Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/bbe

Original Research Article

Assembly of repetitive regions using next-generation sequencing data

Robert M. Nowak *

Electronic Systems Institute, Warsaw University of Technology, Warsaw, Poland

ARTICLE INFO

Article history:
Received 17 June 2014
Received in revised form
21 November 2014
Accepted 10 December 2014
Available online 6 January 2015

Keywords:
Genome assembler
Repetitive sequences
Mathematical model
Next generation sequencing
de Bruijn graph parameters

ABSTRACT

High read depth can be used to assemble short sequence repeats. The existing genome assemblers fail in repetitive regions of longer than average read.

I propose a new algorithm for a DNA assembly which uses the relative frequency of reads to properly reconstruct repetitive sequences. The mathematical model for error-free input data shows the upper limits of accuracy of the results as a function of read coverage. For high coverage, the estimation error depends linearly on repetitive sequence length and inversely proportional to the sequencing coverage. The model depicts, the smaller de Bruijn graph dimensions, the more accurate assembly of long repetitive regions.

The algorithm requires high read depth, provided by the next-generation sequencers and could use the existing data. The tests on errorless reads, generated in silico from several model genomes, pointed the properly reconstructed repetitive sequences, where existing assemblers fail.

The C++ sources, the Python scripts and the additional data are available at http://dnaasm.sourceforge.net.

1. Introduction

Next-generation sequencing (NGS) dramatically reduced the cost of producing genome sequences [1]. Therefore, we observe exponential increase of sequencing data [2]. The whole-genome shotgun method is the most popular sequencing technique, where the computer programs called genome assemblers reconstruct a DNA sequence up to chromosome length. The genome assembly is a challenging task for computer science due to a huge volume and complexity of input data produced by NGS. The huge volume of data results from both higher throughput and higher over-sampling. Computer programs use the de Bruijn graphs [3,4] as well as greedy extensions of overlap-consensus-layout graphs [5] to process the volume of data.

Currently more than 50 genome assemblers are available [6–9], but the assembly products are incomplete due to the repetitive regions, the uncovered areas and the sequencing errors. The feasibility of assembly with short reads generated from completely sequenced genomes [10] shows that there is still room for better algorithms.

The short sequence repeats (SSR) are infrequent in sequences coding proteins, therefore transcriptome analysis

* Corresponding author at: Electronic Systems Institute, Warsaw University of Technology, Nowowiejska 15/19, 00-665 Warsaw, Poland.
E-mail address: r.m.nowak@elka.pw.edu.pl

http://dx.doi.org/10.1016/j.bbe.2014.12.001

use genome sequences without properly restored SSR. However, SSR occur in large quantities in eukaryiotic [11] and prokaryitic cells [12], mainly in extragenic and regulatory regions and these regions are used to study genetic variations between individuals. Older techniques based on micro-array or electrophoresis have been replaced with the NGS data used to detect such variations [13–15], when the reference genome is available.

In the presented approach I propose a new algorithm to retrieve the length of a repetitive section using short reads, designed for *de novo* assembly of NGS data. This algorithm estimates SSR length from the coverage statistics and it is able to properly assemble consecutive repeats, as depicted in Fig. 1.

To my knowledge, only the Euler-SR assembler[16] handles consecutive repeats of longer than average read or de Bruijn graph dimension. It constructs the assembly as a path that traverses the repeat twice, therefore underestimates the copy number. The other assemblers skip such SSR.

The paper is organized as follows: Section 2 describes the new algorithm and the mathematical model used to calculate the accuracy of SSR length estimation. Section 3 shows the numerical experiments on *in silico* generated data. Finally, Section 4 presents the proposals for extensions, the protocol of processing the existing data and the conclusions.

2. Approach

2.1. Algorithm

The algorithm uses a k-dimensional weighted de Bruijn multigraph $G(V, E)$, called A-Bruijn graph [17], where V is a set of vertices, E is a set of edges. The edge $e(u, v)$ represents the sequence $s_0 s_1 \dots s_{k-1}$, the vertex u, the source of edge e, represents the sequence $s_0 \dots s_{k-2}$, the vertex v, the target of e, represents $s_1 \dots s_{k-1}$. The edge weight w may be understood as the number of the parallel edges between the source and the target vertices and it depicts how many times the edge should be used to produce an output path.

The algorithm is built of three steps: the graph construction from reads, the edge weight normalization and the output generation.

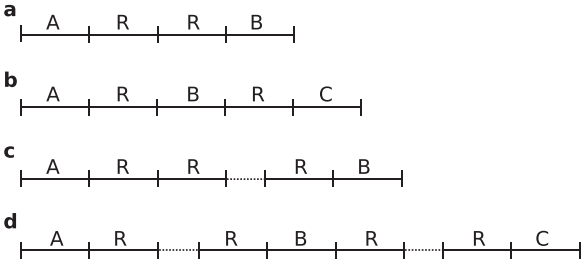


Fig. 1 – SSR, denoted by R, surrounded by unique sequences, denoted by A, B, C, in genome fragment. Cases (a) and (b) are handled by existing sequence assemblers, case (c) is properly solved by the presented algorithm, case (d) is insolvable. Each A, B, C, R are longer than de Bruijn graph dimension.

Algorithm 1 constructs an A-Bruijn graph G from a set of reads R. Every sub-string of length k from R creates an edge in G. Graph dimension k should be an odd number and $k < L$, where L is the average length of read.

Algorithm 1. A-Bruijn graph construction algorithm.

```
Require: R collection of reads
  G ← ∅
  for all r ∈ R : |r| ≥ k do
    for all i : 0 ≤ i ≤ |r| − k do
      u ← r_i ... r_{i+k−2}, v ← r_{i+1} ... r_{i+k−1}
      G.add(edge(u, v)) {increase edge's weight if exists otherwise
add new edge to G with w = 1}
    end for
  end for
```

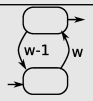
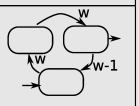
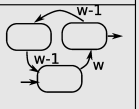
An SSR is a sequence $S = m_0 \dots m_{d-1} m_0 \dots m_{(n-1) \bmod d}$, with of length n, $|S| = n$. S is built of a repeating motif $m = m_0 m_1 \dots m_{d-1}$, $|m| = d$, $d \le (n/2)$. The symbol d denotes the length of the shortest motif of given SSR, therefore two adjacent motifs cannot be treated as a single one. Such SSR create whirls [16], when $n \ge 2(k-1)$. Some whirls are shown in Table 1.

The second step of the algorithm, edge normalization, is a new approach to genome assembly. Eq. (1) converts the edge weight w into w', where c is sequencing coverage and L is read average length. The w' may be understood as edge coverage, because the sequence with length L creates $L - k + 1$ fragments of length k in Algorithm 1.

$$w' = \lfloor \frac{k}{c(L-k+1)} w + 0.5 \rfloor, \quad \text{where} \quad L \ge k \tag{1}$$

The normalization reduces errors, assuming that reads spread uniformly over the sequenced genome. The fragments that occur less frequently than $((c(L-k+1))/(2k))$ are removed from A-Bruijn graph. The normalization plays a similar role to rejecting the sequences that occurs less frequently than predetermined threshold, which is used in other sequence

Table 1 – Examples of whirls in an A-Bruijn graph, w is edge weight, k is graph dimension, d is motif length, n is repetitive sequence length, $n \ge 2(k-1)$.

	$d = 1$	$d = 2$	$d = 3$
$n - k \equiv 0 \pmod d$			
$n - k \equiv 1 \pmod d$			
$n - k \equiv -1 \pmod d$