthetic recording and in situ readout of lineage information in single cells". In: *Nature* 541 (Jan. 2017), pp. 107–111.

[35] S. Jain, F. Farnoud Hassanzadeh, M. Schwartz, and J. Bruck. “Noise and Uncertainty in String-Duplication Systems”. In: *IEEE Int. Symp. Information Theory (ISIT)*. Aachen, Germany, June 2017.

[36] M. Kovačević and V. Y. F. Tan. “Asymptotically Optimal Codes Correcting Fixed-Length Duplication Errors in DNA Storage Systems”. In: *IEEE Communications Letters* 22.11 (2018), pp. 2194–2197. ISSN: 1089-7798. DOI: 10.1109/LCOMM.2018.2868666.

[37] M. Kovačević. “Zero-Error Capacity of Duplication Channels”. In: *arXiv e-prints*, arXiv:1902.06275 (2019), arXiv:1902.06275. arXiv: 1902.06275 [cs.IT].

[38] Y. M. Chee, J. Chrisnata, H. M. Kiah, and T. T. Nguyen. “Efficient Encoding/Decoding of Irreducible Words for Codes Correcting Tandem Duplications”. In: *2018 IEEE International Symposium on Information Theory (ISIT)*. 2018, pp. 2406–2410. DOI: 10.1109/ISIT.2018.8437789.

[39] A. Lenz, A. Wachter-Zeh, and E. Yaakobi. “Duplication-correcting codes”. In: *Designs, Codes and Cryptography* 87.2 (2019), pp. 277–298. ISSN: 1573-7586. DOI: 10.1007/s10623-018-0523-0. URL: https://doi.org/10.1007/s10623-018-0523-0.

[40] R. Rudner, J. D. Karkas, and E. Chargaff. “Separation of B. subtilis DNA into complementary strands. 3. Direct analysis”. In: *Proc Natl Acad Sci U S A* 60.3 (1968). 4970114[pmid], pp. 921–922. ISSN: 0027-8424. URL: https://www.ncbi.nlm.nih.gov/pubmed/4970114.

[41] S. Shporer, B. Chor, S. Rosset, and D. Horn. “Inversion symmetry of DNA k-mer counts: validity and deviations”. In: *BMC Genomics* 17.1 (2016), p. 696. ISSN: 1471-2164. DOI: 10.1186/s12864-016-3012-8. URL: https://doi.org/10.1186/s12864-016-3012-8.

[42] A. Jamie Cuticchia and D. Qi. “Compositional symmetries in complete genomes ”. In: *Bioinformatics* 17.6 (June 2001), pp. 557–559. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/17.6.557. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/17/6/557/760443/170557.pdf. URL: https://dx.doi.org/10.1093/bioinformatics/17.6.557.

[43] S-G. Kong, W-L. Fan, H-D. Chen, Z-T. Hsu, N. Zhou, Bo Zheng, and H-C. Lee. “Inverse Symmetry in Complete Genomes and Whole-Genome Inverse Duplication”. In: *PLOS ONE* 4.11 (Nov. 2009), pp. 1–15. DOI: 10.1371/journal.pone.0007553. URL: https://doi.org/10.1371/journal.pone.0007553.

[44] S. Wang, J. Tu, Z. Jia, and Z. Lu. “High order intra-strand partial symmetry increases with organismal complexity in animal evolution”. In: *Scientific Reports* 4 (2014). Article, 6400 EP –. URL: https://doi.org/10.1038/srep06400.

[45] V. Afreixo, C. A.C. Bastos, Sara P. Garcia, J. M.O.S. Rodrigues, A. J. Pinho, and Paulo J.S.G. Ferreira. "The breakdown of the word symmetry in the human genome". In: *Journal of Theoretical Biology* 335 (2013), pp. 153 –159. ISSN: 0022-5193. DOI: https://doi.org/10.1016/j.jtbi.2013.06.032. URL: http://www.sciencedirect.com/science/article/pii/S0022519313003044.

[46] P. Baldi, P.-François Baisnée, and S. Hampson. "Why are complementary DNA strands symmetric?" In: *Bioinformatics* 18.8 (Aug. 2002), pp. 1021–1033. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/18.8.1021. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/18/8/1021/633125/181021.pdf. URL: https://dx.doi.org/10.1093/bioinformatics/18.8.1021.

[47] S. Jain, N. Raviv, and J. Bruck. "Attaining the 2nd Chargaff Rule by Tandem Duplications". In: *2018 IEEE International Symposium on Information Theory (ISIT)*. 2018, pp. 2241–2245. DOI: 10.1109/ISIT.2018.8437526.

[48] G Levinson and G A Gutman. "Slipped-Strand Mispairing: A Major Mechanism for DNA Sequence Evolution." In: *Molecular Biology and Evolution* 4.3 (1987), pp. 203–221.

[49] C. Schlötterer. "Evolutionary Dynamics of Microsatellite DNA". en. In: *Chromosoma* 109.6 (Sept. 2000), pp. 365–371. ISSN: 0009-5915, 1432-0886. DOI: 10.1007/s004120000089.

[50] J. X. Sun, A. Helgason, G. Masson, S. S. Ebenesersdóttir, H. Li, S. Mallick, S. Gnerre, N. Patterson, A. Kong, D. Reich, and K. Stefansson. "A Direct Characterization of Human Mutation Based on Microsatellites". en. In: *Nature Genetics* 44.10 (Oct. 2012), pp. 1161–1165. ISSN: 1061-4036. DOI: 10.1038/ng.2398.

[51] P. J. Campbell, S. Yachida, L. J. Mudie, P. J. Stephens, E. D. Pleasance, L. A. Stebbings, L. A. Morsberger, C. Latimer, S. McLaren, M-L. Lin, D. J. McBride, I. Varela, S. A. Nik-Zainal, C. Leroy, M. Jia, A. Menzies, A. P. Butler, J. W. Teague, C. A. Griffin, J. Burton, H. Swerdlow, M. A. Quail, M. R. Stratton, C. Iacobuzio-Donahue, and P. A. Futreal. "The patterns and dynamics of genomic instability in metastatic pancreatic cancer". In: *Nature* 467 (2010), 1109 EP –. URL: https://doi.org/10.1038/nature09460.

[52] B. W. Stewart and C. P. Wild. *World Cancer Report*. Lyon, France: IARC, 2014.

[53] C. Tomasetti, L. Li, and B. Vogelstein. "Stem cell divisions, somatic mutations, cancer etiology, and cancer prevention". In: *Science* 6331.355 (2017), pp. 1330–1334.

[54] A. Sud, B. Kinnersley, and R. S. Houlston. "Genome-wide association studies of cancer: current insights and future perspectives". In: *Nature Reviews* 17 (2017), pp. 692–704.

[55] L. Ding et al. “Perspective on Oncogenic Processes at the End of the Beginning of Cancer Genomics”. In: *Cell* 173.2 (2018), 305–320.e10. ISSN: 0092-8674. DOI: 10.1016/j.cell.2018.03.033. URL: https://doi.org/10.1016/j.cell.2018.03.033.

[56] M. Tang, M. Waterman, and S. Yooseph. “Zinc Finger Gene Clusters and Tandem Gene Duplication”. In: *Proceedings of the Fifth Annual International Conference on Computational Biology*. RECOMB ’01. Montreal, Quebec, Canada: ACM, 2001, pp. 297–304. ISBN: 1-58113-353-7. DOI: 10.1145/369133.369241. URL: http://doi.acm.org/10.1145/369133.369241.

[57] National Cancer Institute. *About the Data | NCI Genomic Data Commons*. URL: https://gdc.cancer.gov/about-data.

[58] L. Mason, J. Baxter, P. Bartlett, and M. Frean. “Boosting Algorithms As Gradient Descent”. In: *Proceedings of the 12th International Conference on Neural Information Processing Systems*. NIPS’99. Denver, CO: MIT Press, 1999, pp. 512–518. URL: http://dl.acm.org/citation.cfm?id=3009657.3009730.

