8-7-2018

# Assembly, quantification, and downstream analysis for high trhoughput sequencing data

Igor Mandric

# ASSEMBLY, QUANTIFICATION, AND DOWNSTREAM ANALYSIS OF HIGH THROUGHPUT SEQUENCING DATA

by

IGOR MANDRIC

Under the Direction of Alexander Zelikovsky, PhD

## ABSTRACT

Next Generation Sequencing is a set of relatively recent but already well-established technologies with a wide range of applications in life sciences. Despite the fact that they are constantly being improved, multiple challenging problems still exist in the analysis of high throughput sequencing data. In particular, genome assembly still suffers from inability of technologies to overcome issues related to such structural properties of genomes as single nucleotide polymorphisms and repeats, not even mentioning the drawbacks of technologies

themselves like sequencing errors which also hinder the reconstruction of the true reference genomes. Other types of issues arise in transcriptome quantification and differential gene expression analysis. Processing millions of reads requires sophisticated algorithms which are able to compute gene expression with high precision and in reasonable amount of time. Following downstream analysis, the utmost computational task is to infer the activity of biological pathways (e.g., metabolic). With many overlapping pathways challenge is to infer the role of each gene in activity of a given pathway. Assignment products of a gene to a wrong pathway may result in misleading differential activity analysis, and thus, wrong scientific conclusions. In this dissertation I present several algorithmic solutions to some of the enumerated problems above. In particular, I designed scaffolding algorithm for genome assembly and created new tools for differential gene and biological pathways expression analysis.

INDEX WORDS: Algorithm, Assembly, Genome scaffolding, Scaffolding evaluation, Read overlap graph, Integer linear programming, Expectation maximization, Maximum likelihood, RNA-Seq, Metabolic pathways

# ASSEMBLY, QUANTIFICATION, AND DOWNSTREAM ANALYSIS FOR HIGH THROUGHPUT SEQUENCING DATA

by

IGOR MANDRIC

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2018

# ASSEMBLY, QUANTIFICATION, AND DOWNSTREAM ANALYSIS FOR HIGH THROUGHPUT SEQUENCING DATA

by

IGOR MANDRIC

Committee Chair: Alexander Zelikovsky

Committee: Robert Harrison
Pavel Skums
Ion Măndoiu
Artem Rogovskyy

Electronic Version Approved:

Office of Graduate Studies
College of Arts and Sciences
Georgia State University
August 2018

## DEDICATION

To my parents and my wife Rimma Mandric (Cara).

# ACKNOWLEDGEMENTS

I would like to thank my scientific advisor Dr. Alex Zelikovsky for his wise guidance through my Ph.D. journey. His encouragement and support were very valuable during all these years. My passion and interest for bioinformatics are truly indebted to him.

I would also like to thank the Ph.D. committee members Dr. Pavel Skums, Dr. Ion Mandoiu, Dr. Artem Rogovskyy, and Dr. Robert Harrison.

I am very grateful to Dr. Rajsekhar Sunderraman, Dr. Xiaojun Cao, and Mrs. Tammie Dudley. They offered me invaluable help and support in many difficult situations.

Not of less importance for me was the collaboration with my colleagues. I would like to thank my older fellows Dumitru Brinza and Serghei Mangul. The latter was for me a tremendous example of energy and optimism. I am very happy I was a part of the same scientific group with my lab-mates Ekaterina Nenastyeva, Nick Mancuso, Olga Glebova, Pelin Icer, Sergey Knyazev, Andrew Melnyk, and others.

I would like to express my gratitude to the Faculty of Mathematics and Informatics of Moldova State University and especially to my Master advisor Dr. Dumitru Lozovanu. I can not help but mention my best friend Radu Buzatu who is also now a faculty member at the same university.

Special thanks to my wife Rimma Mandric (Cara) and my parents Elena and Alexandr. Their constant love, support, and motivation made me stronger during the climb for my Ph.D. degree.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

- ORF - Open Reading Frame
- PPV - Positive Predictive Value
- TPR - True Positive Rate
- ILP - Integer Linear Programming
- BCR - B Cell Receptor
- TCR - T Cell Receptor
- NGS - Next Generation Sequencing
- HTS - High Throughput Sequencing
- CDR3 - Complementarity-Determining Region 3
- EM - Expectation Maximization
- FPKM - Fragments Per Kilobase per Million of reads

# PART 1

# INTRODUCTION

One of the most important and challenging biological tasks has been discovery and deciphering of the human genetic code [17]. To understand the process of life, one needs to determine the sequence of the four bases - adenine, guanine, cytosine, and thymine (A, G, C, T) which make up the human genome. One of the earliest technologies for DNA sequencing is so-called Sanger sequencing [78] based on specific chain-terminating inhibitors of DNA polymerase. Its main advantage is very low error rate [92], but it is not practical due to its high costs. Thus, The Human Genome Project (HGP), used a newer technology of *shotgun sequencing* (i.e. shearing DNA into multiple random pieces and then assembling into the genome sequence) which is considerably cheaper than the Sanger sequencing but poses challenging computational problems requiring a lot of resources.

A new era in life sciences has begun with the advent of the Next Generation Sequencing (NGS) technologies: 454, Illumina, SOLiD, Ion Torrent. These new technologies revolutionized the field of bio- and medical sciences, in particularly, bioinformatics, as they allowed generating millions of high quality reads (single- and paired-end) with a significant drop of cost per base pair. The new technologies posed new problems, but at the same time, they allowed expanding the range of bioinformatics applications which can leverage the massive data produced by them: variant calling, discovery of germline and somatic mutations, etc.

After RNA-Seq (transcriptome sequencing) technology emerged, it became a routine task for the researchers to obtain NGS read data from mRNA. Besides the arise of new computational problems, new very important biological applications have become possible: gene expression analysis, immune repertoire discovery, etc. Of the utmost importance for life sciences is the study of the biochemical processes which occur in living organisms. In this dissertation, RNA-Seq as the main tool is used to study the abundance of biological

pathways which represent networks of biochemical reactions.

As the Central Dogma [16] of molecular biology assumes the informational flow from DNA to proteins, most of the bioinformatics pipelines start with analyzing NGS reads and end up with downstream analysis like differential gene expression or estimation of biological pathway activity. This dissertation presents multiple algorithmic contributions on all three informational levels of the Central Dogma unified into one bioinformatics pipeline for NGS data analysis.

## 1.1 Background

In this section, we provide the description of most widely spread Next Generation Sequencing technologies and importance of their multiple applications and also highlight the structure of the bioinformatics pipeline which serves as the skeleton unifying the contributions of this dissertation.

### 1.1.1 NGS technologies

The importance of Next Generation Sequencing technologies cannot be underestimated because they allow for rapid advances in life sciences. They are used in a very large set of applications, the most important of which are [35]:

- Resequencing of human genome for discovery of genes and regulatory elements involved in different diseases;
- Comparative biology studies through whole-genome sequencing of multiple species;
- Sequencing of bacterial and virus species for public health and epidemiological studies;
- Gene expression studies through RNA-Seq technologies etc.

There are different NGS platforms which have their own strategies to generate reads and therefore have different read lengths and throughput: Roche, LifeTechnologies, Illumina, Pacific Biosciences, Helicos. In the Table 1.1 the main characteristics are provided [13].

Table 1.1 NGS technologies and their characteristics [13].

| Sequencing Platform | | Sequence by | Run types | Run time | Read length | Reads per run | Output per run |
|---|---|---|---|---|---|---|---|
| Roche | GS FLX Titanum XL+ | Synthesis | Single end | 23h | 700 | 1 M | 700 Mb |
| | GS Junior System | Synthesis | Single end | 10 h | 400 | 0.1 M | 40 Mb |
| Life-Technologies | Ion Torrent | Synthesis | Single end | 4h | 200-400 | 4 M | 1.5-2 Gb |
| | Proton | Synthesis | Single end | 4h | 125 | 60-80 M | 8-10 Gb |
| | Abi/solid | Ligation | Single end & paired-end | 10d | 75 + 35 | 2.7 B | 300 Gb |
| Illumina | HiSeq | Synthesis | Single end & paired-end | 12d | 2 x 100 | 3 B | 600 Gb |
| | MiSeq | Synthesis | Single end & paired-end | 65h | 2 x 300 | 25 M | 15 Gb |
| PacBio | RSII | Single molecule synthesis | Single end | 2d | 10 K | 0.8 M | 5 Gb |
| Helicos | Heliscope | Single molecule synthesis | Single end | 10d | 30 | 500 M | 15 Gb |

To analyze the massive datasets produced by NGS platforms researchers have to design specialized pipelines for processing raw reads and getting particular bioinformatics results. In this dissertation, the main focus is on the pipeline which leads from NGS reads to biological pathways.

### 1.1.2 Bioinformatics Pipeline for NGS Data Analysis

The bioinformatics pipeline which is the main topic of this dissertation is presented in Figure 1.1. It consists of 6 stages: 1) Sample preparation; 2) Sequencing; 3) Assembly; 4) Post-assembly; 5) Quantification; 6) Further downstream analyses (pathways, immune repertoire profiling). The main contributions refer to the stages 3-6 and are depicted in red.

From a biological sample, NGS DNA and RNA-Seq reads are obtained (stages 1-2). Then, the reads are passed through the assembly processing workflow (stage 3). As a result, DNA reads are assembled into a set of contiguous DNA sequences called *contigs* and RNA-Seq reads are assembled into so-called *transcripts* (i.e., units of transcription). In case if the sample belongs to a model organism (or there is already a reference with a known set of genes), this workflow is not necessary. Additionally, for a targeted RNA-Seq analysis when the reads come from transcripts corresponding to a specific genomic region, a more specific type of assembly may be carried out (for example, CDR3 clonotype assembly). The stages

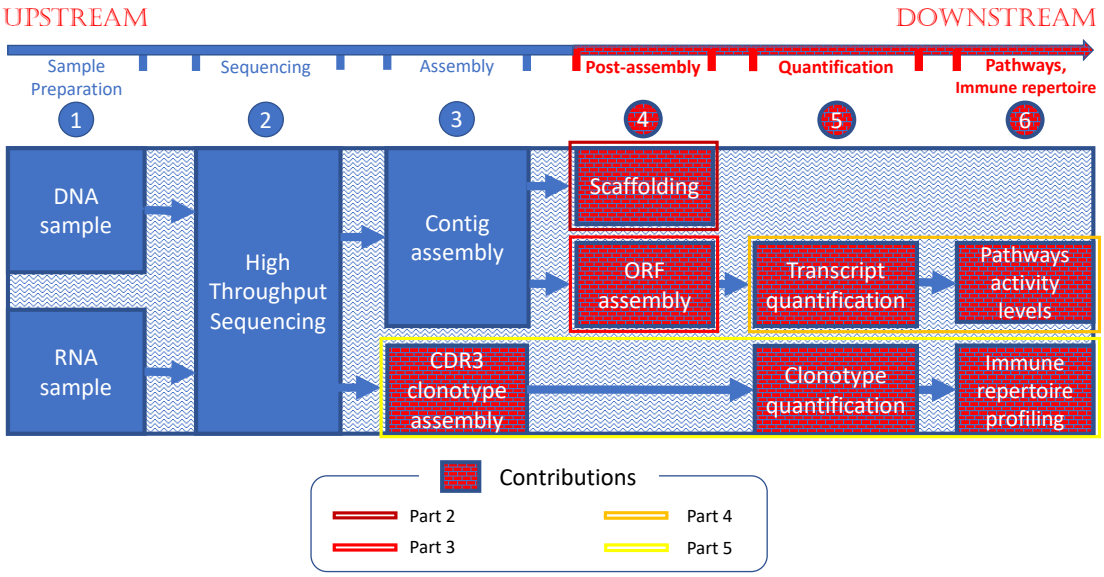



Figure 1.1 The bioinformatics pipeline for NGS data analysis. Compartments drawn in red correspond to the main contributions of this dissertation. Colored frames delimitate group of compartments corresponding to different parts of this dissertation (for example, the main focus of Part 2 is scaffolding).

1-3 described above are referred to as *upstream* analyses and are well studied.

Next three stages referred to as *downstream* analyses constitute the core of this dissertation. In the stage 4 (post-assembly), DNA contigs are joined into chains where each contig is ordered and oriented. This process is referred to as *scaffolding*. Also, RNA sequences corresponding to transcripts without a full open reading frame (ORF) are assembled further to contain a full ORF coding for a protein. In the stage 5, transcripts and CDR3 clonotypes are undergone a *quantification analysis*. In this step, the relative abundance for each transcript/clonotype is determined. In the utmost stage of our pipeline, stage 6, further downstream analyses are performed (such as pathways activity levels or immune repertoire profiling).

## 1.2 Problems

In this dissertation, several bioinformatics problems which arise in the context of the main pipeline described in Section 1.1.2 are solved.

**Scaffolding Problem.** Next Generation Sequencing (NGS) technologies (Illumina, Ion Torrent) produce reads that are several hundreds of nucleotide base pairs long. It is challenging to assemble genomes due to some drawbacks which are still characteristic of these technologies: uneven read coverage, sequencing errors, insufficient read length to span repeated regions in genomes. All these issues hinder the genome assembly problem to be solvable in polynomial time. Thus, overcoming technological issues to allow for better assemblies of different species is still crucial. Several attempts are being made by so-called Third Generation Sequencing technologies (Pacific Biosciences, Oxford Nanopore Technologies) to increase the read length up to several thousand base pairs, but currently they have a very high error rate ($\approx 15\%$ for Pacific Biosciences) compared to NGS (Illumina MiSeq $< 1\%$) and they are still under active development. Therefore, there is still a high demand for developing efficient and scalable assembly algorithms for the genome assembly problem.

Genome assembly pipeline has been traditionally divided into three main stages: assembly, scaffolding, and gap filling. The output of the assembly stage is the set of contiguous DNA sequences called *contigs*. Contig length depends on the assembly tool and the other factors mentioned previously (read length, uniformity of coverage, etc). The set of contigs and the paired-end reads serve as input to the scaffolding stage. The main goal is to join contigs into chains called *scaffolds* to correctly determine the relative orientation, the relative ordering, and the distance between adjacent contigs in each scaffold. So the scaffolding problem in general case can be formulated as follows:

> **Scaffolding Problem.** Given $\mathcal{C}$ - the set of contigs and $\mathcal{P}$ - the set of paired-end reads, build a set of scaffolds $\mathcal{S}$ maximizing the optimization criterion $K$.

Here $K$ may, for example, be the number of correct contig joins. The last stage of genome assembly is gap filling. Here the gaps remaining after the previous step are closed using the reads.

**ORF assembly.** An mRNA transcript consists of several conventional parts: a 5'-cap, a 5'-UTR (untranslated region), an open reading frame (ORF), a 3'-UTR, and a poly-A tail

(see Figure 1.2). The ORF is the most interesting and important transcript part from the biological perspective since it potentially codes for a protein. It begins with a start codon Met (methionine, AUG nucleotide sequence) and ends with a stop codon (either UAA, or UAG, or UGA). State-of-the-art RNA-Seq assemblers like Trinity [34] often output transcripts which do not contain the full ORF and are either missing the 5'-prime, 3'-prime or both ends. The portion of such non-complete transcripts may be considerable resulting in losing much of the information contained in an RNA-Seq sample. The problem of ORF assembly is about having a set of non-complete ORFs extracted from a set of transcripts to join them into as many as possible full ORFs.

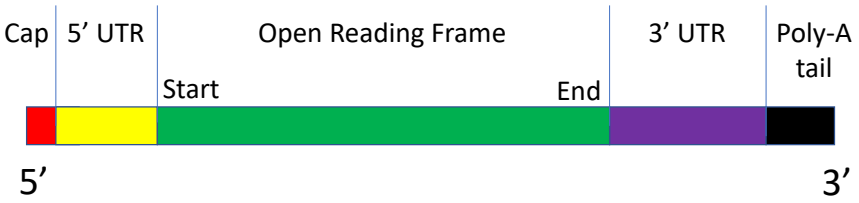



Figure 1.2 The structure of a mRNA transcript. Open reading frame is depicted in green.

**Gene and Isoform Quantification.** Estimating gene and isoform expression levels is a very important bioinformatics problem. Due to recent advances in sequencing technologies, millions of paired-end high-quality RNA-Seq reads can be produced in a usual experiment. The amount of data and the intrinsic properties of transcriptomes (e.g. human) make it a very challenging task. In general, quantification methods are divided into two main categories: count-based (e.g. HTSeq [3]) and FPKM-based (e.g. IsoEM [68], RSEM [51]). Although FPKM-based methods are deemed to usually outperform count-based methods [43] in terms of accuracy, the ultimate goal of any RNA-Seq experiment is to estimate differential gene and/or isoform expression levels. The problem becomes even harder in absence of enough number of replicates.

**Pathway Activity Level Inference.** Pathways can be represented as graphs that use nodes to represent biochemical compounds, with enzymes associated with edges describing biochemical reactions. The expression of the genes coding for enzymes determines the

amount of these enzymes participating in biochemical reactions. As a result, if some enzymes are overrepresented/underrepresented the corresponding reactions occur with more/less intensity, i.e. the activity level of the corresponding pathways increases/decreases. Therefore, in this context, a problem of inference of pathway activity levels from RNA-Seq reads is formulated. One of the main challenges of this problem is the fact that many enzymes work in the context of several pathways, thus creating ambiguities of assignment.

**CDR3 Sequence Assembly.** The adaptive immune system is a very important biological mechanism which eliminates and prevents the growth of pathogens in the host organism. The immune response is carried out by the two main cell types - B cells and T cells. They secrete antibodies known as immunoglobulins. The part of the antibody known to serve as a binding site to antigenes is highly variable and is called Complementary Determining Region 3 (CDR3). The hypermutations and the genetic V(D)J recombination between the V, D, and J genomic segments can code for a virtually unbounded number of antobody types. Due to this fact, assembling the whole immune repertoire from a set of RNA-Seq reads is a challenging task. The assembled CDR3 sequences are usually clustered into so-called *clonotypes*. For a comprehensive immune repertoire profiling one also needs to quantify the relative abundance of each clonotype.

## 1.3 Contributions

- A novel stand-alone scaffolding algorithm ScaffMatch which is based on the representation of the scaffolding problem as the Maximum Weight Acyclic 2-Matching. This approach allows to efficiently use the intrinsic DNA properties (such as self-complementarity) to formulate and solve the scaffolding problem in a reasonable amount of time even for whole genome-scale datasets. Approximate solutions of Maximum Weight Acyclic 2-Matching problem are found using a heuristic based on Maximum Weight Matching which is known to be solvable in polynomial time.

- Designing a novel scaffolding evaluation framework which is able to take into account

repeated contig sequences. It allows to adequately measure the quality of scaffolding assemblies produced by the current state-of-the-art tools as well as the tools adjusted to handle repeats.

- Releasing a novel repeat-aware scaffolding tool called BATISCAF which performs an optimal spurious contig filtering resulting in the scaffolding problem to be reduced to a trivial case. Contigs which are repeated are re-inserted into scaffolding as many times as it can be inferred from the scaffolding graph structure.

- Development of ImReP, a novel computational method for rapid and accurate profiling of adaptive immune repertoire from RNA-Seq data. ImReP can efficiently extract TCR- and BCR- derived reads from the RNA-Seq data and assemble corresponding B and T cell receptor clonotypes.

- A novel tool DORFA for protein database guided ORF (open reading frame) assembly. It takes as input the set of partial ORFs produced by an RNA-Seq assembler and builds from them complete ORFs. The biological value of our tool is very important since it complements the output of a de-novo RNA-Seq assembler. Finding the exhaustive set of ORFs can be crucial for accurate protein activity level estimation or for pathway reconstruction.

- New release of the IsoEM2/IsoDE2 suite for RNA-Seq gene and isoform expression level estimation and differential expression analysis. The main feature of these tools is the fast non-parametric computation of confidence intervals and identification of DE genes based on bootstrapping.

- A novel tool EMPathways for inference of pathways activity levels from RNA-Seq data was developed.

## 1.4 Roadmap

This dissertation is organized as follows. Chapter 1 presents a highlight of the Next Generation Sequencing Technologies and gives a global overview of the algorithms and methods in the context of the bioinformatics pipeline described in Section 1.1.2.

In the following chapters, novel and efficient algorithms related to different parts of the pipeline are presented. In particular, Chapter 2 presents a novel scaffolding algorithm ScaffMatch which outperforms many of the existing approaches. A conceptually new methodology for scaffolding quality assessment is proposed. Finally, a novel repeat-aware scaffolding tool BATISCAF is described. The main focus here is on the first level of the Central dogma flow. Chapter 3 proposes an assembly algorithms at the second level of the Central dogma. Namely, DORFA is designed to assemble open reading frames (ORFs) from partial RNA-Seq contigs. Chapter 4 mainly being at the third level of the Central dogma proposes a maximum likelihood approach for inferring pathways activity levels from RNA-Seq data. Along with this, a differential bootstrap based statistical approach for inferring significant up- and down-regulated pathways is presented. Chapter 5 is completely dedicated to the immune profiling part of the pipeline. The new method for CDR3 sequence assembly and immune profiling called ImReP is proposed.

## 1.5 Scientific Products

### 1.5.1 Publications

- **Book Chapters**

  1. **Igor Mandric**, James Lindsay, Ion Măndoiu, Alex Zelikovsky "Scaffolding Algorithms" Computational Methods for Next Generation Sequencing Data Analysis, 2016.

- **Journal Papers**

1. **Igor Mandric** and Alex Zelikovsky. "Solving scaffolding problem with repeats". *Bioinformatics* (Submitted).
2. **Igor Mandric**, Sergey Knyazev, and Alex Zelikovsky. "Repeat aware evaluation of scaffolding tools". *Bioinformatics (2018).*
3. Pavel Skums, Alex Zelikovsky, Rahul Singh, Walker Gussler, Zoya Dimitrova, Sergey Knyazev, **Igor Mandric**, Sumathi Ramachandran, David Campo, Deeptanshu Jha, Leonid Bunimovich, Elizabeth Costenbader, Connie Sexton, Siobhan O'Connor, Guo-liang Xia, Yury Khudyakov. "QUENTIN: reconstruction of disease transmissions from viral quasispecies genomic data". *Bioinformatics (2017) 34 (1): 163-170.*
4. **Igor Mandric**, Yvette Temate-Tiagueu, Tatiana Shcheglova, Sahar Al Seesi, Alex Zelikovsky, Ion Măndoiu. "Fast bootstrapping-based estimation of confidence intervals of expression levels and differential expression from RNA-Seq data". *Bioinformatics (2017) 33 (20): 3302-3304.*
5. Serghei Mangul, **Igor Mandric**, Harry Taegyun Yang, Nicolas Strauli, Dennis Montoya, Jeremy Rotman, Will Van Der Wey, Jiem R Ronas, Benjamin Statz, Alex Zelikovsky, Roberto Spreafico, Sagiv Shifman, Noah Zaitlen, Maura Rossetti, K Mark Ansel, Eleazar Eskin. "Profiling adaptive immune repertoires across multiple human tissues by RNA Sequencing". *bioRxiv* 2017.
6. Yvette Temate-Tiagueu, Sahar Al Seesi, Meril Mathew, **Igor Mandric**, Alex Rodriguez, Kayla Bean, Qiong Cheng, Olga Glebova, Ion Măndoiu, Nicole B Lopanik, Alexander Zelikovsky. "Inferring metabolic pathway activity levels from RNA-Seq data". *BMC Genomics 2016 17(Suppl 5):542.*
7. **Igor Mandric** and Alex Zelikovsky. "ScaffMatch: scaffolding algorithm based on maximum weight matching". *Bioinformatics (2015) 31 (16): 2632-2638.*

- **Referred Proceedings**

1. **Igor Mandric**, James Lindsay, Ion Măndoiu, Alex Zelikovsky. 'SILP3: Maximum likelihood approach to scaffolding'. *IEEE 4th International Conference on Computational Advances in Bio and Medical Sciences (ICCABS) (2014)*
2. Yvette Temate-Tiagueu, Meril Mathew, **Igor Mandric**, Qiong Cheng, Olga Glebova, Nicole Beth Lopanik, Ion Măndoiu, Alex Zelikovsky. "Metabolic pathway activity estimation from RNA-Seq data". *11th International Symposium on Bioinformatics Research and Applications (ISBRA) (2015).*
3. **Igor Mandric**, Sergey Knyazev, Cory Padilla, Frank Stewart, Ion Măndoiu, Alex Zelikovsky. "Metabolic Analysis of Metatranscriptomic Data from Planktonic Communities". *13th International Symposium on Bioinformatics Research and Applications (ISBRA) (2017).*
4. **Igor Mandric** and Alex Zelikovsky. "ScaffMatch: scaffolding algorithm based on maximum weight matching". *Research in Computational Molecular Biology (RECOMB) 2015.*

- **Conference Abstracts**

1. **Igor Mandric**, Yvette Temate-Tiagueu, Adrian Senatore, Paul Katz, and Alex Zelikovsky. "DORFA: Database-guided ORFeome assembly from RNA-Seq data". *IEEE 5th International Conference on Computational Advances in Bio and Medical Sciences (ICCABS) (2015).*

### 1.5.2 Presentations

1. Alexander Artyomenko, **Igor Mandric**, Pavel Skums, Yury Khudyakov, Alex Zelikovsky. "QUASIM: SIMulating Viral QUAsispecies evolution under immune response". *12th International Symposium on Bioinformatics Research and Applications (ISBRA) (2016).*

2. **Igor Mandric**, Sergey Knyazev, and Alex Zelikovsky. "Repeat aware evaluation of scaffolding tools". *The 7th RECOMB Satellite Workshop on Massively Parallel Sequencing (RECOMB-Seq) (2017).*
3. Serghei Mangul, **Igor Mandric**, Alex Zelikovsky and Eleazar Eskin. "Profiling adaptive immune repertoires across multiple human tissues by RNA Sequencing". *The 21st Annual International Conference on Research in Computational Molecular Biology (RECOMB) (2017).*
4. **Igor Mandric** and Alex Zelikovsky. "BATISCAF: solving scaffolding problem with repeats". *The 8th RECOMB Satellite Workshop on Massively Parallel Sequencing (RECOMB-Seq) (2018)*

### 1.5.3 Software Packages

- **ScaffMatch** - Scaffolding by Maximum Weight Matching. https://github.com/mandricigor/ScaffMatch.
- **IsoEM2** - Inferring Gene and Isoform Expression through Expectation-Maximization. https://github.com/mandricigor/isoem2.
- **DORFA** - Database-guided ORFeome Assembly. https://github.com/mandricigor/DORFA.
- **ImReP** - Profiling Immune Repertoire from RNA-Seq data. https://github.com/mandricigor/imrep.
- **EMPathways** - Estimation of Pathways activity levels from RNA-Seq data. https://github.com/mandricigor/EMPathways.
- **BATISCAF** - BAd conTIg filtering SCAFfolding. https://github.com/mandricigor/batiscaf.
- **Repeat-aware evaluation framework**. - https://github.com/mandricigor/repeat-aware.

# PART 2

# ALGORITHMS FOR SCAFFOLDING AND SCAFFOLDING EVALUATION

## 2.1 Introduction

Due to rapid advances in *High-Throughput Sequencing* (HTS) technologies the interest in the problem of *de novo* genome assembly has been renewed. These powerful technologies, also referred to as Next-Generation Sequencing (NGS), can produce millions of short paired-end reads covering whole genome; thus, the throughput is a magnitude higher than the classic *Sanger* sequencing. It is worth noticing that the cost of producing reads keeps a trend of decreasing making NGS a very attractive tool for a high range of applications. For example, Illumina HiSeq platform is able to produce billions of read pairs in a single run at a cost of cents per megabase. Although the number of reads (*shotgun* reads) is significant, due to their short length, genome assembly still represents a challenging problem. Assembled genomes are frequently highly fragmented and consist of contigs of highly variable length. The connectivity information coming from read pairs mapped to contigs can be used to merge them into a *scaffold* which is a set of chains of oriented ordered contigs with estimated gaps between all pairs of adjacent elements.

Current assemblers (Velvet [101], ALLPATHS-LG [33]) output a set of contiguous DNA chunks, usually referred to as *contigs*. Contig lengths can vary from hundreds to hundreds of thousands of base pairs. The advantage of NGS read pairs is the possibility to use them for joining contigs into some larger DNA chunks. Two reads coming from a read pair, which are mapped to two different contigs, due to the constant insert size of the library (it can be deemed as constant, although the mean insert size and its standard deviation are used), infer a certain connectivity information between the contigs. Thus, such a read pair suggests a certain relative order, relative orientation and an estimation of gap length between the two contigs. A set of contigs joined into a chain, where relative order and orientation, as well as

the estimation of the gap length between neighboring contigs is provided, is called *scaffold*.

Software programs, usually referred to as *scaffolders*, construct scaffolds based on contigs output by an assembler and the connectivity information provided by the NGS read pairs. Due to misassemblies in contigs, repeats and chimeric reads, the information about relative ordering and orientation of two contigs connected with a set of read pairs can be contradictory and not reliable. Thus, choosing a wrong subset of read pairs as an evidence for a connection between two contigs, can result in inferring a wrong relative ordering and/or orientation as well as the gap estimation between them. Edges that comply with the true orientation of contigs and the distance between them are usually called *concordant*, otherwise *discordant* edges.

In [30] it is proven that the Scaffolding Problem is an NP-hard problem. Thus, all state-of-the-art scaffolders use different heuristic approaches. A recent comprehensive evaluation of available software tools has shown that the scaffolding problem still does not have an adequate solution [40]. This means that most of the available scaffolders are not able to optimally (or at least close to an optimal solution) solve the scaffolding problem and there is still a lot of room for further improvements. In this subsection, a brief overview of the most important scaffolding tools is provided.

The most straightforward way of solving the scaffolding problem is implemented by SSPACE [9]. SSPACE (SSAKE-based Scaffolding of Pre-Assembled Contigs after Extension) is a stand-alone scaffolder of pre-assembled contigs. SSPACE is based on the short-read assembler SSAKE [93]. It greedily builds scaffolds using contigs as seeds. It starts from the largest contig and iteratively extends the scaffold based on the number of supporting read pairs. If a contig has multiple alternatives, then the ratio between two best alternatives is computed. If the ratio is less than 0.7 (default value), the scaffold is extended with the best alternative, otherwise, the extension is stopped. The process is repeated until the scaffold cannot be extended in either direction. Once the scaffold is constructed, SSPACE starts building a new scaffold from the remaining contigs using exactly the same procedure. Thus, the scaffolds are greedily built until no extension is possible.

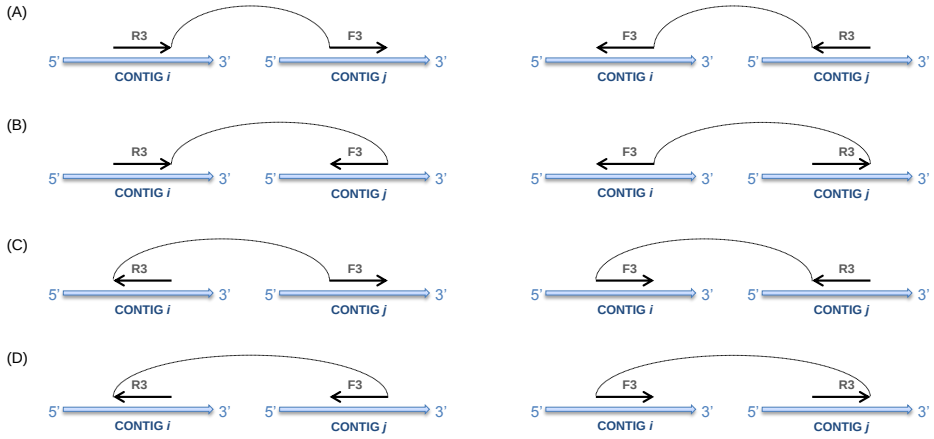



Figure 2.1 4 possibilities of connecting two contigs by a paired-end read [55].

