BIOINFORMATICS APPLICATIONS NOTE

Vol. 28 no. 1 2012, pages 123–124
doi:10.1093/bioinformatics/btr617

Genome analysis

Advance Access publication November 13, 2011

rNA: a fast and accurate short reads numerical aligner

Francesco Vezzi[1,2,*,†], Cristian Del Fabbro[1,2,3,*,†], Alexandru I. Tomescu[1,4] and Alberto Policriti[1,2]

[1]Department of Mathematics and Computer Science, University of Udine, Udine, [2]Istituto di Genomica Applicata, Udine, [3]Department of Agriculture and Environmental Sciences, University of Udine, Udine, Italy and [4]Faculty of Mathematics and Computer Science, University of Bucharest, Bucharest, Romania

Associate Editor: Ivo Hofacker

ABSTRACT

Summary: The advent of high-throughput sequencers (HTS) introduced the need of new tools in order to analyse the large amount of data that those machines are able to produce. The mandatory *first* step for a wide range of analyses is the alignment of the sequences against a reference genome. We present a major update to our rNA (randomized Numerical Aligner) tool. The main feature of rNA is the fact that it achieves an accuracy greater than the majority of other tools in a feasible amount of time. rNA executables and source codes are freely downloadable at http://iga-rna.sourceforge.net/.

Contact: vezzi@appliedgenomics.org; delfabbro@appliedgenomics.org

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on May 23, 2011; revised on October 26, 2011; accepted on November 4, 2011

1 INTRODUCTION

State of the art high-throughput sequencers (HTS) are able to produce up to 200 million reads of length between 30 and 400 bp, in a single run. In almost every usage of such data—notably, single nucleotide polymorphisms (SNP) calling and copy number variation (CNV) identification—the first analysis step consists in aligning these huge datasets against a reference genome.

Recently a significant number of tools able to align the HTS-short reads have been proposed (Li and Homer, 2010). The main efforts in the design of such tools are on improving *speed* and *correctness*. On the one, we need fast tools in order to keep the pace with data production, and on the other hand, we need to maximize the number of *correctly* placed reads and to be sure to align them in *every* possible location. Usually tools sacrifice correctness over speed allowing only few mismatches between reads and reference. To maximize such trade-off, tools like BOWTIE (Langmead *et al.*, 2009) and BWA (Li and Durbin, 2009) make use of the *seed-and-extend* heuristic: in order to align a read r, an almost exact match of the first $l < |r|$ bases of the read is a necessary condition. BFAST (Homer *et al.*, 2009) moves towards favouring correctness over speed, allowing alignments with a high number of mismatches and insertions/deletions (indels), but is one order of magnitude slower than previous tools.

In this work, we present a major update to rNA (randomized Numerical Aligner) (Policriti *et al.*, 2010) able to align in a reasonable amount of time the multitude of reads produced by HTS. With regard to the previous implementation, the present one supports FASTA and FASTQ input formats as well as the SAM/BAM output format. Moreover, it supports the alignment of single and paired-end reads, and it can run on both parallel and distributed architectures. Finally, a graphical user interface (GUI) allows an easy interaction with the various components of the tool. rNA is a highly accurate tool able to align reads in the presence of extensive polymorphisms, high error rates and small indels. As a further contribution, we introduce a new alignment classification designed to better align reads belonging to repetitive regions. rNA is mainly designed for Illumina data, but it can also be used with Solid (together with a suitable conversion from color-space) or 454 data.