[59] S. Jain, B. Mazaheri, N. Raviv, and J. Bruck. “Disease Risk Estimation from Mutation Profile of the Genome”. In: *Provisional Patent Filed with Caltech* ().

[60] S. Jain, B. Mazaheri, N. Raviv, and J. Bruck. “Cancer Classification from Healthy DNA using Machine Learning”. In: *bioRxiv* (2019). DOI: 10.1101/517839. eprint: https://www.biorxiv.org/content/early/2019/01/11/517839.full.pdf. URL: https://www.biorxiv.org/content/early/2019/01/11/517839.

[61] K. Usdin. “The Biological Effects of Simple Tandem Repeats: Lessons from the Repeat Expansion Diseases”. In: *Genome research* 18.7 (2008), pp. 1011–1019.

[62] J. W. Fondon and Harold R. Garner. “Molecular Origins of Rapid and Continuous Morphological Evolution”. In: *Proceedings of the National Academy of Sciences* 101.52 (2004), pp. 18058–18063. DOI: 10.1073/pnas.0408118101.

[63] P. Leupold, C. Martín-Vide, and V. Mitrana. “Uniformly bounded duplication languages”. In: *Discrete Applied Mathematics* 146.3 (2005), pp. 301 –310. ISSN: 0166-218X. DOI: https://doi.org/10.1016/j.dam.2004.10.003. URL: http://www.sciencedirect.com/science/article/pii/S0166218X04003531.

[64] J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2006. ISBN: 0321455363.

[65] J. Shallit. *A Second Course in Formal Languages and Automata Theory*. 1st ed. New York, NY, USA: Cambridge University Press, 2008. ISBN: 0521865727, 9780521865722.

[66] N. G. De Bruijn. "A Combinatorial Problem". In: *Proceedings of the Section of Sciences of the Koninklijke Nederlandse Akademie van Wetenschappen te Amsterdam* 49.7 (1946). Available: http://repository.tue.nl/415282b7-6c10-4b9f-9624-4437629cc621, pp. 758–764.

[67] G. Benson and L. Dong. "Reconstructing the Duplication History of a Tandem Repeat." In: *ISMB*. 1999, pp. 44–53.

[68] O. Gascuel, D. Bertrand, and O. Elemento. *Mathematics of Evolution and Phylogeny*. Ed. by Olivier Gascuel. Oxford: Oxford University Press, 2005.

[69] O. Elishco, F. Farnoud Hassanzadeh, M. Schwartz, and J. Bruck. "The Capacity of Some Pólya String Models". In: *IEEE Int. Symp. Information Theory (ISIT)*. Barcelona, Spain, July 2016, pp. 270–274. DOI: 10.1109/ISIT.2016.7541303.

[70] F. Farnoud, M. Schwartz, and J. Bruck. "Estimation of duplication history under a stochastic model for tandem repeats". In: *BMC Bioinformatics* 20.1 (2019), p. 64. ISSN: 1471-2105. DOI: 10.1186/s12859-019-2603-1. URL: https://doi.org/10.1186/s12859-019-2603-1.

[71] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error Correcting Codes*. New York: Elsevier/North-Holland Inc., 1977.

[72] J M. Steele. *Probability Theory and Combinatorial Optimization*. Society for Industrial and Applied Mathematics, 1997.

[73] R. C. Entringer, D. E. Jackson, and J.A. Schatz. "On Nonrepetitive Sequences". In: *J. Combinatorial Theory, Series A* 16.2 (1974), pp. 159–164. ISSN: 0097-3165. DOI: http://dx.doi.org/10.1016/0097-3165(74)90041-7.

[74] A. Lindenmayer. "Mathematical models for cellular interactions in development". In: *Theoretical Biology* 18 (1968), pp. 300–315.

[75] P. Prusinkiewicz and A. Lindenmayer. *The algorithmic beauty of plants*. Springer–Verlag, 1990.

[76] R. Roth. *Introduction to Coding Theory*. Cambridge University Press, 2006.

