DE-NOVO ASSEMBLY OF SHORT READS IN MINIMAL OVERLAP MODEL

SHASHANK SHARMA

AMAR NATH AND SHASHI KHOSLA SCHOOL OF INFORMATION TECHNOLOGY

INDIAN INSTITUTE OF TECHNOLOGY DELHI

AUGUST 2023

DE-NOVO ASSEMBLY OF SHORT READS IN MINIMAL OVERLAP MODEL

by

SHASHANK SHARMA

AMAR NATH AND SHASHI KHOSLA SCHOOL OF INFORMATION TECHNOLOGY

Submitted

in fulfillment of the requirements of the degree of **Doctor of Philosophy**

to the

INDIAN INSTITUTE OF TECHNOLOGY DELHI

AUGUST 2023

DEDICATED TO

Abhishek Sharma

Certificate

This is to certify that the thesis titled **DE-NOVO ASSEMBLY OF SHORT READS IN MINIMAL OVERLAP MODEL** being submitted by **Mr. SHASHANK SHARMA** for the award of **Doctor of Philosophy** in Information Technology is a record of bonafide work carried out by her under my guidance and supervision at the **Amar Nath and Shashi Khosla School of Information Technology, Indian Institute of Technology Delhi**. The work presented in this thesis has not been submitted elsewhere, either in part or full, for the award of any other degree or diploma.

Kolin Paul
Professor
Department of Computer Science and Engineering
Indian Institute of Technology Delhi
New Delhi - 110016

Acknowledgements

The thesis would not be in its current shape if not for the efforts of Chinmay, Nirwan and Dhanak. Chinmay painstakingly reviewed the entire thesis and made numerous suggestions for its improvement. He was involved in my PhD from the beginning and was very kind in discussing ideas and sharing his experiences about the research. With Nirwan and Dhanak, I experienced the joy of the writing process. They explained the philosophical aspect of the research, thereby defining the *Abstract, Introduction* and *Future and Conclusion* chapters of the thesis. My sincere thanks to all of you for your constant support.

I want to thank Vijay for formatting the whole thesis in a few hours. The figures and tables would neither be at correct places nor be visible without his help.

I would also like to thank Prof. Naveen Garg, Prof. Parag Singla, Dinesh Khandelwal and Janamjay Kumar for giving me perspective while facing the toughest test in my PhD. I will be forever grateful for the clarity that you provided.

I would also like to acknowledge the *High Performance Computing* group for providing an excellent computing infrastructure to run all kinds of experiments necessary for the work.

The talks hosted by the department were a constant source of motivation and intellectual stimulation. I have also enjoyed the series *Reflections on Nobel Prizes* hosted by the institute yearly.

I am extremely grateful to Rajesh Sir, Suresh, Arun Sir, Hemant and Rekha Ma'am for the support they provided. You all are the backbone of the department.

I am fortunate to make wonderful friends at IIT. I want to thank Chinmay, Janamjay, Yashwani, Devesh, Ravi, Anuj, Rajinder, Abhimanyu, Dinesh, Shibashis, Symantak, Rijurekha, Anamitra, Anoop, Suvam, Britty, Ankur, Navdeep, Vikas, Rajeshwar, Abhishek, Amit, Gaurav, Namita, Parul, Divyanshu, Nikhil, Prashant, Jatin, Rajesh, John, Gulshan and Vijay for their constant support. I apologise if I have forgotten anyone.

Without the emotional support provided by my family, the PhD was a distant dream. I want to

thank my parents and sister for their courage, strength and patience in tough times. I thank my niece, *Poorvisha*, for all her smiles. They fill my heart. I would also like to thank my friend Neeraj for looking after the family in my absence.

Finally, I borrow from Gabriel García Márquez to express my immense gratitude for the problem herself.

"She knew that he loved her above all else, more than anything in the world, but only for his own sake."
—Love in the Time of Cholera by **Gabriel García Márquez**

Shashank Sharma

Abstract

Next Generation Sequencing (NGS) Technologies are relatively high throughput as they produce millions of short reads compared to First Generation Sequencing Technologies that produce relatively fewer but longer reads at orders of magnitude higher cost. Development of new algorithms that can cohere millions of NGS reads while coping up with the sheer volume of the data is presently required to fully harness the potential of NGS technologies for societal benefit, for example, by analyzing sequencing data for predicting, diagnosis and treatment of diseases through personalized medicines and genetic modifications of plants to produce disease-resistant crops.