OPERA implements an elegant dynamic programming scaffolding algorithm that solves the Scaffolding Problem of maximizing the number of concordant edges in the scaffold. SOPRA (Scaffolding algorithm for paired reads via statistical optimization) [18] solves the Scaffolding Problem by using methods inspired from statistical physics. The SOPRA algorithm consists of several stages. In the first stage, the problem of contig orientation is solved. For a pair of contigs $i$ and $j$ a value $J_{ij}$ is introduced, which is equal to the signed number of contigs supporting a certain relative orientation. The sign of $J_{ij}$ depends on the orientations of the contigs $i$ and $j$: if they have the same orientation, then the sign of $J_{ij}$ is positive, otherwise, it is negative. SOPRA searches such a configuration of the contigs, that minimizes the sum

$$-\sum_{(i,j)\in E(G)} J_{ij}S_iS_j,$$

where $S_i \in \{-1, +1\}$ stands for the contig orientation.

After the relative orientation is determined, the next stage is to solve the problem of relative contig positioning. This is achieved by maximizing a joint probability distribution associated with adjacent contigs. SOPRA provides a nice physical interpretation for the statistical optimization problem it solves at this stage.

SILP2 [55] and its improved version SILP3 [60] propose an Integer Linear Programming (ILP) formulation based on the four possibilities of a paired read to join two contigs (See Figure 2.1).

The flow of SILP3 (further, we will refer only to SILP3) consists of the following steps (which are very similar to many other tools):

1. mapping reads onto contigs
2. scaffolding graph construction
3. maximum likelihood contig orientation via ILP
4. decomposition into paths of orientation compatible edges via bipartite matching
5. maximum likelihood gap estimation

The scaffolding graph $G = (V, E)$ is constructed as to connect the vertices-contigs with the edges-read pairs. Maximum likelihood contig orientation is formulated as the following ILP [55]. Let $S_i$ be a boolean variable with value being set to 0 if the orientation of contig $i$ remains unchanged. The 4 states $A$, $B$, $C$, $D$ (Figure 2.1) in which a pair $(i, j)$ of contigs can be based on their orientation and relative ordering. The 4 boolean variables $A_{ij}$, $B_{ij}$, $C_{ij}$, $D_{ij} = \{0, 1\}\ \forall(i, j)$ correspond to the 4 states. For each state a weight $A^w_{ij}$, $B^w_{ij}$, $C^w_{ij}$, $D^w_{ij}$ using a maximum likelihood approach is calculated. The number of concordant contig pairs is then maximized:

$$Max \sum_{(i,j)\in E} A^w_{ij} \cdot A_{ij} + B^w_{ij} \cdot B_{ij} + C^w_{ij} \cdot C_{ij} + D^w_{ij} \cdot D_{ij}$$

subject to constraints connecting $A_{ij}$, $B_{ij}$, $C_{ij}$, $D_{ij}$ with $S_i$. SILP3 applies the technique of non-serial dynamic programming [82] in order to solve the optimization problem. The output of this step is a directed graph $G' = (V, E', w, g)$ in which each contig and each edge has the most likely orientation, $w : E' \to R^+$ is the weight of an edge contributing to the ILP objective, and $g : E' \to R^+$ is the estimated gap between adjacent contigs.

An *ordering* $O$ is a graph consisting of a set of disjoint directed chains of contigs together with estimations of gaps between all pairs of adjacent contigs. Note that in the ordering $O$ the estimation of gaps between adjacent contigs uniquely defines the gap $g_O(i,j)$ between any pair of connected contigs $i$ and $j$. SILP2 [55] extracts the maximum subgraph-ordering out of $G'$. Instead, SILP3 is aimed to find an ordering with the maximum support of $G'$-edges. A directed edge $e = (i,j) \in E'$ is *concordant* with an ordering $O$ if $i$ precedes $j$ in $O$ and the gap $|g(e) - g_O(i)| \leq 5\sigma$, where $\sigma$ is the standard deviation of read pair fragment length. The maximum likelihood ordering is approximated with the ordering concordant with the edges of $G'$ of the maximum total $w$-weight.

SILP3 first runs depth-first search (DFS) on $G'$ recursively deleting least-weight edge resulting in $G'$ being a directed acyclic graph (DAG). Any topological order of $G'$ is order-concordant with all edges remaining in $G'$. Let $C$ be the set of contigs between $i$ and $j$ in $G'$. $C$ is sorted based on their distance estimations to $i$ and $j$, $C_{sorted} = \{i_1, i_2, ..., i_k\}$. All edges in the graph are replaced with the directed path $P_0 = \{i, i_1, i_2, ..., i_k, j\}$. Then SILP3 determines the maximum weight bipartite matching similar to SILP2.

**Contributions.** In this chapter, a novel scaffolding algorithm called ScaffMatch [62] will be presented in details. Its main advantage over all other existing algorithms is that it formulates the Scaffolding Problem as a combinatorial optimization problem of finding the maximum-weight acyclic 2-matching problem (in other words, a subset of edges of the graph such that each vertex has degree at most 2). With a suitable reduction to the maximum weight matching problem, the almost optimal scaffolding can be determined in polynomial time ($O(n^3)$). By using the greedy heuristic, ScaffMatch is able to solve the Scaffolding Problem in linear time. As multiple experimental results show, ScaffMatch outperforms most of the standard state-of-the-art scaffolding tools.

Another very important contribution of this dissertation is the result showing that not only the Scaffolding Problem is in general computationally intractable, but such as well is the problem of evaluation of scaffolding tools. It has been shown that the Scaffolding Evaluation Problem (see Section 2.3) is NP-hard. The presence of multiple repeats in genomes can

hinder significantly the process of evaluation of scaffolding results. A novel repeat aware evaluation framework is presented in this chapter. Its advantage is that it is able to compare scaffolding results making abstraction of whether repeats are handled by the scaffolding tool or not. Also, it provides insight into scaffolding results by making a comprehensive classification of incorrect joins. This can leverage further research in genome scaffolding.

Not of less importance is the contribution which relates to repeat aware scaffolding. In this thesis, a novel repeat aware tool called *BATISCAF* is presented. BATISCAF tackles the scaffolding problem differently than all previous tools. Namely, it is focused not on finding paths corresponding to scaffolds (as most of the tools do) but on an optimal spurious contig filtering. The two types of spurious contigs complicating the task of scaffolding are (i) short (ii) repeated contigs. By eliminating them from the scaffolding graph BATISCAF reduces the computationally hard problem to a trivial case. Afterward, the previously filtered contigs are re-inserted back into the scaffolds by using a maximum likelihood approach. The repeated contigs may be re-inserted multiple times (as many times as it can be inferred by the algorithm from the scaffolding graph structure).

## 2.2 ScaffMatch: combinatorial optimization approach to genome scaffolding

We proposed a novel optimization formulation representing scaffolding as a maximum-weight *acyclic* 2-matching problem. Since the Hamiltonian path problem can be reduced to this problem, this formulation is also NP-complete. The presented algorithm ScaffMatch efficiently finds the maximum-weight 2-matching and iteratively destroys all cycles. This approach works very well since, usually, number of cycles is very small.

Our experimental study follows the evaluation of state-of-the-art scaffolders in [40] performed on 5 scaffolding datasets (including 4 from the GAGE (Genome Assembly Gold-standard Evaluations) project [76]). We have run the majority of up-to-date versions of stand-alone scaffolders such as OPERA, SOPRA [18], SSPACE and MIP as well the recently published ones, SILP2 and BESST. We have also included the results for scaffolding modules of SGA [83] and SOAPdenovo2 [57] run independently of de novo assembly follow-

ing [40]. Our matching-based tool ScaffMatch compares favorably with the state-of-the-art tools in terms of the widely accepted N50 metric, the metrics introduced in [40], as well as sensitivity, PPV (positive predictive value), and F-score in predicting contig junctions.

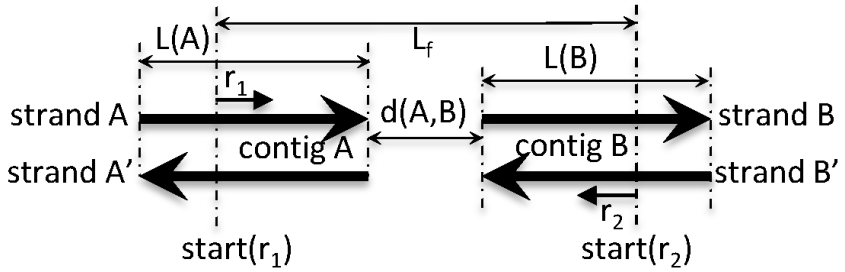



Figure 2.2 Gap estimation $d$ is calculated in conformity with the formula: $d = L_f - (L(A) - start(r_1)) - (L(B) - start(r_2))$, where $L_f$ is the fragment length, $L(A), L(B)$ are lengths of contigs $A$ and $B$, $start(r_1)$ (resp. $start(r_2)$) is the distance from the starting position of $r_1$ (resp. $r_2$) to the beginning of the strand $A$ (resp. $B'$).

**Methods.** Below we describe the problem formulation and algorithmic details in the following main scaffolding steps:

- Preprocessing of read pairs including read mapping, handling repeats and gap estimation for each read pair.
- Scaffolding graph construction with vertices corresponding to contig strands and edges corresponding to read pairs.
- Matching scaffold finding near-maximum weight paths in the scaffolding graph and the corresponding orientation and ordering contigs.
- Insertion of skipped contigs into the matching scaffold.

We conclude with the implementation details of ScaffMatch.

**Read Preprocessing.** Each contig has two reverse complement strands and each read from a pair is mapped to one of the strands. We discard reads aligned to (suspected) repeats. First, we filter out read pairs in which at least one read has multiple alignments. Then for each contig, we compute its read coverage and filter out contigs with coverage greater by

$2.5\sigma$ than the average where $\sigma$ is the standard deviation of contig coverage. This empirically chosen threshold allows to remove the majority of repeats while keeping almost 99% of correct contigs. Although assemblers may give chimeric contigs or produce two contigs for the same genomic region (representing the two haplotypes) ScaffMatch does not attempt to identify or modify any contigs.

Each read is mapped only to one of the two contig strands (palindromes are discarded). For each read pair connecting two distinct contig strands, we estimate the gap according to Figure 2.2 (for an alternative gap estimation model see [73]). Among all read pairs connecting the same contig strands, we find a read pair $p$ with the median gap estimation and then bundle $p$ with all read pairs whose gap estimation is at most $3\sigma$ away from $p$'s estimation. Since we want to keep only reads that agree with each other, the reads outside of this bundle are discarded.

**Scaffolding graph.** Each vertex of the scaffolding graph $G = (V, E)$ corresponds to one of the contig strands and each *inter-contig* edge corresponds to a bundle of read pairs connecting two strands of different contigs (see Figure 2.3). The weight of an inter-contig edge is equal to the size of the corresponding bundle. Also for each contig, we have a *dummy* edge connecting its two strands.

**Matching Scaffolding.** Ideally, we expect that the scaffolding graph would consist of a set of paths each corresponding to a different chromosome (see Figure 2.4(a)). Unfortunately, repeats introduce noisy edges connecting unrelated contigs even from different chromosomes. Additionally, the paths corresponding to chromosomes may skip short contigs (especially contigs which are shorter than the insert length). Therefore, any set of paths passing through all dummy edges in the scaffolding graph $G$ corresponds to a plausible scaffold (see Figure 2.4(b)). The most likely scaffold would be supported by the largest number of read pairs. Therefore, we can formulate the following

> **Scaffolding Problem.** Given a scaffolding graph $G$, find a set of paths passing through all dummy edges with maximum total weight of all inter-contig edges.

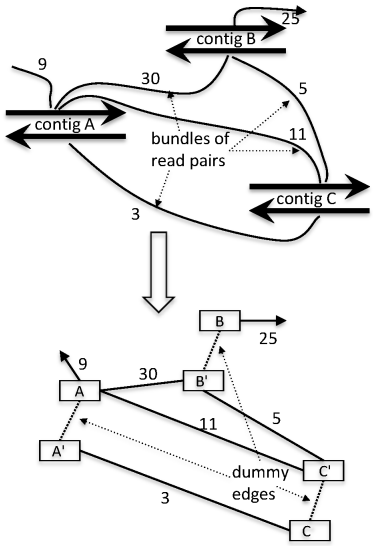



Figure 2.3 Contigs $A$, $B$, and $C$ with connecting bundles of read pairs and the corresponding scaffolding graph. Each contig is split into two nodes connected with a dummy edge. Each bundle of read pairs corresponds to an inter-contig edge connecting respective strands with the weight equal to the size of the bundle. A plausible scaffold corresponds to the path $C' - C - A' - A - B' - B$ supported by two inter-contig edges $CA'$ and $AB'$.

By setting the weight of each dummy edge to a large number (e.g., number of all read pairs), we reduce the scaffolding problem to the following

> **Maximum-Weight Acyclic 2-Matching (MWA2M) Problem.** Given a weighted graph $G = (V, E, w)$, find a maximum weight acyclic subset of edges $M \subseteq E$ such that each vertex $v \in V$ is incident to at most 2 edges in $M$.

The MW2AM of an $n$-vertex graph $G$ with all edge weights 1 has the weight $n - 1$ if and only if $G$ has a Hamiltonian path. Therefore, the MWA2M problem is NP-complete since it includes the Hamiltonian path problem. A similar well-known problem, the Maximum Weight 2-Matching (MW2M), allows chosen edges to form cycles. In contrast to the MWA2M problem, the MW2M problem can be efficiently solved [70].

**Maximum-Weight Matching heuristic for the MWA2M problem.** We propose to almost optimally solve the MWA2M problem with the following iterative heuristic based on the well-known blossom algorithm [23]) for finding the maximum-weight matching in

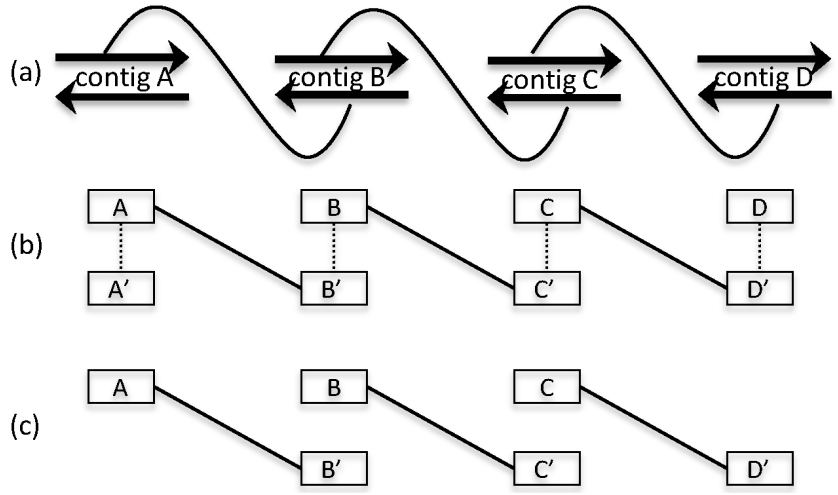



Figure 2.4 (a) A scaffold $A - B - C - D$: the connection of each pair of adjacent contigs is supported by bundles of read pairs. (b) A path $A' - A - B' - B - C' - C - D' - D$ in the scaffolding graph corresponding to the scaffold $A - B - C - D$. (c) The matching of the scaffolding graph corresponding to the bunches of read pairs supporting adjacent contigs.

weighted graphs. It starts with finding the maximum-weight matching $M$ among the inter-contig edges. All the dummy edges also form a matching $D$. If the union of these two matchings $M \cup D$ does not contain cycles, then the heuristic reaches the optimal collection of paths. Otherwise, a negative weight $-1$ is assigned to the least weight inter-contig edge in each cycle. The above steps of finding the inter-contig matching $M$ and destroying cycles in $M \cup D$ are repeated until the union $M \cup D$ becomes a collection of paths. The output of this heuristic will be called the Matching Scaffold.

In general, the deletion of least-weight edges may significantly reduce (as much as twice in the worst case) the total weight of the collection of paths. Fortunately, the erroneous heavy inter-contig edges are very rare in real data. Our experiments show that for each scaffolding example there is no more than a single cycle in the initial union $M \cup D$ of the maximum-weight matching $M$ and the dummy edges solution and after the second iteration, $M \cup D$ does not contain any cycles at all.

**Greedy heuristic for the the MWA2M problem.** The maximum weight matching can be computed efficiently even for larger genomes. Still, the runtime can be dramatically

decreased using the Greedy Heuristic repeatedly choosing the heaviest feasible edge, i.e., an edge which does not make a vertex degree higher than 2 and does not form cycles with the previously chosen edges. Note that the Greedy Heuristic picks the *globally* heaviest edge in contrast to greedy scaffolders (such as SSPACE) greedily extending existing chains. We provide an option that allows ScaffMatch to run with the Greedy Heuristic reducing the run-time complexity from $O(n^3)$ to $O(n \cdot log\ n)$ as we use max heap in our implementation. Our experiments show that the Greedy Matching performs very well in practice but sacrificing not much in quality to the Maximum-Weight Matching heuristic.

**Contig ordering and orientation.** The Matching Scaffold is represented by a collection of disjoint chains of contig strands. The sequence of edges along each chain alternates: two strands of the same contig are connected with a dummy edge, two strands of different contigs are connected with an inter-contig edge. When traversing the strands along the paths in the matching scaffold, the order of traversing ends of dummy edges gives us the orientation of the corresponding contigs and the order of traversing inter-contig edges gives us the relative order of contigs.

**Insertion of skipped contigs.** The Matching Scaffold can skip short contigs whose length is less than the read pair insert size. For example, let the true scaffold contain a triple of consecutive contigs $A$, $B$ and $C$ such that $l_A > l_{ins}$, $l_C > l_{ins}$, but $l_B \ll l_{ins}$. Then instead of picking both edges $AB$ and $BC$, the Matching Scaffold may choose one single edge $AC$ since the edge weight between short contigs depends almost linearly on the length of the contigs. Thus, even though the contig $B$ must follow $A$ in the final scaffold, the weight of the edge between $A$ and $B$ is much smaller than the weight of the edge between $A$ and $C$, which "jumps" over $B$.

Below we describe the insertion of skipped contigs into the Matching Scaffold (See Algorithm 1). A contig is identified as *skipped* only if it is isolated or is a part of a 2-chain in the Matching Scaffold. Scaffolds with more than 2 contigs are kept intact. For each skipped contig we identify the most bundle-supported slot in the matching scaffold satisfying the gap estimations and insert it in this slot (see Figure 2.5). If several skipped contigs are assigned

to the same slot, their relative order and orientation is decided based on the gap estimations as follows. For each skipped contig $(X', X)$, we estimate the distance to the left contig and orient it according to the adjacent strands. Then we sort all contigs with respect to this distance and insert them according to this order.

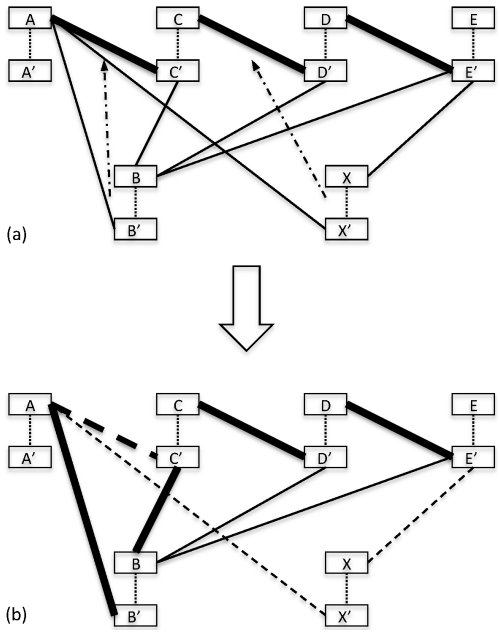



Figure 2.5 Insertion procedure: (a) The matching scaffold $A - C - D - E$ is obtained with the maximum weight matching; the contig $B$ is connected with edges to all 4 contigs of the matching, the contig $X$ is connected to $A$ and $C$; $B$ should be placed between $A$ and $C$ according to the consensus of connecting edges and $X$ should be placed between $C$ and $D$; (b) Since there is a sufficient distance between contigs $A$ and $C$, $B$ is placed there, i.e., the edge $(A, C')$ from the matching is replaced with $(A, B')$ and $(B, C')$ (the sum of weights of $(A, B')$ and $(B, C')$ is less than the weight of $(A, C')$); since there is no sufficient room for $X$ between contigs $C$ and $D$, the edges $(A, X')$ and $(X, E')$ are removed. The resulted scaffold is $A - B - C - D$.

**Software implementation.** The described scaffoding algorithm is implemented as a stand-alone software tool called ScaffMatch. We separately provide a UNIX shell script for mapping reads to contigs. As a short read aligner, bowtie2 is used [48]. The scaffolder takes as input a fasta file containing the contigs and two SAM files produced by mapping the two read files to the contigs. We keep a small set of mandatory parameters: the mean insert size, the standard deviation and the orientation (forward-reverse, reverse-forward or SOLiD-style) of

**Algorithm 1** Insertion of skipped contigs

$\mathcal{SLOTS} \leftarrow \{\}$
$SKIPPED \leftarrow$ the set of skipped contigs
$M = \{m_1, m_2, ..., m_n\} \leftarrow$ the Matching Scaffold
$G = (V, E, w) \leftarrow$ the Scaffolding Graph
$l \leftarrow$ insert length
**for all** $\mathcal{X} = (X', X) \in SKIPPED$ **do**
  **for all** $m \in M$ **do**
    **if** $\exists$ contigs $\mathcal{A} = (A, A'),\ \mathcal{B} = (B, B') \in m$
    s.t. $(A, X') \in E$ and $(B', X) \in E$ &
    $gap(\mathcal{A}, \mathcal{X}) + l(\mathcal{X}) + gap(\mathcal{X}, \mathcal{B}) \leq l$ **then**
      **for** each edge $e = (\mathcal{C}_i, \mathcal{C}_{i+1}) \in m(\mathcal{A}, \mathcal{B})$ **do**
        **if** $gap(\mathcal{A}, \mathcal{C}_i) \leq gap(\mathcal{A}, \mathcal{X})$ **and** $gap(\mathcal{C}_{i+1}, \mathcal{B}) \leq gap(\mathcal{X}, \mathcal{B})$ **then**
          $\mathcal{SLOTS}[e, \mathcal{X}]$ += $w(A, X') + w(X, B')$
        **end if**
      **end for**
    **end if**
  **end for**
**end for**
**for all** $\mathcal{X} \in SKIPPED$ **do**
  $e \leftarrow \max\{\mathcal{SLOTS}[e, \mathcal{X}] \mid e \in E\}$
  **insert** $\mathcal{X}$ **into** $e$
**end for**
**return** $\mathcal{SLOTS}$

the paired-end reads. The program outputs a fasta file with scaffolds. We use Networkx python library for graph computations [38].

### 2.2.1 Results

**Datasets.** We validate and compare the scaffolding tools on the collection of scaffolding datasets used in [40] including 4 datasets from the GAGE project [76] (*Staphylococcus aureus*, *Rhodobacter sphaeroides* and *Homo sapiens (chr14)*) and one additional dataset *Plasmodium falciparum* following [40]. All contigs were assembled by Velvet [100]. The Table 2.9 gives the parameters of all used scaffolding datasets.

**Quality metrics.** The main metric that is used for evaluation of scaffolding tools is N50 [91]. However, this metric may not be representative enough as mentioned in [40] where a comprehensive evaluation of scaffolders was performed. In that evaluation, state-of-the-art

Table 2.1 Scaffolding Datasets

| Datasets | insert size | read length | # contigs | # reads |
|---|---|---|---|---|
| *S. aureus* | 3600 | 37 | 170 | 3494070 |
| *R. sphaeroides* | 3700 | 101 | 577 | 2050868 |
| *H. sapiens (chr14)* | | | | |
| (short insert size) | 2865 | 101 | 19936 | 22669408 |
| (long insert size) | 35000 | 80 | 19936 | 2405064 |
| *P. falciparum* | | | | |
| (short insert size) | 645 | 76 | 9318 | 52542302 |
| (long insert size) | 2705 | 75 | 9318 | 12010344 |

tools were compared based on multiple criteria, such as the number of correct junctions between two adjacent contigs, the number of junctions with incorrect relative order, relative orientation, gap estimation and their combinations (e.g., incorrect relative order + incorrect gap estimation). The scores assigned to the scaffolders, however, can be misleading. For example, MIP on *S. aureus* (using bowtie2) got a high score despite the fact that it joined no contigs. Thus, we introduce an F-score based metric in order to compare the results of our tool ScaffMatch with other de-novo stand-alone scaffolding tools.

Various quality metrics have been proposed up to date. Rather than coming up with our own metrics, we have decided to follow the most recent evaluation paper [40] which besides N50 and corrected N50 also reports the number of correctly and erroneously predicted joins between contigs in the reference genome. Following [40], we do not distinguish between links connecting long and short contigs as well as contigs from different chromosomes. Let $P$ be the number of potential contigs that can be joined in scaffold which is the number of contigs minus the number of chromosomes, let $TP$ be the number of correct contig joins in the output of the scaffolder (true positives) and $FP$ be the number of erroneous joins (false positives). We compute the following quality metrics: $TPR = \frac{TP}{P}$, $PPV = \frac{TP}{TP+FP}$, $F$-score $= \frac{2 \cdot TPR \cdot PPV}{TPR+PPV}$, where $TPR$ is sensitivity, $PPV$ is positive predictive value.

**Evaluation and Comparison.** ScaffMatch is compared with well established scaffolders SSPACE, OPERA, SOPRA, MIP, SCARPA, two recently published scaffolders SILP2 and BESST [75] scaffolding modules of SGA [83] and SOAPdenovo2 [57]. SSPACE, OPERA,

Table 2.2 Performance of different algorithms on the scaffolding datasets from GAGE.

| Dataset | Scaffolder | Correct links | Error links | Skipped contigs | N50 | Corr. N50 | Sensitivity | PPV | F-score |
|---|---|---|---|---|---|---|---|---|---|
| *S. aureus* | ScaffMatch | 139 | 14 | 23 | 1476925 | 351546 | **0.832** | 0.908 | **0.869** |
| | SSPACE | 105 | 13 | 13 | 332784 | 261710 | 0.629 | 0.890 | 0.737 |
| | OPERA | 112 | 11 | 22 | 1084108 | **686577** | 0.671 | 0.911 | 0.845 |
| | SOPRA | 40 | 2 | 7 | 112278 | 112083 | 0.240 | 0.952 | 0.383 |
| | MIP | 0 | 0 | 0 | 46221 | 46221 | 0 | 0 | 0 |
| | SCARPA | 77 | 16 | 10 | 112264 | 112083 | 0.461 | 0.828 | 0.592 |
| | SILP2 | 121 | 3 | 34 | 645780 | 284980 | 0.725 | 0.976 | 0.832 |
| | BESST | 112 | 11 | 21 | **1716351** | 335064 | 0.671 | 0.911 | 0.772 |
| | SGA | 83 | 1 | 10 | 309286 | 309153 | 0.497 | **0.988** | 0.661 |
| | SOAPdenovo2 | 131 | 12 | 13 | 643384 | 621109 | 0.784 | 0.916 | 0.845 |
| *R. sphaeroides* | ScaffMatch | 482 | 18 | 40 | **2547706** | **2528248** | **0.845** | 0.964 | **0.901** |
| | SSPACE | 357 | 7 | 49 | 109776 | 108410 | 0.626 | 0.981 | 0.764 |
| | OPERA | 316 | 1 | 23 | 108172 | 108172 | 0.554 | **0.997** | 0.713 |
| | SOPRA | 242 | 15 | 24 | 32232 | 30492 | 0.425 | 0.942 | 0.585 |
| | MIP | 419 | 37 | 16 | 488095 | 487941 | 0.735 | 0.919 | 0.817 |
| | SCARPA | 209 | 5 | 23 | 37667 | 37581 | 0.367 | 0.977 | 0.533 |
| | SILP2 | 425 | 24 | 87 | 471077 | 422445 | 0.746 | 0.947 | 0.834 |
| | BESST | 367 | 2 | 15 | 1021151 | 1020921 | 0.644 | 0.995 | 0.782 |
| | SGA | 232 | 1 | 26 | 42825 | 42722 | 0.407 | 0.996 | 0.578 |
| | SOAPdenovo2 | 468 | 8 | 26 | 2522483 | 2522482 | 0.821 | 0.983 | 0.895 |
| *H. sapiens* (chr 14) short insert size | ScaffMatch | 12411 | 252 | 3480 | 131135 | 80329 | 0.622 | 0.980 | 0.761 |
| | SSPACE | 9566 | 43 | 2754 | 78552 | 77361 | 0.487 | 0.986 | 0.652 |
| | OPERA | 12291 | 112 | 2991 | 214972 | 207047 | 0.616 | 0.991 | 0.760 |
| | SOPRA | 14761 | 381 | 1441 | 100768 | 96436 | 0.740 | 0.975 | 0.841 |
| | MIP | 13899 | 954 | 2735 | 244064 | **235731** | 0.697 | 0.936 | 0.799 |
| | SCARPA | 9938 | 162 | 1829 | 58330 | 55760 | 0.498 | 0.984 | 0.661 |
| | SILP2 | 10548 | 124 | 4918 | 126689 | 77421 | 0.529 | 0.988 | 0.689 |
| | BESST | 7970 | 355 | 2165 | 146749 | 80218 | 0.400 | 0.957 | 0.564 |
| | SGA | 9761 | 6 | 3214 | 134574 | 133192 | 0.490 | **0.999** | 0.657 |
| | SOAPdenovo2 | 15740 | 390 | 2378 | **282437** | 234561 | **0.790** | 0.976 | **0.873** |
| *H. sapiens* (chr 14) long insert size | ScaffMatch | 5938 | 443 | 5198 | 148412 | 42523 | **0.298** | 0.933 | **0.452** |
| | SSPACE | 2750 | 23 | 2539 | 77832 | 30449 | 0.138 | **0.992** | 0.242 |
| | OPERA | 3687 | 677 | 3226 | 73477 | 20677 | 0.185 | 0.845 | 0.303 |
| | SOPRA | 2938 | 166 | 2622 | 79517 | 34750 | 0.147 | 0.947 | 0.255 |
| | MIP | 5898 | 1092 | 4861 | **272440** | 49800 | 0.296 | 0.844 | 0.438 |
| | SCARPA | 1603 | 31 | 1466 | 43969 | 17786 | 0.080 | 0.981 | 0.149 |
| | SILP2 | 3899 | 65 | 3732 | 74094 | 38810 | 0.196 | 0.984 | 0.326 |
| | BESST | 123 | 13 | 98 | 13815 | 8828 | 0.006 | 0.904 | 0.012 |
| | SGA | 0 | 0 | 0 | 12211 | 12211 | 0 | 0 | 0 |
| | SOAPdenovo2 | 4516 | 294 | 3301 | 220644 | **86679** | 0.227 | 0.939 | 0.365 |
| *H. sapiens* (chr 14) combined library short + long insert size | ScaffMatch | 12658 | 348 | 3874 | 802755 | 195239 | 0.635 | 0.973 | 0.769 |
| | SSPACE | 9249 | 36 | 2677 | 66271 | 65222 | 0.464 | 0.996 | 0.633 |
| | OPERA | 12853 | 58 | 3409 | **1692782** | **1062031** | 0.645 | 0.996 | 0.783 |
| | SOPRA | 10418 | 238 | 3322 | 112239 | 75046 | 0.523 | 0.978 | 0.681 |
| | MIP | 8534 | 696 | 3213 | 44372 | 31148 | 0.428 | 0.925 | 0.585 |
| | SCARPA | 10712 | 161 | 2376 | 134364 | 106654 | 0.537 | 0.985 | 0.695 |
| | BESST | 8287 | 286 | 2347 | 295976 | 114434 | 0.416 | 0.967 | 0.581 |
| | SGA | 9764 | 3 | 3214 | 134574 | 133192 | 0.490 | **0.999** | 0.657 |
| | SOAPdenovo2 | 15748 | 382 | 2575 | 561198 | 447849 | **0.790** | 0.976 | **0.873** |