[77] M. Arita and Y. Ohashi. "Secret Signatures Inside Genomic DNA". In: *Biotechnology Progress* 20.5 (2004), pp. 1605–1607.

[78] D. Heider and A. Barnekow. "DNA-based watermarks using the DNA-Crypt algorithm". In: *BMC Bioinformatics* 8.1 (2007), pp. 1–10.

[79] M. Liss, D. Daubert, K. Brunner, K. Kliche, U. Hammes, A. Leiherer, and R. Wagner. "Embedding Permanent Watermarks in Synthetic Genes". In: *PLoS ONE* 7.8 (Aug. 2012), e42465.

[80] P. C. Wong, K-k. Wong, and H. Foote. "Organic Data Memory Using the DNA Approach". In: *Commun. ACM* 46.1 (Jan. 2003), pp. 95–98.

[81] D. C. Jupiter, T. A. Ficht, J. Samuel, Q-M. Qin, and P. de Figueiredo. "DNA Watermarking of Infectious Agents: Progress and Prospects". In: *PLoS Pathog* 6.6 (June 2010), e1000950.

[82] F. Balado. "Capacity of DNA Data Embedding Under Substitution Mutations". In: 59.2 (Feb. 2013), pp. 928–941.

[83] C. T. Clelland, V. Risca, and C. Bancroft. "Hiding messages in DNA microdots". In: *Nature* 399.6736 (June 1999), pp. 533–534.

[84] N. Yachie, Y. Ohashi, and M. Tomita. "Stabilizing synthetic data in the DNA of living organisms". In: *Systems and Synthetic Biology* 2.1-2 (2008), pp. 19–25.

[85] D. Haughton and F. Balado. "BioCode: Two biologically compatible Algorithms for embedding data in non-coding and coding regions of DNA". In: *BMC Bioinformatics* 14.1 (2013), pp. 1–16.

[86] S. M. H. T. Yazdi, H. M. Kiah, E. Garcia-Ruiz, J. Ma, H. Zhao, and O. Milenkovic. "DNA-Based Storage: Trends and Methods". In: *IEEE Transactions on Molecular, Biological and Multi-Scale Communications* 1.3 (Sept. 2015), pp. 230–248. DOI: `10.1109/TMBMC.2016.2537305`.

[87] F. Farnoud, M. Schwartz, and J. Bruck. "A stochastic model for genomic interspersed duplication". In: *Proceedings of the 2015 IEEE International Symposium on Information Theory (ISIT2015), Hong Kong, China SAR*. June 2015, pp. 1731–1735.

[88] L. Dolecek and V. Anantharam. "Repetition error correcting sets: explicit constructions and prefixing methods". In: 23.4 (2010), pp. 2120–2146.

[89] P. Leupold, C. Martín-Vide, and V. Mitrana. "Uniformly bounded duplication languages". In: 146.3 (2005), pp. 301–310.

[90] J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Prentice Hall, 2004.

[91] P. Leupold. "Duplication roots". In: 4588 (2007), pp. 290–299.

[92] D. Lind and B. H. Marcus. *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, 1985.

[93] K. A. S. Immink. *Coding Techniques for Digital Recorders*. Prentice-Hall, 1991.

[94] A. Kato and K. Zeger. "On the capacity of two-dimensional run-length constrained channels". In: 45 (July 1999), pp. 1527–1540.

[95] R. M. Roth and P. H. Siegel. "Lee-metric BCH codes and their application to constrained and partial-response channels". In: 40.4 (July 1994), pp. 1083–1096.

[96] J. H. van Lint and R. M. Wilson. *A Course in Combinatorics, 2nd Edition*. Cambridge Univ. Press, 2001.

[97] E. Chargaff. "Chemical specificity of nucleic acids and mechanism of their enzymatic degradation". In: *Experientia* 6.6 (1950), 201–209. DOI: 10.1007/bf02173653.

[98] E. Chargaff. "Structure and function of nucleic acids as cell constituents." In: *Federation proceedings* 10 3 (1951), pp. 654–9.