One of the main applications of NGS in microbiology is the identification of microorganisms such as bacteria, viruses, etc., in a sample. However, a large number of microorganisms have not yet been discovered and hence, their reference genome is not available. Therefore, assembling the entire genome *de novo* remains the most difficult problem for biologists and bioinformaticians. While NGS reads present new challenges because of their quantitative and qualitative nature, they provide new opportunities for assembling and identifying new organisms. Thus, the high throughput provided by NGS technologies can be argued to be an advantage from a computational perspective. In this thesis, it is postulated that the complete overlap information of a NGS read is not required for performing assembly. It is expected that comparable or improved results may be obtained by processing a limited amount of overlap information. Moreover, working with small overlaps avoids time-consuming string operations. This becomes especially relevant when dealing with mammalian size genomes size as human or mouse and large repeat rich model plant genomes like Arabidopsis thaliana. This thesis presents a novel model for assembling NGS reads and an assembler based on it.

In Chapter 2, a novel assembly model based on the above hypothesis is developed and an assembler that works in this model is also developed. Additionally, assembly of ideal reads which are error-free, forward oriented, uniformly distributed over the genome was performed. The results indicated that the hypothesis was valid for ideal data.

In Chapter 3, the problem of error correction in NGS reads is addressed on real datasets. An algorithm tuned to the requirements of the assembler (developed in Chapter 2) was developed. The results indicate improved or comparable performance to contemporary algorithms on real datasets obtained from NGS.

In Chapter 4, the error correction algorithm developed in Chapter 3 is used to correct errors in real NGS reads. The assembly algorithm developed in Chapter 2 is extended to handle corrected NGS reads

of 50 bp (base pair) length. This is followed by extending the assembly algorithm to work with reads of 75 and 100 bp lengths. We also showed that our assembly algorithm is able to handle NGS dataset for large repeat rich model plant genome Arabidopsis thaliana. This indicate that the ideas proposed in the thesis, as they are, can handle genome of such a scale.

Finally, in Chapter 5, the problem of unknown orientations of NGS reads is addressed. For this, a force-based labelling algorithm was developed that assigned orientations to the assemblies and their constituent reads using the assembly overlap graph of the assemblies reported in Chapter 4. Lastly, the assembler was used to generate fewer and longer assemblies using these labelled assemblies. The results indicate that the hypothesis was valid for NGS data.

In conclusion, this thesis reports a novel computational method to analyze datasets with high redundancy. The ideas presented here may also apply not only in genomics but also in problems dealing with a large amount of redundant data.

संक्षेप

नेक्स्ट जेनरेशन सीक्वेंसिंग (एनजीएस) प्रौद्योगिकियां अपेक्षाकृत उच्च थ्रूपुट हैं। वे फर्स्ट जेनरेशन सीक्वेंसिंग टेक्नोलॉजीज की तुलना में लाखों कम रीड्स उत्पन्न करते हैं जो अपेक्षाकृत कम लेकिन उच्च लागत के ऑर्डर पर लंबे समय तक रीड्स उत्पन्न करते हैं। नए एल्गोरिदम का विकास जो सामना करते समय लाखों एनजीएस रीड्स को सुसंगत कर सकता है वर्तमान में क्षमता का पूरी तरह से उपयोग करने के लिए डेटा की विशाल मात्रा की आवश्यकता है। सामाजिक लाभ के लिए एनजीएस प्रौद्योगिकियां, उदाहरण के लिए, व्यक्तिगत दवाओं और आनुवंशिकी के माध्यम से बीमारियों की भविष्यवाणी, निदान और उपचार के लिए अनुक्रमण डेटा का विश्लेषण करके रोग-प्रतिरोधी फसलें पैदा करने के लिए पौधों में संशोधन। सूक्ष्म जीव विज्ञान में एनजीएस के मुख्य अनुप्रयोगों में से एक नमूने में बैक्टीरिया, वायरस आदि जैसे सूक्ष्मजीवों की पहचान करना है। हालाँकि, बड़ी संख्या में सूक्ष्मजीवों की अभी तक खोज नहीं हुई है और इसलिए, उनका संदर्भ जीनोम नहीं है उपलब्ध। इसलिए, संपूर्ण जीनोम डे नोवो को असेंबल करना सबसे कठिन बना हुआ है जीवविज्ञानियों और जैव सूचना विज्ञानियों के लिए समस्या। जबकि एनजीएस नई चुनौतियाँ प्रस्तुत करता है । अपनी मात्रात्मक और गुणात्मक प्रकृति के कारण, वे नए अवसर प्रदान करते हैं नए जीवों को इकट्ठा करना और उनकी पहचान करना। इस प्रकार, एनजीएस द्वारा उच्च थ्रूपुट प्रदान किया जाता है कम्प्यूटेशनल परिप्रेक्ष्य से प्रौद्योगिकियों को एक लाभ होने का तर्क दिया जा सकता है। में इस थीसिस में, यह माना गया है कि एनजीएस रीड की पूरी ओवरलैप जानकारी नहीं है असेंबली करने के लिए आवश्यक है. उम्मीद है कि परिणाम तुलनीय या बेहतर हो सकते हैं ओवरलैप जानकारी की सीमित मात्रा को संसाधित करके प्राप्त किया जा सकता है। इसके अलावा, काम कर रहे हैं छोटे ओवरलैप के साथ समय लेने वाली स्ट्रिंग ऑपरेशंस से बचा जाता है। ये खास हो जाता है मानव या चूहे और बड़े जैसे स्तनधारी आकार के जीनोम के आकार से निपटने के दौरान प्रासंगिक एराबिडोप्सिस थालियाना जैसे समृद्ध मॉडल पादप जीनोम को दोहराएं। यह थीसिस एक उपन्यास प्रस्तुत करती है एनजीएस रीड्स को असेंबल करने के लिए मॉडल और उस पर आधारित एक असेंबलर।

