SeqUMich
Software development to index and search large DNA databases

Sam Lu, Charles Wang, & Hui Jiang
Undergraduate Research Opportunity Program, University of Michigan, Ann Arbor, Michigan 48109

Abstract

With the advent of the next-generation sequencing technologies that allow an entire genome to be sequenced in a matter of days, cancer researchers have sequenced thousands of patients' genomes in an attempt to find the cause of any particular form of cancer. These genomes are stored in an enormous database, waiting to be analyzed. Current software take too long to sift through the hundreds of gigabytes of data to search for a read. Our goal was to develop a new program, SeqUMich, that quickly searches for a gene in the database.

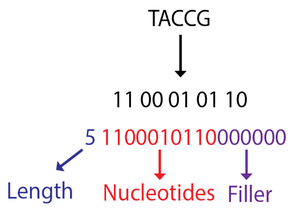
We did this by creating a hash table of k-mers that appear in the reads. With this, we then compared the k-mers with a query sequence, while accounting for error. Our hash table was similar to the one created in the SeqAlto assembler, which only stores a random sample of k-mers, and has a prefix table that significantly reduces memory consumption.

We tested SeqUMich by generating a random genomic reference set, introducing various errors, and testing our algorithm using these random strings. We then tested on real genome sequences.

Objectives & Motives

Our goal is to assist cancer research by improving DNA string querying to isolate mutated genes. Specifically, our goal was to develop a program that can index a set of reads and determine if a query string is contained within this set in the quickest and most efficient way.

Methods



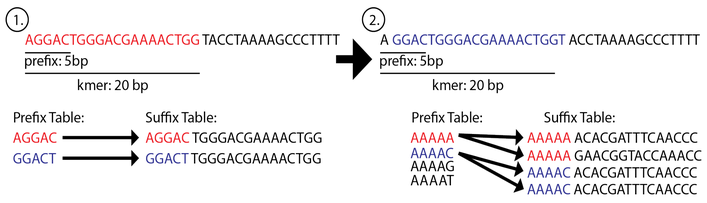
Binary File

- Problem:
  - Reads have too much information to store in memory
  - Still need random access to reads
- Solution: Binary File
  - Unlike sequential (text) file, this file can be accessed at any point
  - Created a file with indexes for future access
- File System: Save all data while using less space
  - Avoid overflow using char array
  - First Char: number of nucleotides
  - Remaining chars: nucleotides converted to 2-bit number and fitted four per char (8 bit char)

Methods (cont.)

Hash Table

- K-mers – DNA nucleotides of a certain length
  - ~20 – 30 nucleotides
  - Stored in the hash table rather than reads
  - Many permutations means there is usually a unique k-mer in a read
  - If there is a k-mer match, it is highly likely that the two strands they come from are similar
  - Each k-mer has an associated read location and read stored in the binary file
  - Easy to search for k-mers and pull the associated index to the binary file
- Hash table – Two-Stage lookup
  - Prefix table
    - Short, 4-6 nucleotides
    - All permutations are present
    - Index the 'suffix' table to facilitate faster lookup
    - Breaks up the search area of the suffix table so that the program does not have to look through the whole array
  - Suffix table
    - Sorted array
    - Stores all the found k-mers (but not every permutation is stored)
    - Also maps the k-mers to their respective indexes in the binary file
    - If a k-mer is found, then the read is pulled from the binary file



Alignment

- Scored using a hybrid of local and global alignment
- One side (the columns) have to be global
  - Every nucleotide must match
- Rows are local
  - The read can appear anywhere and there will be no penalty
- Dynamic algorithm that accounts for insertions, deletions, and mismatches
- A high score usually means that two reads are similar
- High scores are calculated using the 'read' as the row and finding string as the column