[99] J. D. Watson and F. H. C. Crick. "Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid". In: *Nature* 171.4356 (1953), pp. 737–738. ISSN: 1476-4687. DOI: 10.1038/171737a0. URL: https://doi.org/10.1038/171737a0.

[100] D. Mitchell and R. Bridge. "A test of Chargaff's second rule". In: *Biochemical and Biophysical Research Communications* 340.1 (2006), pp. 90 –94. ISSN: 0006-291X. DOI: https://doi.org/10.1016/j.bbrc.2005.11.160. URL: http://www.sciencedirect.com/science/article/pii/S0006291X05027130.

[101] V. Afreixo, João M O S Rodrigues, and C. A C Bastos. "Analysis of single-strand exceptional word symmetry in the human genome: new measures". In: *Biostatistics (Oxford, England)* 16.2 (2015), 209—221. ISSN: 1465-4644. DOI: 10.1093/biostatistics/kxu041. URL: https://doi.org/10.1093/biostatistics/kxu041.

[102] B.R. Powdel, M. Borah, S. S. Satapathy, A. K., A. K. Buragohain, Pankaj K. Jha, and Suvendra K. Ray. "A Study in Entire Chromosomes of Violations of the Intra-strand Parity of Complementary Nucleotides (Chargaff's Second Parity Rule)". In: *DNA Research* 16.6 (Oct. 2009), pp. 325–343. ISSN: 1340-2838. DOI: 10.1093/dnares/dsp021. eprint: http://oup.prod.sis.lan/dnaresearch/article-pdf/16/6/325/1039502/dsp021.pdf. URL: https://dx.doi.org/10.1093/dnares/dsp021.

[103] F. Farnoud, M. Schwartz, and J. Bruck. "A Stochastic Model for Genomic Interspersed Duplication". In: *IEEE Int. Symp. Information Theory (ISIT)*. June 2015, pp. 904–908. DOI: 10.1109/ISIT.2015.7282586.

[104] L. C. Tsui, M. Buchwald, D. Barker, J. C. Braman, R. Knowlton, J. W. Schumm, H. Eiberg, J. Mohr, D. Kennedy, N. Plavsic, and al. et. "Cystic fibrosis locus defined by a genetically linked polymorphic DNA marker". In: *Science* 230.4729 (1985), pp. 1054–1057. ISSN: 0036-8075. DOI: 10.1126/science.2997931. eprint: http://science.sciencemag.org/content/230/4729/1054.full.pdf. URL: http://science.sciencemag.org/content/230/4729/1054.

[105] M. E. MacDonald, C. M. Ambrose, M. P. Duyao, R. H. Myers, C. Lin, L. Srinidhi, G. Barnes, S. A. Taylor, M. James, N. Groot, H. MacFarlane, B. Jenkins, M. A. Anderson, N. S. Wexler, J. F. Gusella, G. P. Bates, S.

Baxendale, H. Hummerich, S. Kirby, M. North, S. Youngman, R. Mott, G. Zehetner, Z. Sedlacek, A. Poustka, A-M. Frischauf, H. Lehrach, A. J. Buckler, D. Church, L. Doucette-Stamm, M. C. O'Donovan, L. Riba-Ramirez, M. Shah, V. P. Stanton, S. A. Strobel, K. M. Draths, J. L. Wales, P. Dervan, D. E. Housman, M. Altherr, R. Shiang, L. Thompson, T. Fielder, J. J. Wasmuth, D. Tagle, J. Valdes, L. Elmer, M. Allard, L. Castilla, M. Swaroop, K. Blanchard, F. S. Collins, R. Snell, T. Holloway, K. Gillespie, N. Datson, D. Shaw, and P. S. Harper. "A novel gene containing a trinucleotide repeat that is expanded and unstable on Huntington's disease chromosomes". In: *Cell* 72.6 (1993), pp. 971 –983. ISSN: 0092-8674. DOI: `https://doi.org/10.1016/0092-8674(93)90585-E`. URL: `http://www.sciencedirect.com/science/article/pii/009286749390585E`.