अध्याय 2 में, उपरोक्त परिकल्पना पर आधारित एक नवीन असेंबली मॉडल विकसित किया गया है इस मॉडल में काम करने वाला एक असेंबलर भी विकसित किया गया है। इसके अतिरिक्त, आदर्श की विधानसभा पढ़ता है जो त्रुटि रहित, आगे उन्मुख, जीनोम पर समान रूप से वितरित होता है प्रदर्शन किया। परिणामों ने संकेत दिया कि परिकल्पना आदर्श डेटा के लिए मान्य थी।

अध्याय 3 में, एनजीएस रीड्स में त्रुटि सुधार की समस्या को वास्तविक रूप से संबोधित किया गया है डेटासेट असेंबलर की आवश्यकताओं के अनुरूप एक एल्गोरिदम (अध्याय में विकसित)। 2) का विकास किया गया। परिणाम एनजीएस से प्राप्त वास्तविक डेटासेट पर समकालीन एल्गोरिदम के बेहतर या तुलनीय प्रदर्शन का संकेत देते हैं।

अध्याय 4 में, अध्याय 3 में विकसित त्रुटि सुधार एल्गोरिदम का उपयोग सही करने के लिए किया जाता है वास्तविक एनजीएस रीड्स में त्रुटियाँ। अध्याय 2 में विकसित असेंबली एल्गोरिदम का विस्तार किया गया है 50 बीपी (बेस पेयर) लंबाई के संशोधित एनजीएस रीड्स को संभालें। इसके बाद विस्तार किया जाता है 75 और 100 बीपी लंबाई की रीडिंग के साथ काम करने के लिए असेंबली एल्गोरिदम। हमने यह भी दिखाया कि हमारा असेंबली एल्गोरिदम बड़े रिपीट

रिच मॉडल के लिए एनजीएस डेटासेट को संभालने में सक्षम है पादप जीनोम एराबिडोप्सिस थालियाना। इससे पता चलता है कि थीसिस में प्रस्तावित विचार, जैसे वे हैं, ऐसे पैमाने के जीनोम को संभाल सकते हैं।

अंत में, अध्याय 5 में, एनजीएस रीड्स के अज्ञात अभिविन्यास की समस्या का समाधान किया गया है। इसके लिए, एक बल-आधारित लेबलिंग एल्गोरिदम विकसित किया गया था जो कि ओरिएंटेशन निर्दिष्ट करता था असेंबली और उनके घटक असेंबली के असेंबली ओवरलैप ग्राफ़ का उपयोग करके पढ़ते हैं। अध्याय 4 में बताया गया है। अंत में, असेंबलर का उपयोग कम और अधिक समय उत्पन्न करने के लिए किया गया था इन लेबल वाली असेंबलियों का उपयोग करने वाली असेंबलियाँ। परिणाम दर्शाते हैं कि परिकल्पना थी एनजीएस डेटा के लिए मान्य।