Table 2.3 Performance of different algorithms on the scaffolding datasets for *P. falciparum*.

| Dataset | Scaffolder | Correct links | Error links | Skipped contigs | N50 | Corr. N50 | Sensitivity | PPV | F-score |
|---|---|---|---|---|---|---|---|---|---|
| *P. falciparum* short insert size | ScaffMatch | 5648 | 287 | 37 | **8626** | 5872 | 0.607 | 0.952 | 0.741 |
| | SSPACE | 5746 | 127 | 12 | 6011 | 5845 | **0.612** | 0.978 | **0.757** |
| | OPERA | 3706 | 116 | 371 | 5035 | 4824 | 0.398 | 0.967 | 0.565 |
| | SOPRA | 4897 | 174 | 34 | 4954 | 4632 | 0.526 | 0.966 | 0.681 |
| | MIP | 5544 | 359 | 15 | 6158 | 5485 | 0.596 | 0.939 | 0.730 |
| | SCARPA | 4830 | 221 | 38 | 4912 | 4628 | 0.519 | 0.956 | 0.673 |
| | SILP2 | 5496 | 498 | 48 | 3109 | 2601 | 0.591 | 0.917 | 0.719 |
| | BESST | 2632 | 462 | 84 | 7471 | 3931 | 0.283 | 0.851 | 0.425 |
| | SGA | 4940 | 46 | 100 | 5324 | 5104 | 0.531 | **0.991** | 0.691 |
| | SOAPdenovo2 | 5540 | 84 | 47 | 6234 | **5981** | 0.596 | 0.985 | 0.742 |
| *P. falciparum* long insert size | ScaffMatch | 6970 | 260 | 1751 | 41564 | 25380 | 0.749 | 0.964 | 0.843 |
| | SSPACE | 4610 | 21 | 1235 | 17796 | 15553 | 0.496 | 0.995 | 0.662 |
| | OPERA | 6257 | 97 | 1339 | 44667 | 40170 | 0.673 | 0.985 | 0.799 |
| | SOPRA | 7247 | 181 | 656 | 49671 | 44158 | 0.779 | 0.976 | **0.866** |
| | MIP | 7754 | 707 | 731 | 88297 | 78672 | **0.834** | 0.916 | 0.873 |
| | SCARPA | 4882 | 117 | 714 | 14037 | 9708 | 0.525 | 0.977 | 0.683 |
| | SILP2 | 5996 | 266 | 2839 | 45407 | 29399 | 0.645 | 0.957 | 0.771 |
| | BESST | 1307 | 46 | 327 | 4133 | 2813 | 0.141 | 0.966 | 0.245 |
| | SGA | 2902 | 2 | 652 | 4438 | 4096 | 0.312 | **0.999** | 0.476 |
| | SOAPdenovo2 | 7659 | 351 | 803 | **167570** | **83851** | 0.635 | 0.869 | 0.734 |
| *P. falciparum* combined library short + long insert size | ScaffMatch | 8223 | 425 | 654 | **78627** | **47662** | **0.884** | 0.951 | **0.916** |
| | SSPACE | 5889 | 123 | 76 | 6383 | 5982 | 0.633 | 0.980 | 0.769 |
| | OPERA | 6434 | 177 | 1171 | 42450 | 38409 | 0.692 | 0.973 | 0.809 |
| | SOPRA | 7018 | 60 | 171 | 16366 | 15511 | 0.754 | **0.992** | 0.857 |
| | MIP | 8082 | 513 | 75 | 56672 | 38704 | 0.869 | 0.940 | 0.903 |
| | SCARPA | 7336 | 370 | 251 | 36945 | 23951 | 0.789 | 0.952 | 0.863 |
| | BESST | 3929 | 541 | 384 | 25300 | 7621 | 0.422 | 0.879 | 0.571 |
| | SGA | 4910 | 44 | 419 | 6606 | 6134 | 0.528 | 0.991 | 0.689 |
| | SOAPdenovo2 | 5977 | 228 | 254 | 12076 | 10629 | 0.643 | 0.963 | 0.771 |

and SOPRA used bowtie [48] mapping, SOAPdenovo2 used its own mapping, and all other scaffolders used bowtie2 mapping. All software has been run with the same versions and options as in [40] except SILP2 and BESST for which default parameters were used from the respective websites. Results for SILP2 are not given for the combined insert size datasets since it does not have an option to process datasets with multiple insert size.

For computing the number of correct and erroneous links we used scripts provided in [40]. Note that MIP and SGA did not give meaningful results respectively for the *S. aureus* and *H. sapiens* (long insert size).

We compared 3 different versions of ScaffMatch: ScaffMatch (Maximum Weight Matching with insertion), ScaffMatch_G (Greedy Matching with insertion), and ScaffMatch_B (Maximum Weight Matching) (The results are available in the Table S1 of Supplementary Data [62]). ScaffMatch usually has the best $F$-score among all three versions. ScaffMatch_G also can be very different from ScaffMatch since it may choose to match completely differ-

ent contigs. ScaffMatch_B has usually the highest PPV and corrected N50 among all three versions implying that insertion of skipped contigs might be erroneous. Unexpectedly, the number of contigs skipped by ScaffMatch_B is not much greater than for ScaffMatch showing that the solution for the scaffolding/MWA2M problem does not skip over many contigs.

The results for GAGE scaffolding testcases are in Table 2.2 and results for *P. falciparum* are in Table 2.3. The entries in the bold font are the best among all 10 scaffolders with respect to the corresponding quality metric. ScaffMatch has the top $F$-score for 4 testcases and the top corrected N50 for 2 testcases. Additionally, ScaffMatch_B has the the top corrected N50 for *S. aureus*. SOAPdenovo2 has the top $F$-score for 2 testcases and the top corrected N50 for 3 testcases. MIP is a top performer once for $F$-score and once for corrected N50. Finally, OPERA is the best in corrected N50 for 2 testcases (still losing to ScaffMatch_$B$ on *S. aureus*) and SSPACE has the best $F$-score for one testcase.

The runtime of all compared scaffolders are in Table 2.10. The runtime growth rate with respect to the dataset size is similar for all scaffolders. The fastest scaffolder is SSPACE and the slowest is SOPRA.

Table 2.4 The wall clock runtime in seconds for the largest and the smallest datasets. All scaffolders were run on 2.5GHz 16-core AMD Opteron 6380 processors with 256Gb RAM running under Ubuntu 12.04 LTS. SM is ScaffMatch and SM_G is ScaffMatch_G.

| Tool / Dataset | SM | SM_G | SSPACE | OPERA | SOPRA | MIP | SCARPA | SILP2 | BESST | SGA | SOAP2 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *S. aureus* | 60 | 56 | 20 | 178 | 676 | 154 | 111 | 64 | 68 | 64 | 40 |
| *H.sapiens* (long) | 754 | 226 | 248 | 308 | 8852 | 528 | 538 | 334 | 728 | - | 264 |

Although the methodology [40] we used to compare ScaffMatch with all other tools is a widely accepted one, it can not exactly evaluate the number of correct links produced by a scaffolding tool due to repeats in the reference genome. Namely, repeated contigs may be aligned to multiple locations on the reference; thus, such ambiguities pose an additional problem when comparing different tools on a benchmarking dataset. The next part of this chapter proposes a novel approach for evaluation of scaffolding tools accounting for repeats.

## 2.3 Methodologies for performance evaluation of scaffolding tools

Genome assembly is one of the oldest, yet still one of the most relevant problems in bioinformatics even nowadays. Traditionally, any genome assembly pipeline is divided into three stages: contig assembly, scaffold assembly, and gap filling. Scaffold assembly is the problem of building chains of contigs from the information provided by paired-end reads. Since early 2000s many scaffolding problem formulations and algorithms were proposed: OPERA [30], SSPACE[9], BESST [75], ScaffMatch [62]. Most of the scaffolding formulations imply the construction of so-called scaffolding graph $G = (V, E)$, where $V$ is usually the set of contigs (or contig strands [62]) and $E$ is the set of links obtained by grouping multiple paired-end reads aligning to different contigs. Some scaffolding tools provide heuristics for building paths in $G$ (SSPACE, BESST), where each path corresponds to a scaffold. Optimization approaches reduce scaffolding to maximizing the number of correct links or minimizing the number of erroneous links. Such formulations are usually NP-hard [30, 62].

Finding true scaffolding is computationally challenging due to different factors: misassemblies, inconsistent coverage across the genome, but the most important challenge though is the presence of genome repeats. For example, the human genome is reported to contain up to 50% of repeated sequences. Contig assembly tools are not able to distinguish different copies of the same repeat, therefore, all the copies of the same repeated DNA region are usually collapsed into one contig. This creates multiple erroneous links in the scaffolding graph. On the other hand, the reference scaffolding should split such contig into several copies in order to correctly correspond to the reference genome. Therefore, an accurate evaluation of an inferred scaffolding should take into account multiple locations of the same contig on the reference scaffolding rather than matching a repeat to a single best location. This makes mapping of an inferred scaffolding onto the reference scaffolding a nontrivial problem.

There are numerous slightly conflicting parameters for evaluating scaffolding quality – N50, corrected (or true) N50, number of correct and incorrect links between contigs with

corresponding sensitivity and PPV, number of inverted contigs, number of correctly assembled genes, etc. (see e.g., [40, 62, 55, 5, 60]). Notoriously, the most difficult to maximize is true N50, which is the minimum length of inferred scaffolds which cover 50% of the genome. Here we focus on a simpler objective of maximizing the number of correctly inferred links between contigs:

> **The Scaffolding Evaluation Problem**. Find a mapping of the inferred scaffolding onto the reference maximizing the number of correct links.

Let us view contigs as genes, repeats as gene families, inferred scaffolds and reference scaffoldings as two genomes over the same set of gene families. Then the Scaffolding Evaluation (SE) problem is equivalent to finding the breakpoint distance [79, 4] between two signed sequences corresponding to the inferred and reference scaffoldings (see Figure 2.6). Computing the breakpoint distance is NP-hard [8] in presence of nontrivial gene families and, therefore, the SE problem is also NP-hard in presence of repeated contigs.

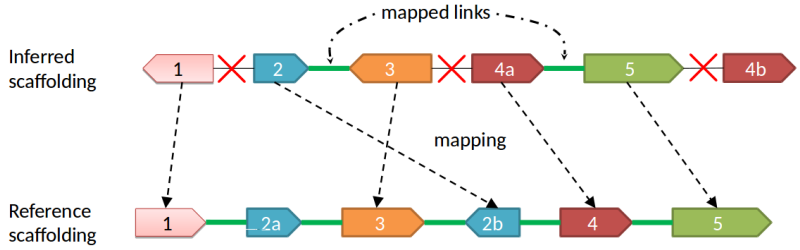



Figure 2.6 Matching of contigs in the scaffolding $S$ and in the reference $R$. Only the links (2,3) and (4a,5) are mapped correctly. Contig 2 has two copies 2a and 2b in the reference scaffolding, contig 4 is inferred to have two copies 4a and 4b. Assigning contig 2 to either of the copy 2a or 2b, as well as assigning either 4a or 4b to the reference contig 4 affects the number of correctly inferred contigs links. Indeed, assigning contig 2 to the reference contig 2a and contig 4b to 4 will mistakenly undercount the number of correct links. The optimal assignment (2 to 2b, 4a to 4) allows to detect two correctly linked contig pairs.

The rest of the section is organized as follows: Subsection 2.3.1 provides background and motivation for a new scaffolding validation framework, in Subsection 2.3.2 we give Integer

Linear Program based formulation of the Scaffolding Evaluation Problem. Subsections 2.3.3-2.3.3 explain the validation pipeline and provide the comparison results of most state-of-the-art scaffolding tools.

### 2.3.1 Background

The traditional way of measuring the quality of a scaffold assembly used by practitioners is N50 which is the shortest sequence length at 50% of the genome. Although this measure is intuitively clear and allows one to estimate the contiguity of the scaffolds, it is rather meaningless: all the contigs randomly joined together produce a scaffolding with highest N50. An alternative way to measure contiguity of scaffolds is to apply corrected N50 metric which is N50 computed after removing all the wrong contig connections.

One of the most recent frameworks for scaffolding evaluation [40] proposed a strategy for evaluating the output of scaffolding tools based on a known ground truth. In [40], the so-called "assembly" contigs (i.e., produced by an assembly tool, for example, Velvet [101]) were used to produce "perfect" contigs. The procedure for obtaining the set of perfect contigs is to align the "assembly" contigs to the reference genome with nucmer [20], merge all overlapping hits corresponding to one contig and removing contigs which are completely contained inside any other contig. Then, from each perfect contig, a unique sequence tag is extracted as an anchor for identification of mutual ordering, orientation, and distance of contigs in the output of scaffolding by aligning the tags to them. In [40], the perfect contigs were not assumed to be repeated ones or to have a sub-part which is repeated in the genome. The only best nucmer hit of each assembly contig was considered. Evaluation [40] did not take into account repeated contigs, though a solid evidence for a high number of repeats exists.

For example, if not only the best nucmer hit was taken into account, but rather all the nucmer hits of assembly contigs with at least 97% identity to the reference in the benchmarks from [40] then the percentage of repeated contigs would be considerable. Thus, the percentage of repeated contigs in the *S. aureus* dataset would be 14%.

Table 2.5 The number of unique contigs and the number of repeats in three GAGE [76] datasets (*S. aureus*, *R. sphaeroides*, *H. sapiens (chr 14)*) and two fungi datasets (*M. fijiensis*, *M. graminicola*) obtained with the aid of iWGS pipeline [102]. Each contig dataset is obtained with the scripts from [40] from Velvet assembly contigs.

| Dataset | Total | Unique | Number of copies | | | Avg len |
|---|---|---|---|---|---|---|
| | | | 2 | 3 | $\geq 4$ | |
| *S. aureus* | 200 | 172 | 4 | 5 | 4 | 16573 |
| *R. sphaeroides* | 601 | 577 | 12 | 1 | 3 | 7780 |
| *H. sapiens (chr 14)* | 43960 | 43372 | 490 | 35 | 8 | 1675 |
| *M. fijiensis* | 18111 | 17485 | 326 | 64 | 28 | 2006 |
| *M. graminicola* | 6442 | 6285 | 95 | 16 | 7 | 5188 |

Taking into consideration only the best hit creates the possibility of mis-calling the right link between two contigs in a scaffolding output dataset. Consider the following example. We ran SSPACE on the *S. aureus* dataset from the GAGE [76] project. All the contigs in the ground truth answer were enumerated from 1 to 170 following framework [40]. In the scaffolding, SSPACE placed the contig 102 (463 bp long) between contigs 19 (57049 bp long) and 20 (132320 bp long) and this situation was classified as producing two errors, namely, incorrect orientation and incorrect distance (see Figure 2.7). If we re-align contig 102 to the *S. aureus* genome using nucmer with the parameter of similarity score set to 97%, we notice that contig 102 has three potential placements: a) between contigs 19 and 20 with similarity score 98.49% and it is reversed, b) between contigs 79 and 80 with similarity score 99.78%, and c) between contigs 101 and 103 with similarity score 100% - the actual "best hit" placement chosen as a perfect contig in [40]. Thus, contig 102, in fact, has 3 copies under the similarity level of 97%, and the decision taken by SSPACE to place 102 between 19 and 20 is "not that wrong". Exactly the same mis-classification of the contig links in SSPACE output is done with contig 100 which is placed between contigs 22 and 23 by SSPACE. In fact, under the 97% threshold, one can observe 5 copies: a) between 22 and 23 with similarity 98.23%, b) between 94 and 95 with similarity 99.05%, c) between 97 and 98 with similarity 98.10%, d) between 99 and 101 with similarity 100%, and e) between 130 and 131 with similarity 98.10%. Again, contig 100 added two more errors to the evaluation. Additionally, [40] may report errors when repeated contigs are shared between different chromosomes.

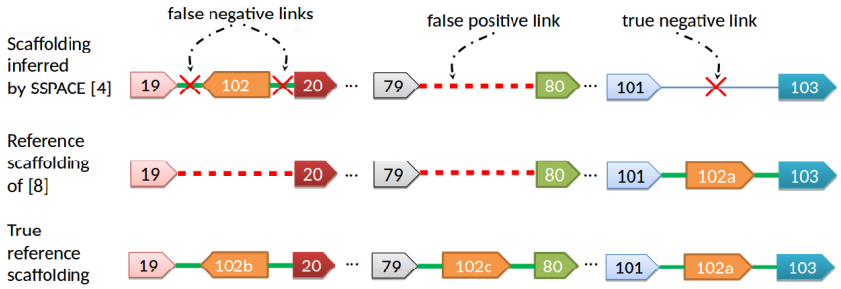



Figure 2.7 The reference scaffolding contains three copies of contig 102 (namely, 102a, 102b, and 102c). In the reference scaffolding of Hunt et al. [40] only the best hit is considered and two copies with a high identity level ($> 97\%$) are discarded. As a result the contig 102 is treated as erroneously placed by SSPACE between contigs 19 and 20 resulting in two false negative links. Similarly, since the contig 102c is missing, the link between contigs 79 and 80 is falsely treated as correct. Finally, the missing contig 102a is correctly classified.

Table 2.6 The number of links classified as correct and incorrect by [40] when applied to the perfect scaffoldings for three datasets *S. aureus*, *R. sphaeroides*, *H. sapiens (chr14)*.

| Dataset | Correct links | Incorrect links | Missclassified links |
|---|---|---|---|
| *S. aureus* | 157 | 15 | 8.7% |
| *R. sphaeroides* | 547 | 42 | 7.1% |
| *H. sapiens* (chr14) | 44219 | 969 | 2.1% |

As a result, the [40] framework will penalize even the "perfect" scaffolding treating as incorrect links connecting repeated contigs. Table 2.6 gives the number of correct and incorrect links inferred by the framework of [40]. For example, on *S. aureus* dataset, the mis-classification rate is almost 9%.

Although each contig, as is shown above, may have multiple copy numbers in genomes, most of the state-of-the-art scaffolding tools treat contigs produced by assembly tools as having only a single copy. Thus, in order to correctly evaluate scaffolding, one needs to consider all the possible placements of each repeated contig and do not report an error in case when a different contig copy is selected within the defined level of tolerance (we use 97%) rather than the best hit (100%).

As far as we are aware of, OPERA-LG [29] is the only scaffolding tool which handles repeated contigs, i.e. it outputs scaffolds where some of the contigs have multiple copies.

As for each contig, its copy number in the scaffolding and in the reference may differ, an additional challenge is emerging for evaluating the scaffolding. In [29], the performance of repeat aware scaffolding was assessed by the ability of the scaffolding tool to correctly infer the sequence of gaps where OPERA-LG placed repeats.

In this chapter, we propose a novel scaffolding evaluation framework which presents a unified approach for assessment of scaffoldings with and without repeated contigs. Our framework as compared to [40] provides an only quantitative measure (number of correct contig links) which is an advantage for an easier comparison of different tools performances. For a more detailed analysis of scaffolding results, one can use more other metrics/scores (see Section 2.3.3).

### 2.3.2 Exact scaffolding evaluation

In this section we formulate Scaffolding Evaluation Problem as an Integer Linear Program (ILP). We start with the formal definitions.

**Definition.** Let $C = \{c_i\}_{i \in I}$ be a set of instances of contigs. Note that multiple elements of $C$ may represent the same contig. Sequence of contig instances $S = (c_1^{o_1}, c_2^{o_2}, c_3^{o_3}, ..., c_n^{o_n})$ with the orientation $o_i \in \{+1, -1\}, i = 1, \ldots, n$, is called a *scaffold*. A *link* is an ordered pair of consecutive instances of contigs together with the estimated distance $d_{i,i+1}$ between them $(c_i^{o_i}, c_{i+1}^{o_{i+1}}, d_{i,i+1})$. We say that two links $l_i = (c_i^{o_i}, c_{i+1}^{o_{i+1}}, d_{i,i+1})$ and $l_j = (c_j^{o_j}, c_{j+1}^{o_{j+1}}, d_{j,j+1})$ are *equivalent*, $l_i \equiv l_j$, if $c_i^{o_i}$ and $c_j^{o_j}$ (respectively, $c_{i+1}^{o_{i+1}}$ and $c_{j+1}^{o_{j+1}}$ ) are instances of the same contig and $|d_{i,i+1} - d_{j,j+1}| < \delta \times b$, where $b$ is the library insert size and by default $\delta = 1$. Reverse complement links $(c_{i+1}^{-o_{i+1}}, c_i^{-o_i}, d_{i,i+1})$ and $(c_i^{o_i}, c_{i+1}^{o_{i+1}}, d_{i,i+1})$ are also defined to be equivalent.

Let *scaffolding* be a set of scaffolds. Consider a genome of a model organism (or any other organism for which a reliable reference exists). Let $R$ be the reference scaffolding and let $S$ be a scaffolding produced by a scaffolding tool. Ideally, a scaffolding tool should output a scaffolding $S$ which will be identical with $R$, or at least "as close as possible" to $R$. Intuitively, this means that one wants to maximize the number of *valid* links in $S$, i.e., links which are identical to links in $R$ or shortcuts of several sequential links in $R$, such that each

$S$-contig gets assigned to at most one $R$-contig and vice versa.

Let $L_S$ be the set of links of the scaffolding $S$. In contrast, let $L_R$ be the set of links between all pairs of contigs in the same scaffold of the reference scaffolding $R$. Then an $S$-link is valid if it is equivalent to an $R$-link with possibly skipped reference contigs.

We find optimal assignment between $S$ and $R$ using the following ILP:

$$
\begin{aligned}
&\sum_{k,p} z_{kp} \quad \to max \\
&\text{s.t.} \qquad \sum_{j} x_{ij} \le 1,\ \forall i \in R && (a) \\
&\qquad\quad \sum_{i} x_{ij} \le 1,\ \forall j \in S && (b) \\
&\qquad\quad 2z_{kp} \le x_{k_1p_1} + x_{k_2p_2},\ \forall k = (k_1, k_2) \in L_R, && (2.1) \\
&\qquad\quad \forall p = (p_1, p_2) \in L_S,\ p \equiv k && (c) \\
&\qquad\quad \sum_{k} z_{kp} \le 1,\ \forall p \in L_S,\ p \equiv k && (d) \\
&\qquad\quad x_{ij} \in \{0, 1\},\ \forall i \in R,\ \forall j \in L && (e) \\
&\qquad\quad z_{kp} \in \{0, 1\},\ \forall k \in L_R,\ \forall p \in L_S && (f)
\end{aligned}
$$

Binary variables $x_{ij} \in \{0, 1\}$ have the following meaning:

$$
x_{ij} = \begin{cases} 1, & \text{if } R\text{-contig } i \text{ and } S\text{-contig } j \text{ are instances of the same} \\ & \text{contig and are assigned to each other} \\ 0, & \text{otherwise.} \end{cases} \qquad (2.2)
$$

Conditions (a) and (b) from the ILP (2.4) guarantee that each contig $j \in S$ is assigned to at most one contig $i \in R$ and vice versa.

For any links $k = (k_1, k_2) \in L_R$ and $p = (p_1, p_2) \in L_S$, $k \equiv p$,

$$z_{kp} = \begin{cases} 1, & \text{if } x_{k_1p_1} = x_{k_2p_2} = 1 \\ 0, & \text{otherwise.} \end{cases} \tag{2.3}$$

Constraints (c) correspond to the relations between the variables $z_{kp}$ and the variables $x_{k_1p_1}$ and $x_{k_2p_2}$. It may happen that for a given link $k \in R$ there may be multiple links $p_1, p_2, ..., p_n \in S$ identical with $k$. To guarantee that at most one link $p_i$ from the scaffolding $S$ is matched with $k$, constraint (d) is introduced to the problem.

Maximizing the objective $\sum_{k,p} z_{kp}$ means to find an assignment of scaffold contigs to the reference contigs, i.e. assignment of variables $x_{ij}$ such that to maximize the number of contig links $k \in R$ matched with identical to them contig links from $S$.

### 2.3.3 Repeat-aware validation of scaffolders

**Benchmarks with repeats** In practice, it is frequently necessary to scaffold incorrectly assembled contigs. Since it is impossible to decide whether scaffolding of incorrect contigs is correct, validation tools (see, [40]) change the original contigs before feeding them to scaffolding tools. As a result, repeated contigs may interfere with validation and our first benchmark *B_rep* removes this interference by making repeated contigs have the same id.

Formally, following [40], we align the contigs produced by the Velvet assembler to the reference genome using nucmer [47]. But unlike [40], we consider all the nucmer alignment hits with $\alpha$-identity ($\alpha = 97\%$ by default) including partial alignments of at least $\lambda$ base pairs long ($\lambda = 200$ by default). The default value 97% is chosen in conformity with the common definition of a DNA repeat [88]. We use $\alpha$-threshold for assigning the same id to two similar contigs in the reference scaffolding. The benchmark *B_rep* consists of the same contigs as [40] but $\alpha$-similar contigs get the same id and the resulted reference scaffolding contains contig repeats.

Assemblers may produce contigs consisting of parts having different copy numbers. Therefore, we suggest modifying original contigs by splitting out partial repeats from contigs. The resulted benchmark *B_split*, on one hand, simplifies scaffolding since all mapping of

paired reads become correct and, on the other hand, it magnifies the problem of scaffolding repeats.

Formally, the benchmark *B_split* splits the alignments of *B_rep*-contigs into segments defined by alignment endpoints (see Figure 2.9). The segments of length less than $\lambda$ bp are dropped. The remaining segments form contigs without partial repeats. Then multiple repeated contigs get the same id. The *B_split* contigs are at least $\lambda$ bp long and no contig contains $\lambda$-long repeats. The pseudocode for the procedure generating *B_split* contigs is provided in Algorithm 2. Note that the algorithm for creating *B_split* benchmarks is stable since it outputs similar contig datasets for input assembly contigs produced by different tools. In Table 2.7 some basic information regarding Velvet and Allpaths-LG *B_split* contigs for *S. aureus* is presented. Figure 2.8 shows similar copy number distributions across the Allpaths-LG and Velvet contig datasets.

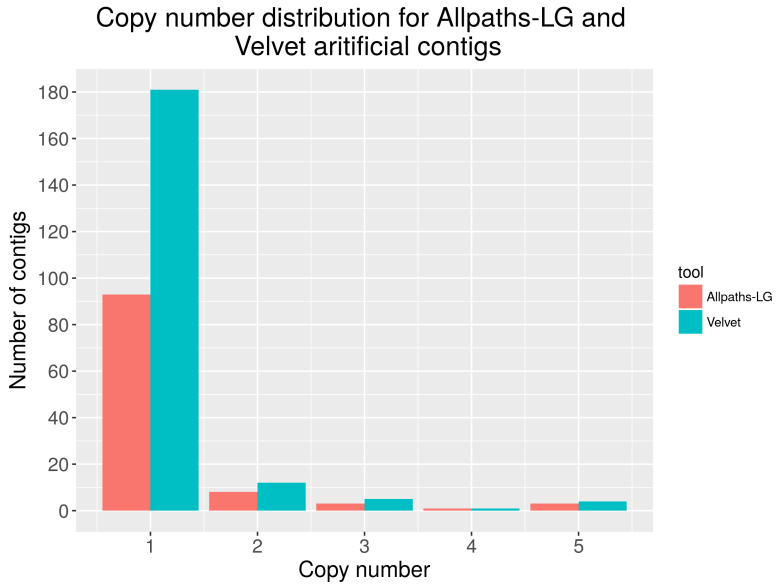



Figure 2.8 Copy number distribution

We built three repeat aware benchmarks based on Velvet assembly from the GAGE project: *S. aureus*, *R. sphaeroides*, *H. sapiens (chr 14)*. Also, we included two novel *in silico* generated (with the aid of iWGS pipeline [102]) fungi datasets: *M. fijiensis* and *M. graminicola*. In Table 2.5 copy number distribution and the total number of contigs for the

**Algorithm 2** Construction of artificial contigs

$\lambda \leftarrow$ minimum artificial contig length
$\alpha \leftarrow$ repeat similarity threshold
$\mathcal{R} \leftarrow \{r_1, r_2, ..., r_n\}$
$\mathcal{C} \leftarrow \{c_1, c_2, ..., c_n\}$
$\mathcal{C}_{art} \leftarrow \emptyset$
$D \leftarrow \{\}$
**for all** $r \in \mathcal{R}$ **do**
  $\mathcal{A} \leftarrow nucmer(r, \mathcal{C}, \alpha, \lambda)$
  $\mathcal{E} \leftarrow \emptyset$
  **for all** $a = (start, end) \in \mathcal{A}$ **do**
    $\mathcal{E} \leftarrow \mathcal{E} \cup \{start, end\}$
  **end for**
  **for all** $c \in \mathcal{C}$ **do**
    $D[c] \leftarrow D[c] \cup \{e \in \mathcal{E} |$ $e$ lays within an alignment $a \in \mathcal{A}$ of contig $c$ to the reference $r\}$
  **end for**
**end for**
**for all** $c \in \mathcal{C}$ **do**
  $p \leftarrow sort(D[C])$
  $p_1 \leftarrow p[1]$
  **for** $i = 2, 3, ..., |\mathcal{P}|$ **do**
    $p_2 \leftarrow p[i]$
    **if** $p_2 - p_1 \geq \lambda$ **then**
      $\mathcal{C}_{art} \leftarrow \mathcal{C}_{art} \cup c[p_1 : p_2]$
    **end if**
    $p_1 \leftarrow p_2$
  **end for**
**end for**
remove duplicate sequences from $\mathcal{C}_{art}$

*B_rep* dataset is presented. The same information for the *B_split* dataset is given in Table 2.8.

