Washington University in St. Louis
Washington University Open Scholarship

Volume 12

Washington University
Undergraduate Research Digest

Spring 2017

Fast K-mer Counting Using the Bi-Directional Burrows-Wheeler Transform

Rishil Mehta
Washington University in St. Louis

Follow this and additional works at: https://openscholarship.wustl.edu/wuurd_vol12

Recommended Citation

Mehta, Rishil, "Fast K-mer Counting Using the Bi-Directional Burrows-Wheeler Transform" (2017). *Volume 12*. 133.
https://openscholarship.wustl.edu/wuurd_vol12/133

This Abstracts J-R is brought to you for free and open access by the Washington University Undergraduate Research Digest at Washington University Open Scholarship. It has been accepted for inclusion in Volume 12 by an authorized administrator of Washington University Open Scholarship. For more information, please contact digital@wumail.wustl.edu.

Fast *K*-mer Counting Using the Bi-Directional Burrows-Wheeler Transform

Rishil Mehta

Mentor: Jeremy Buhler

With faster DNA sequence analysis techniques, biologists and clinicians will be able to analyze more DNA in a shorter period of time, allowing them to conduct faster research and better serve their patients. I focused on increasing the speed and lowering the RAM usage of a DNA analysis tool called *k*-mer counting (enumeration of the number of distinct substrings of size *k* within a text). Unlike traditional alignment-based sequencing techniques, *k*-mer counting allows rapid estimation of the similarity between large genomes and/or large unassembled sequence read sets. One of the most efficient counting implementations uses hashing strategies that are fast but require 10 to 100 gigabytes of RAM for genome-sized sequence comparisons. An alternative data structure, the suffix tree, may achieve better running times. Although suffix trees are also memory-intensive, the bi-directional Burrows-Wheeler transform of a DNA string can emulate the behavior of a suffix tree without the overhead of storing the entire tree in memory. I investigated whether *k*-mer counting using virtual suffix trees and the bi-directional Burrows-Wheeler transform could achieve better speeds and/or more efficient RAM usage. My implementation achieved competitive runtimes versus the top competitor program. I also recognized a number of optimizations that will improve the running time/lower RAM usage even further. My virtual suffix tree implementation is indeed a promising candidate to improve the efficiency of *k*-mer counting.