निष्कर्ष में, यह थीसिस डेटासेट का विश्लेषण करने के लिए एक नवीन कम्प्यूटेशनल विधि की रिपोर्ट करती है उच्च अतिरेक के साथ। यहां प्रस्तुत विचार न केवल जीनोमिक्स में भी लागू हो सकते हैं बड़ी मात्रा में अनावश्यक डेटा से निपटने में भी समस्याएँ आती हैं।

Contents

Certificate **i**

Acknowledgements **iii**

Abstract **v**

1 Introduction **1**

1.1 DNA and its significance 1

1.2 DNA sequencing technologies 2

1.3 Chain termination method and first generation sequencing technologies . . 2

1.4 Pyrosequencing and next generation sequencing technologies 4

1.5 Problem definition 4

1.6 A brief history of sequence assembly 5

1.6.1 Greedy approach 5

1.6.2 Overlap–Layout–Consensus approach 6

1.6.3 de Bruijn graph approach 7

1.6.4 String graph approach 8

1.7 Challenges and opportunities in assembling next generation sequencing reads 10

2 *De novo* assembly of ideal reads **13**

2.1 Introduction 13

2.2 Overlap graph construction 14

2.2.1 Coverage requirement of the overlap graph 16

2.3 Similarity hypothesis for short reads 18

2.3.1 Motivating similarity using an example 18

2.3.2 Similarity 18

2.3.3 Anchor nodes 19

2.3.4 Properties of similarity relation 19

2.3.5 Experimental validation of similarity hypothesis 21

2.3.6 Characterizing true overlaps using similar sets . . . 26
2.3.7 Strategy to combine similar sets using common reads . . . 27
2.4 Bushy structures in an overlap graph . . . 31
2.4.1 Relation between bushy structure and similar sets . . . 33
2.5 Clustered graph: data structure to store bushy structures . . . 33
2.5.1 Representing multiple bushy structure using a clustered graph . . . 36
2.5.2 Random overlaps dropped while building clustered graph . . . 37
2.6 Algorithm for building the clustered graph from an overlap graph . . . 39
2.6.1 Algorithm to construct groups from children nodes . . . 42
2.6.2 Check Consistency . . . 43
2.6.3 Create Group . . . 45
2.6.4 Selecting child groups . . . 47
2.7 Generating assemblies . . . 48
2.8 Results . . . 51
2.9 Conclusions . . . 54

3 Error correction of NGS reads 57
3.1 Introduction . . . 57
3.2 k-mer based error correction . . . 59
3.2.1 BLESS [30] . . . 61
3.2.2 MUSKET [52] . . . 63
3.2.3 LIGHTER [100] . . . 64
3.2.4 BFC [46] . . . 64
3.3 Multiple sequence alignment based . . . 65
3.3.1 CORAL [84] . . . 65
3.4 Error correction algorithm . . . 66
3.5 Computing threshold λ to validate an overlap . . . 69
3.6 Results . . . 71
3.6.1 Results on simulated datasets . . . 71
3.6.2 Results on real datasets . . . 80
3.7 Conclusions . . . 84

4 *De novo* assembly of NGS reads 87
4.1 Introduction . . . 87
4.2 Experimental validation of similarity hypothesis for simulated erroneous reads and real data . . . 89
4.3 Assembling reads of length 50 bp . . . 95

4.4 Assembling reads of length greater than 50 bp 98
4.4.1 Splitting long reads into sub-reads to capture long overlaps 100
4.5 Modifications in *MOBS* to assemble with long NGS reads 101
4.5.1 Merging nodes on the basis of common reads 102
4.5.2 Merging nodes on the basis of overlaps among sub-reads 106
4.6 Results 108
4.6.1 Analysis of the misassemblies for the read length 50 bp 110
4.6.2 Analysis of the misassemblies for the read length greater than 50 bp 111
4.7 Assembling Arabidopsis thaliana using *MOBS* 114

5 Orientation problem of NGS reads 117
5.1 Introduction 117
5.2 Assembly overlap graph construction 119
5.2.1 Error model for inexact overlaps 119
5.2.2 Computing overlaps among assemblies 122
5.3 Force-based labelling algorithm 122
5.4 Results 124

6 Conclusion 127
6.1 Contributions 127
6.2 Limitations of the work 128
6.3 Future work 129

Appendix 145

List of Publications 151

Biography 153

List of Figures

1.1 Overlap graph and de Bruijn graph [92] 7

2.1 Read u and v make a true overlap and u and w make a false overlap. . . . 14
2.2 Read u make 3 true overlaps with v, w and x. The length of these overlaps is different. 15
2.3 Coverage 5× 17
2.4 Coverage 10× 17
2.5 Coverage 15× 17
2.6 Coverage 20× 17
2.7 Coverage 25× 17
2.8 Coverage 30× 17
2.9 Coverage 35× 17
2.10 Reads u and v and the corresponding similar reads a and b. The corresponding overlap graph is also shown. 18
2.11 Anchor reads u and v and the corresponding similar reads. 20
2.12 Reads u and v are similar and reads v and w are similar but u and w are not similar as the length of inferred overlap is 10 bp. 20
2.13 Frequency distribution of out-degree of the left anchor nodes. 27
2.14 A set of similar reads with two anchor reads w and x at each ends. Red colored reads are the new nodes found similar to c, d and e. 28
2.15 Combining anchors in Figure 2.11 and 2.14, we get a larger set of similar reads. 29
2.16 S_1 can be combined with both S_2 and S_3 using common read c. But S_2 and S_3 give different extensions to similar string of S_1. 30
2.17 **Alternate paths and bushiness:** The horizontal line segments represent reads. Arrows represent an overlap between reads. There are multiple alternate paths between source and destination; three of them are depicted. 31

2.18 Bushy structure (shown in blue) in an overlap graph between nodes 1 and 18. Dotted edges are random overlaps and Bold edges are true overlaps. Assuming all the reads are of length 50 bp, all paths between node 1 and 18 corresponds to assemblies of length 202 bp. 32
2.19 Hierarchical View of bushy structure. Nodes on same level are enclosed in dashed rectangle. Nodes at each level are ordered as per the overlap they make with parent group. 34
2.20 Read equivalent of a group. We can see the order in the reads. 35
36figure.caption.81
2.22 Bushy structure and the corresponding clustered graph is shown. The weight on the edges of clustered graph represents inferred overlap length between last read of parent group and first read of child group. 36
2.23 Two bushy structures and the corresponding clustered graph is shown. Both the structures merge at group def. 37
2.24 Clustered Graph showing the contents of a node and the parent child relationship. Last read of parent similar set makes a suffix-prefix overlap of 49 bp with first read of child similar set. 38
2.25 Read A and B form a group. Read D makes an random overlap with A but do not overlap with B. C makes a true overlap with A and B. 38
2.26 Children reads of node u are mapped to genome. Three of them map to same region and other two map to different region. Red colored portion represents small repeat of say length 18 bp. 40
2.27 First level of BFS exploration. 40
2.28 Grey nodes 7, 8, 9 and 10 are common children of nodes 2,3,4. So nodes 2,3 and 4 are clustered. 41
2.29 Similarly 7, 8, 9 and 10 are clustered on the basis of common children 13 and 14. In the next iteration, 13 and 14 will be clustered and we will have a clustered graph. 41
2.30 **Repeat Exit:** Read A is the last read of the repeat and should be added to both the groups. 42
2.31 Consider the maximum confident group g' that contains largest number of reads. The heuristic checks if it is possible to combine reads of a smaller group g with some reads of a g' and form a bigger group. We pick the first read of g and check if it is consistent with some read r of g'. If such a read r exists, then we add all the reads of g' before (and including) r to g, thus forming a larger group. 48
2.32 Assembly Generation 50

3.1 Consider Read A, B and C with sequencing error in B marked as a red colored cross. The corresponding character in Read C is denoted by red bar. Due to this error, the overlap relationship between Read A and B is not captured. On the other hand, a false overlap relationship is established between Read C and B. Overlap Graph capturing the relationships among reads is shown. Dashed edge represents the overlap that was concealed due to the presence of sequencing error in Read B and solid edge represents the false overlap caused by same sequencing error. 58

3.2 Sequencing errors in reads are marked as a red colored cross and the corresponding correct base is denoted as blue colored check mark. Sequencing error can be corrected with the help of correct bases from the neighboring reads. Chances of same error occurring in all the reads simultaneously is low. Notice that frequency of errors is high towards the end of the reads. . 59

3.3 Sequencing errors in reads are marked as a red colored cross. The blue bars are k-mers of reads. As you can see the presence of a sequencing error in first k-mer makes it different from the corresponding k-mers of the following reads. So the total number of occurrences of the k-mer with error is 1 and the number of occurrences of k-mers which are error free is 5. Compare this situation to the case in which the error in the k-mer of first read does not occur. Then there will be on only one k-mer and its number of occurrences or frequency will be 6 instead of 5. 60