**Evaluation framework** We implemented our evaluation framework using Python programming language (version 2.7). We used nucmer version 3.1 for aligning contigs to the reference. Integer Linear Program (2.4) is solved with the aid of IBM CPLEX solver version 12.7. In order to evaluate a scaffolding $S$ given the reference genome, we perform the following sequence of steps:

Table 2.7 Comparison between artificial contigs produced based on Velvet and Allpaths-LG contigs. "# all contigs" – total number of contigs in the dataset; "# common contigs" – in Velvet (Allpaths-LG) column, number of Velvet (Allpaths-LG) contigs which are either identical, or contain one or more Allpaths-LG (Velvet) contigs, or are contained in an Allpaths-LG (Velvet) contig; "total contig length" – total length of all contigs in the dataset; "common contig length" – in Velvet (Allpaths-LG) column, total length of Velvet (Allpaths-LG) contigs which are either identical, or contain one or more Allpaths-LG (Velvet) contigs, or are contained in an Allpaths-LG (Velvet) contig; "reference length" – the length of the *S. aureus* genome; "JS divergence between copy number profiles" – Jensen-Shannon divergence computed between the distributions of copy numbers for the two datasets.

| | Velvet | Allpaths-LG |
|---|---|---|
| # all contigs | 203 | 108 |
| # common contigs | 176 | 98 |
| common contig length (Mbp) | 2.82 | 2.83 |
| total contig length (Mbp) | 2.83 | 2.84 |
| reference length | 2.94 | |
| JS divergence between copy number profiles | 0.005 | |

1. Take assembly contigs and produce perfect contigs as described in Section 4.1.

2. Build the reference scaffolding $R$

3. Run a scaffolding tool on the perfect contigs and a paired-end read dataset

4. Align the perfect contigs to the output scaffolding using nucmer (with exactly the same identity level which was used for building the artificial contigs in step 1).

5. Build the output scaffolding $S$

6. Solve the Scaffolding Evaluation Problem with input data $(S, R)$.

The flowchart of the evaluation tool is presented in Figure 2.10.

**Validated scaffolding tools** We ran the following scaffolding tools: OPERA-LG (version 2.0.6), OPERA (version 1.4), ScaffMatch (version 0.9), SOAPdeNovo2 [58] (version 2.04), BESST (version 2.2.5), BOSS [56] (latest version from GitHub), SSPACE (version 3.0) on the three artificial contig datasets (*S. aureus*, *R. sphaeroides*, *H. sapiens (chr 14)*) from

Table 2.8 The number of unique contigs and the number of repeats in three GAGE [76] datasets (*S. aureus*, *R. sphaeroides*, *H. sapiens (chr 14)*) and two fungi datasets (*M. fijiensis*, *M. graminicola*) obtained with the aid of iWGS pipeline [102]. Each contig dataset is obtained as described in Section 2.3.3 ($\alpha = 0.97, \lambda = 200$).

| Dataset | Total | Unique | Number of copies | | | Avg len |
|---|---|---|---|---|---|---|
| | | | 2 | 3 | $\geq 4$ | |
| *S. aureus* | 244 | 203 | 12 | 5 | 5 | 11720 |
| *R. sphaeroides* | 652 | 612 | 24 | 4 | 2 | 6845 |
| *H. sapiens (chr 14)* | 45035 | 44350 | 557 | 45 | 11 | 1635 |
| *M. fijiensis* | 19357 | 17781 | 686 | 177 | 132 | 1818 |
| *M. graminicola* | 7261 | 6875 | 218 | 40 | 22 | 4490 |

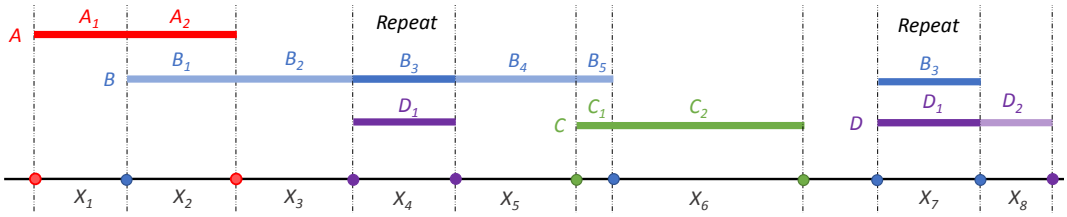



Figure 2.9 Original contigs $A$, $B$, $C$ are aligned to the reference with nucmer. Contig $A$ significantly overlaps with contig $B$, contig $B$ overlaps with contig $C$. Contig $B$ contains a repeated region $B_3$ which also aligns to a different place on the reference. Overlap length of $A$ and $B$ (which is $A_2 \equiv B_1 \equiv X_2$) is greater than the minimum contig length parameter $\lambda$, therefore we retain $X_2$. Overlap of $B$ and $C$ (which is $B_5 \equiv C_1$) is not included in the artificial contig dataset. Contig $D$ has a partial alignment to $X_4$ as well as contig $B$ has a partial alignment to $X_7$. The two segments $X_4$ and $X_7$ are collapsed into a single artificial contig $X_4$. Finally, the artificial contig dataset consists of $X_1, X_2, X_3, X_4, X_5, X_6, X_8$.

the GAGE project and two fungi artificial contig datasets (*M. fijiensis*, *M. graminicola*). The following Illumina paired-end read datasets were used: *S. aureus* - read length 37, insert size 3600; *R. sphaeroides* - read length 101, insert size 3700; *H. sapiens (chr 14)* - read length 101, insert size 2700, *M. fijiensis* - read length 100, insert size 1800; *M. graminicola* - read length 100, insert size 1800 (see Table 2.9 for more details). Most of the tools accept a user specified read aligner (for example, BWA [53], Bowtie [49] or Bowtie2 [48]. As BESST is better to be used with BWA (as per BESST documentation, `https://github.com/ksahlin/BESST`), we used it for all our experiments. ScaffMatch was run with Bowtie2 alignments).

**Evaluation metrics** Following [40] we provide a classification of erroneous links and report the corrected N50 and the number of correct links. Following [62, 56] we report

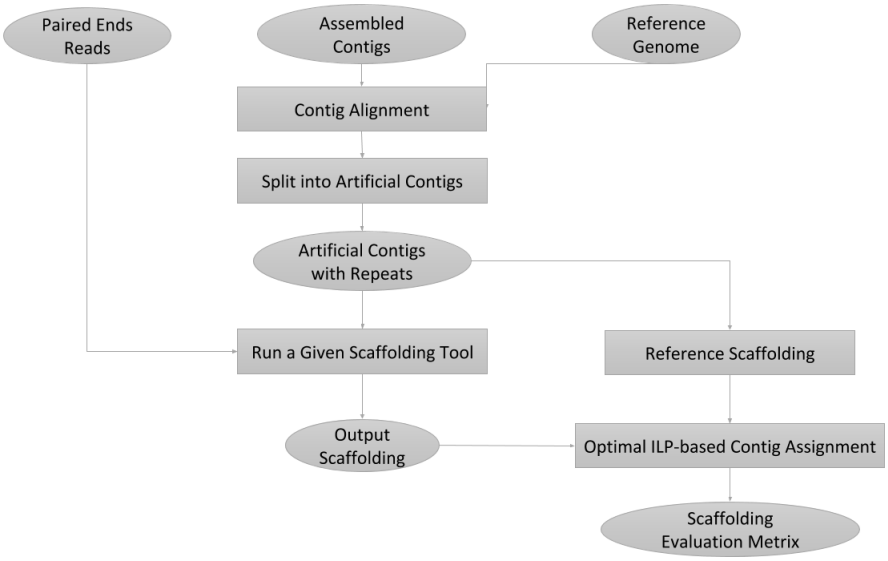



Figure 2.10 The flowchart of the scaffolding evaluation tool.

Table 2.9 The main parameters of the five datasets used in the experiments.

| | **Read length (bp)** | **Insert size (bp)** | **Std of insert size (bp)** | **Library orientation** |
|---|---|---|---|---|
| *S. aureus* | 37 | 3600 | 300 | outtie |
| *R. sphaeroides* | 101 | 3700 | 200 | outtie |
| *H. sapiens (chr 14)* | 101 | 2700 | 400 | outtie |
| *M. fijiensis* | 100 | 1800 | 150 | innie |
| *M. graminicola* | 100 | 1800 | 150 | innie |

sensitivity, PPV and F-score as well as a novel metric of corrected chain N50. Below we give details of each reported metric.

**Number of correct links** is the solution of the ILP (2.4) representing the number of optimally assigned correct contigs joins.

**Sensitivity, PPV, F-score.** Sensitivity is the number of correctly predicted links over the total number of reference links, positive predictive value (PPV) is the number of correctly predicted links over the total number of predicted links. F-score is the harmonic mean of sensitivity and PPV. Although it is widely used in the context of machine learning, it is also one of the metrics of choice in the evaluation of scaffolding tools [62, 56].

**Corrected N50**. Traditionally, corrected N50 is obtained by breaking the incorrect links and computing N50 metric on the resulting scaffolds using scripts from [76] or [36]. Unfortunately, neither of these scripts are repeat-aware and simply drop repeated contigs. Instead, we

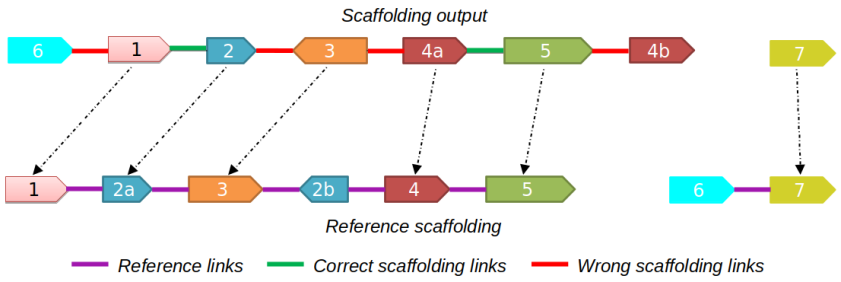



Figure 2.11 Classification of incorrect links. Contig 2 in the scaffolding output is assigned to contig 2a in the reference, contig 4a in the scaffolding output is assigned to contig 4 in the reference (marked with arrows). There are two correct links (marked with green) and 4 wrong links (marked with red). Link $(6, 1)$ connects contigs 6 and 1 coming from different reference sequences. Link $(2, 3)$ connects contigs 2 and 3 which are not in correct order/orientation. Jumping link $(3, 4a)$ connects contigs 3 and 4a which are not in correct order/orientation. Link $(5, 4b)$ connects contig 5 with an "extra" copy of contig 4 (namely 4b).

compute repeat aware corrected N50 using the optimal assignment of contigs and links produced by the ILP (2.4). We break all the unassigned links, sort the resulting scaffolds in descending order of their length, and output the length of the scaffold at which half of the genome is covered. Note that gaps between correctly linked contigs are counted in the length of the scaffold.

**Corrected chain N50** is a novel metric counting the number of contigs in the smallest corrected scaffold necessary to cover 50% of all contigs. The corrected chain N50 allows setting apart assessments of scaffolding and assembly.

**Classification of incorrect links** It is of particular interest for genome assembly practitioners to classify incorrect connections between contigs for an additional insight into drawbacks of scaffolding algorithms. Since the framework is based on links between contigs, we propose to classify errors exclusively in terms of incorrect links. This is in contrast to the previous approach of [40]. We distinguish the following types of incorrect links:

1. Linking an unassigned copy (i.e., a contig copy in the output scaffolding for which there is no contig assigned by ILP in the reference) - connecting an unassigned copy

of a contig (see link $(5, 4b)$ in the Figure 2.11);

2. Linking different references - connecting contigs coming from different reference sequences (see link $(6, 1)$ in the Figure 2.11);

3. Linking incorrect ends of contigs that should be connected. For example, link $(2, 3)$ in Figure 2.11 is mapped into link $(2a, 3)$ but the right end of 2a should be connected to the opposite end of contig 3. In general, there are 4 possible ways to connect between two contigs and these ways should be consistent between output and reference scaffoldings.

4. Links skipping contigs - the two linked contigs are in the correct order and orientation, but at least one contig is skipped between them. This corresponds to the case of "skipped" contigs in [40]. It is treated as a correct link in [40] and in our framework, it is treated as correct if the distance in bp between the two contigs is smaller than the library insert size $\delta$ and incorrect otherwise.

5. Links skipping contigs and linking incorrect contig ends - the combination of cases 3 and 4 (see link $(3, 4a)$ in the Figure 2.11).

6. Links in which contigs are joined correctly, but the number of base pairs between contigs is more than $\delta$ away from the real distance.

**Validation results** We applied our ILP (2.4) to obtain the number of inferred correct contig links by each scaffolder. For the largest data set *H. sapiens (chr14)* the ILP was scalable enough taking less than 40 min for the largest dataset (for *H. sapiens* dataset, ILP consists of $\geq$ 60000 variables and $\geq$ 20000 constraints, most of the time spent on building the ILP, and only about 0.1 seconds in average was enough for the solver to complete) on 2.5GHz 16-core AMD Opteron 6380 processors with 256Gb RAM running under Ubuntu 16.04 LTS (see Table 2.10).

Table 2.11 compares all scaffolding metrics for 5 *B_split* datasets for all 8 evaluated scaffolding tools (links skipping contigs are considered correct if the distance between two

Table 2.10 The average wall clock runtime of the scaffolding validation and the average wall clock time of the ILP (1) in seconds on the output scaffoldings obtained for the five datasets. As it can be clearly seen, most of the validation time is spent on building the ILP itself and a non-significant part is required for ILP to be solved.

| Dataset | *S. aureus* | *R. sphaeroides* | *H. sapiens (chr 14)* | *M. fijiensis* | *M. graminicola* |
|---|---|---|---|---|---|
| Runtime (s) | 0.150 | 0.630 | 2180 | 182 | 54.200 |
| Runtime ILP (s) | 0.160 | 0.002 | 0.119 | 0.112 | 0.159 |

link contigs is within one library insert size). The best results are achieved by SOAPdeNovo2 for 3 datasets and by ScaffMatch for 2 datasets. OPERA-LG with enabled repeated scaffolding (OPERA-LG/Rep.) finds consistently more correct contig links than OPERA-LG discounting repeats (OPERA-LG/No rep.). Note that the corrected N50 and the corrected chain N50 correlate well with the number of correct links – the median correlations over 5 samples are 0.97 and 0.95 respectively. This in contrast with much weaker correlation for the repeat unaware corrected N50.

The results on 5 *B_split* benchmarks on a more stringent version of our framework when all links skipping contigs are treated as incorrect can be found in Table S3 (see Supplementary materials of the paper online). We also evaluated all 8 scaffolding tools on 5 *B_rep* benchmarks (the contigs of *B_rep* benchmarks are generated by the scripts from [40]). The results for both versions (i.e., when links skipping contigs are treated and when they are treated as incorrect) on *B_rep* benchmarks are presented in Tables S5 and S6 correspondingly (see Supplementary materials of the paper online).

Repeated and short contigs significantly complicate scaffolding. In order to assess the performance of the scaffolding tools in presence of multiple repeat and short contigs we use two parameters $\alpha$ and $\lambda$ to control the complexity of the artificial datasets. When $\alpha$ decreases, then more contigs are assigned the same id and the number of repeats in the reference scaffolding grows. Similarly, when $\lambda$ decreases, then more contigs get fragmented and the number of short contigs grows. For example, artificial *M. graminicola* contigs for $\alpha = 0.91$ and $\lambda = 100$ contain almost 25% of repeats.

For each of the five datasets we generated 25 artificial contig datasets for different

values of $\alpha$ and $\lambda$ ($\alpha \in \{0.91, 0.93, 0.95, 0.97, 0.99\}$, $\lambda \in \{100, 200, 300, 400, 500\}$). Figure 2.12 illustrates the performance of the scaffolders in terms of F-score as a function of $\alpha$ and $\lambda$ for *S. aureus* (see also Supplementary Figures F2-F6). Clearly, as the complexity of the dataset grows, the performance of all the tools rapidly drops. Nonetheless, SOAPdeNovo2 and ScaffMatch handle the highly repeated datasets more efficiently starting with $\lambda = 200$. The performance of OPERA-LG is low for $\lambda = 100$ but as $\lambda$ grows it achieves a dramatically high level of F-score. Comparing the results of OPERA-LG with and without handling repeats, we can conclude that repeat handling is useful and it can efficiently boost up the performance of a scaffolding tool. In Table 2.12 we provide the classification of incorrect links for the *S. aureus* dataset ($\alpha = 0.97, \lambda = 200$). The most widely spread errors for each scaffolder are links skipping contigs and linking incorrect contig ends. It is obvious that these types of errors are caused by short and repeated contigs. In the next part of this chapter, we are building upon this idea. Namely, after the elimination of short and repeated contigs the scaffolding problem becomes a trivial task. But then, the repeated contigs may be re-inserted back into the scaffolds.

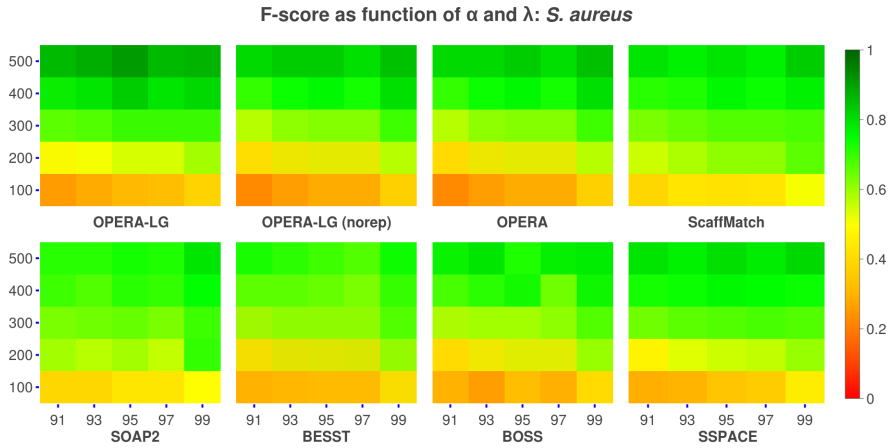



Figure 2.12 Performance (F-score) of the scaffolding tools depends both on $\lambda$ and $\alpha$. The $x$ axes of the heat maps denote the similarity level used to obtain the artificial contig datasets and the $y$ axes of the heat maps denote the minimum length of contig in the dataset. This figure displays results for *S. aureus*.

Table 2.11 The evaluation metrics on the five *B_split* benchmarks ($\alpha = 0.97$, $\lambda = 200$) as obtained from solving the ILP (1). Links skipping contigs are considered correct if their gap estimate is within one insert size. The bold font marks the best results.

| Datasets | OPERA-LG Rep. | OPERA-LG No rep. | OPERA | ScaffMatch | SOAP2 | BESST | BOSS | SSPACE |
|---|---|---|---|---|---|---|---|---|
| *S. aureus* | | | | | | | | |
| # correct | 151 | 134 | 134 | **180** | 142 | 124 | 146 | 135 |
| Sensitivity | 0.63 | 0.56 | 0.56 | **0.75** | 0.59 | 0.51 | 0.61 | 0.56 |
| PPV | 0.95 | **0.98** | **0.98** | **0.98** | 0.95 | 0.95 | 0.96 | 0.97 |
| F-score | 0.76 | 0.71 | 0.71 | **0.85** | 0.73 | 0.66 | 0.75 | 0.71 |
| Scaffold corrected N50 (K bp) | **579.4** | 574.4 | 574.4 | 568.9 | 180.0 | 564.4 | 573.9 | 563.8 |
| Corrected chain N50 | 25 | 23 | 23 | **45** | 8 | 22 | 11 | 9 |
| *R. sphaeroides* | | | | | | | | |
| # correct | 371 | 323 | 325 | **496** | 470 | 380 | 246 | 144 |
| Sensitivity | 0.58 | 0.5 | 0.5 | **0.77** | 0.73 | 0.59 | 0.38 | 0.22 |
| PPV | 0.93 | **1.0** | **1.0** | 0.94 | 0.98 | 0.99 | 0.99 | 0.96 |
| F-score | 0.71 | 0.67 | 0.67 | **0.85** | 0.84 | 0.74 | 0.55 | 0.36 |
| Scaffold corrected N50 (K bp) | 108.3 | 102.4 | 108.3 | **2521.3** | 904.5 | 644.9 | 32.9 | 23.1 |
| Corrected chain N50 | 8 | 7 | 7 | **55** | 30 | 22 | 2 | 1 |
| *H. sapiens (chr14)* | | | | | | | | |
| # correct | 29916 | 29666 | 29943 | 35672 | **39573** | 15504 | 10566 | 19943 |
| Sensitivity | 0.66 | 0.66 | 0.66 | 0.79 | **0.88** | 0.34 | 0.23 | 0.44 |
| PPV | 0.99 | **1.0** | **1.0** | 0.98 | 0.98 | 0.98 | **1.0** | **1.0** |
| F-score | 0.79 | 0.8 | 0.8 | 0.87 | **0.93** | 0.5 | 0.37 | 0.61 |
| Scaffold corrected N50 (K bp) | 102.1 | 102.2 | 114.5 | 42.0 | **120.0** | 51.1 | 5.9 | 16.8 |
| Corrected chain N50 | 23 | 22 | 25 | 13 | **49** | 1 | 1 | 2 |
| *M. fijiensis* | | | | | | | | |
| # correct | 9450 | 9362 | 9370 | 11877 | **12938** | 6037 | 7666 | 6333 |
| Sensitivity | 0.49 | 0.48 | 0.49 | 0.61 | **0.67** | 0.31 | 0.4 | 0.33 |
| PPV | 0.98 | 0.99 | 0.99 | 0.96 | 0.99 | 0.99 | 0.98 | **1.0** |
| F-score | 0.65 | 0.65 | 0.66 | 0.75 | **0.8** | 0.47 | 0.57 | 0.5 |
| Scaffold corrected N50 (K bp) | 30.0 | 30.1 | 30.1 | 28.4 | **31.6** | 23.4 | 15.7 | 14.8 |
| Corrected chain N50 | 4 | 4 | 4 | 7 | 9 | 1 | 2 | 1 |
| *M. graminicola* | | | | | | | | |
| # correct | 4996 | 4973 | 4981 | 5580 | **5762** | 4417 | 4864 | 3949 |
| Sensitivity | 0.69 | 0.69 | 0.69 | 0.77 | **0.8** | 0.61 | 0.67 | 0.55 |
| PPV | 0.99 | **1.0** | **1.0** | 0.99 | 0.99 | 0.99 | 0.99 | **1.0** |
| F-score | 0.81 | 0.82 | 0.82 | 0.87 | **0.88** | 0.75 | 0.8 | 0.71 |
| Scaffold corrected N50 (K bp) | 120.7 | 120.7 | 120.7 | 114.1 | **122.2** | 114.1 | 86.5 | 51.3 |
| Corrected chain N50 | 13 | 13 | 13 | 15 | **17** | 11 | 9 | 5 |

Table 2.12 Classification of incorrect links on *S. aureus* dataset.

| Datasets | OPERA-LG Rep. | OPERA-LG No rep. | OPERA | ScaffMatch | SOAP2 | BESST | BOSS | SSPACE |
|---|---|---|---|---|---|---|---|---|
| Linking an unassigned copy | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Linking different references | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Links skipping contigs | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 1 |
| Linking incorrect ends of contigs that should be connected | 0 | 0 | 0 | 1 | 2 | 1 | 2 | 0 |
| Links skipping contigs and linking incorrect contig ends | 3 | 3 | 3 | 2 | 5 | 3 | 4 | 3 |
| Links are joined correctly but distance is wrong | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 2.4 Repeat aware scaffolding

Increasing the length of sequencing reads allows assembling short genomes but assembly of long genomes remains one of the most interesting and challenging problems in bioinformat-

ics. As it was mentioned in the introduction of this dissertation, conventionally, the genome assembly process is divided into three stages: contig assembly, scaffolding, and gap filling. The task of scaffolding tools consists of orienting contigs, joining them into chains (called *scaffolds*) and providing distance estimates between neighboring contigs. Significant portions of long genomes are repeated confusing the assembly of the limited length reads which is manifested in numerous contig mis-assemblies and scaffolding errors. Repeats negatively affect scaffolding in two ways: (i) they cause fragmentation of contigs, forcing skipping short contigs and (ii) they produce false connections between non-adjacent contigs, significantly complicating the task of finding the neighboring contigs.

The common strategy to reduce the amount of incorrect joins caused by repeats consists of the following steps (ScaffMatch [62] and BESST [75]):

(1) filtering out repeats before the scaffolding process based on their coverage,

(2) scaffolding the remaining contigs

(3) inserting some of the filtered contigs in the scaffolding.

Usually the most effort is devoted to step (2), i.e., identification of correct joins between remaining contigs. There are several optimization formulations for step (2), e.g., maximizing the number of "concordant" read pairs (ScaffMatch [62], SILP3 [60], etc) or minimizing the number of "discordant" read pairs (OPERA [30], OPERA-LG [29]). Usually these formulations are NP-complete [30] and solved either heuristically by greedy-like approaches (SSPACE [9], ScaffMatch [62]), or exactly by Integer Linear Programming (ILP) or dynamic programming (OPERA and OPERA-LG).

The first repeat aware scaffolding tool OPERA-LG [29] instead adds as many copies of repeated contigs as necessary. The original OPERA problem formulation of minimizing the number of discordant read pairs is modified to include the repeated contigs in a parsimonious fashion. Thus, OPERA-LG simultaneously scaffolds both unique and repeated contigs. The potential repeats are determined based on the read coverage analysis which may not be accurate enough [29].

The previous part of this chapter showed that the state-of-the-art validation framework in [40] does not avoid significant errors in developing scaffolding benchmarks as well as estimating scaffolding quality [59]. One of the interesting discoveries about the first repeat-aware scaffolder OPERA-LG was that it does not exhibit significant improvement over the original OPERA and thus, it can not really compete with the best non-repeat-aware tools like ScaffMatch and SOAP2. In this chapter, we propose a new scaffolding tool *BATISCAF* (BAd conTIg filtering SCAFfolding) which follows the same steps (1-3) but emphasizing filtering out repeats (step (1)) instead of step (2). More precisely, we remove suspected repeats and short contigs which offer multiple alternatives for scaffolding chains. After removal of all confusing contigs, the scaffolding step (2) becomes trivial since no alternatives left. In the final step (3) of inserting removed repeat and short contigs, multiple copies of repeats are added to the appropriate slots between scaffolded contigs.

We validated BATISCAF on 5 benchmarks: three from the GAGE project [76]: *S. aureus*, *R. sphaeroides*, and *H. sapiens (chr14)* along with two fungi datasets: *M. fijiensis* and *M. graminicola*. Our experimental results show advantage of BATISCAF over existing scaffolding tools.

### 2.4.1 Repeat and Short Contig Filtering Problem

BATISCAF is a novel repeat aware scaffolding tool. The high-level idea behind it consists of 3 steps:

1. Filtering out potential repeats via ILP
2. Constructing backbone scaffolding for potentially unique contigs
3. Insertion of multiple copies of potential repeats into backbone scaffolds

**Repeat and Short Contig Filtering Problem.** Let $C$ be the set of contigs. For each contig $c_i \in C$ we produce two strands – the sequence for the first strand $c_i^+$ coincides with the contig sequence and the sequence for the second strand $c_i^-$ coincides with the reverse complement of $c_i$. We connect the corresponding vertices $c_i^+$ and $c_i^-$ with an *intra-contig*

edge $e_d = (c_i^+, c_i^-)$ of weight $\infty$ (a big enough number) (see Figure 1(a)). The set of all intra-contig edges is denoted as $E_d$. Note that each contig $c_i \in C$ has one and only one corresponding intra-contig edge $e_d \in E_d$.

We say that a paired-end read $r$ (e.g., Illumina technology) connects strands $c_i^{s_i}$ and $c_j^{s_j}$ where $s_i, s_j \in \{+, -\}$, $i \neq j$, of two distinct contigs if the forward read of $r$ is aligned to $c_i^{s_i}$ and the reverse read of $r$ is aligned to $c_j^{s_j}$. The vertices corresponding to these strands are connected with an inter-contig edge $e = (c_i^{s_i}, c_j^{s_j})$ of weight $\omega_{ij}$ if and only if they are connected with $\omega_{ij}$ paired-end reads supporting similar gap estimate (statistically inferred value for the distance in base pairs, gap estimation in BATISCAF follows [74]) between the contigs $c_i$ and $c_j$. Let $E$ denote the set of such edges $e$. Finally, the *scaffolding graph* $G = (V = C^+ \cup C^-, E \cup E_d, \omega)$ consist of vertices $V = C^+ \cup C^-$ corresponding to the contig strands connected with weighted intra- and inter-contig edges.

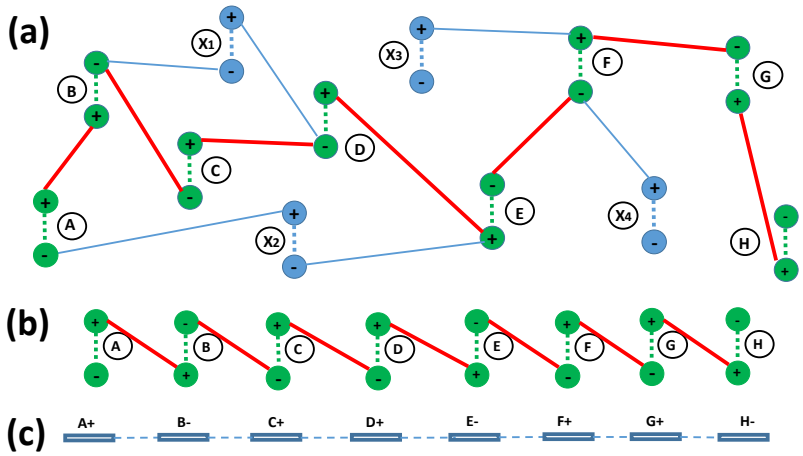



Figure 2.13 (a) The scaffolding graph $G$. Each contig is represented by two vertices (+ and −) corresponding to forward and backward strands. The intra-contig edges are dashed, the inter-contig edges are solid. (b) The scaffolding graph corresponding to a valid scaffold. The graph is a chain of alternating intra- and inter-contig edges. (c) The chain of contigs corresponding to the scaffolding graph of a scaffold. Each contig is either in the original orientation (+) or inverted (−).

Any valid scaffold corresponds to a scaffolding graph in which each vertex (strand) is incident to exactly one intra-contig edge and at most one inter-contig edge (see Figure

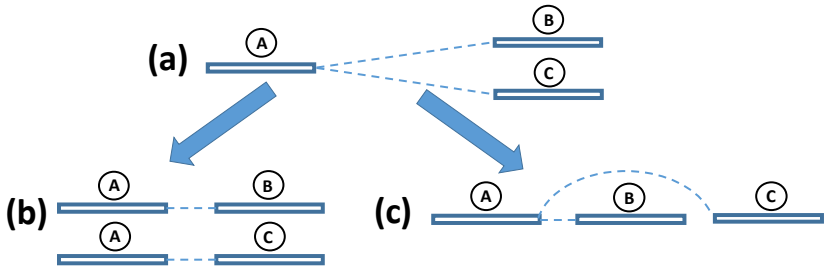



Figure 2.14 (a) A confusion triple: the same strand of contig $A$ is connected with two strands of contigs $B$ and $C$; The two possible scenarios causing the confusion: (b) The contig A is a repeat and another copy of contig $A$ is connected with $C$; (c) The connection from $A$ to $C$ jumps over the short contig $B$.

2.13(b)). Therefore, if a vertex in $G$ is incident to at least two inter-contig edges, then either one or both of them should be disregarded. Assuming no contig mis-assemblies such confusing edges are caused by either repeat or short contigs (see Figure 2.14). If there is no such confusion, the scaffolding graph $G$ is a set of valid scaffolds (with potentially missing links). So in order to avoid confusion and keep only reliable contigs we need to delete suspected repeat and short contigs. Usually, repeat contigs are also short (which is defined by the repeat monomer length of 150-400 bp [64]). Thus, the problem of repeat and short contig removal can be formulated as the following

> **Repeat and Short Contig Filtering (RSCF) Problem.** Given a scaffolding graph $G = (V, E \cup E_d, \omega)$ find minimum total length subset of contigs $X \subseteq V$ such that in subgraph $G'$ induced by $V \setminus X$, any vertex $v$ is incident to at most one inter-contig edge.