2 METHODS

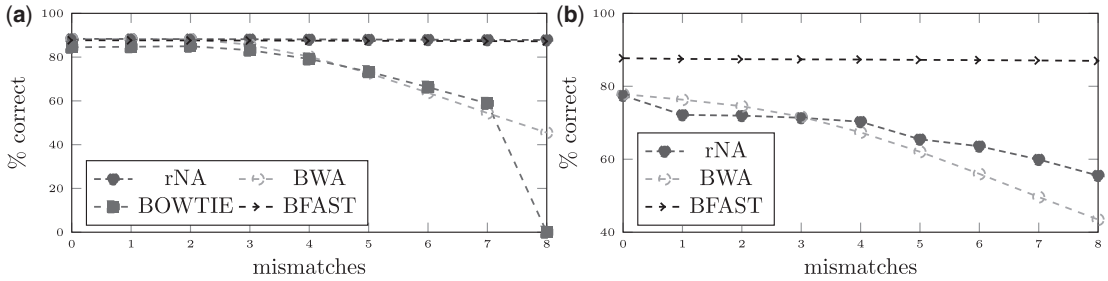
rNA, like the vast majority of aligners, is divided into two distinct computational steps: reference genome pre-processing phase and alignment phase. The former phase builds a hash table over the reference genome given as input, while the latter aligns the reads against the reference employing the hash table. The second step can be parallelized and distributed over several nodes of a cluster.

Indexing and alignment strategies: rNA is based on a simple yet efficient idea originally proposed by Karp and Rabin (1987). A pattern of length l over the alphabet Σ can be encoded by a number in base $|\Sigma|$. For practical values of l, this number is usually too large to fit into a memory word, therefore they proposed to compute the modulo q of this number (q being an adequate prime), obtaining in this way what is called a *fingerprint* of the pattern. Karp and Rabin showed that by computing the fingerprints of all possible length l substrings of the reference T, the pattern can be searched with average complexity $O(|T|)$. In Policriti *et al.* (2010), this approach was extended to efficiently deal also with mismatches; there, q was chosen a Mersenne number. rNA computes and saves the fingerprints of all substrings of length bl present in the reference. To align a read r with k mismatches, we first divide it into $t = \lfloor |r|/bl \rfloor$ non-overlapping blocks; then, the technique of Policriti *et al.* indicates the fingerprints of all the positions in the reference where one of the blocks may occur at Hamming distance at most $\lfloor k/t \rfloor$, which are subsequently checked. This search strategy guarantees a *correct* and *complete* solution to the *best-k-mismatch problem*, i.e. the problem of finding the best alignments with at most k mismatches. The hash table of the fingerprints is built and stored in space proportional to $|T|$ (reference size) and q (hash table size), requiring $4q+5|T|$ bytes.

Indels: due to the increasing length of Illumina reads (grown from 30 bp to 150 bp in a few years) and to the need of aligning reads against biologically distant individuals, it is becoming of foremost importance to align in the presence of small indels. When a read is not aligned allowing only mismatches, rNA can try to align it with indels. This is done through a memory-efficient implementation of a variant of the *Smith–Waterman* algorithm (Smith and Waterman, 1981).

*To whom correspondence should be addressed.

†The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First authors.



		BFAST	BFAST^{indel}	BOWTIE	BWA	BWA^{indel}	rNA	rNA^{indel}
Human	time (hh:mm:ss)	510:33:32	822:11:45	01:21:02	8:05:30	14:06:22	20:10:35	314:58:00
	% align	74.12% (94.03%)	84.24% (94.15%)	69.19%	76.25%	76.91%	79.4 %	85.04%
	RAM	17.4GB	19.8GB	3.5GB	3.7GB	3.7GB	19.8GB	19.8GB

Fig. 1. The two top figures compare the ability to correctly align simulated reads in the presence of mismatches (**a**) and indels (**b**). The Table shows the time needed to align 166 622 914 Illumina reads (SRX027713) against the human reference genome hg18 and the percentage of aligned reads.

Δ-*search*: it is of significant biological importance to understand if a read occurs in single or multiple copy throughout the genome. However, the *best-k-mismatch* problem can give biologically misleading results, by identifying a read as single-copy only because the best hit has such multiplicity. Therefore, we decided to introduce a $\Delta \geq 1$ parameter, such that if the best occurrence of a read is found with k mismatches then the read is searched also allowing $k+1, k+2, \ldots, k+\Delta$ mismatches. With the Δ-option active, a read that aligns with k mismatches is declared single-copy only if no other occurrences with up to $k+\Delta$ mismatches are found.