3.4 Black line represents a read and the sequencing error is marked as a red colored cross. The blue lines represents k-mer derived from the read. As you can see the presence of a sequencing error in the read generates k erroneous k-mers. In general, a sequencing error occurring at position i of a read of l bases creates $min\{k, i, l-i+1\}$ erroneous k-mers. Using solid k-mers sequencing error can be localized in a read as one can check (starting from leftmost k-mer) whether the k-mer of reads is solid or weak using bloom filter. The first weak k-mer found has the last base as potentially erroneous. 60

3.5 Black lines are reads, blue lines are k-mers, crosses are sequencing errors and tick marks are correct bases. Consider read A whose last two bases are erroneous and let read C truly overlap A in genome and hence C's k-mers corresponds to correct extension of A. Let read B be another read which although does not overlaps with A in genome but have two solid k-mers that may constitute alternative extension for A. As per this alternative extension provided by B only second last base of A is erroneous and last base is correct as it not a sequencing error in B (red tick in B). If the extension provided by read B is taken then last base of A will remain erroneous. 62

3.6 Black lines are reads, blue lines are k-mers, crosses are sequencing errors and tick marks are correct bases. Zig zag portion of read B represents high difference between A and B. Consider read A whose last base is erroneous and let read C truly overlap A in genome and hence C's k-mers corresponds to right correction (blue tick) for A. Let read B be another read which although does not overlaps with A in genome but have a solid k-mers that may alternative correction (green tick) for A. Due to the presence of two possibilities for correction last base of A will be termed ambiguous but it was possible to resolve it if MUSKET have interpreted the situation in terms of reads and not k-mers. 62

3.7 Consider Reads A and B and sequencing errors of Read A's suffix is to be corrected. Sequencing errors are marked as a red colored cross. The blue bar represents the portion of read A matched with blue bar region of read B using look up table. Note that the look up is performed with 2 mismatches in Read B. The unmatched region of the two overlapping reads is matched using standard dynamic programming based method. Threshold on the number of errors allowed λ, is computed using *Chebyshev*'s inequality. If the threshold is not crossed, then the overlap is retained and Read B becomes the part of neighboring reads set. 68

3.8 Sequencing errors of Read A's suffix is to be corrected. Sequencing errors are marked as a red colored cross. The number of the overlapping reads are quite less, 2 in this case. So Read A is dropped as sufficient coverage required for correction is not found. 68

4.1 Consider Read A and B with Read A sequenced from the forward strand of the DNA and Read B sequenced from the reverse strand of the DNA. The direction of sequencing for Read A will be from left to right (that is in the direction of arrow). The direction of sequencing for Read B will be from right to left. So the sequenced version of Read A and B will not share an overlap though they do overlap in genome itself. Due to this unknown relative orientation of A and B, the overlap relationship between them is not captured (missing edge is shown as dashed). 88

4.2 Case A: Mismatch between reads R2 and R4 is highlighted. R2 and R4 share same length overlap with R1. Hence the overlap between R2 and R4 is equal to the read length. Case B: The overlap between R2 and R4 is less than the read length. 95

4.3 Sequencing errors in reads are marked as a red colored cross. After correction, the ends of the reads are clean. Read v have an error in the middle region. When a group is formed it will contain all these reads. But due to the presence of the error it will be marked as invalid. 96

4.4 Black line inside a node represents constituent reads and the read r common to both nodes u and v is colored using blue. The constituent reads of w, that is, $r_1, r_2, r_3, r_4 \ldots r$ are merged to node v with their original shifted configuration in w is maintained in v after merging. The original shifted configuration of constituent reads of v is also maintained after merging. . . 102

4.5 Consider a Read A split into two smaller overlapping pieces A_1 and A_2. After Clustered graph generation process, let A_1 be a constituent read of node u and A_2 be a constituent read of node v. Since we know the positions of A_1 and A_2 in original read A, we can use this information to find relative shift between node strings u_s and v_s. 103

4.6 Black line inside a node represents constituent reads and the node string is drawn just below the node as thick line. The read r common to both nodes is colored using blue and is used to compute offset between two node strings. The overlapping portion of the two node strings u_s and v_s is compared and if the number of mismatches exceeds a certain threshold (chosen 5) the u_s and v_s are declared as not similar else they are declared similar. 104