The solution of the RSCF problem represents a set of contigs whose removal from $G$ leaves a set of simple alternating paths and/or cycles consisting of intra-contig edges interspersed with inter-contig ones. The paths can be easily transformed into scaffolds. If there are no cyclic chromosomes, we need to transform them into paths. Therefore, from each cycle, we remove the least weight edge. After the least weight edge is removed from each cycle the resulting alternating paths can be easily transformed into a set of scaffolds

using a procedure similar to [62] (see Figure 2.13(c)). Such scaffolds are highly reliable since there is no confusion during their extraction. Clearly, solving the RSCF problem does not guarantee to exhaustively remove all the repeated contigs from $C$. Indeed, a contig with a high degree of its strands which is not necessarily a repeated one in the scaffolding graph would be a candidate for removal. The objective of RSCF problem ensures that the minimal length contigs are removed.

The RSCF problem can be viewed as a set cover problem in which elements correspond to confusion triples of contigs, i.e., contig triples with two $E$-edges connecting a single strand with two different strands of other contigs (see Figure 2.14) and sets corresponding to contigs – each contig $c$ covers all confusion triples containing $c$. Therefore, this is a set cover problem where each element belongs to at most three sets. Although such a problem is NP-complete, it can be 3-approximated with a primal-dual algorithm [90].

The RSCF problem can be solved efficiently if the scaffolding graph does not contain cycles. Therefore, instead of solving this problem exactly or approximately, we also propose a fast heuristic consisting of finding the maximum spanning tree $T(G)$ of the scaffolding graph $G$ (note that edges connecting the strands of the same contig will belong to $T$) and then finding the exact solution for $T$ (BATISCAF-MST).

**ILP Formulation for the RSCF Problem.** Since the RSCF problem is NP-hard, we propose to find the exact optimal solution using an Integer Linear Programming (ILP) approach.

Let $G = (V, E \cup E_d, \omega)$ be the scaffolding graph, where $V$ is the set of contig strands.

Let the length (in bp) of contig $u$ be denoted as $l_u$. We formulate the following ILP:

$$\begin{aligned}
&\sum_v l_v x_v \quad \to min \\
\text{s.t.} \quad & x_u + x_v \geq y_{uv},\ \forall e = (u,v) \in E \cup E_d && (a) \\
& y_{uv} \geq x_u,\ \forall e = (u,v) \in E \cup E_d && (b) \\
& y_{uv} \geq x_v,\ \forall e = (u,v) \in E \cup E_d && (c) \\
& \sum_{v:(u,v)\in E\cup E_d} y_{uv} \geq deg(u)\ -\ 2,\ \forall u \in V && (d) \\
& (\sum_{v:(u,v)\in E\cup E_d} y_{uv}) - x_u \leq deg(u) - 1,\ \forall u \in V && (e) \\
& x_u - x_v = 0,\ \forall e = (u,v) \in E_d && (f) \\
& x_u \in \{0,1\},\ \forall u \in V && (g) \\
& y_{uv} \in \{0,1\},\ \forall e = (u,v) \in V && (h)
\end{aligned} \tag{2.4}$$

Binary variables $x_u \in \{0,1\}$ have the following meaning:

$$x_u = \begin{cases} 1, & \text{if contig } u \text{ is marked as either a repeat} \\ & \text{or a possibly skipped short contig} \\ 0, & \text{otherwise.} \end{cases} \tag{2.5}$$

**Definition.** Let $u$ be a contig belonging to the scaffolding graph $G$. If by solving the ILP (2.4) we obtain $x_u = 1$ we call such contig *untrusted* (otherwise, *trusted*). The set of all untrusted contigs is denoted as $U$.

Binary variables $y_{uv} \in \{0,1\}$ have the following meaning:

$$y_{uv} = \begin{cases} 1, & \text{if the link } e = (u,v) \text{ is incident to} \\ & \text{an untrusted contig (either } u \text{ or } v) \\ 0, & \text{otherwise.} \end{cases} . \tag{2.6}$$

**Definition.** Let $e = (u,v)$ be an edge in the scaffolding graph $G$. We call the edge $e$

*untrusted* if it is incident to at least one untrusted contig.

The meaning of all the constraints is the following:

- Conditions (a), (b), (c) from the ILP (2.4) ensure that edges incident to an untrusted contig are also marked as untrusted.
- Condition (d) means that the degree of a trusted contig $u$ is at most 2 in the resulting scaffolds, i.e. no more than two edges which are not marked as untrusted are incident to it.
- Condition (e) means that if all the edges incident to a node $u$ are marked as untrusted than $u$ is untrusted as well.
- Condition (f) guarantees that two strands of any contig are both are either marked as trusted or untrusted.

Finally, the objective of the ILP (2.4) is to minimize the total length of the untrusted contigs while requesting that all the trusted contigs be connected into chains or cycles (i.e. their degree in the graph $G \setminus U$ is either 0, 1, or 2).

The solution of ILP (2.4) represents a set of untrusted contigs $U$. After we remove them from the scaffolding graph we obtain a graph $T = G \setminus U$ with all its connected components being either paths or cycles. We remove the least weight edge from each cycle. As a result, we obtain a set of alternating paths which can be translated into scaffolds following a procedure similar to [62]. In each scaffold, $S$ relative ordering and orientation of each contig is established.

**Most Likely Repeat and Short Contig Insertion.** The second stage of our algorithm which is constructing scaffolding corresponding to contigs left after removing confusing contigs (see Figure 2.13(c)) is trivial. The third stage of our approach is to insert the removed contigs from the set $U$ back into the scaffolds. Potential repeats identified in the first stage are inserted into the scaffolds as many times as it can be inferred from the scaffolding graph structure. For each scaffold $s \in S$ we create a *surrounding* graph $G_S = (S \cup N, E)$ which is a

subgraph of the scaffolding graph $G$ on the set of nodes representing contig strands in $S$ and all their neighboring nodes $N$, i.e. $\forall n \in N, \exists u \in S$, such that $e = (n, u) \in E(G)$ (see Figure 2.15a). In $G_S$, the orientation of each contig in $S$ is known and the relative order between contigs in $S$ is established. The same information is to be determined for the contigs in $N$. Surrounding graphs $G_{s_i}$ corresponding to different scaffolds $s_i \in S$ may share neighboring nodes, i.e. $N_k \cap N_p \neq \emptyset$ for some scaffolds $s_k$ and $s_p$. This fact determines the copy number of each repeated contig.

Next, we build a directed surrounding graph $\overrightarrow{G_S}$ where nodes represent oriented contigs and arcs encode the relative ordering of neighboring contigs. The orientation of each contig in $N$, as well as the relative ordering between any contig $n \in N$ and its neighbors from $S$ in $G_S$, can be determined in the following way. Let $e$ be an edge in the surrounding graph $G_S$ between a strand of contig $n \in N$ and a strand of a trusted contig $u \in S$. If the orientation of $u$ is determined to be "+" and $e$ is incident to the negative strand of $u$ then if $e$ is also incident to the positive strand of $n$ the orientation of $n$ is assigned to be "+" (otherwise "–"). A new arc from $n$ to $u$ is added to $\overrightarrow{G_S}$. In the same manner, the direction of each arc and the orientation of each contig is determined. For example, in the graph $G_S$ depicted in Figure 2.15a, the positive strand of the contig $X_1 \in N$ is connected to the negative strand of contig $B \in S$ (which has orientation "+"). Therefore, in the graph $\overrightarrow{G_S}$ depicted in Figure 2.15b, contig $X_1$ is assigned orientation "+" and there is an outgoing arc connecting it with $B$.

The directed graph $\overrightarrow{G_S}$ may not be acyclic because some of the newly introduced contigs (from the set $N$) into the scaffold are repeats. We have to identify the set of repeated contigs $R$ and for each contig $r \in R$ we replace it with several copies of itself. The resulting graph is acyclic, i.e. it represents a partial order. A minimal set of repeated contigs in $\overrightarrow{G_S}$ can be determined by solving the Minimum Feedback Vertex Set problem or any of its weighted versions (for example, using contig lengths as weights). It is known that this problem is APX-hard on directed graphs [26], i.e. it does not admit any polynomial time approximation schemes (PTAS). Therefore, we apply a simple greedy heuristic to determine the feedback

vertex set. Namely, we randomly pick a cycle $\mathcal{C}$ in $\overrightarrow{G_S}$, find the smallest length contig $c \in \mathcal{C}$ and remove $c$ from the graph. We assign a copy of $c$ to each of its neighbors in $\overrightarrow{G_S}$. We continue this procedure until the graph is acyclic.

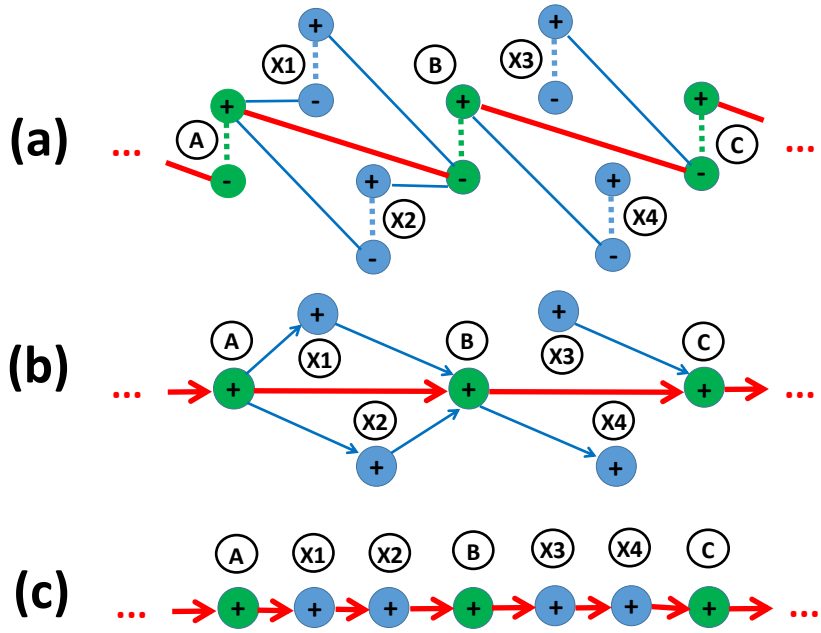



Figure 2.15 Insertion procedure. Contigs belonging to a backbone scaffold $S$ have green color; contigs which are candidates for insertion have blue color. a) A fragment of surrounding graph $\overrightarrow{G_S}$ with the chain of trusted contigs $A$, $B$, $C$ and neighboring contigs $X_1 - X_4$. b) The directed surrounding graph $\overrightarrow{G_S}$ corresponding to $G_S$. c) The scaffold $S$ augmented with contigs $X_1$, $X_2$, $X_3$, and $X_4$.

We refer to the transitive reduction of the directed acyclic graph (DAG) $\overrightarrow{G_S}$ as *spine*. The spine consists of all the nodes in the scaffold $S$ and some or all the $N$ nodes.

We define a *slot* $\mathcal{S} = (u, v)$ as a set of nodes between a pair of articulation nodes $u$ and $v$ in the spine of $\overrightarrow{G_S}$ which does not contain any other articulation point. For a slot $\mathcal{S}$ there can be only two cases (or a combination of them):

1. It is composed of a set of directed paths from $u$ to $v$ (e.g., in the Figure 2.15b, the slot ($A$, $B$) comprises two paths: $A \to X_1 \to B$ and $A \to X_2 \to B$);

2. It contains only "dangling" nodes attached either to $u$ or to $v$ (e.g., in the Figure 2.15b,

the slot ($B$, $C$) contains a dangling node $X_4$ attached to $B$ and another dangling node $X_3$ attached to $C$);

In the first case, for all the contigs belonging to $\mathcal{S}$ we identify their relative ordering by sorting them according to the distance from either $u$ or $v$. In the second case, a "dangling" node $\mathcal{D}$ may not necessarily belong to $\mathcal{S}$. For example, in Figure 2.15b, contig $X_3$ may belong to either slot $(B, C)$ or $(A, B)$ depending on the distance estimates between $X_3$ and $C$ and between $B$ and $C$. Without loss of generality, let's consider $\mathcal{D}$ to be connected with a contig $S_i \in S$ by an outgoing arc (e.g. contig $X_3$ has an outgoing arc to $C$). Contig $\mathcal{D}$ may be inserted into one of the slots $(S_{i-1}, S_i)$, $(S_{i-2}, S_{i-1})$, etc. For all such slots $\mathcal{S}_k = (S_{i-k-1}, S_{i-k})$, we estimate the probability $P_{\mathcal{S}_k} = P(\mathcal{D} \in \mathcal{S}_k)$ defined as

$$P_{\mathcal{S}_k} = F(x \leq d(\mathcal{D}, S_i) \leq x + y),$$

where

$$x = \sum_{p=0}^{k-1} (d(S_{i-p-1}, S_{i-p}) + l(S_{i-p})),$$

$$y = d(S_{i-k-1}, S_{i-k}) - l(\mathcal{D}),$$

$l(z)$ is the length of contig $z$, $d(z_1, z_2)$ is distance estimate between two contigs $z_1$ and $z_2$, $F$ is the normal distribution $N(\mu, \sigma^2)$ with $\mu$, $\sigma^2$ being the mean and standard deviation of the library insert size.

The dangling contig $d$ is assigned to the slot $\mathcal{S}_{k_0}$, where $k_0 = argmax_k P_{S_k}$. After all the contigs in $N$ are assigned to the corresponding slots, we get the set of scaffolds $S'$ augmented with repeated and short contigs (see Figure 2.15c).

### 2.4.2 Results

**Datasets with repeats.** We used the five *B_split* datasets described in the previous part of theis chapter (see Validation results in 2.3.3). The following Illumina paired-end read

datasets were used: *S. aureus* - read length 37, insert size 3600; *R. sphaeroides* - read length 101, insert size 3700; *M. fijiensis* - read length 100, insert size 1800; *H. sapiens (chr14)* - read length 101, insert size 2700; *M. graminicola* - read length 100, insert size 1800. In the Table 2.13 the basic characteristics of the contig datasets are presented.

Table 2.13 The basic characteristics of the simulated contig datasets: "avg len"- average contig length, "# unique" - the number of unique contigs (no copies counted), "# total" - the total number of contigs including copies, "# repeats" - the number of repeated contigs, "# max CN" - the maximum copy number, i.e. the number of times the most abundant contig is encountered in the dataset.

| | avg len | # unique | # total | # repeats | # max CN |
|---|---|---|---|---|---|
| *S. aureus* | 13.9 K | 203 | 244 | 22 | 5 |
| *R. sphaeroides* | 7.2 K | 612 | 652 | 30 | 5 |
| *H. sapiens (chr14)* | 1.7 K | 44350 | 45035 | 613 | 5 |
| *M. graminicola* | 4.7 K | 6875 | 7261 | 280 | 9 |
| *M. fijiensis* | 2.0 K | 17781 | 19357 | 995 | 28 |

**Performance metrics.** We used the following evaluation metrics in our comparison:

- number of correct contig links;
- sensitivity (or recall) and PPV (positive predictive value) - two scaffolding quality metrics introduced in [62] and used in [56]. They are defined as $TPR = \frac{TP}{P}$, $PPV = \frac{TP}{TP+FP}$, where $TP$ is the number of correct contig joins in the output of the scaffolder (true positives), $FP$ be the number of erroneous joins (false positives), and $P$ is the number of potential contigs that can be joined in scaffold (equal to the number of contigs minus the number of reference sequences). We also report F-score equal to the harmonic mean of $TPR$ and $PPV$;
- Corrected N50 which is the length of contigs in the smallest corrected scaffold necessary to cover 50% of all contigs [76].

**Validation results.** We compared our tool with other five state-of-the-art stand-alone scaffolders: OPERA-LG, SSPACE, BESST, ScaffMatch, and BOSS. On each of the five datasets BATISCAF outperforms all other tools (Table 2.15). Notably, a large gap between BATISCAF and the remaining tools is observed on the GAGE datasets *S. aureus*, *R. sphaeroides*,

and *H. sapiens (chr14)* in terms of both F-score and corrected N50 metrics. Indeed, on *S. aureus* it identified 33 %, on *R. sphaeroides* 23 %, and on *H. sapiens (chr14)* 34 % more correct contig links than the next top competitor (ScaffMatch on the first two datasets and OPERA-LG on the third one) (see Table 2.15).

BATISCAF scaffolds for the fungi datasets are also of a better quality, although it did not aggressively join contigs as on the GAGE datasets. However, a small improvement over ScaffMatch in terms of F-score is compensated by more contiguous scaffolds as it is suggested by the corrected N50 results (see Table 2.15).

The runtime of BATISCAF on the largest *H. sapiens (chr14)* dataset containing 44350 distinct contigs is reasonable ($\approx$ 80 minutes) and comparable with runtime of other tools and the wall clock time spent on solving the ILP (2.4) is 16 seconds. We used CPLEX (version 12.7) for solving the ILP (2.4). All the experiments were run on 2.5GHz 16-core AMD Opteron 6380 processors with 256Gb RAM running under Ubuntu 16.04 LTS.

Further, we also used the well-established standard evaluation framework [40] to confirm the advantage of BATISCAF over the competitor tools. As we mentioned previously, it does not take into account repeats and chooses only the "best" placement of each contig in the reference ground truth scaffolding. We generated artificial contigs for the same five datasets using the scripts from [40] (`https://github.com/martinghunt/Scaffolder-evaluation`). Note, that these contigs are different from the ones we used in the repeat aware evaluation. The results in Table 2.14 suggest that BATISCAF is a top performer even in repeat unaware settings.

**Conclusions.** We presented a novel highly performing repeat-aware scaffolding tool BATISCAF (and its faster version BATISCAF-MST). Our tool solves the scaffolding problem in an innovative way. Instead of tackling it directly, BATISCAF first solves the problem of minimal length repeat and short contig removal after which the problem of scaffolding becomes trivial. The remaining contigs are connected into very reliable scaffolds which are then augmented with the previously removed contigs. The procedure for insertion of short and repeated contigs into the scaffolds detects the necessary number of times each repeated

contig to be inserted. For each contig copy, it finds the most likely slot for insertion.

We validated BATISCAF on 5 benchmarking datasets and compared it to other state-of-the-art scaffolders. Our experiments showed that BATISCAF is the top performer in terms of F-score and corrected N50 on all the datasets.

The future work on BATISCAF will be focused on providing the users the possibility to scaffold contigs using multiple libraries (including second and third generation sequencing).

Table 2.14 The evaluation results using the standard repeat unaware framework [40].

| Datasets | BATISCAF | | OPERA-LG | ScaffMatch | BESST | BOSS | SSPACE |
|---|---|---|---|---|---|---|---|
| | NO-MST | MST | | | | | |
| *S. aureus* | | | | | | | |
| # correct | **147** | 145 | 119 | 141 | 105 | 128 | 107 |
| F-score | **0.90** | 0.89 | 0.81 | 0.88 | 0.75 | 0.85 | 0.75 |
| *R. sphaeroides* | | | | | | | |
| # correct | 492 | **496** | 333 | 483 | 371 | 231 | 141 |
| F-score | 0.89 | **0.90** | 0.73 | 0.89 | 0.78 | 0.57 | 0.39 |
| *H. sapiens (chr14)* | | | | | | | |
| # correct | **35593** | 34498 | 29172 | 34741 | 15201 | 27432 | 19397 |
| F-score | **0.88** | 0.86 | 0.80 | **0.88** | 0.52 | 0.77 | 0.62 |
| *M. fijiensis* | | | | | | | |
| # correct | 11519 | 11392 | 9376 | 11373 | 6053 | **11644** | 6597 |
| F-score | 0.76 | 0.76 | 0.70 | 0.77 | 0.51 | **0.79** | 0.55 |
| *M. graminicola* | | | | | | | |
| # correct | 5145 | 5114 | 4687 | 5126 | 4184 | **5154** | 3846 |
| F-score | **0.89** | 0.88 | 0.85 | **0.89** | 0.80 | **0.89** | 0.76 |

Table 2.15 The evaluation metrics on the five datasets ($\alpha = 0.97$, $\lambda = 200$) as obtained from solving the ILP (2.4). The bold font marks the best results.

| Datasets | BATISCAF NO-MST | BATISCAF MST | OPERA-LG | ScaffMatch | BESST | BOSS | SSPACE |
|---|---|---|---|---|---|---|---|
| *S. aureus* | | | | | | | |
| # correct | **172** | 169 | 109 | 129 | 87 | 94 | 105 |
| Sensitivity | **0.71** | 0.70 | 0.45 | 0.54 | 0.36 | 0.39 | 0.44 |
| PPV | **0.85** | 0.84 | 0.69 | 0.70 | 0.66 | 0.62 | 0.76 |
| F-score | **0.77** | 0.76 | 0.54 | 0.61 | 0.47 | 0.48 | 0.56 |
| Scaffold corrected N50 (K bp) | **245.3** | 228.4 | 113.3 | 159.8 | 112.2 | 112.2 | 159.4 |
| *R. sphaeroides* | | | | | | | |
| # correct | 500 | **501** | 333 | 405 | 358 | 196 | 114 |
| Sensitivity | 0.77 | **0.78** | 0.52 | 0.63 | 0.56 | 0.30 | 0.18 |
| PPV | 0.87 | 0.86 | 0.84 | 0.91 | **0.94** | 0.79 | 0.76 |
| F-score | **0.82** | **0.82** | 0.64 | 0.69 | 0.70 | 0.43 | 0.29 |
| Scaffold corrected N50 (K bp) | **679.2** | 361.5 | 85.7 | 173.1 | 209.8 | 28.5 | 21.2 |
| *H. sapiens (chr14)* | | | | | | | |
| # correct | **30916** | 30092 | 23168 | 20780 | 12376 | 5280 | 12096 |
| Sensitivity | **0.69** | 0.67 | 0.51 | 0.46 | 0.27 | 0.12 | 0.27 |
| PPV | **0.79** | 0.76 | 0.77 | 0.57 | **0.79** | 0.50 | 0.61 |
| F-score | **0.74** | 0.71 | 0.61 | 0.51 | 0.40 | 0.19 | 0.37 |
| Scaffold corrected N50 (K bp) | **33.4** | 25.1 | 17.4 | 19.4 | 11.0 | 4.3 | 9.9 |
| *M. fijiensis* | | | | | | | |
| # correct | 10397 | **11019** | 7913 | 10095 | 5442 | 6093 | 5099 |
| Sensitivity | 0.54 | **0.57** | 0.41 | 0.52 | 0.28 | 0.32 | 0.26 |
| PPV | 0.84 | 0.85 | 0.82 | 0.82 | **0.89** | 0.78 | 0.80 |
| F-score | 0.66 | **0.68** | 0.55 | 0.64 | 0.43 | 0.45 | 0.39 |
| Scaffold corrected N50 (K bp) | **26.9** | 25.9 | 16.9 | 24.5 | 15.6 | 11.3 | 11.0 |
| *M. graminicola* | | | | | | | |
| # correct | 5330 | **5455** | 4490 | 5237 | 4184 | 4354 | 3522 |
| Sensitivity | 0.74 | **0.75** | 0.62 | 0.72 | 0.58 | 0.60 | 0.49 |
| PPV | 0.93 | 0.93 | 0.89 | 0.93 | **0.94** | 0.89 | 0.89 |
| F-score | 0.82 | **0.83** | 0.73 | 0.81 | 0.72 | 0.72 | 0.63 |
| Scaffold corrected N50 (K bp) | **101.9** | 97.5 | 64.8 | 89.2 | 75.7 | 56.2 | 38.2 |

# PART 3

# ORF ASSEMBLY

## 3.1 Introduction

RNA-Seq has become one of the most popular technologies due to its broad applicability to many biological problems. The most widespread application of RNA-Seq is transcriptome expression estimation; however, due to the fact of being relatively cheap, it can be successfully used for transcriptome sequencing, for example. Other emerging applications of RNA-Seq are the estimation of protein activity levels and estimation of pathway expression (see Figure 3.1). For a more accurate estimation of protein/pathway expression, one needs to accurately reconstruct the whole set of transcripts and their frequencies.

A study that sets the pace for transcript reconstruction using database was carried by [97]. Their whole genome assembly tool ORFome consists of three steps: first, each read is assessed individually and the putative open reading frames (ORFs) are annotated; then the annotated ORFs are assembled into a collection of peptides using a modified EULER assembly method and finally the assembled peptides are used for the database searching of homologs. A major difference between the ORFome assembly approach and the conventional whole genome assembly is that the former approach conducts gene annotation using all six frame translations, followed by the assembly of identified short peptides, whereas the latter approach conducts gene annotation after assembly of DNA sequences. The algorithm implemented in ORFome takes a different approach compared to just clustering a set of sequence reads like in [95]). However, the unsupervised clustering of sequence reads also has the limitation of clustering the input reads based only on k-mer frequencies in the 'short' reads without assembly.

In [80], the authors take a completely new approach proposing a method to reconstruct new mRNA transcripts from short sequencing reads with reference information of known

transcripts in existing databases. They used some prior knowledge in the form of transcript annotations in RefSeq database to define exon boundaries and fill in the transcript regions not covered by sequencing data. Their approach was validated on short sequences reads data from transcriptome and they were able to identify thousands of transcripts not previously annotated in RefSeq. Those transcripts were much longer than the one created by methods such as Trinity[34] which assume no prior knowledge and 73% of these new transcripts found supports from UCSC Known Genes [39], Ensembl [27] or EST transcript annotations.

The state-of-the-art RNA-Seq assembler Trinity [34] is able to produce long contigs which contain open reading frames (ORFs) (Trinity is bundled with Transdecoder, a script that inputs a set of contigs and outputs a set of ORFs). However, as experiments show [14], many ORFs produced by Trinity + Transdecoder are non-complete (see Figure 3.2). Trinity as a De Bruijn graph-based assembler uses 25-mers for constructing contigs; therefore, it may not join some of the reads into a connected component (see [34] for more details). Using $k$-mers with $k < 25$ may result in much noisy output, i.e. in this case an RNA-Seq assembler can produce a lot of non-existing contigs.

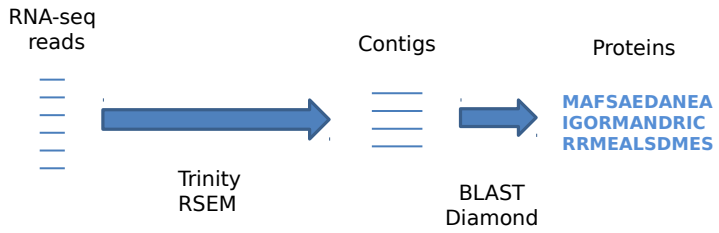



Figure 3.1 RNA-Seq data analysis flow.

We can ask the following question: is it possible to assemble full ORFs from partial ones? In this part of the dissertation, we propose DORFA, a tool for assembly of complete ORFs from partial ones based on a protein database.

We applied DORFA to a mollusk dataset (*Dendronotus iris*) and obtained 337 new full ORFs. Validation of the new ORFs was performed by comparison with the complete ORFs obtained by Trinity + Transdecoder.


Figure 3.2 Full ORF, 3'-partial (i.e. missing 3'-end), 5'-partial (i.e. missing 5'-end), and internal ORF.

## 3.2 DORFA: Database-guided ORFeome Assembly method from RNA-Seq Data

### 3.2.1 Methods

Two partial ORFs with an overlap may be concatenated (joined) if mapping to a protein database (ORF database) they correspond to the same protein (ORF of a related species). However, if the alignment of the two overlapping partial ORFs doesn't include significant parts from both ORFs then it is unlikely that these two ORFs came from the same real complete ORF. Thus, a chain of partial ORFs starting from a 3'-partial and ending with a 5'-partial ORF (with possibly some intermediate partial internal ORFs) with overlaps between adjacent (neighbor) partial ORFs represents a candidate complete ORF.

We must have a measure of certainty that a join of two partial ORFs is meaningful, i.e. that there exists a protein to which potentially a newly assembled complete ORF may be translated. We fully rely on the $E$-value which is a measure that assesses the significance of a local alignment of a sequence query to a database. Recall the definition of the E-value:

$$E = Kmne^{-\lambda S'} \tag{3.1}$$

Thus, the significance of a match linearly depends on the size of the query and the database and it exponentially increases when a higher bit score is achieved. $E$-value is attributed to a pair of segments (one is from the query and the other is from the database) whose score cannot be increased by extension or trimming, usually referred to as *high-scoring segment*

*pair* (HSP). We will call the query segment of an HSP *matching segment*.

We define a pair of partial ORFs *significant* if the significance $E_{AB}$ of their join exceeds the significance of each of the partial ORFs $E_A$ and $E_B$, or more formally:

$$E_{AB} < min\{E_A, E_B\} \tag{3.2}$$

For a significant ORF pair we additionally require the following two conditions:

1. the length of the overlap be a multiple of 3. This condition ensures that the resulting ORF is a valid ORF (it can be translated to a protein) and both partial ORFs are in-phase. It is very important to maintain this condition because otherwise, we lose the initial translational information (for example, a 3'-partial ORF after shifting to a different phase may lose the start codon).

2. the overlap between two partial ORFs be mapped to a protein in the database, i.e. it must be a part of a matching segment. This condition is crucial for a meaningful join because otherwise, two partial ORFs can have an overlap by chance.

Denote by $\alpha$ the significance threshold used for mapping ORFs to a database. A matching segment is called $\alpha$-significant if the $E$-value of its corresponding HSP is less than $\alpha$. A significant ORF pair matching to a protein $P$ is of one of the following three types:

1. Both partial ORFs have $\alpha$-significant matching segments with $P$ that get fused after the ORF concatenation. The formula 3.3 must hold; otherwise, we do not consider the match as a strong evidence of joining the two ORFs.

2. Only one of the two partial ORFs has an $\alpha$-significant matching segment $M$ with $P$. Without loss of generality, consider that $M$ belongs to the ORF $A$ and $A$ and $B$ have an overlap. In this case, $M$ is extended after the concatenation of $A$ and $B$ and the ORF pair gets a higher bit score $S'$ than the bit score of $A$. However, the condition $S' > S$ is not strong enough since we have to account for the lengths of both $A$ and $B$. Thus, if we put $E_B = \infty$, the formula 3.3 still holds.

3. Neither of the two ORFs have $\alpha$-significant matching segments with $P$. However, when joined together, the two partial ORFs $A$ and $B$ may become an $\alpha$-significant pair. Note that in this case, the matching segment does not have a high $E$-value score. Set $E_A = E_B = \infty$ and the formula 3.3 is satisfied.

We call a significant ORF pair *valid* if it consists of either i) 3'-partial, 5'-partial; ii) 3'-partial, internal; iii) internal, internal; iv) internal, 5'-partial. ORFs. So a significant pair of a 3' - partial and a 5' - partial ORF is a candidate to be a full ORF that was not constructed by the RNA-Seq assembler. As well, a sequence of partial ORFs, for which the first and the last ORFs are 3'- and 5'-partial correspondingly, but all the intermediary ORFs are internal ones (in other words, a sequence of partial ORFs for which every pair of adjacent ORFs is a valid ORF pair) could be not assembled due to many factors, such as inconsistent coverage or the limitations of the assembly algorithm. We propose a new method called *DORFA* which improves on the top of RNA-Seq assembly algorithms. Namely, it uses non-complete ORFs to produce new complete ORFs that were missed by assembly programs such as Trinity, Bridger, etc.