[106] I. Oberle, F. Rousseau, D. Heitz, D. Kretz C.and Devys, A. Hanauer, J. Boue, M. F. Bertheas, and J. L. Mandel. "Instability of a 550-base pair DNA segment and abnormal methylation in fragile X syndrome". In: *Science* 252.5009 (1991), pp. 1097–1102. ISSN: 0036-8075. DOI: `10.1126/science.252.5009.1097`. eprint: `http://science.sciencemag.org/content/252/5009/1097.full.pdf`. URL: `http://science.sciencemag.org/content/252/5009/1097`.

[107] J. N. Hirschhorn and M. J. Daly. "Genome-wide association studies for common diseases and complex traits". In: *Nature Reviews Genetics* 6 (2005), pp. 95–108.

[108] M. Bossert. *Channel Coding for Telecommunications*. 1st. New York, NY, USA: John Wiley & Sons, Inc., 1999. ISBN: 0471982776.

[109] E. Vilar and S. B. Gruber. "Microsatellite instability in colorectal cancer-the stable evidence". In: *Nat Rev Clin Oncol* 7 (2010), pp. 153–162.

[110] L.A. Meyer, R. R. Broaddus, and K. H. Lu. "Endometrial cancer and Lynch syndrome: clinical and pathologic considerations". In: *Cancer Control* 16 (2009), pp. 14–22.

[111] G. Singer, T. Kallinowski, A. Hartmann, W. Dietmaier, P. J. Wild, P. Schraml, G. Sauter, M. J. Mihatsch, and H. Moch. "Different types of microsatellite instability in ovarian carcinoma". In: *Int J Cancer* 112 (2004), pp. 643–646.

[112] C. C. Pritchard, C. Morrissey, A. K., X. Zhang, C. Smith, I. Coleman, S. J. Salipante, J. Milbank, M. Yu, and W. M. Grady et al. "Complex MSH2 and MSH6 mutations in hypermutated microsatellite unstable advanced prostate cancer". In: *Nat. Commun.* 5 (2014), p. 4998.

[113] T. Bilgin Sonay, M. Koletou, and A. Wagner. "A Survey of Tandem Repeat Instabilities and Associated Gene Expression Changes in 35 Colorectal Cancers". In: *BMC Genomics* 16.1 (Sept. 2015). ISSN: 1471-2164. DOI: `10.1186/s12864-015-1902-9`.

[114] R. J. Hause, C. C. Pritchard, J. Shendure, and S. J. Salipante. “Classification and Characterization of Microsatellite Instability across 18 Cancer Types”. en. In: *Nature Medicine* 22.11 (Nov. 2016), pp. 1342–1350. ISSN: 1078-8956. DOI: `10.1038/nm.4191`.

[115] L. B. Alexandrov, S. Nik-Zainal, D. C. Wedge, S. A. J. R. Aparicio, S. Behjati, A. V. Biankin, G. R. Bignell, N. Bolli, A. Borg, A.-L. Børresen-Dale, S. Boyault, B. Burkhardt, A. P. Butler, C. Caldas, H. R. Davies, C. Desmedt, R. Eils, J. E. Eyfjörd, J. A. Foekens, M. Greaves, F. Hosoda, B. Hutter, T. Ilicic, S. Imbeaud, M. Imielinski, N. Jäger, D. T. W. Jones, D. Jones, S. Knappskog, M. Kool, S. R. Lakhani, C. López-Otín, S. Martin, N. C. Munshi, H. Nakamura, P. A. Northcott, M. Pajic, E. Papaemmanuil, A. Paradiso, J. V. Pearson, Xose S. Puente, K. Raine, M. Ramakrishna, A. L. Richardson, J. Richter, P. Rosenstiel, M. Schlesner, T. N. Schumacher, P. N. Span, J. W. Teague, Y. Totoki, A. N. J. Tutt, R. Valdés-Mas, M. M. van Buuren, L. van ’t Veer, A. Vincent-Salomon, N. Waddell, L. R. Yates, Australian Pancreatic Cancer Genome Initiative, ICGC Breast Cancer Consortium, ICGC MMML-Seq Consortium, I. C. G. C. PedBrain, J. Zucman-Rossi, P. Andrew Futreal, U. McDermott, P. Lichter, M. Meyerson, S. M. Grimmond, R. Siebert, E. Campo, T. Shibata, S. M. Pfister, P. J. Campbell, and M. R. Stratton. “Signatures of mutational processes in human cancer”. In: *Nature* 500 (2013), pp. 415–421. URL: `https://doi.org/10.1038/nature12477`.

