CHAPTER

1 Background

1.1 Introduction

1.1.1 Motivation

This work has been motivated by recent advances of molecular genetics. The human genome has been sequenced in 2001. Also mouse, drosophila, etc. Nowadays # reference model genomes are available in genbank.

Next-generation sequencing has been the second revolution. NGS produces billions of reads for 1000$ dollars. Why should one re-sequence a known genome? Resequencing applications include variant calling, etc. So NGS impacts biomedicine.

Given a set of reads, two approaches are possible: assembly and mapping.

Assembly methods are based on overlaps, de brujin graphs, or...

Read mapping methods work on a previously assembled reference genome.

The typical SNPs analysis pipeline **??** consists of...

In this work we focus on read mapping, although many core algorithms considered are also applicable to assembly, as well as to later pipeline stages.

Figure 1.1: *NGS pipeline.*

1.1.2 Problem definition

We define formally the basic objects alphabet and word. We call alphabet a finite set of symbols, we denote it as Σ. We call word a sequence of symbols over an alphabet. We denote a word w over Σ^*.

Define distance. Let d be a distance function over words, defined as $d : (\Sigma^*, \Sigma^*) \rightarrow \mathcal{N}$. Common distance metrics are the Hamming and Levenshtein or edit distance.

$d_H =$

$d_E =$

Define occurrence. We say that a word p of length m occurs in another word t at position e if $t_{e-m,e} = p$. The notion of occurrence can be generalized to metric distances...

Define approximate string matching. Given two words t and p such that $t \gg p$, respectively called text and pattern. Given a distance threshold $k \in \mathcal{N}$. The approximate string matching problem is to find all occurrences of p into t within the error threshold k.

These definitions allow us to model basic biological.

1.2 Overview of existing methods

Existing methods can be classified in three categories: online, indexed and filtering. Online and filtering in [?]. Indexed in [?].

1.2.1 Online methods

Dynamic Programming

The edit distance between two strings can be efficiently computed via dynamic programming.

The recurrence relation is as follows: $D[i, j] = ...$

Answering the question whether the distance $d(x, y) \leq k$ is an easier problem: a band of size $k + 1$ is sufficient.

Finding all occurrences of a pattern p in a text t...

Change initialization.

Change traceback to start from all $j : D[m, j] \leq k$.

DP Bit-parallelism

Automata

NFA

1.2.2 Indexed methods

Suffix tree

Backtracking

1.2.3 Filtering methods

Why filtering

Pigeonhole principle

q-Gram lemma

1.3 Related problems

1.3.1 Local similarity search

Define score and scoring scheme.

Define local similarity.

Online methods

Give dynamic programming solution.

Indexed methods

Backtracking over substring index. BWT-SW.

Filtering methods

SWIFT/Stellar is based on the q-gram lemma. Lastz resembles a suffix filter.

1.3.2 Dictionary search

Define problem.

Trie

Backtracking

1.3.3 Overlaps computation

Define problem.
DP solution.
Indexed solution, exact and approximate.

CHAPTER

2 Online Methods

CHAPTER

3 Indexed Methods

3.1 Classic Full-Text Indices

3.1.1 Trie

3.1.2 Suffix trie and suffix tree

3.1.3 Suffix array

3.1.4 q-Gram index

3.2 Compressed Full-Text Indices

3.2.1 Burrows-Wheeler transform

3.2.2 FM-index

3.2.3 Rank dictionaries

3.3 Backtracking

3.3.1 Pruning methods

3.3.2 Multiple backtracking

CHAPTER 4

Filtering Methods

4.1 q-Gram filters

4.1.1 Exact seeds

4.1.2 Gapped seeds

4.2 Factor filters

4.2.1 Exact seeds

4.2.2 Approximate seeds

4.3 Suffix filters

CHAPTER 5

Read Mapping

5.1 Related work

5.1.1 Best mappers

5.1.2 All mappers

5.2 The Masai mapper

5.2.1 Single-end mapping

5.2.2 Paired-end mapping

5.3 The Masai 2 mapper

5.3.1 Single-end mapping

5.3.2 Paired-end mapping

5.3.3 Parallelization

5.3.4 Hardware acceleration

5.4 Experimental results

5.4.1 Comparison of filtration strategies

5.4.2 Rabema benchmark results

5.4.3 Variant detection results

5.4.4 Runtime results

5.5 Discussion

CHAPTER 6

String Similarity Search / Join