Read filtering: alignment is of primal importance also in *de novo* assembly. The first step of every *de novo* project consists in filtering the reads by trimming low-quality regions and removing contaminated reads (e.g. reads belonging to chloroplast or mitochondrion). A simple pipeline that trims, aligns and extracts the results, keeping track of paired read information, is slow and produces huge and useless intermediate files. rNA has a module that performs read trimming, aligns reads on a set of contamination references and saves the trimmed and filtered reads, by preserving the read pairing constraints and without producing intermediate files. This module can be run on multiple processors to enhance performances.

3 RESULTS AND DISCUSSION

We compared rNA with three widely used HTS aligners: BFAST (Homer *et al.*, 2009), BOWTIE (Langmead *et al.*, 2009) and BWA (Li and Durbin, 2009). We worked on simulated as well as real data, in order to fully evaluate the performances of the different tools. All the experiments have been run on an 8 core 2.5 GHz Intel(R) Xeon(R) 32 GB RAM machine, always using eight threads. An extended description of the input datasets as well as the command lines used are presented in the Supplementary Materials.

We defined a read *correctly* aligned if the best alignment is unique and the alignment position is ± 5 bases away from the real sampling position. We produced two different simulated datasets: the first one was composed of 9 files containing 1M simulated reads of length 100 such that file i contains reads with exactly i mismatches. The second dataset was similar but with the presence of a contiguous indel of length at most 5 in each read. Reads were simulated and aligned on human chromosome one.

In Figure 1a, we plotted the ratio between the correctly aligned reads and the total reads varying the number of errors introduced in the reads. A similar graph is presented in Figure 1b where also a continuous indel is introduced in each read. When only mismatches are present, rNA and BFAST are the two best tools, with almost the same performances. When also indels are present, BFAST is the most precise tool, while the sensitivity of rNA and BWA decreases as the number of mismatches increases.

We also tested all the aligners on a real Illumina dataset of 100 bp-length reads to reinforce our analysis. The dataset consisted of 166 622 914 paired reads downloaded from the Sequence Read Archive (SRX027713). We aligned all the reads against the 3.2 Gb human genome hg18. Results are presented in the Table of Figure 1. We ran the tools with default parameters, the only exception being rNA for which we disabled the auto trimming option. Since BFAST is designed for align reads at a high distance, we report two sets of results for that: we filtered out reads aligned with >7 mismatches—without limit on the number of indels—and in the second, in parenthesis, we report its original output.

BFAST is the tool that aligns most reads, but it requires an amount of time too large for practical uses. On the opposite side, BOWTIE is the fastest tool, but its performances are obtained at the price of a lower ability to place reads. rNA and BWA achieve similar results: when only mismatches are allowed, rNA aligns almost 3% more reads than BWA that, on the other hand, is faster than rNA. When indels are allowed, rNA aligns 7% more reads than BWA. If rNA is run with the auto trimming option, then the required time of aligning with indels is reduced to 100 h.

Conflict of Interest: none declared.

REFERENCES

Homer,N. *et al.* (2009) BFAST: an alignment tool for large scale genome resequencing. *PloS one*, **4**, e7767.

Karp,R. and Rabin,M. (1987) Efficient randomized pattern-matching algorithms. *IBM J. Res. Develop.*, **31**, 249–260.

Langmead,B. *et al.* (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol.*, **10**, R25.

Li,H. and Durbin,R. (2009) Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, **25**, 1754–1760.

Li,H. and Homer,N. (2010) A survey of sequence alignment algorithms for next-generation sequencing. *Brief. Bioinformatics*, **11**, 473–483.

Policriti,A. *et al.* (2010) A randomized numerical aligner (rNA) *Lang. and Auto. Theory and Appl., LNCS*, **6031**, 512–523. Extended version to appear on Journal of Computer and Systems Sciences.

Smith,T. and Waterman,M. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.