[116] G. Benson. “Tandem repeats finder: a program to analyze DNA sequences. Nucleic Acids Res.” In: *Nucleic Acids Research* 27.2 (1999), pp. 573–580.

[117] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study (Princeton Series in Applied Mathematics)*. Princeton, NJ, USA: Princeton University Press, 2007. ISBN: 0691129932, 9780691129938.

[118] D. Aran, R. Camarda, J. Odegaard, H. Paik, B. Oskotsky, G. Krings, A. Goga, M. Sirota, and A. J. Butte. “Comprehensive analysis of normal adjacent to tumor transcriptomes”. In: *Nature Communications* 8.1 (2017), p. 1077. ISSN: 2041-1723. DOI: `10.1038/s41467-017-01027-z`. URL: `https://doi.org/10.1038/s41467-017-01027-z`.

[119] E. Persi, D. Prandi, Y. I. Wolf, Y. Pozniak, C. Barbieri, P. Gasperini, H. Beltran, B. M. Faltas, M. A. Rubin, T. Geiger, E. V. Koonin, F. Demichelis, and D. Horn. “Proteomic and Genomic Signatures of Repeat-instability in Cancer and Adjacent Normal Tissues”. In: *bioRxiv* (2018). DOI: `10.1101/491423`. eprint: `https://www.biorxiv.org/content/early/2018/12/09/491423.full.pdf`. URL: `https://www.biorxiv.org/content/early/2018/12/09/491423`.

[120] C. F. A. de Bourcy, I. De Vlaminck, J. N. Kanbar, J. Wang, C. Gawad, and S. R. Quake. “A quantitative comparison of single cell whole genome

amplification methods". In: *PLoS One* 9.8 (2014), e105585. DOI: https://doi.org/10.1371/journal.pone.0105585.

[121] S. Jain, B. Mazaheri, N. Raviv, and J. Bruck. "Short Tandem Repeats Information in TCGA is Statistically Biased by Amplification". In: *bioRxiv* (2019). DOI: 10.1101/518878. eprint: https://www.biorxiv.org/content/early/2019/01/11/518878.full.pdf. URL: https://www.biorxiv.org/content/early/2019/01/11/518878.

[122] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, and 1000 Genome Project Data Processing Subgroup. "The Sequence Alignment/Map format and SAMtools". In: *Bioinformatics* 25.16 (2009). 19505943[pmid], pp. 2078–2079. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btp352. URL: https://www.ncbi.nlm.nih.gov/pubmed/19505943.

[123] T. Willems, D. Zielinski, J. Yuan, A. Gordon, M. Gymrek, and Y. Erlich. "Genome-wide profiling of heritable and de novo STR variations". In: *Nat Methods.* 14.6 (2017), pp. 590–592.

[124] S. A. Forbes, D. Beare, H. Boutselakis, S. Bamford, N. Bindal, J. Tate, C. G. Cole, S. Ward, E. Dawson, L. Ponting, R. Stefancsik, B. Harsha, C. Y. Kok, M. Jia, H. Jubb, Z. Sondka, S. Thompson, T. De, and P. J. Campbell. "COSMIC: somatic cancer genetics at high-resolution". In: *Nucleic Acids Research* 45.D1 (2017), pp. D777–D783. DOI: 10.1093/nar/gkw1121. eprint: /oup/backfile/content_public/journal/nar/45/d1/10.1093_nar_gkw1121/3/gkw1121.pdf. URL: http://dx.doi.org/10.1093/nar/gkw1121.