4.7 Consider nodes u and v of Clustered Graph that share a child node w. The edge weight k denote the length of the suffix-prefix overlap between node strings u_s and w_s. Similarly edge weight l denotes the length of the suffix-prefix overlap between node strings v_s and w_s. On the basis of k and l, the relative shift between u_s and v_s can be easily found. Similarly the relative shift between u_s and v_s can be found if they share common parent. 104
4.8 Various merging cases that may arise 105
4.9 **Tandem Repeats:** As the reads spans a tandem repeat region, we can see there are two different ways of arranging these reads to form a group of a bushy structure. 111
4.10 Black line represents an assembly and the blue lines represents the constituent reads. As you can see a large offset of length k between two consecutive reads A and B. We call the largest offset between two consecutive reads as maximum relative offset. If the value of the maximum relative offset if greater than some experimentally chosen threshold than it naturally leads to a cut-point. A cut point is defined as the starting position of the second read participating in maximum relative offset in the assembly, in this case B and the cut point is i. 112
4.11 Misassembly formation due to reverse complement repeat in genome. . . . 113

5.1 Overlap not captured due to opposite orientation of reads. 118
5.2 Consider a read u which is a part of two different assemblies lying in two different components (shown in red and blue). The assemblies are not shown. Our procedure assigns $forward$ label to u in red component and this label is forced on blue component due to u. Similarly read u's reverse complement, read u^{RC}, is part of an assembly which lies in purple component. So label $reverse$ is forced on the purple component. 123
5.3 Conflict Case: Consider a read u from which our labelling algorithm starts and the $forward$ label reaches purple component and further propagated beyond it. Now consider the $reverse$ label being propagated due to u^{RC} and the purple component receives $reverse$ label and hence it is marked as *conflicted*. 124

List of Tables

2.1 H. acinonychis: Similar set size distribution at various coverages level . . . 21
2.2 H. acinonychis: False out-degree distribution of the left anchors at coverage level 5× . . . 22
2.3 H. acinonychis: False out-degree distribution of the left anchors at coverage level 10× . . . 22
2.4 H. acinonychis: False out-degree distribution of the left anchors at coverage level 15× . . . 22
2.5 H. acinonychis: False out-degree distribution of the left anchors at coverage level 20× . . . 23
2.6 H. acinonychis: False out-degree distribution of the left anchors at coverage level 25× . . . 23
2.7 H. acinonychis: False out-degree distribution of the left anchors at coverage level 30× . . . 23
2.8 H. acinonychis: False out-degree distribution of the left anchors at coverage level 35× . . . 24
2.9 H. acinonychis: Distribution of majority size in a similar set at coverage level 35× . . . 24
2.10 H. acinonychis: Distribution of false and true overlaps within a majority at coverage level 35× . . . 25
2.11 Percentage of true and false overlaps dropped by dropping similar sets of size 2. . . . 27
2.12 Percentage of **random** overlaps removed from overlap graph and remaining overlaps. . . . 39
2.13 Percentage of **invalid** nodes/groups of a clustered graph constructed for various genomes. . . . 42
2.14 Helicobacter acinonychis. . . . 52
2.15 Agrobacterium Tumefaciens. . . . 52
2.16 Arcanobacterium Haemolyticum. . . . 53

2.17 Bacillus Licheniformis. 53
2.18 Helicobacter acinonychis analysis after removal of contained assemblies. . . 54
2.19 Agrobacterium Tumefaciens analysis after removal of contained assemblies. 54
2.20 Arcanobacterium Haemolyticum analysis after removal of contained assemblies. 55
2.21 Bacillus Licheniformis analysis after removal of contained assemblies. . . . 55

3.1 Datasets description with name of the organism, reference genome accession number and NGS reads dataset accession number from Sequence Reads Archive. 72
3.2 D1 72
3.3 H2 72
3.4 H4 72
3.5 M1 73
3.6 M2 73
3.7 M3 73
3.8 M5 73
3.9 M9 73
3.10 Error distribution at the ends of reads post correction by MOBS. 76
3.11 **LIGHTER**: Error distribution at the ends of reads post correction. 77
3.12 **CORAL**: Error distribution at the ends of reads post correction. 78
3.13 **BFC**: Error distribution at the ends of reads post correction. 79
3.14 Running time and Memory Usage of OUR error correction algorithm on simulated datasets 80
3.15 Percentage of reads matched (with **0 errors**) against genome post correction. 81
3.16 Percentage of reads matched (with **1 errors**) against genome post correction. 82
3.17 Percentage of reads matched (with **2 errors**) against genome post correction. 82
3.18 Percentage of reads matched (with **3 errors**) against genome post correction. 83
3.19 Percentage of reads matched (with **4 errors**) against genome post correction. 83
3.20 Number of reads matched exactly by both LIGHTER and MOBS algorithm, by MOBS algorithm only and by LIGHTER only. 84
3.21 Number of reads matched exactly by both CORAL and MOBS algorithm, by MOBS algorithm only and by CORAL only. 84
3.22 Number of reads matched exactly by both BFC and MOBS algorithm, by MOBS algorithm only and by BFC only. 85
3.23 Reads that are matched with 0 errors due to LIGHTER correction (and not by MOBS algorithm) are matched with 1,2,3 and 4 errors. 85