Consider a digraph $G = (V, E)$, where $V$ is the set of all non-complete ORFs and the set of edges $E$ includes all valid ORF pairs, i.e. two vertices $x$ and $y$ in $G$ are connected by an edge $e$ if and only if:

1. $x$ and $y$ are two overlapping ORFs with the length of the overlap multiple of 3;

2. $(x,\ y)$ is a significant pair, i.e. there exists a protein in the database for which there exists a matching segment of the joined ORF $xy$ covering the overlap;

3. $(x, y)$ is valid.

We call graph $G$ *ORF overlap graph*, or OOG (see Figure 3.4).

In the OOG $G$ we search for all valid paths, i.e we search for paths starting with a 3'-partial, ending with a 5'-partial, and possibly having one or more intermediate internal ORFs. Note that the OOG is not a tripartite graph since we admit edges between internal


Figure 3.3 A 3'-partial ORF has overlap of 9 nt with a 5'-partial ORF.

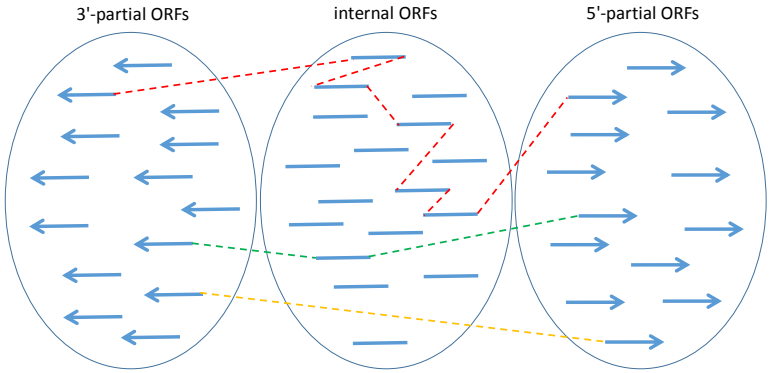



Figure 3.4 (a) Red path with 6 hops; (b) Green path with 2 hops; (c) Yellow path with 1 hop.

ORFs. Hopefully, the problem of enumeration of valid paths in an OOG is tractable, since its main bottleneck is in finding paths between all internal ORFs and in practice the subgraph of the OOG consisting of internal ORFs is not dense.

The set of valid paths in the OOG is then undergone a rigorous filtering procedure. Since we aim at keeping paths corresponding to an existing complete ORF and at filtering out paths corresponding to chimeric ORFs, we do the following:

- Map newly constructed ORFs (the ones that correspond to valid paths in OOG) to the protein/ORF database;
- Check whether the significance of the map to a protein of the new full ORF is higher (i.e. the $E$-value is less) than the significance of each of the significant ORF pairs comprising this full ORF. In other words, if the newly assembled full ORF corresponds to a path in the OOG with $E$-value of its edges $(E_1, E_2, ..., E_m)$, we require that for its $E$-value $E_{full}$ the following inequality to be satisfied:

$$E_{full} < min\{E_1, E_2, ..., E_m\} \tag{3.3}$$

### 3.2.2 Results

For our experiments we used 6 mollusk datasets (*Dendronotus*, *Flabellina*, *Hermissenda*, *Melibe*, *Pleurobranchaea*, *Tritonia*) consisting of RNA-Seq reads with read length 100 and insert size 180. We ran Trinity on these reads with the default $k$-mer length ($k = 25$). ORFs were searched using Transdecoder. The number of ORFs are presented in the Table 3.1.

Table 3.1 Number of complete and non-complete ORFs (3'-partial, internal, and 5'-partial) for 6 mollusk datasets.

| | 3'-partial | internal | 5'-partial | complete |
|---|---|---|---|---|
| *Dendronotus* | 4173 | 6376 | 6849 | 15826 |
| *Flabellina* | 4198 | 5080 | 6989 | 22189 |
| *Hermissenda* | 4594 | 7896 | 7777 | 15051 |
| *Melibe* | 4887 | 6596 | 8178 | 20684 |
| *Pleurobranchaea* | 4602 | 6101 | 8417 | 25569 |
| *Tritonia* | 4410 | 5616 | 6822 | 18107 |

We ran DORFA on all 6 datasets with the minimum overlap between two contigs 6 nt against two databases: first, we used Swiss-Prot protein database and, second, for each mollusk we used ORF database consisting of full and 5'-partial ORFs of the remaining 5 mollusks. ORF databases were transformed into protein databases using EMBOSS Transeq. We used DIAMOND to map ORFs to Swiss-Prot and the ORF databases. The Table 3.2 presents the number of new full ORFs reconstructed by DORFA against Swiss-Prot and the ORF databases.

Table 3.2 Number of reconstructed full ORFs for each of the 6 mollusk datasets.

| | Swiss-Prot | ORF database | intersection |
|---|---|---|---|
| *Dendronotus* | 1118 | 2015 | 472 |
| *Flabellina* | 1757 | 2630 | 734 |
| *Hermissenda* | 2066 | 3673 | 762 |
| *Melibe* | 1923 | 3235 | 823 |
| *Pleurobranchaea* | 2234 | 4014 | 1012 |
| *Tritonia* | 1737 | 3087 | 732 |

In order to evaluate DORFA performance from the functional point of view, we consid-

ered the increase in the number of protein homological groups after running DORFA on each mollusk dataset (against Swiss-Prot). We used Mnemonic IDs (MIDs) from the Swiss-Prot as the lowest homology level. The results are presented in the Table 3.3.

Table 3.3 Increase of number of Mnemonic IDs after running DORFA on Trinity contigs.

| | Trinity MIDs | Trinity + DORFA MIDs | MIDs increase (%) |
|---|---|---|---|
| *Dendronotus* | 4858 | 314 | 6.46 |
| *Flabellina* | 5336 | 380 | 7.12 |
| *Hermissenda* | 4423 | 470 | 10.62 |
| *Melibe* | 5058 | 435 | 8.60 |
| *Pleurobranchaea* | 5239 | 465 | 8.88 |
| *Tritonia* | 5052 | 463 | 9.16 |

# PART 4

# INFERENCE OF GENE EXPRESSION AND PATHWAY ACTIVITY LEVELS FROM RNA-SEQ DATA

## 4.1 Introduction

In this part, we are concentrated on the downstream RNA-Seq analyses (see Figure 1.1.2). RNA-Seq experiments use high-throughput sequencing to generate both sequence and abundance information about expressed gene isoforms. The two most common applications of RNA-Seq are to quantify gene/isoform expression levels in single samples and identify genes/isoforms that are differentially expressed between samples. Both applications are affected by noise introduced by library preparation and sequencing errors as well as ambiguities in read mapping. Chapter 4.2 is dedicated to describing IsoEM2 which is an improved combination of previously published tools IsoEM [68] and IsoDE [2].

Chapter 4.3 presents a novel quantification approach for the inference of pathways activity levels from RNA-Seq data. This approach is built upon a simple assumption which states that a pathway activity is positively correlated (or, in some sense, proportional) with the expression levels of all the enzymes which are active in it. In other words, the more enzymes belonging to a particular pathway are produced (i.e., the genes coding for those enzymes are more expressed), the more active that pathway is. The challenge in quantification of pathways activity levels is ambiguity created by some enzymes which work in the context of several pathways simultaneously. We present a novel tool called *EMPathways* which overcomes these challenges. EMPathways takes as input an RNA-Seq sample and outputs activity levels of pathways expressed in that sample. It is also designed to conduct differential pathway activity level analysis between several samples.

The ambiguities which were pointed out to create challenges for solving the above described quantification problems are of similar abstract nature. Both tools formalize them

as a maximum likelihood (ML) problem. Efficient algorithms based on the Expectation-Maximization (EM) approach are proposed.

## 4.2 IsoEM2: Quantification and Differential Expression Analysis of Gene and Isoform Expression from RNA-Seq data

Numerous tools for RNA-Seq quantification have been developed. A comprehensive assessment study [43] recently compared the most commonly used tools BitSeq [32], CEM [54], Cufflinks [87], eXpress [71], IsoEM [68], MMSEQ [89], RSEM [52], rSeq [77], Sailfish [69], Scripture [37], and TIGAR2 [67]. The results in [43] show that IsoEM [68] has one of the highest accuracies in all experiments (see also Supplementary Table 1) while being orders of magnitude faster than the other best-performing methods.

IsoEM is based on the *Expectation-Maximization (EM)* algorithm. Its probabilistic model takes into account the fragment length distribution (with mean/standard deviations specified by the user or automatically inferred when using paired-end reads) and incorporates base quality scores and strand information (if available). IsoDE [2] performs differential gene expression analysis using FPKM/TPM values estimated for bootstrap samples generated by re-sampling alignments. Although bootstrapping is computationally expensive, the high speed of IsoEM makes the running time of IsoDE practical.

Here we introduce IsoEM2, a new version of the IsoEM package that uses bootstrapping to infer confidence intervals for gene and isoform expression level estimates. The accompanying differential expression tool IsoDE2 has also been updated to take advantage of the fast in-memory bootstrapping of IsoEM2, resulting in speedups of over 200× over the original version in [2].

Compared to the previous versions, the main enhancements are the addition of confidence intervals for FPKM and TPM estimates produced by IsoEM2, the substantially faster running time for performing bootstrapping with IsoDE2, and the development of Galaxy wrappers making both IsoEM2 and IsoDE2 easy to use via a user-friendly web interface.

Table 4.1 Feature-based comparison of state-of-the-art RNA-Seq quantification tools. In the reference row, G stands for genome and T for transcriptome.

| Feature \ Tool | IsoEM2 | Kallisto | BitSeq | CEM | Cufflinks | eXpress | MMSEQ | RSEM | rSeq | Sailfish | Scripture | TIGAR2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Alignment free | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ |
| Reference | G/T | T | T | G | G | T | T | G/T | T | T | G | T |
| Confidence intervals | ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ |
| Indels | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ |
| Integrated DE | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ |
| GUI | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ |
| Multi-threading | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ |
| Frag. length distribution | ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |
| Sequence bias | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ |

### 4.2.1 **Software Features of IsoEM2**

Table 4.1 provides a feature-based comparison of the tools included in the assessment of [43] and the subsequently published Kallisto [11]. IsoEM2 offers a broad range of features and achieves one of the highest accuracies (Supplementary Table 1). It is also significantly faster than the other best-performing methods with the exception of Kallisto. On real datasets with over 100M read pairs, the HISAT2/IsoEM pipeline requires just over 1 hour to perform *both* read alignment and RNA-Seq quantification with 200 bootstraps using 16 CPU cores (Supplementary Table 3). Although the alignment-free Kallisto is 5-10× faster, its confidence intervals are substantially less reliable than those generated by IsoEM2 (see Tables 4.2 and 4.3).

**IsoEM2** IsoEM2 takes as input aligned RNA-Seq reads in (compressed) SAM format and outputs FPKM and TPM estimates of gene and isoform expression levels. Unlike the original implementation in [68], IsoEM2 computes confidence intervals for the estimates using the bootstrap method [24]. In each run IsoEM2 generates $N$ bootstrap estimates by in-memory re-sampling of the compatible read alignments. For each genomic feature (gene or isoform) and given confidence level $C \in (0, 1)$, the confidence interval $[c_{low}, c_{hi}]$ is

computed from the $N$ bootstrap estimates $\mathcal{B} = \{b_1, \ldots, b_N\}$ by setting $c_{low}$ and $c_{hi}$ equal to the $k$-th smallest, respectively $k$-th largest element of $\mathcal{B}$, where $k = \lfloor N(1-C)/2 \rfloor$. By default IsoEM2 uses $C = 0.95$ and $N = 200$, but these settings can be changed by the user. IsoEM2 generates four tab delimited output files for gene/isoform FPKM/TPM estimates. Each file includes a point estimate and the confidence interval for each feature. Additionally, it generates a compressed archive containing the bootstrap estimates used to compute the confidence intervals; these archives can be used for DE analysis using IsoDE2.

Besides the command-line version, IsoEM2 is also available with a user-friendly GUI through a Galaxy wrapper (Supplementary Figure 1). The wrapper can be downloaded from the Galaxy Tool Shed and installed on any local installation of Galaxy. The Galaxy tool is designed to work with both single-end and paired-end Illumina RNA-Seq reads as well as single-end Ion Torrent reads. It takes as input unaligned RNA-Seq reads and it maps them to a transcriptome reference selected by the user through the wrapper interface. The aligned reads are then automatically processed by IsoEM2. In addition to IsoEM2, the wrapper needs HISAT2 [45] to be installed on the Galaxy server.

**IsoDE2** IsoDE2, which is an extension of IsoDE [2], performs differential expression (DE) analysis using bootstrap samples generated by IsoEM2. To test for differential expression, the bootstrap expression level estimates generated for the two conditions by IsoEM2 are paired and used to compute for each gene a set of fold change estimates. A confident fold change $f$ is then computed for a user-specified significance level under the null hypothesis that fold changes obtained from bootstrap estimates are equally likely to be greater or smaller than $f$. For details on the format of IsoDE2 output files see Supplementary data.

### 4.2.2 Experimental Results

We conducted experiments to assess both the running time and the accuracy of confidence intervals of the updated IsoEM2/IsoDE2 suite and of the newly published Kallisto [11]. We only included Kalisto in this comparison since IsoEM was already shown to dominate in

accuracy and/or running time the methods included in the comparative assessment of [43].

The running time of IsoEM2 is much smaller compared to the bootstrapping step (called IsoBoot) of the old IsoDE. This is achieved by implementing the re-sampling in IsoEM2 based on internal data structures representing connected components of the read-isoform compatibility graph [68]. To assess the runtime improvement, we used two mouse retina RNA-Seq datasets from [44] with ∼100M unaligned read pairs each. On each dataset, generating 200 bootstrap samples with IsoEM2 has a speed-up of over 200× compared to IsoBoot (Supplementary Table 2). Although Kallisto is 5-10× faster (Supplementary Table 3), the HISAT2/IsoEM pipeline remains very practical, requiring just over 1 hour to perform read alignment and RNA-Seq quantification with 200 bootstraps using 16 CPU cores.

**Accuracy Comparison** To assess the accuracy of gene/isoform expression level estimates we computed the Pearson correlation with the known ground truth. To assess the quality of confidence intervals we used the percentage of genes for which confidence intervals contained the known ground truth. Since Kallisto does not output explicit confidence intervals, we ran it with the "-B 200" option to generate 200 bootstrap estimates and computed confidence intervals using the approach described in Section 4.2.1 for IsoEM2.

Tables 4.2 and 4.3 give Pearson correlations and confidence interval coverages for gene, respectively isoform expression level estimates obtained by IsoEM2 and Kallisto on datasets with 1M-10M simulated single-end reads from [43]. The confidence interval coverage for $C = 95\%$ reports how frequently the 95% CI estimated by IsoEM2 or Kallisto contains the true gene expression value. The accuracy metrics are computed both over the subset of genes/isoforms with non-zero ground truth, as in [43], and over all genes/isoforms. We note that, although Kallisto has similar Pearson correlations to IsoEM2 over the genes and isoforms with non-zero truth, its Pearson correlation is significantly lower than that of IsoEM2 when including isoforms with zero ground-truth. More importantly, for all considered sets of genes and isoforms, the coverage of 95%-confidence intervals computed by Kallisto is substantially lower than that of IsoEM2.

Table 4.2 Gene expression level estimation accuracy on simulated RNA-Seq datasets with 1M-10M single-end reads from [43].

.

| Number of reads | | 1M | 3M | 10M |
|---|---|---|---|---|
| All genes | | | | |
| Pearson correlation | IsoEM2 | 0.995 | 0.996 | 0.996 |
| | Kallisto | 0.84 | 0.84 | 0.84 |
| Confidence interval coverage for $C$=95% | IsoEM2 | 0.94 | 0.95 | 0.94 |
| | Kallisto | 0.80 | 0.78 | 0.78 |
| Genes with non-zero ground-truth | | | | |
| Pearson correlation | IsoEM2 | 0.96 | 0.98 | 0.98 |
| | Kallisto | 0.96 | 0.98 | 0.98 |
| Confidence interval coverage for $C$=95% | IsoEM2 | 0.74 | 0.77 | 0.72 |
| | Kallisto | 0.33 | 0.27 | 0.38 |

Table 4.3 Transcript expression level estimation accuracy on simulated RNA-Seq datasets with 1M-10M single-end reads from [43].

| Number of reads | | 1M | 3M | 10M |
|---|---|---|---|---|
| All isoforms | | | | |
| Pearson correlation | IsoEM2 | 0.98 | 0.98 | 0.98 |
| | Kallisto | 0.89 | 0.89 | 0.89 |
| Confidence interval coverage for $C$=95% | IsoEM2 | 0.95 | 0.95 | 0.94 |
| | Kallisto | 0.89 | 0.86 | 0.82 |
| Isoforms with non-zero ground truth | | | | |
| Pearson correlation | IsoEM2 | 0.90 | 0.94 | 0.96 |
| | Kallisto | 0.90 | 0.94 | 0.96 |
| Confidence interval coverage for $C$=95% | IsoEM2 | 0.59 | 0.64 | 0.61 |
| | Kallisto | 0.44 | 0.38 | 0.28 |

## 4.3 Metabolic Pathways Activity Levels: Inferring Relative Abundance and Differentially Expressed Pathways from RNA-Seq data

RNA-seq is a standard method for comparative analysis of gene transcription across different conditions. It supplanted a widely used microarray approach, enabling analysis of a much larger number of genes, including those represented in pools of transcripts from complex multi-species communities (metatranscriptomes). RNA-seq allows researchers to determine and compare gene transcription levels, as well as the transcriptional activity of distinct metabolic pathways. Diverse bioinformatic tools have been developed to facilitate

comparisons of RNA-seq data [22, 25, 42, 46, 66, 81, 84, 85, 86, 96]. Such tools include web-based services with automated pipelines that allow assessment of the metabolic properties represented in RNA-seq datasets. For example, the MAP platform [41] predicts genes expressed in samples, while also provides information about gene classification into orthology groups (see figure 4.1). Unfortunately, such pipelines fail to quantify transcripts in concert with the annotation step. We, therefore, propose an enhanced pipeline that combines the biochemical annotation with quantification analysis. For this purposes, we propose to use an expectation-maximization (EM) technique similar to one from IsoEM2 [61] (see the previous Chapter 4.2). We tested our algorithm using metatranscriptome data from marine bacterioplankton sampled during both the day and nighttime, and therefore likely exhibiting predictable variation in community transcription patterns.

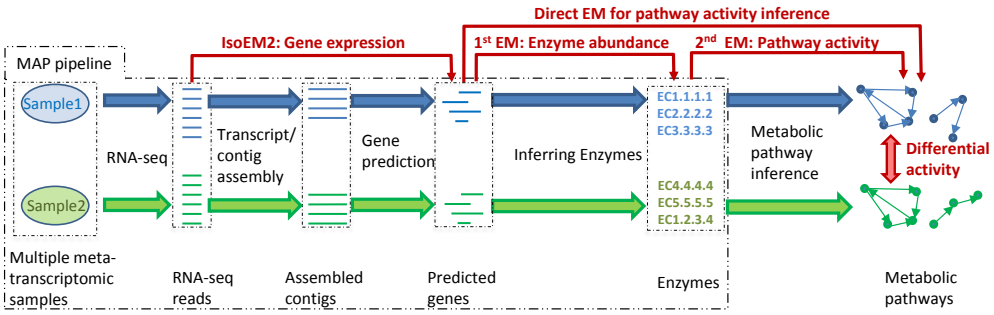



Figure 4.1 The pipeline MAP and the enhanced pipeline for quantification and differential analysis of the metabolic pathway activity. The quantification enhancements are drawn in red.

### 4.3.1 Methods

In this section, we describe the procedure of inferring metabolic pathway activity levels from RNA-Seq data for naturally occurring microbial communities. We also apply differential pathway activity level analysis similar to the non-parametric statistical approach described in [2], which was successfully applied for gene differential expression. The methods described here were implemented in the tool EMPathways using Python programming language.

A general meta-omic pipeline is described in Figure 4.1. Several metatranscriptomic

samples are sequenced on an Illumina Hi-Seq (2x150 bp) and the resulting reads are assembled into a set of contigs. Genes detected on the contigs are mapped to protein databases and enzymatic functions are inferred. Finally, the representation of metabolic pathways is inferred based on the presence/absence of enzymes within each pathway. The above generic pipeline has been described in [41]. This paper proposes to enhance the above pipeline with the inference of metabolic pathway activity levels using repeated maximum likelihood inference and resolution by the Expectation - Maximization (EM) algorithm. The proposed inferences are depicted in red in Figure 4.1.

**Inference of pathway activity levels** The first step is to estimate the abundances of the assembled contigs. The abundances can be inferred by any RNA-seq quantification tool. Here, we suggest using IsoEM2, as this method is sufficiently fast to handle Illumina Hiseq data and more accurate than kallisto [12]. The next proposed step is to estimate the abundance of enzymes based on contig abundances. For this step we propose so-called *1-st EM*. The *2-nd EM* is used to infer metabolic pathway activity levels based on inferred enzyme abundances and databases of metabolic pathways. The 1-st and the 2-nd EM's can be also integrated into a single *direct EM* that directly infers pathway activity levels from contig abundances. All components (1-st EM, 2-nd EM and direct EM) are built with similarities to IsoEM2 methodology.

**Expectation-Maximization approach.** Let $w$ be a pathway that is considered to be a set of enzymes. Traditionally, pathway maps are drawn as graphs with Enzyme Commission number nodes. Enzyme Commission numbers (EC numbers) have been widely used as a primary identifier for reconstructing the metabolic pathway from the complete genome. A more recent attempt to reconcile metabolic pathways with non-metabolic ones resulted in the introduction of the so-called KEGG Orthology. As in this paper, we are only interested in quantifying the activity of metabolic pathways, our primary goal of interest will be considering EC numbers and their contribution to pathways activity levels. We will, therefore, refer to the pathway $w$ as a set of EC numbers as the signature describing the biochemical activity occurring in a given microbial/viral community. A well-known fact is that different

EC numbers may take part in multiple pathways. Therefore, it is a challenging task to quantify the activity of each pathway in the condition of the uncertainty of whether enzymes belonging to a particular EC number participate in one particular metabolic pathway and not in another one.

Let $T$ be a random variable with values from the set of observed transcripts/contigs, and let $W$ be a random variable whose values belong to the set of pathways from the KEGG Pathway database. The probability of observing a contig $t$ is given by the following formula: $P(T = t) = \sum_{w \in W} f_w \; P(T = t \mid W = w)$, where $f_w$ stands for the frequency of the pathway $w$ which will be also referred as the *activity level* of $w$. We are interested in computing the distribution of frequencies on the set of pathways: $f_W = (f_{w_1}, f_{w_2}, ..., f_{w_{|W|}})$ Thus, in our model we adopt the following likelihood function:

$$L(f_W) = \prod_{t \in T} \left( \sum_{w \in W} f_w \; P(T = t \mid W = w) \right)^{a_t}$$

where $a_t$ denotes the abundance of $t$ estimated by IsoEM2. The corresponding log-likelihood is

$$l(f_W) = \sum_{t \in T} a_t \; log \left( \sum_{w \in W} f_w P(T = t | W = w) \right)$$

For each transcript, we associate a set of EC numbers. Namely, transcripts are aligned to a protein database and the set of all EC numbers $E$ corresponding to the matching proteins is retrieved. In general, more than one EC number is associated with every transcript (otherwise stated, $|E| \geq 1$). We apply the law of total probability to decompose further each term $P(T = t|W = w)$ participating in the log-likelihood:

$$P(T = t|W = w) = \sum_{e, t \in e} P(T = t, E = e|W = w) = \sum_{e : t \in e} P(E = e|W = w) \cdot P(T = t|E = e) \tag{4.1}$$

We use the uniform probability distribution over the set of EC numbers participating

in each pathway. This means the following:

$$P(E = e|W = w) = \begin{cases} \frac{1}{|w|}, & \text{if } e \in w \\ 0, & \text{otherwise} \end{cases}$$

Therefore, each probability term from the log-likelihood function may be written in the following form:

$$P(T = t|W = w) = \frac{1}{|w|} \cdot \sum_{e:t \in e, e \in w} P(T = t|E = e)$$

Further, the log-likelihood is transformed into the following:

$$l(f_W) = \sum_{t \in T} a_t \; log(\sum_{w \in W} f_w \cdot (\frac{1}{|w|} \cdot \sum_{e:t \in e, e \in w} P(T = t|E = e)))$$

Finally:

$$l(f_W) = \sum_{t \in T} a_t \; log(\sum_{w \in W} \frac{f_w}{|w|} \cdot \sum_{e:t \in e, e \in w} p_{te}),$$

where

$$p_{te} = P(T = t|E = e) = \frac{b_{te}}{\sum_{t' \in e} b_{t'e}}$$

In the last formula, $b_t$ are the bit-scores obtained from the alignment of assembled transcripts to the proteins of EC number $e$. We use the bit-score measure as the degree of reliability of each alignment. In other words, the probability of assigning a transcript $t$ to an EC number $e$ is proportional to the bit-score of the alignment $(t, e)$. Finally, we obtain:

$$l(f_W) = \sum_{t \in T} a_t \; log(\sum_{w \in W} \alpha_{tw} f_w),$$

where

$$\alpha_{tw} = \frac{1}{|w|} \cdot \sum_{e:t \in e, e \in w} p_{te}$$

In the log-likelihood function $l(f_W)$ the values $a_t$ are obtained by running IsoEM2 (or any other tool for transcript quantification). The values $\alpha_{tw}$ are computed from the corresponding tripartite graph (see Figure 4.2). The only values to be determined are $f_W$. We aim at finding the values $f_W$ which maximize the log-likelihood $l(f_W)$.

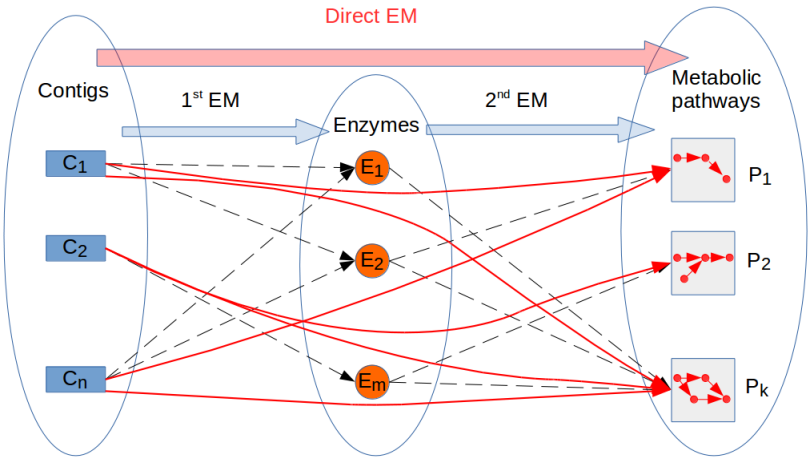



Figure 4.2 Tripartite graph: transcripts → EC numbers → pathways.

We apply the EM-type algorithm [21] for determining the values $f_W$. We initialize each of the abundance estimates for each pathway with a random number $f_w \in [0, 1]$, $w \in W$. Then, we iterate the following two steps until a convergence criteria is satisfied:

**The E-step**. We first compute the expected number of reads $n_w$ emitted by each pathway $w$ through the following formula:

$$n_w = \sum_{t \in T} a_t \cdot \frac{\alpha_{tw} f_w}{\sum_{w' \in W} \alpha_{tw'} f_{w'}}$$

**The M-step**. The new estimates are provided based on a standard maximization EM step:

$$f_w^{new} = \frac{n_w}{\sum_{w' \in W} n_{w'}}$$

The algorithm halts when the new estimates are "close" to the ones from the previous step: $||f_W^{new} - f_W|| \leq \epsilon$, where $\epsilon \ll 1$

**Differential analysis of pathway activity** Using the estimates of pathway activity levels in the differential pathway activity analysis requires estimating uncertainty. The extension of our bootstrapping approach introduced in [1] is useful for the direct maximum likelihood model since the pathway activity levels are inferred directly from RNA-seq reads that can be resampled. The current version of IsoEM2 allows the user to generate bootstrapped samples from the RNA-Seq reads and to infer abundance estimates, based on Fragments Per Kilobase of transcript per Million mapped reads (FPKM). We estimate pathway activity level for each of the bootstrapped samples and then run a differential expression (DE) analysis similar to the one described in [2].

### 4.3.2 Results

In this section, we apply our analysis pipeline to two conditions (day. night) of a planktonic marine microbial community. We describe a subset of the most abundant pathways and conduct a differential pathway activity level analysis that highlights statistically significant functional features from the repertoire of metabolic processes occurring in the community.

**Datasets.** The samples were collected from surface waters (2 m depth) at 12:30 and 23:55 (local time) at a station on the Northern Louisiana Shelf (Gulf of Mexico) in July 2015. Seawater ($\approx$ 1 L) was pumped directly onto a 0.22 $\mu$m Sterivex filter, preserved in 1.8 ml of RNA-later and flash frozen. Samples were stored a -80°C until extraction. RNA was isolated from the samples by a phenol-chloroform method following the Mirvana RNA kit protocol. Samples were treated with DNase to remove residual DNA signal from the metatranscriptome. The RNA-Seq data were generated via Illumina HiSeq 2500 sequencing at the Department of Energy – Joint Genome Institute (DOE-JGI). Detailed information about the two samples is provided in Table 1.

Table 4.4 Dataset description

| Sample | | | | Reads | | | Contigs | |
|---|---|---|---|---|---|---|---|---|
| Name | Depth | Code | Time | Length | Count | Insert size | Total | Total length |
| Day | 2m | 177_2m | 12:30 PM | 2× 151 bp | 89.4 M | 195±49 | 94.7 k | 58.3 MB |
| Night | 2m | 240_2m | 11:55 PM | 2× 151 bp | 91.4 M | 187±49 | 108 k | 68.1 MB |

**MAP pipeline.** A preliminary annotation of RNA-seq data was obtained using the DOE-JGI Metagenome Annotation Pipeline (MAP v.4) (JGI portal) [41]. The MAP processing consists of feature prediction including identification of protein-coding genes. In this pipeline, the MEGAHIT metagenome assembler is used to first assemble RNA-Seq reads into scaffolds. Further, several software suites (GeneMark.hmm, MetaGeneAnnotator, Prodigal, FragGeneScan) are used to predict genes on assembled scaffolds. The MAP pipeline also annotates genes according to EC numbers, which are a necessary input in our maximum likelihood model. The annotations are obtained via homology searches (using USEARCH) against a non-redundant proteins sequence database (maxhits=50, e-value=0.1) where each protein is assigned to a KEGG Orthology group (KO). The top 5 hits for each KO, with the condition that the identity score is at least 30% and 70% of the protein length is matched, are used. The KO IDs are translated into EC numbers using KEGG KO to EC mapping.

**The enhanced quantification pipeline.** Our enhanced pipeline is depicted in red in Figure 4.1. We start our analysis from the RNA-Seq metatranscriptomic reads. First, we find the abundance estimates (frequencies) for each metatranscriptomic gene/transcript by applying Maximum Likelihood abundance estimation. For this purpose we use IsoEM2. The custom GTF annotation file needed for supplying each run of IsoEM2 was prepared by using the fastaToGTF script from the same software suite. Next, we use FPKM estimates as the weights of each transcript for inferring abundances of each EC number. We use transcripts to EC notation alignments as provided by the MAP pipeline.

**Highly active pathways.** Table 4.5 shows the 10 most active pathways in the Day sample sorted in descending order of their activity level, i.e., the number of reads attributed to the proposed maximum likelihood model. The 11th pathway listed (ko0061) is among the 10 most active at night but is not among the 10 most active in the day. Similarly, the pathway

Table 4.5 10 most abundant pathways in the Day and Night samples.

| Pathway | | Abundance reads $\times 10^3$ | |
|---|---|---|---|
| Code | Description | Day | Night |
| ko00190 | Oxidative phosphorylation (Energy metabolism) | 2260 | 2700 |
| ko00710 | Carbon fixation in photosynthetic organisms (Energy metabolism) | 837 | 422 |
| ko00240 | Pyrimidine metabolism (Nucleotide metabolism) | 644 | 1110 |
| ko00270 | Cysteine and methionine metabolism (Amino acid metabolism) | 568 | 176 |
| ko00020 | Citrate cycle - TCA cycle (Carbohydrate metabolism) | 525 | 411 |
| ko00900 | Terpenoid backbone biosynthesis (Metabolism of terpenoids and polyketides) | 508 | 261 |
| ko01230 | Biosynthesis of amino acids | 333 | 471 |
| ko00195 | Photosynthesis (Energy metabolism) | 327 | 63 |
| ko00230 | Purine metabolism (Nucleotide metabolism) | 318 | 618 |
| ko00630 | Glyoxylate and dicarboxylate metabolism (Carbohydrate metabolism) | 299 | 530 |
| ko00061 | Fatty acid biosynthesis (Lipid metabolism) | 37 | 179 |

ko00195 is among the most 10 actives at night but is not among the 10 most active in the day. All other 9 pathways are among the most active during both night and day.

**Differential pathway analysis.** In Table 4.6 there is a list of all metabolic pathways which are up-regulated at noon with at least 1.7 fold change, 95% confidence and at least 1000 reads assigned by EM. The values of abundances are given at 95% confidence interval upper boundary (therefore, they are slightly greater than in the Table 4.5). In Table 4.7 there is a list of all metabolic pathways which are up-regulated at noon with at least 1.7 fold change, 95% confidence and at least 1000 reads assigned by EM.

**Results.** The results in Tables 4.5-4.7 are reflective of planktonic microbial communities driven by a diurnal cycle. During the daytime, pathways mediating photosynthesis, carbon fixation, and the building blocks for amino acid biosynthesis are the most abundant. At night there is an increase in nucleotide and lipid generation, probably for new cell production. In general, the community appears to be gaining energy and substrates during the day and expending them at night by generating crucial cellular components. This is supported by the differential expression between the day and night transcript pools, with energy (photosynthesis) and small organic molecule synthesis (e.g, fructose, glutamine-glutamate, glycosaminoglycan, etc.) being up-regulated during the day and the synthesis of larger biomolecules at night (e.g. lipid metabolism, amino acids, and carotenoids). There is a clear shift in energy sources between day and night. While oxidative phosphorylation is highly transcribed at both time points, it is clear that photosynthesis elevates some of this

energy requirement. This is evidenced by a slight decrease of oxidative phosphorylation and increase of TCA-related transcripts during the day, potentially replenishing the NADH/-NADPH reserves for the use of the electron transport chain at night. As predicted, these results indicate a community undergoing diel cycling, thereby providing validation of our proposed EM-based pipeline and suggesting this method as a valuable tool for coupled annotation and quantification of metabolic pathways in community RNA-seq data.

Table 4.6 Up-regulated pathways in the Day sample

| Pathway | | reads in $10^3$ | |
|---|---|---|---|
| Code | Description | Day | Night |
| ko00051 | Fructose and mannose metabolism (Carbohydrate metabolism) | 326 | 34.1 |
| ko00195 | Photosynthesis (Energy metabolism) | 488 | 93.1 |
| ko00261 | Monobactam biosynthesis (Biosynthesis of other secondary metabolites) | 237 | 44.5 |
| ko00410 | beta-Alanine metabolism (Metabolism of other amino acids) | 10.0 | 0.01 |
| ko00471 | D-Glutamine and D-glutamate metabolism | 6.79 | 0 |
| ko00532 | Glycosaminoglycan biosynthesis - chondroitin sulfate / dermatan sulfate | 28.8 | 3.65 |
| ko00533 | Glycosaminoglycan biosynthesis - keratan sulfate | 22.9 | 0.609 |
| ko00604 | Glycosphingolipid biosynthesis - ganglio series | 4.17 | 0 |
| ko00660 | C5-Branched dibasic acid metabolism (Carbohydrate metabolism) | 4.39 | 0.01 |
| ko00930 | Caprolactam degradation (Xenobiotics biodegradation and metabolism) | 3.80 | 0.883 |
| ko00332 | Carbapenem biosynthesis (Biosynthesis of other secondary metabolites) | 10.3 | 1.54 |
| ko00565 | Ether lipid metabolism (Lipid metabolism) | 10.4 | 0.682 |
| ko00590 | Arachidonic acid metabolism (Lipid metabolism) | 51.8 | 19.4 |
| ko00270 | Cysteine and methionine metabolism (Amino acid metabolism) | 787 | 246 |
| ko00514 | Other types of O-glycan biosynthesis (Glycan biosynthesis and metabolism) | 7.75 | 2.96 |
| ko00450 | Selenocompound metabolism (Metabolism of other amino acids) | 201 | 80.2 |
| ko00710 | Carbon fixation in photosynthetic organisms(Energy metabolism) | 1000 | 487 |
| ko00983 | Drug metabolism - other enzymes (Xenobiotics biodegradation & metabolism) | 58.3 | 16.5 |
| ko00520 | Amino sugar and nucleotide sugar metabolism (Carbohydrate metabolism) | 265 | 123 |

Table 4.7 Up-regulated pathways in the Night sample

| Pathway | | reads in $10^3$ | |
|---|---|---|---|
| Code | Description | Day | Night |
| ko00053 | Ascorbate and aldarate metabolism (Carbohydrate metabolism) | 0 | 1.88 |
| ko00061 | Fatty acid biosynthesis (Lipid metabolism) | 55.9 | 270 |
| ko00120 | Primary bile acid biosynthesis (Lipid metabolism) | 2.75 | 116 |
| ko00140 | Steroid hormone biosynthesis (Lipid metabolism) | 0 | 4.11 |
| ko00232 | Caffeine metabolism (Biosynthesis of other secondary metabolites) | 0 | 1.05 |
| ko00260 | Glycine, serine and threonine metabolism (Amino acid metabolism) | 49.3 | 227 |
| ko00311 | Penicillin and cephalosporin biosynthesis | 0 | 2.74 |
| ko00365 | Furfural degradation (Xenobiotics biodegradation and metabolism) | 0 | 2.12 |
| ko00430 | Taurine and hypotaurine metabolism (Metabolism of other amino acids) | 3.19 | 62.3 |
| ko00472 | D-Arginine and D-ornithine metabolism (Metabolism of other amino acids) | 0 | 1.25 |
| ko00780 | Biotin metabolism (Metabolism of cofactors and vitamins) | 7.05 | 48.6 |
| ko00906 | Carotenoid biosynthesis (Metabolism of terpenoids and polyketides) | 0 | 26.2 |
| ko00984 | Steroid degradation (Xenobiotics biodegradation and metabolism) | 0 | 2.07 |
| ko00362 | Benzoate degradation (Xenobiotics biodegradation and metabolism) | 3.58 | 16.7 |
| ko00592 | alpha-Linolenic acid metabolism (Lipid metabolism) | 0.19 | 2.89 |
| ko00072 | Synthesis and degradation of ketone bodies (Lipid metabolism) | 2.67 | 11.6 |
| ko00364 | Fluorobenzoate degradation (Xenobiotics biodegradation and metabolism) | 0.180 | 2.96 |
| ko01051 | Biosynthesis of ansamycins (Metabolism of terpenoids and polyketides) | 0 | 3.38 |
| ko00760 | Nicotinate and nicotinamide metabolism (Mcofactors and vitamins) | 30.2 | 103 |
| ko00281 | Geraniol degradation (Metabolism of terpenoids and polyketides) | 1.57 | 170 |
| ko00627 | Aminobenzoate degradation (Xenobiotics biodegradation and metabolism) | 0.949 | 4.06 |
| ko00730 | Thiamine metabolism (Metabolism of cofactors and vitamins) | 10.4 | 35.4 |
| ko00643 | Styrene degradation (Xenobiotics biodegradation and metabolism) | 0.958 | 22.6 |
| ko01200 | Carbon metabolism | 13.7 | 86.9 |
| ko00220 | Arginine biosynthesis (Amino acid metabolism) | 3.53 | 11.0 |
| ko00440 | Phosphonate and phosphinate metabolism | 1.30 | 5.33 |
| ko00905 | Brassinosteroid biosynthesis (Metabolism of terpenoids and polyketides) | 2.00 | 35.6 |
| ko00941 | Flavonoid biosynthesis (Biosynthesis of other secondary metabolites) | 2.84 | 6.03 |
| ko00720 | Carbon fixation pathways in prokaryotes (Energy metabolism) | 1.36 | 15.9 |
| ko00290 | Valine, leucine and isoleucine biosynthesis (Amino acid metabolism) | 68.0 | 193 |
| ko00403 | Indole diterpene alkaloid biosynthesis | 0 | 2.68 |
| ko01053 | Biosynthesis of siderophore group nonribosomal peptides | 0 | 1.16 |
| ko00920 | Sulfur metabolism (Energy metabolism) | 47.7 | 135 |
| ko00625 | Chloroalkane and chloroalkene degradation | 24.3 | 51.8 |

# PART 5

# IMMUNE REPERTOIRE PROFILING

## 5.1 Introduction

A key function of the adaptive immune system, which is composed of B cells and T cells, is to mount protective memory responses to a given antigen. B cell and T cells recognize their specific antigens through their surface antigen receptors (B-cell and T-cell receptors, BCR and TCR, respectively), which are unique to each cell and its progeny. BCR and TCR are diversified through somatic recombination, during which variable (V), diversity (D) and joining (J) gene segments are randomly joined, and non-templated bases are inserted or deleted at the recombination junctions [31]. The resulting diverse DNA sequences are then translated into the antigen receptor proteins. The random recombination process enables [missing noun here] to reach an astonishing diversity of the lymphocyte repertoire (i.e., the collection of antigen receptors of a given individual), with >1013 theoretically possible distinct immunological receptors [31]. This diversity is key for the immune system to confer protection against a wide variety of potential pathogens. Furthermore, BCRs are subject to additional diversification in their variable region through somatic hypermutation. These changes are mostly single base substitutions occurring at extremely high rates ($10^5$ to $10^3$ mutations per base pair per generation). Another mechanism contributing to the B cell functional diversity is isotype switching. V(D)J recombination produces BCR expressed as IgM and IgD isotypes. Isotype switching changes the immunological properties of a BCR without changing its specificity by joining the heavy chain VDJ regions with different constant (C) regions that encode IgG, IgA, or IgE isotype antibodies.

High-throughput technologies enable accurate profiling of B- and T-cell repertoires. Commonly used assay-based approaches provide a detailed view of the adaptive immune system with deep sequencing of amplified DNA and RNA from the variable region of BCR or

TCR loci [7]. Those technologies have been successfully applied to characterize the immune repertoire of the peripheral blood[28]. However, little is known about the immunological repertoires of other human tissues, including barrier tissues like the skin and the mucosae. Studies involving assay-based protocols usually have small sample sizes and are not suitable to study the intra-individual variation of immunological receptors across diverse human tissues. RNA Sequencing (RNA-Seq) traditionally uses the reads mapped onto human genome references to study the transcriptional landscape of both entire cellular populations and single cells. However, due to the repetitive nature of loci encoding for BCRs and TCRs, as well as the extreme level of diversity in BCR and TCR transcripts, most mapping tools are ill-equipped to handle immune repertoire sequences. Despite this, BCR and TCR transcripts often occur in sufficient numbers within the transcriptome to characterize the human immunological repertoires[63].

In this study, we developed ImReP, a novel computational method for rapid and accurate profiling of adaptive immune repertoire from RNA-Seq data. We applied it to 8,555 samples across 544 individuals from 53 tissues obtained from Genotype-Tissue Expression study (GTEx v6)[15]. The data was derived from 38 solid organ tissues, 11 brain subregions, whole blood, and three cell lines. This provides a rich resource to study lymphatic tissues, including secondary lymphoid organs ($n = 4$) and sub-mucosa membrane sites ($n = 21$) such as gastrointestinal tract, urogenital tract, thyroid, breast, lung, salivary glands, and skin. ImReP uses unmapped RNA-Seq reads to reconstruct the CDR3 sequences. Our analysis of this dataset identified a typical tissue-specific immunological profile and defined the phylogenetic relation of clonal lineages across various tissues. We found significant differences in immune profiles across the tissues and between individuals.

## 5.2 ImReP: CDR3 sequence assembly and profiling immune repertoires from RNA-Seq data

### 5.2.1 Methods

In contrast to assay-based protocols that produce reads from the amplified variable region of BCR or TCR loci, RNA-Seq is able to capture the entire cellular population of the sample, including B and T cells.

ImReP first prepares the candidate receptor-derived reads. It extracts reads mapped to the TCR and BCR genes. Second, ImReP prepares the high-quality unmapped reads using ROP (step1 – step3) [63] by filtering out low quality, low complexity reads and reads that match rRNA repeats. It also filters out lost human reads, reads unmapped to the reference genome, and lost repeat reads, unmapped reads mapped to the repeat sequences. The reads mapped to the BCR and TCR loci and high-quality unmapped reads were merged, and ImReP used this data to assemble CDR3 sequences and corresponding V(D)J recombinations.

ImReP is a two-stage approach aimed to assemble CDR3 sequence and detect corresponding V(D)J recombinations (Fig 5.1.b). In the first stage, ImReP utilizes the reads that simultaneously overlap V and J gene segments to infer the CDR3 sequences, which are the result of the read compassed by cysteine on the right and phenylalanine (for TCR) or tryptophan (for BCR) on the left. In the second stage, ImReP utilizes the reads overlapping a single gene segment that contains a partial CDR3 sequence. Then, ImReP uses suffix tree to perform the pairwise comparison of the reads and join the reads based on overlap in the CDR3 region. Further, ImReP uses a CAST [6] (Cluster Affinity Search Technique) clustering technique to correct assembled clonotypes for PCR and sequencing errors. CDR3 amino acid sequences produced by ImReP (see Algorithm 3) are represented as vertices $V$ in a complete graph $G = (V, E, \omega)$, where $\omega$ weights are computed as edit distance between all the pair of sequences. For each $v \in V$ we also assign weights $w$ equal to the count of each sequence corresponding to $v$. CAST eliminates the minimal number of edges from $G$ such that the resulting graph is a union of cliques. In each clique, we choose a representative vertex

with the highest weight $w$. The set of such representatives constitutes our assembled CDR3 sequences. We map D genes (for IGH, TCRB, TCRG) onto assembled CDR3 sequences and infer corresponding V(D)J recombination.

**Algorithm 3** ImReP algorithm

$CDR3\_regions \leftarrow \{\}$
$Partial\_CDR3\_V \leftarrow \{\}$
$Partial\_CDR3\_J \leftarrow \{\}$
$V\_genes \leftarrow$ V genes sequences from IMGT database
$J\_genes \leftarrow$ J genes sequences from IMGT database
$\mathcal{R} \leftarrow$ RNA-Seq reads
**for all** $r \in \mathcal{R}$ **do**
  **for all** $(v,\ j) \in V\_genes \times J\_genes$ **do**
    **if** $intersects(v, r)$ **and** $intersects(r, j)$ **then**
      $CDR3\_regions \leftarrow CDR3\_regions \cup \{r\}$
    **else if** $intersects(v, r)$ **then**
      $Partial\_CDR3\_V \leftarrow Partial\_CDR3\_V \cup \{r\}$
    **else if** $intersects(r, j)$ **then**
      $Partial\_CDR3\_J \leftarrow Partial\_CDR3\_J \cup \{r\}$
    **end if**
  **end for**
**end for**
$SF \leftarrow$ suffix tree on $Partial\_CDR3\_V$
**for all** $j \in Partial\_CDR3\_J$ **do**
  **if** $overlap(v, j) > 10$ for a $v \in SF$ **then**
    $CDR3\_regions \leftarrow CDR3\_regions \cup \{concatenation(v, j)\}$
  **end if**
**end for**
**return** $CDR3\_regions$

**Simulation of RNA-Seq and BCR (TCR)-Seq data** We performed in-silico simulations to investigate the feasibility of using RNA-Seq to study the clonal adaptive immune repertoire. We first checked the ability of the ImReP to extract the receptor-derived reads from the RNA-Seq reads. First, we simulated the TCR and BCR transcripts, which are composed of recombined VDJ segment with non-template insertion at the V(D)J junction. We used the IMGT database [50] of V and J gene segments. We randomly selected a pair of VJ segments and inserted a sequence of random nucleotides. The length of the inserted se-

quence was sampled from the Gaussian-like distribution with mean 15 [65]. We also exclude the simulated transcripts with the random insertions leading to out-of-frame proteins. We used LymAnalizer [99] to validate CDR3 sequences of the transcript.

We used SimNGS (https://www.ebi.ac.uk/goldman-srv/simNGS/) to simulate $x1$, $x2$, $x3$ number of paired-end reads from BCR and TCR transcripts in order to achieve $y1$, $y2$, $y3$ infiltration levels of B and T cells. Next, we simulated the 50 million reads from the human transcriptome reference (GRCh37). We mixed reads derived from BCR and TCR transcripts with transcriptomic reads into an RNA-Seq mixture. We then apply ImReP to a simulated RNA-Seq mixture to check the ability of ImReP to extract CDR3-derived reads from the RNA-Seq mixture. We notice some CDR3-derived reads were partially mapped (i.e., reads having N and/or S in the CIGAR strings of their alignments) to the genome by STAR. This highlights the importance of the ImReP's approach of complementing the unmapped reads with mapped reads from BCR and TCR loci to extract CDR3-derived reads.

After we proved the ability of ImReP to reliably extract the CDR3-derived reds from the RNA-Seq mixture, we studied the effects of the coverage and read length on the ability to reconstruct CDR3 sequences. In total, we simulated 1,000 BCR or TCR transcripts. We simulated paired-end reads of various read length ($l = 50, 75, 100$) with use various coverage of TCR and BCR transcripts ($c = 1, 2, 4, 8, 16, 32, 64, 128$). We used the power law distribution to assign frequencies to simulated T and B cell transcripts [94]. The CDR3 protein sequences assembled by ImReP were compared to simulated transcripts to evaluate the recall and precision for various read length and coverage (Figures 5.2 and 5.3).

We define recall and precision in the following way:

$$Recall = \frac{TP}{TP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

Where $TP$ is the number of correctly assembled CDR3 sequence features (exact match to the simulated CDR3), $FN$ is the number of simulated CDR3 sequence features not assembled

by the method, and $FP$ is the number of incorrectly assembled CDR3 sequences.

### 5.2.2 Results

**Feasibility of using RNA-Seq to study adaptive immune repertoire** To validate the feasibility of using RNA-Seq to study the adaptive immune repertoire, we simulated RNA-seq and BCR(TCR)-Seq data. We then compare methods and investigate the sequencing depth and read length required to reliably assemble TCR and BCR sequences. We compared ImReP to the existing approaches designed for assay-based data based on simulated RNA-Seq data generated as described in Section 5.2.1. We report recall and precision rates for each of the methods. MiXCR [10] cannot be applied to RNA-Seq reads, because it was originally designed for deep sequencing of amplified DNA and RNA from the variable region of BCR or TCR loci (BCR-Seq or TCR-Seq). We prepared the candidate receptor-derived reads for MiXCR using the ImReP strategy.

Read length has a strong effect on the performance of MiXCR (Figures 5.2(a) and 5.3(b)). ImReP is able to tolerate short read length of RNA-Seq reads and has consistent results for different read length including a short length of 50bp. Notably, ImReP is able to reconstruct significantly more CDR3 clonotypes than MiXCR for read length 50bp with a higher precision rate. High precision-recall rates even for short reads are achieved by ImReP due to the second stage implying an assembly step using a suffix tree. The increase of coverage has a positive effect on the number of assembled clonotypes. At the same time, the precision of MiXCR drops to 10-20% with the increase in coverage. We observe a slight drop in the precision of ImReP with the increase of coverage.

We further validated the ability of ImReP to accurately infer the fraction of the reads derived from B and T cell receptor by comparing it to the proportion of B and T cell inferences that are based on the gene expression profile. B and T cell signature inferred by SaVant (http://pathways.mcdb.ucla.edu/savant/) shows high correlation (Pearson r=0.67) across all tissues except the spleen and EBV-transformed lymphocytes (LCLs).

We applied ImReP to 0.6 trillion paired-end reads (92 Tbp) produced by RNA-Seq

for 8,555 samples across 500 individuals from 53 tissues obtained from Genotype-Tissue Expression study (GTEx v6) to assemble CDR3 sequences of B and T cell receptors. First, we mapped RNA-Seq reads to the human reference by the short-read aligner (performed by GTEx consortium [15]. To identify reads spanning the V(D)J junction of B and T cell receptors and assemble clonotype sequences, ImReP used 0.02 trillion high-quality unmapped reads that failed to map to the human reference in conjunction with reads mapped to BCR and TCR genes.

ImReP was able to identify over 26 million reads overlapping 6.1 million CDR3 sequences that originate from diverse human tissues. Spleen tissue produced 1.5 million CDR3 sequences. The majority of assembled CDR3 sequences were derived from BCRs, including 2.2 million from immunoglobulin heavy chain (IGH), 1.9 million from immunoglobulin kappa chain (IGK), and 1.4 million from immunoglobulin lambda chain (IGL) BCRs. A smaller fraction of CDR3 sequences was derived from TCRs, including 0.2M sequences from alpha and beta TCRs (TCRA and TCRB). On average, we observe 500 CDR3 sequences per sample, with a range of 0-15,544. We observed the highest number of secondary lymphoid organs, followed by tissues with mucosa membrane sites. Tissues not related to the immune system (e.g., brain and esophagus) contain on average 12.1 CDR3 sequences per sample, which are most likely due to the blood content of the tissues[98]. This is supported by an increased number of CDR3 sequences shared between the blood and various tissues within the same individuals. The highest number of CDR3 sequences among non-lymphoid tissues is present in the omentum, a membranous double layer of adipose tissue corresponding to fat-associated lymphoid clusters. As expected, Epstein bar virus (EBV)-transformed lymphocytes (LCL) contain a high number of immune receptor-derived reads corresponding with a small number of clonotypes (a group of clones with identical CDR3 nucleotide sequences) due to the conserved clonotypic pattern of LCL cell lines [19].

We compared the length and amino acid composition (WebLogo 3, http://weblogo.threeplusone.com/manual.html) of the assembled CDR3 sequences of immunoglobulin and T cell receptor chains across the tissues. Consistent with previous studies, we observe that

immunoglobulin light chains have notably shorter and less variable CDR3 lengths compared to heavy chains [72].

Histological images of tissue cross-sections and pathologists' notes (available at GTEx portal, http://www.gtexportal.org/home/histologyPage) have been used to validate the ImReP's ability to detect the samples with a high activity of lymphocytes, which often correspond to a disease state. ImReP was able to identify Thyroid sample with chronic inflammation corresponding to 15,219 assembled clonotypes. We have also compared the number of CDR3 sequences inferred by ImReP from sigmoid colon. As expected, we observe a significantly smaller number of CDR3 sequences inferred from muscularis externa layer compared to Mucosa layer(p-value $< 10^{-16}$).

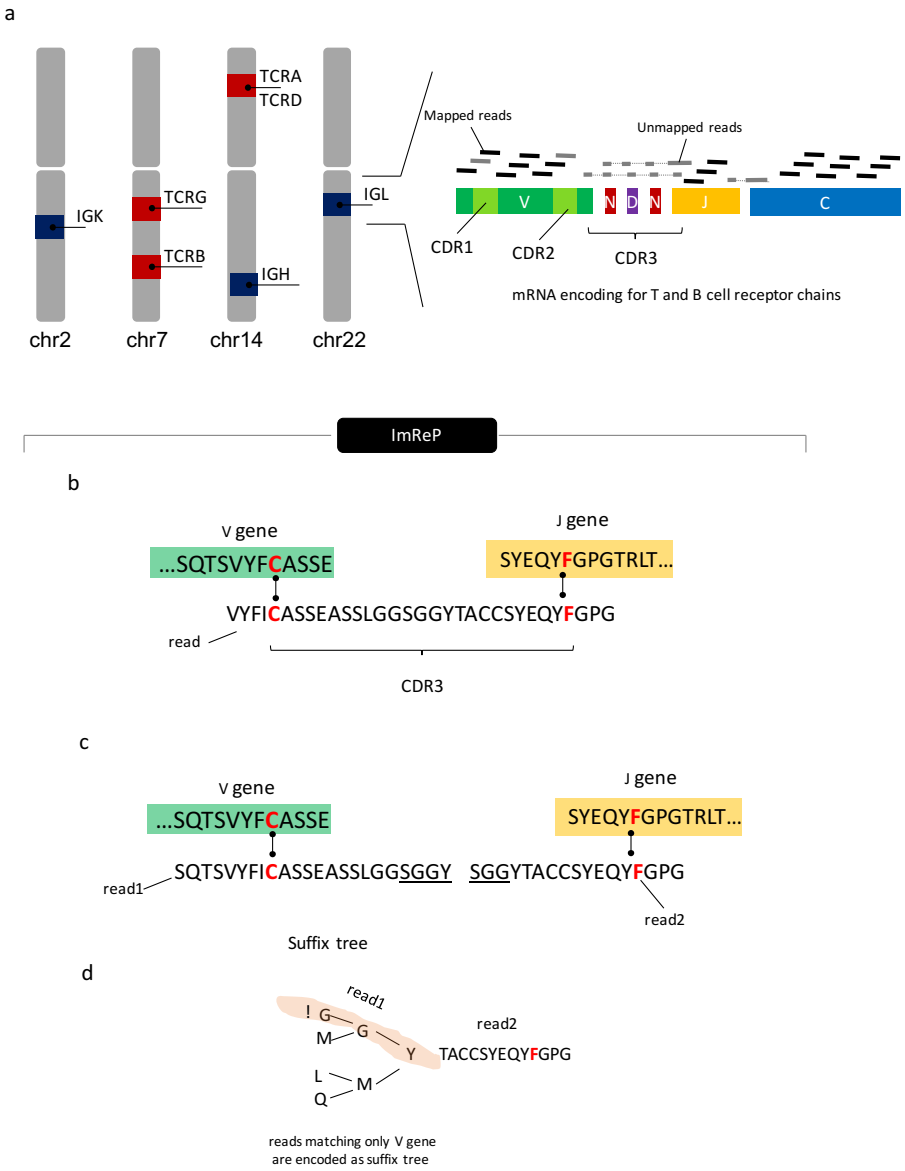
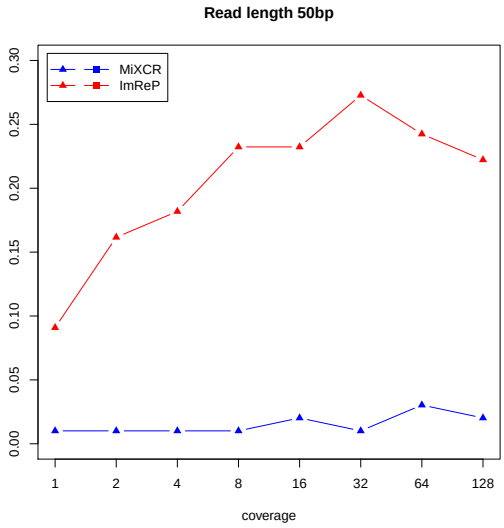
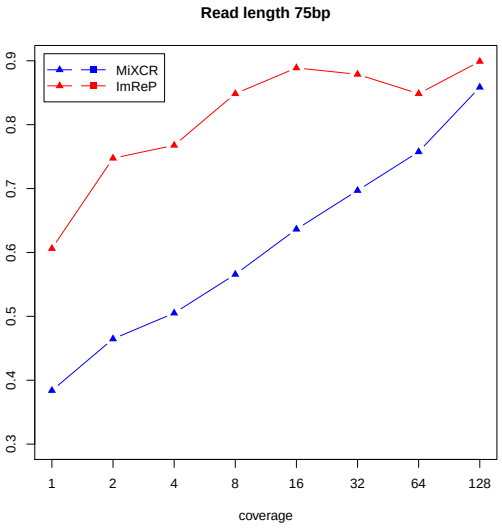


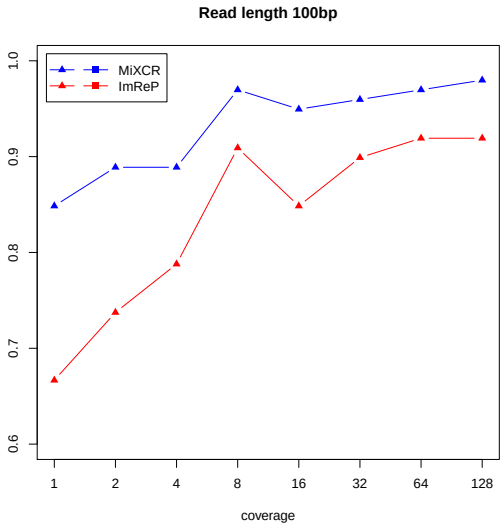

Figure 5.1 Overview of ImReP. (a) Schematic representation of human adaptive immune repertoire. Adaptive immune repertoire consists of four T cell receptor loci (blue color, T-cell receptor alpha locus (TCRA); T-cell receptor beta locus (TCRB); T-cell receptor delta locus (TCRD); and T-cell receptor gamma locus (TCRG)) and three immunoglobulin loci (red color, Immunoglobulin heavy locus (IGH); Immunoglobulin kappa locus (IGK) ; Immunoglobulin lambda locus (IGL). Alternative name – BCR, B cell receptor). B and T cell receptors contain multiple variable (V, green color), diversity (D, present only in IGH, TCRB, TCRG, violet color), joining (J, yellow color) and constant (C, blue color) gene segments. V(D)J gene segments are randomly jointed and non-templated bases (N, dark red color) are inserted at the recombination junctions. The resulting spliced T or B cell repertoire transcript incorporates the C segment and is translated into the antigen receptor proteins. RNA-Seq reads are derived from the rearranged immunoglobulin IG and TCR loci. Reads entirely aligned to genes of B and T cell receptors are inferred from mapped reads (black color). Reads with extensive somatic hypermutations and reads spanning the V(D)J recombination are inferred from the unmapped reads (grey color). Complementarity determining region 3 (CDR3) is the most variable region of the three CDR regions and is used to identify T/B cell receptor clonotypes—a group of clones with identical CDR3 amino acid sequences. (b) Receptor derived reads spanning V(D)J recombinations are identified from unmapped reads and assembled into the CDR3 sequences. We first scan the amino acid sequences of the read and determine the putative CDR3 boundaries defined by last conserved cysteine encoded by the V gene and the conserved phenylalanine (for TCR) or tryptophan (for BCR) of J gene. Given the putative CDR3 boundaries, we check the prefix and suffix of the read to match the suffix of V and prefix of J genes, respectively. (c-d) In case a read overlaps with only the V or J gene, we perform the second stage of ImReP to match such reads based on the overlap of CDR3 sequence using suffix tree. We map D genes (for IGH, TCRB, TCRG) onto assembled CDR3 sequences and infer corresponding V(D)J recombination.




(a) Read length 50 bp

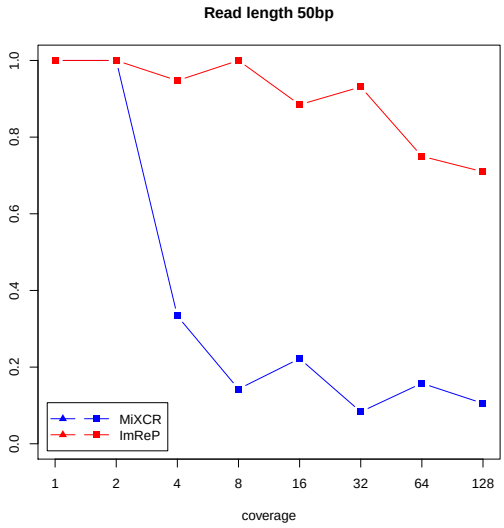



(b) Read length 75 bp

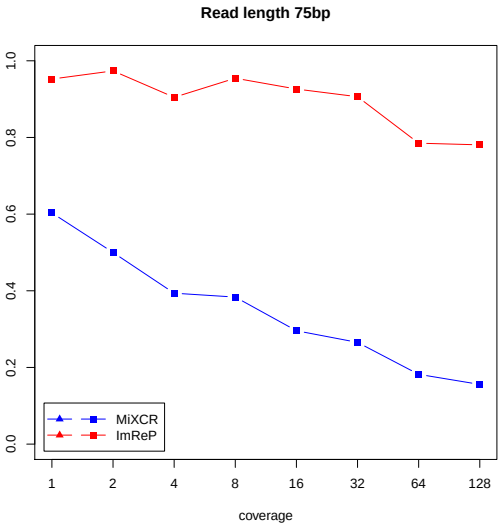



(c) Read length 100 bp

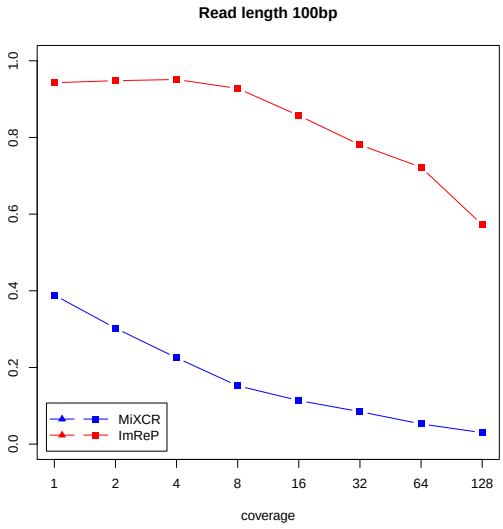
Figure 5.2 ImReP vs MiXCR on simulated data: recall plots for coverages 1, 2, 4, 8, 16, 32, 64, 128: a) Read length 50 bp; b) Read length 75 bp; c) Read length 100 bp.




(a) Read length 50 bp




(b) Read length 75 bp




(c) Read length 100 bp

Figure 5.3 ImReP vs MiXCR on simulated data: precision plots for coverages 1, 2, 4, 8, 16, 32, 64, 128: a) Read length 50 bp; b) Read length 75 bp; c) Read length 100 bp.

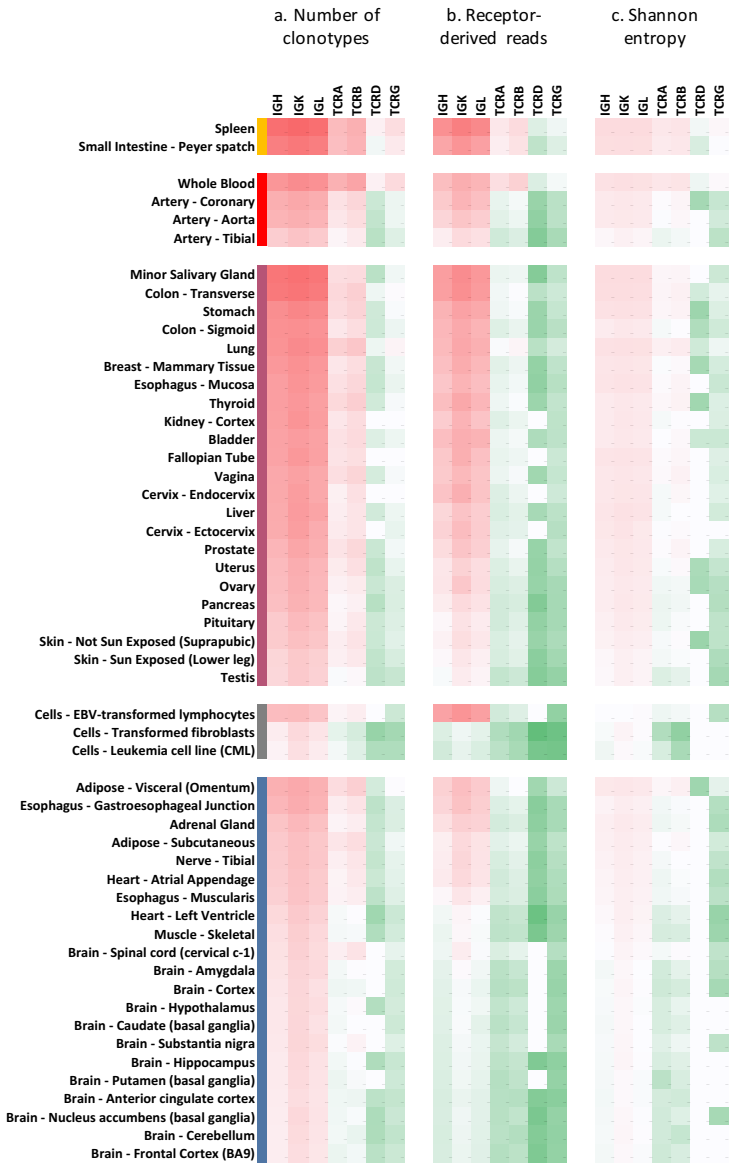



Figure 5.4 Diversity of adaptive immune repertoire across multiple human tissues. Heatmaps depicting the T and B cell repertoires of 8,555 samples across 544 individuals from 53 body sites obtained from Genotype-Tissue Expression study (GTEx v6). We group the tissues by their relationship to the immune system. The first group includes the lymphoid tissues (n=2, orange colors). The second group includes the tissues associated with the blood (n=4, red color ). The Third group includes the tissues that contain mucosal membrane sites (n=21, violet color). The fourth group are the cell lines (n=3, grey color). The fifth group are the tissues not related to the immune system (n=24, blue color). Inside each group the tissues are sorted based on median number of CDR3 sequences per sample of each tissue. (a) Each column report the median number of distinct CDR3 protein sequences of immunoglobulin (IG) or T cell receptor (TCR) chains: immunoglobulin heavy chain (IGH), immunoglobulin kappa chain (IGK, immunoglobulin lambda chain (IGK), T cell receptor alpha chain (TCRA), T cell receptor beta chain (TCRB), T cell receptor delta chain (TCRD), and T cell receptor gamma chain (TCRG). (a) Each row corresponds to a tissue, and each column corresponds to a mean number of distinct CDR3 sequences. (b) Each row corresponds to a tissue, and each column corresponds to a mean number of receptor-derived reads per one million RNA-Seq reads (c) Each row corresponds to a tissue, and each column corresponds to a mean Shannon entropy per tissue. Shannon entropy incorporates total number of CDR3 clonotypes and their relative proportions.

# REFERENCES

[1] Sahar Al Seesi, Serghei Mangul, Adrian Caciula, Alex Zelikovsky, and Ion Măndoiu. Transcriptome reconstruction and quantification from rna sequencing data. *Genome Analysis: Current Procedures and Applications*, page 39, 2014.

[2] Sahar Al Seesi, Yvette T Tiagueu, Alexander Zelikovsky, and Ion I Măndoiu. Bootstrap-based differential gene expression analysis for RNA-Seq data with and without replicates. *BMC genomics*, 15(Suppl 8):S2, 2014.

[3] Simon Anders, Paul Theodor Pyl, and Wolfgang Huber. Htseq—a python framework to work with high-throughput sequencing data. *Bioinformatics*, 31(2):166–169, 2015.

[4] Vineet Bafna and Pavel A Pevzner. Genome rearrangements and sorting by reversals. *SIAM Journal on Computing*, 25(2):272–289, 1996.

[5] Anton Bankevich, Sergey Nurk, Dmitry Antipov, Alexey A Gurevich, Mikhail Dvorkin, Alexander S Kulikov, Valery M Lesin, Sergey I Nikolenko, Son Pham, Andrey D Prjibelski, et al. Spades: a new genome assembly algorithm and its applications to single-cell sequencing. *Journal of computational biology*, 19(5):455–477, 2012.

[6] Amir Ben-Dor, Ron Shamir, and Zohar Yakhini. Clustering gene expression patterns. *Journal of computational biology*, 6(3-4):281–297, 1999.

[7] Jennifer Benichou, Rotem Ben-Hamo, Yoram Louzoun, and Sol Efroni. Rep-seq: uncovering the immunological repertoire through next-generation sequencing. *Immunology*, 135(3):183–191, 2012.

[8] Guillaume Blin, Guillaume Fertin, and Cedric Chauve. The breakpoint distance for signed sequences. In *1st Conference on Algorithms and Computational Methods for biochemical and Evolutionary Networks (CompBioNets' 04)*, volume 3, pages 3–16. King's College London publications, 2004.

[9] Marten Boetzer, Christiaan V Henkel, Hans J Jansen, Derek Butler, and Walter Pirovano. Scaffolding pre-assembled contigs using sspace. *Bioinformatics*, 27(4):578–579, 2011.

[10] Dmitriy A Bolotin, Stanislav Poslavsky, Igor Mitrophanov, Mikhail Shugay, Ilgar Z Mamedov, Ekaterina V Putintseva, and Dmitriy M Chudakov. Mixcr: software for comprehensive adaptive immunity profiling. *Nature methods*, 12(5):380–381, 2015.

[11] Nicolas Bray, Harold Pimentel, Páll Melsted, and Lior Pachter. Near-optimal RNA-Seq quantification. *Nature Biotechnology*, 34:525–527, 2016.

[12] Nicolas L Bray, Harold Pimentel, Páll Melsted, and Lior Pachter. Near-optimal probabilistic rna-seq quantification. *Nature biotechnology*, 34(5):525–527, 2016.

[13] HPJ Buermans and JT Den Dunnen. Next generation sequencing technology: advances and applications. *Biochimica et Biophysica Acta (BBA)-Molecular Basis of Disease*, 1842(10):1932–1941, 2014.

[14] Zheng Chang, Guojun Li, Juntao Liu, Yu Zhang, Cody Ashby, Deli Liu, Carole L Cramer, and Xiuzhen Huang. Bridger: a new framework for de novo transcriptome assembly using rna-seq data. *Genome biology*, 16(1):30, 2015.

[15] GTEx Consortium et al. The genotype-tissue expression (gtex) pilot analysis: Multitissue gene regulation in humans. *Science*, 348(6235):648–660, 2015.

[16] Francis Crick. Central dogma of molecular biology. *Nature*, 227(5258):561–563, 1970.

[17] Francis HC Crick. The origin of the genetic code. *Journal of molecular biology*, 38(3):367–379, 1968.

[18] Adel Dayarian, Todd P Michael, and Anirvan M Sengupta. Sopra: Scaffolding algorithm for paired reads via statistical optimization. *BMC bioinformatics*, 11(1):345, 2010.

[19] Anita De Rossi, Silvio Roncella, Maria Luisa Calabro, Emma D'Andrea, Marcella Pasti, Marina Panozzo, Fabrizio Mammano, Manlio Ferrarini, and Luigi Chieco-Bianchi. Infection of epstein-barr virus-transformed lymphoblastoid b cells by the human immunodeficiency virus: evidence for a persistent and productive infection leading to b cell phenotypic changes. *European journal of immunology*, 20(9):2041–2049, 1990.

[20] Arthur L Delcher, Steven L Salzberg, and Adam M Phillippy. Using mummer to identify similar regions in large sequence sets. *Current Protocols in Bioinformatics*, pages 10–3, 2003.

[21] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[22] Michele Donato, Zhonghui Xu, Alin Tomoiaga, James G Granneman, Robert G MacKenzie, Riyue Bao, Nandor Gabor Than, Peter H Westfall, Roberto Romero, and Sorin Draghici. Analysis and correction of crosstalk effects in pathway analysis. *Genome research*, 23(11):1885–1893, 2013.

[23] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of mathematics*, 17(3):449–467, 1965.

[24] Bradley Efron and B Efron. *The jackknife, the bootstrap and other resampling plans*, volume 38. SIAM, 1982.

[25] Bradley Efron and Robert Tibshirani. On testing the significance of sets of genes. *The annals of applied statistics*, pages 107–129, 2007.

[26] Guy Even, J Seffi Naor, Baruch Schieber, and Madhu Sudan. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica*, 20(2):151–174, 1998.

[27] Paul Flicek, M Ridwan Amode, Daniel Barrell, Kathryn Beal, Konstantinos Billis, Simon Brent, Denise Carvalho-Silva, Peter Clapham, Guy Coates, Stephen Fitzgerald, et al. Ensembl 2014. *Nucleic acids research*, page gkt1196, 2013.

[28] J Douglas Freeman, René L Warren, John R Webb, Brad H Nelson, and Robert A Holt. Profiling the t-cell receptor beta-chain repertoire by massively parallel sequencing. *Genome research*, 19(10):1817–1824, 2009.

[29] Song Gao, Denis Bertrand, Burton KH Chia, and Niranjan Nagarajan. Opera-lg: Efficient and exact scaffolding of large, repeat-rich eukaryotic genomes with performance guarantees. *Genome biology*, 17(1):102, 2016.

[30] Song Gao, Wing-Kin Sung, and Niranjan Nagarajan. Opera: reconstructing optimal genomic scaffolds with high-throughput paired-end sequences. *Journal of Computational Biology*, 18(11):1681–1691, 2011.

[31] George Georgiou, Gregory C Ippolito, John Beausang, Christian E Busse, Hedda Wardemann, and Stephen R Quake. The promise and challenge of high-throughput sequencing of the antibody repertoire. *Nature biotechnology*, 32(2):158–168, 2014.

[32] Peter Glaus, Antti Honkela, and Magnus Rattray. Identifying differentially expressed transcripts from RNA-seq data with biological variation. *Bioinformatics*, 28(13):1721, 2012.

[33] Sante Gnerre, Iain MacCallum, Dariusz Przybylski, Filipe J Ribeiro, Joshua N Burton, Bruce J Walker, Ted Sharpe, Giles Hall, Terrance P Shea, Sean Sykes, et al. High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proceedings of the National Academy of Sciences*, 108(4):1513–1518, 2011.

[34] MG. Grabherr, BJ. Haas, M. Yassour, JZ. Levin, DA. Thompson, I. Amit, X. Adiconis, L. Fan, R. Raychowdhury, Q. Zeng, Z. Chen, E. Mauceli, N. Hacohen, A. Gnirke, N. Rhind, F. di Palma, BW. Birren, C. Nusbaum, K. Lindblad-Toh, N. Friedman, and A. Regev. Basic local alignment search tool. *Nat Biotechnol*, 29(7):644–652, 2011.

[35] Ayman Grada and Kate Weinbrecht. Next-generation sequencing: methodology and application. *Journal of Investigative Dermatology*, 133(8):1–4, 2013.

[36] Alexey Gurevich, Vladislav Saveliev, Nikolay Vyahhi, and Glenn Tesler. Quast: quality assessment tool for genome assemblies. *Bioinformatics*, 29(8):1072–1075, 2013.

[37] Mitchell Guttman, Manuel Garber, Joshua Z Levin, Julie Donaghey, James Robinson, Xian Adiconis, Lin Fan, Magdalena J Koziol, Andreas Gnirke, Chad Nusbaum, John L Rinn, Eric S Lander, and Aviv Regev. Ab initio reconstruction of cell type-specific transcriptomes in mouse reveals the conserved multi-exonic structure of lincRNAs. *Nature Biotechnology*, 28:503–510, 2010.

[38] A Hagberg, D Schult, and P Swart. Networkx: Python software for the analysis of networks. Technical report, Technical report, Mathematical Modeling and Analysis, Los Alamos National Laboratory, 2005. http://networkx. lanl. gov, 2005.

[39] Fan Hsu, W James Kent, Hiram Clawson, Robert M Kuhn, Mark Diekhans, and David Haussler. The ucsc known genes. *Bioinformatics*, 22(9):1036–1046, 2006.

[40] Martin Hunt, Chris Newbold, Matthew Berriman, and Thomas D Otto. A comprehensive evaluation of assembly scaffolding tools. *Genome Biology*, 15(3):R42, 2014.

[41] Marcel Huntemann, Natalia N Ivanova, Konstantinos Mavromatis, H James Tripp, David Paez-Espino, Kristin Tennessen, Krishnaveni Palaniappan, Ernest Szeto, Manoj Pillay, I-Min A Chen, et al. The standard operating procedure of the doe-jgi metagenome annotation pipeline (map v. 4). *Standards in genomic sciences*, 11(1):17, 2016.

[42] Daniel H Huson, Suparna Mitra, Hans-Joachim Ruscheweyh, Nico Weber, and Stephan C Schuster. Integrative analysis of environmental sequences using MEGAN4. *Genome research*, 21(9):1552–1560, 2011.

[43] Alexander Kanitz, Foivos Gypas, Andreas J Gruber, Andreas R Gruber, Georges Martin, and Mihaela Zavolan. Comparative assessment of methods for the computational inference of transcript isoform abundance from RNA-seq data. *Genome Biology*, 16(1):1–26, 2015.

[44] D.K.P. Karunakaran, S. Al Seesi, A.R. Banday, M. Baumgartner, A. Olthof, C. Lemoine, I.I. Mandoiu, and R.N. Kanadia. Network-based bioinformatics analysis of spatio-temporal RNA-Seq data reveals transcriptional programs underpinning normal and aberrant retinal development. *BMC Genomics*, 17(Suppl 5):495:477–492, 2016.

[45] Daehwan Kim, Ben Langmead, and Steven L Salzberg. HISAT: a fast spliced aligner with low memory requirements. *Nature Methods*, 12:357–350, 2015.

[46] Kishori M Konwar, Niels W Hanson, Antoine P Pagé, and Steven J Hallam. Metapathways: a modular pipeline for constructing pathway/genome databases from environmental sequence information. *BMC bioinformatics*, 14(1):202, 2013.

[47] Stefan Kurtz, Adam Phillippy, Arthur L Delcher, Michael Smoot, Martin Shumway, Corina Antonescu, and Steven L Salzberg. Versatile and open software for comparing large genomes. *Genome biology*, 5(2):R12, 2004.

[48] Ben Langmead and Steven L Salzberg. Fast gapped-read alignment with bowtie 2. *Nature methods*, 9(4):357–359, 2012.

[49] Ben Langmead, Cole Trapnell, Mihai Pop, and Steven L Salzberg. Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome biology*, 10(3):R25, 2009.

[50] Marie-Paule Lefranc, Véronique Giudicelli, Patrice Duroux, Joumana Jabado-Michaloud, Géraldine Folch, Safa Aouinti, Emilie Carillon, Hugo Duvergey, Amélie Houles, Typhaine Paysan-Lafosse, et al. Imgt®, the international immunogenetics information system® 25 years on. *Nucleic acids research*, page gku1056, 2014.

[51] Bo Li and Colin N Dewey. Rsem: accurate transcript quantification from rna-seq data with or without a reference genome. *BMC Bioinformatics*, 12(1):323, 2011.

[52] Bo Li and Colin N. Dewey. RSEM: accurate transcript quantification from rna-seq data with or without a reference genome. *BMC Bioinformatics*, 12(1):323, 2011.

[53] Heng Li and Richard Durbin. Fast and accurate long-read alignment with burrows–wheeler transform. *Bioinformatics*, 26(5):589–595, 2010.

[54] Wei Li and Tao Jiang. Transcriptome assembly and isoform expression level estimation from biased RNA-Seq reads. *Bioinformatics*, 28(22):2914, 2012.

[55] James Lindsay, Hamed Salooti, Ion Măndoiu, and Alex Zelikovsky. Ilp-based maximum likelihood genome scaffolding. *BMC bioinformatics*, 15(9):S9, 2014.

[56] Junwei Luo, Jianxin Wang, Zhen Zhang, Min Li, and Fang-Xiang Wu. Boss: a novel scaffolding algorithm based on an optimized scaffold graph. *Bioinformatics*, page btw597, 2016.

[57] Ruibang Luo, Binghang Liu, Yinlong Xie, Zhenyu Li, Weihua Huang, Jianying Yuan, Guangzhu He, Yanxiang Chen, Qi Pan, Yunjie Liu, et al. Soapdenovo2: an empirically improved memory-efficient short-read de novo assembler. *Gigascience*, 1(1):18, 2012.

[58] Ruibang Luo, Binghang Liu, Yinlong Xie, Zhenyu Li, Weihua Huang, Jianying Yuan, Guangzhu He, Yanxiang Chen, Qi Pan, Yunjie Liu, et al. Soapdenovo2: an empirically improved memory-efficient short-read de novo assembler. *Gigascience*, 1(1):18, 2012.

[59] Igor Mandric, Sergey Knyazev, and Alex Zelikovsky. Repeat aware evaluation of scaffolding tools. *bioRxiv*, 2017.

[60] Igor Mandric, James Lindsay, Ion Măndoiu, and Alex Zelikovsky. Silp3: Maximum likelihood approach to scaffolding. In *Computational Advances in Bio and Medical Sciences (ICCABS), 2014 IEEE 4th International Conference on*, pages 1–1. IEEE, 2014.

[61] Igor Mandric, Yvette Temate-Tiagueu, Tatiana Shcheglova, Sahar Al Seesi, Alex Zelikovsky, and Ion Mandoiu. Fast bootstrapping-based estimation of confidence intervals of expression levels and differential expression from rna-seq data. *Bioinformatics*, to appear.

[62] Igor Mandric and Alex Zelikovsky. Scaffmatch: scaffolding algorithm based on maximum weight matching. *Bioinformatics*, page btv211, 2015.

[63] Serghei Mangul, Harry Taegyun Yang, Nicolas Strauli, Franziska Gruhl, Timothy Daley, Stephanie Christenson, Agata Wesolowska Andersen, Roberto Spreafico, Cydney Rios, Celeste Eng, et al. Dumpster diving in rna-sequencing to find the source of every last read. *bioRxiv*, page 053041, 2016.

[64] Shweta Mehrotra and Vinod Goyal. Repetitive sequences in plant nuclear dna: types, distribution, evolution and function. *Genomics, proteomics & bioinformatics*, 12(4):164–171, 2014.

[65] Patrick Miqueu, Marina Guillet, Nicolas Degauque, Jean-Christophe Doré, Jean-Paul Soulillou, and Sophie Brouard. Statistical analysis of cdr3 length distributions for the assessment of t and b cell repertoire biases. *Molecular immunology*, 44(6):1057–1064, 2007.

[66] Cristina Mitrea, Zeinab Taghavi, Behzad Bokanizad, Samer Hanoudi, Rebecca Tagett, Michele Donato, Călin Voichita, and Sorin Drăghici. Methods and approaches in the topology-based analysis of biological pathways. *Frontiers in physiology*, 4, 2013.

[67] Naoki Nariai, Kaname Kojima, Takahiro Mimori, Yukuto Sato, Yosuke Kawai, Yumi Yamaguchi-Kabata, and Masao Nagasaki. TIGAR2: sensitive and accurate estimation of transcript isoform expression with longer RNA-Seq reads. *BMC Genomics*, 15(10):S5, 2014.

[68] Marius Nicolae, Serghei Mangul, Ion I Măndoiu, and Alex Zelikovsky. Estimation of

alternative splicing isoform frequencies from RNA-Seq data. *Algorithms for Molecular Biology*, 6(1):1, 2011.

[69] Rob Patro, Stephen M Mount, and Carl Kingsford. Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms. *Nature Biotechnology*, 32:462–464, 2014.

[70] Wo Pulleyblank. *Dual integrality in b-matching problems*. Springer, 1980.

[71] Adam Roberts and Lior Pachter. Streaming fragment assignment for real-time analysis of sequencing experiments. *Nature Methods*, 10:71–73, 2013.

[72] Edwin P Rock, Peter R Sibbald, Mark M Davis, and YUEH-HSIU Chien. Cdr3 length in antigen-specific immune receptors. *The Journal of experimental medicine*, 179(1):323–328, 1994.

[73] Kristoffer Sahlin, Nathaniel Street, Joakim Lundeberg, and Lars Arvestad. Improved gap size estimation for scaffolding algorithms. *Bioinformatics*, 28(17):2215–2222, 2012.

[74] Kristoffer Sahlin, Nathaniel Street, Joakim Lundeberg, and Lars Arvestad. Improved gap size estimation for scaffolding algorithms. *Bioinformatics*, 28(17):2215–2222, 2012.

[75] Kristoffer Sahlin, Francesco Vezzi, Björn Nystedt, Joakim Lundeberg, and Lars Arvestad. Besst-efficient scaffolding of large fragmented assemblies. *BMC bioinformatics*, 15(1):281, 2014.

[76] Steven L Salzberg, Adam M Phillippy, Aleksey Zimin, Daniela Puiu, Tanja Magoc, Sergey Koren, Todd J Treangen, Michael C Schatz, Arthur L Delcher, Michael Roberts, et al. Gage: A critical evaluation of genome assemblies and assembly algorithms. *Genome research*, 22(3):557–567, 2012.

[77] Julia Salzman, Hui Jiang, and Wing Hung Wong. Statistical modeling of RNA-Seq data. *Statistical Science*, 26(1):62–83, 2011.

[78] Frederick Sanger, Steven Nicklen, and Alan R Coulson. Dna sequencing with chain-terminating inhibitors. *Proceedings of the national academy of sciences*, 74(12):5463–5467, 1977.

[79] David Sankoff. Genome rearrangement with gene families. *Bioinformatics*, 15(11):909–917, 1999.

[80] Junhee Seok, Weihong Xu, Hui Jiang, Ronald W Davis, and Wenzhong Xiao. Knowledge-based reconstruction of mrna transcripts with short sequencing reads for transcriptome research. *PloS one*, 7(2), 2012.

[81] Itai Sharon, Sivan Bercovici, Ron Y Pinter, and Tomer Shlomi. Pathway-based functional analysis of metagenomes. *Journal of Computational Biology*, 18(3):495–505, 2011.

[82] Oleg Shcherbina. Nonserial dynamic programming and tree decomposition in discrete optimization. In *OR*, pages 155–160. Springer, 2006.

[83] Jared T Simpson and Richard Durbin. Efficient de novo assembly of large genomes using compressed data structures. *Genome research*, 22(3):549–556, 2012.

[84] Aravind Subramanian, Pablo Tamayo, Vamsi K Mootha, Sayan Mukherjee, Benjamin L Ebert, Michael A Gillette, Amanda Paulovich, Scott L Pomeroy, Todd R Golub, Eric S Lander, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(43):15545–15550, 2005.

[85] Adi L Tarca, Sorin Draghici, Gaurav Bhatti, and Roberto Romero. Down-weighting overlapping genes improves gene set analysis. *BMC bioinformatics*, 13(1):136, 2012.

[86] Yvette Temate-Tiagueu, Sahar Al Seesi, Meril Mathew, Igor Mandric, Alex Rodriguez, Kayla Bean, Qiong Cheng, Olga Glebova, Ion Măndoiu, Nicole B. Lopanik, and Alexan-

der Zelikovsky. Inferring metabolic pathway activity levels from rna-seq data. *BMC Genomics*, 17(5):542, 2016.

[87] Cole Trapnell, Brian A Williams, Geo Pertea, Ali Mortazavi, Gordon Kwan, Marijke J van Baren, Steven L Salzberg, Barbara J Wold, and Lior Pachter. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nature Biotechnology*, 28:511–515, 2010.

[88] Todd J Treangen and Steven L Salzberg. Repetitive dna and next-generation sequencing: computational challenges and solutions. *Nature Reviews Genetics*, 13(1):36–46, 2012.

[89] Ernest Turro, Shu-Yi Su, Angela Goncalves, Lachlan JM Coin, Sylvia Richardson, and Alex Lewin. Haplotype and isoform specific expression estimation using multi-mapping RNA-seq reads. *Genome Biology*, 12(2):R13, 2011.

[90] Vijay V. Vazirani. *Approximation algorithms*. Springer, 2001.

[91] Francesco Vezzi, Giuseppe Narzisi, and Bud Mishra. Feature-by-feature–evaluating de novo sequence assembly. *PloS one*, 7(2):e31002, 2012.

[92] Xin Victoria Wang, Natalie Blades, Jie Ding, Razvan Sultana, and Giovanni Parmigiani. Estimation of sequencing error rates in short reads. *BMC bioinformatics*, 13(1):185, 2012.

[93] René L Warren, Granger G Sutton, Steven JM Jones, and Robert A Holt. Assembling millions of short dna sequences using ssake. *Bioinformatics*, 23(4):500–501, 2006.

[94] Joshua A Weinstein, Ning Jiang, Richard A White, Daniel S Fisher, and Stephen R Quake. High-throughput sequencing of the zebrafish antibody repertoire. *Science*, 324(5928):807–810, 2009.

[95] Bin Yang, Yu Peng, Henry CM Leung, Siu-Ming Yiu, Jing-Chi Chen, and Francis YL Chin. Unsupervised binning of environmental genomic fragments based on an error robust selection of l-mers. *BMC bioinformatics*, 11(Suppl 2):S5, 2010.

[96] Yuzhen Ye and Thomas G Doak. A parsimony approach to biological pathway reconstruction/inference for genomes and metagenomes. *PLoS computational biology*, 5(8):e1000465, 2009.

[97] Yuzhen Ye and Haixu Tang. An orfome assembly approach to metagenomics sequences analysis. *Journal of bioinformatics and computational biology*, 7(03):455–471, 2009.

[98] Hsiu-Ping Yu, Yuh-Wen Chiu, Hsin-Hong Lin, Tien-Chun Chang, and Yu-Zen Shen. Blood content in guinea-pig tissues: correction for the study of drug tissue distribution. *Pharmacological research*, 23(4):337–347, 1991.

[99] Yaxuan Yu, Rhodri Ceredig, and Cathal Seoighe. Lymanalyzer: a tool for comprehensive analysis of next generation sequencing data of t cell receptors and immunoglobulins. *Nucleic acids research*, page gkv1016, 2015.

[100] D. R. Zerbino and E. Birney. Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. *Genome Research*, 18:821–829, 2008.

[101] Daniel R Zerbino and Ewan Birney. Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome research*, 18(5):821–829, 2008.

[102] Xiaofan Zhou, David Peris, Jacek Kominek, Cletus P Kurtzman, Chris Todd Hittinger, and Antonis Rokas. In silico whole genome sequencer and analyzer (iwgs): a computational pipeline to guide the design and analysis of de novo genome sequencing studies. *G3: Genes, Genomes, Genetics*, 6(11):3655–3662, 2016.