[125] *Cancer Gene Census COSMIC*. https://cancer.sanger.ac.uk.

[126] T. J. Treangen and Steven L. Salzberg. "Repetitive DNA and next-generation sequencing: computational challenges and solutions". In: *Nature Reviews Genetics* 13 (2011). Review Article, 36 EP –. URL: https://doi.org/10.1038/nrg3117.

[127] A. D. Ewing. "Transposable element detection from whole genome sequence data". In: *Mob DNA* 6 (2015). 26719777[pmid], pp. 24–24. ISSN: 1759-8753. DOI: 10.1186/s13100-015-0055-3. URL: https://www.ncbi.nlm.nih.gov/pubmed/26719777.

[128] M. Gymrek. "A genomic view of short tandem repeats". In: *Current Opinion in Genetics Development* 44 (2017). Molecular and genetic bases of disease, pp. 9 –16. ISSN: 0959-437X. DOI: https://doi.org/10.1016/j.gde.2017.01.012. URL: http://www.sciencedirect.com/science/article/pii/S0959437X16301538.

[129] *db GaP*. https://dbgap.ncbi.nlm.nih.gov/aa/wga.cgi?page=login.

[130] M. H. Bailey et al. “Comprehensive Characterization of Cancer Driver Genes and Mutations”. In: *Cell* 173.2 (2018), 371–385.e18. ISSN: 0092-8674. DOI: 10.1016/j.cell.2018.02.060. URL: https://doi.org/10.1016/j.cell.2018.02.060.

[131] *Driver Gene*. http://www.tumorportal.org/. WebContent.

[132] R.N. Rosenberg, D. Lambracht-Washington, G. Yu, and W. Xia. “Genomics of alzheimer disease: A review”. In: *JAMA Neurology* 73.7 (2016), pp. 867–874. DOI: 10.1001/jamaneurol.2016.0301. eprint: /data/journals/neur/935408/nrv160001.pdf. URL: +http://dx.doi.org/10.1001/jamaneurol.2016.0301.

[133] V. Pascual, D. Chaussabel, and J. Banchereau. “A genomic approach to human autoimmune diseases”. In: *Annu Rev Immunol.* 28 (2010), pp. 535–571.

[134] *NIAGADS Data Sharing Service*. URL: https://dss.niagads.org/datasets/adsp-umbrella/.

[135] A. F. A. Smit, R. Hubley, and P. Green. *RepeatMasker Open-4.0*. 2013-2015. URL: http://www.repeatmasker.org.

[136] T. Warnow. *Computational Phylogenetics: An Introduction to Designing Methods for Phylogeny Estimation*. Cambridge University Press, 2017. DOI: 10.1017/9781316882313.

[137] V. S Borkar. “Stochastic Approximation”. In: *Cambridge Books* (2008).

[138] I. Shomorony, T. A. Courtade, and D. Tse. “Fundamental Limits of Genome Assembly Under an Adversarial Erasure Model”. In: *IEEE Transactions on Molecular, Biological and Multi-Scale Communications* 2.2 (2016), pp. 199–208. ISSN: 2372-2061. DOI: 10.1109/TMBMC.2016.2641440.

[139] N. Mousavi, S. Shleizer-Burko, and M. Gymrek. “Profiling the genome-wide landscape of tandem repeat expansions”. In: *bioRxiv* (2018). DOI: 10.1101/361162. eprint: https://www.biorxiv.org/content/early/2018/07/03/361162.full.pdf. URL: https://www.biorxiv.org/content/early/2018/07/03/361162.

[140] E. L. Van Dijk, Y. Jaszczyszyn, D. Naquin, and C. Thermes. “The third revolution in sequencing technology”. In: *Trends in Genetics* 34.9 (2018), pp. 666–681.

[141] V. I. Levenshtein. “Efficient reconstruction of sequences”. In: *IEEE Transactions on Information Theory* 47.1 (2001), pp. 2–22. ISSN: 0018-9448. DOI: 10.1109/18.904499.