3.24 Reads that are matched with 0 errors due to CORAL correction and (not by MOBS algorithm) are matched with 1,2,3 and 4 errors. 86
3.25 Reads that are matched with 0 errors due to BFC correction and (not by MOBS algorithm) are matched with 1,2,3 and 4 errors. 86

4.1 H. acinonychis: Similar set size distribution of simulated data 89
4.2 H. acinonychis: False Out-degree distribution of the left anchors at coverage level 35× (seeded errors) 89
4.3 H. acinonychis: False Out-degree distribution of the left anchors at coverage level 35× 90
4.4 H. acinonychis (Seeded Errors): Distribution of Majority size in a similar set at coverage level 35× with up to 0 errors allowed in pairwise matching 90
4.5 H. acinonychis: Distribution of Majority size in a similar set at coverage level 35× with up to 0 errors allowed in pairwise matching 91
4.6 H. acinonychis (Seeded Errors): Distribution of Majority size in a similar set at coverage level 35× with up to 1 errors allowed in pairwise matching 91
4.7 H. acinonychis: Distribution of Majority size in a similar set at coverage level 35× with up to 1 error allowed in pairwise matching 91
4.8 H. acinonychis (Seeded Errors): Distribution of Majority size in a similar set at coverage level 35× with up to 2 errors allowed in pairwise matching 92
4.9 H. acinonychis: Distribution of Majority size in a similar set at coverage level 35× with up to 2 errors allowed in pairwise matching 92
4.10 H. acinonychis (Seeded Errors): Distribution of Majority size in a similar set at coverage level 35× with up to 3 errors allowed in pairwise matching 92
4.11 H. acinonychis: Distribution of Majority size in a similar set at coverage level 35× with up to 3 errors allowed in pairwise matching 93
4.12 Similar set size distribution of real dataset for Helicobacter bizzozeronii . . 94
4.13 Real Data: Distribution of Majority size in a similar set with 0 errors allowed in pairwise matching 94
4.14 Real Data: Distribution of Majority size in a similar set with up to 1 errors allowed in pairwise matching 94
4.15 Real Data: Distribution of Majority size in a similar set with up to 2 errors allowed in pairwise matching 94
4.16 Real Data: Distribution of Majority size in a similar set with up to 3 errors allowed in pairwise matching 95

4.17 Distribution of length of the assemblies generated by *MOBS*. *rl* stands for read length and max is the length of longest assembly generated. $3 \times rl$ means number of assemblies with length less than thrice the read length but greater than or equal to twice the read length. 99
4.18 Distribution of length of assemblies after breaking long read into small sub-reads generated by *MOBS*. 100
4.19 Distribution of length of assemblies generated by *MOBS*. 108
4.20 Comparison of different algorithms on dataset Helicobacter bizzozeronii. . . 109
4.21 Comparison of different algorithms on dataset Bacillis subtilis. 109
4.22 Comparison of different algorithms on dataset E. coli a. 109
4.23 Comparison of different algorithms on dataset H2. 110
4.24 Distribution of length of misassemblies generated by *MOBS*. 110
4.25 Number of tandem repeats not covered by *MOBS* assemblies. 111
4.26 Distribution of maximum relative offset of large misassemblies generated by *MOBS*. 112
4.27 Distribution of maximum relative offset of large correct assemblies generated by *MOBS*. 112
4.28 Distribution of position of the cut point for large misassemblies generated by *MOBS*. 114
4.29 Distribution of position of the cut point for large correct assemblies generated by *MOBS*. 114
4.30 Comparison of different algorithms on dataset Arabidopsis thaliana. 115

5.1 Statistics for the error rate and the length of the erroneous region being matched. 121
5.2 Comparison of length of assemblies generated by *MOBS* post orientation assignment with other assemblers. 126
5.3 Comparison of different algorithms on dataset Helicobacter bizzozeronii. . . 126