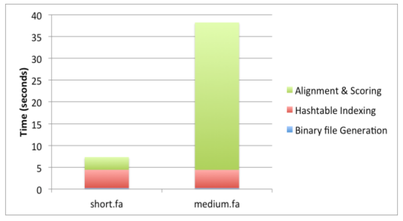
- The below graph demonstrates a sample of the scoring algorithm, which is a hybrid of Needleman-Wunsch and Smith-Waterman

		A	C	G	T	A	A	C
	0	-1	-2	-3	-4	-5	-6	-7
A	0	1	0	-1	-2	-1	0	-1
G	0	0	0	1	0	-1	-1	-2
T	0	-1	-1	0	2	1	0	-1
A	0	1	0	-1	1	3	4	3
A	0	2	1	0	0	4	5	4
T	0	0	0	-1	0	3	4	4
C	0	-1	1	0	-1	2	3	5

Data & Results

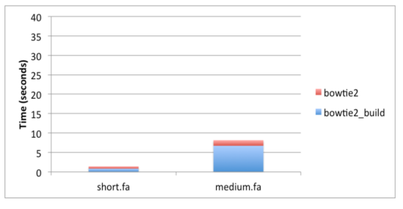
SeqUMich

	short.fa	medium.fa
Binary file Generation	0.23	0.23
Hashtable Indexing	4.2	4.18
Alignment & Scoring	2.91	33.81
Total Time	7.34	38.22



Bowtie2

	short.fa	medium.fa
bowtie2_build	0.854	6.728
bowtie2	0.568	1.434
Total Time	1.422	8.162



- File information
  - Both comparisons were made using a .fastq file that was 6.5 MB.
  - Short.fa was 483.3 KB and medium.fa was 4.7 MB (medium.fa is about 10x larger than short.fa).
- Indexing
  - bowtie2_build on medium.fa took 7.8x longer than short.fa (Bowtie2 indexes the .fa files).
- Aligning
  - bowtie2 (alignment & scoring algorithm) with medium.fa and short.fastq took 3x longer than short.fa and short.fastq.
  - SeqUMich's alignment and scoring for medium.fa took 11.6x longer than short.fa.
  - SeqUMich took 5.2x longer to align short.fa than Bowtie2 and 4.7x longer to align medium.fa than Bowtie2.
- SeqUMich had a constant binary file generation time and hashtable indexing time (we index the .fastq files).

Conclusions

To test the capabilities of SeqUMich, we compared our program against Bowtie2, a well-known sequencing program that aligns reads to long references sequences much like SeqUMich.

The data & results section shows that Bowtie2 is currently a much faster aligner than SeqUMich for small files. Bowtie2 was able to align reads around 5x faster than our own program. It is worth noting that we tested our program with parameters set to give perfect alignment scores as opposed to restricting score calculation, which significantly reduces alignment time. Bowtie2 also showed improvement in alignment time with larger comparison files (i.e. its complexity is better than linear), whereas SeqUMich showed linear increase in alignment with file size.

While SeqUMich is far from the standards of popular sequencers, our program still has room for many improvements. To improve upon our program, we plan to:

- Implement saving indexed fastq files
- Eliminate redundant comparisons of the same reads and regions of query strings
- Optimize our hash function
- Improving our scoring algorithm.

Literature cited

Langmead B, Salzberg S.
Fast gapped-read alignment with Bowtie 2. *Nature Methods. 2012, 9:357-359.*

John C. Mu, Hui Jiang, Amirhossein Kiani, Marghoob Mohiyuddin, Narges Bani Asadi and Wing H. Wong, *Fast and Accurate Read Alignment for Resequencing*, Bioinformatics, 2012

Acknowledgments

We would like to thank our UROP sponsor Hui Jiang, Assistant Professor in the Department of Biostatistics, for introducing us to the field of gene sequencing and providing us with invaluable guidance throughout the development of SeqUMich during the 2013-2014 academic year.

For further information

Contact: Sam Lu (samlu@umich.edu) or
Charles Wang (cvwang@umich.edu) or
Hui Jiang (jianghui@umich.edu)

More information on Bowtie2 can be found at:
http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml