Sequence analysis

The GNUMAP algorithm: unbiased probabilistic mapping of oligonucleotides from next-generation sequencing

Nathan L. Clement1, Quinn Snell1, Mark J. Clement1, Peter C. Hollenhorst3, Jahvi Purwar2, Barbara J. Graves3, Bradley R. Cairns3 and W. Evan Johnson2,3

1Department of Computer Science, 2Department of Statistics, Brigham Young University, Provo, UT 84602 and 3Department of Oncological Sciences, Huntsman Cancer Institute, University of Utah, Salt Lake City, UT 84105, USA

Received on May 18, 2009; revised on September 24, 2009; accepted on October 16, 2009
Advance Access publication October 27, 2009
Associate Editor: John Quackenbush

ABSTRACT

Motivation: The advent of next-generation sequencing technologies has increased the accuracy and quantity of sequence data, opening the door to greater opportunities in genomic research.

Results: In this article, we present GNUMAP (Genomic Next-Generation Universal MAPper), a program capable of overcoming two major obstacles in the mapping of reads from next-generation sequencing runs. First, we have created an algorithm that probabilistically maps reads to repeat regions in the genome on a quantitative basis. Second, we have developed a probabilistic Needleman-Wunsch algorithm which utilizes prb.txt and int.txt files produced in the Solexa/Illumina pipeline to improve the mapping accuracy for lower quality reads and increase the amount of usable data produced in a given experiment.

Availability: The source code for the software can be downloaded from http://dna.cs.byu.edu/gnumap.

Contact: nathanclement@gmail.com

Supplementary information: Supplementary data are available at Bioinformatics online.

1 INTRODUCTION

Next-generation sequencing technologies produce fast and accurate results with base pair level resolution. These technologies are currently being applied to a diverse set of biological applications including creating chromatin state maps (Mikkelsen et al., 2007), performing whole-genome evolutionary analyses (McCutcheon and Moran, 2007), determining miRNA expression (Morin et al., 2008), identifying protein–DNA interactions (Johnson et al., 2007), illuminating histone modification sites (Barski et al., 2007) and translocation breakpoint identification (Chen et al., 2008). Speed and quantity of sequenced bases are two areas where next-generation sequencing outperforms traditional sequencing; but the area in which next-generation approaches such as the Solexa platform, developed by Illumina, Inc. (San Diego, CA, USA), provide the greatest benefit is in chemical processes often result in ambiguous bases, such as assigning nearly equal probabilities to an A and a C. In addition, the first few bases in a Solexa sequencing read are very high in quality, but towards the end of each read, the error rate increases, often dramatically. Many algorithms will only use the nucleotide with the highest probability and ‘call’ that location in the read, ignoring the other three probabilities and the decreasing accuracy near the end of the read. By following this procedure, if there are more than a few bases that have lower probability values, the entire read is discarded. However, if all four probabilities are used in later stages of the mapping process, even the less-confident reads can be mapped, resulting in more usable information from the experiment.

Several applications have attempted to solve the mapping problem. SeqMap (Jiang and Wong, 2008), RMAP (Smith et al., 2008) and ELAND (included as part of the Solexa/Illumina pipeline) all create a hash from the reads. This hash is then used to find matching reads to regions of the genome. Because the genome is not hashed, there is no way to fairly allocate a read across repeat regions. MAQ (Li,H. et al., 2008), SOAP (Li,R. et al., 2008) and Novocraft (unpublished data, http://www.novocraft.com/index.html) also use a hash map, but the reference genome is hashed instead of the reads. MAQ allows up to two mismatches in the first 28 bp. More mismatches are allowed if the Phred-quality score of the entire read is sufficient (Li,H. et al., 2008). Slider (Malhis et al., 2009)
Accurately mapping reads to repetitive genomic elements is essential for next-generation sequencing data to be used to draw valid biological conclusions. For example, a ChIP-seq experiment attempts to accurately identify small DNA regions interacting with a protein of interest. Binding motifs often appear in or near repeat regions (Park et al., 2002; van Helden, 2004), reducing the ability to identify motifs in these regions. Other applications such as transcription mapping, alternative splicing analysis and miRNA identification may also suffer from inaccuracies using such a mapping method. Several programs (such as RMAP, SeqMap and ELAND) have attempted to significantly speed up this mapping process through creating a hash map to efficiently map reads to the genome. Reads are broken into short, 9–15 bp segments and assigned a numerical value in the hash map to efficiently map reads to the genome. Reads are broken into short, 9–15 bp segments and assigned a numerical value in the hash map to efficiently map reads to the genome.

A rigorous probabilistic approach to mapping repeat regions and reads with lower quality scores can result in a significantly larger number of mapped reads. This can often lead to the identification of regions of interest on the genome that otherwise would have been overlooked—for example, mapping to the large number repetitive genomic elements in mammalian genomes. This article, along with the Genomic Next-generation Universal MAPper (GNUMAP) program, focuses on overcoming the aforementioned inaccuracies for an overall increase in data usage and more accurate read mapping to a reference genome.

2 APPROACH

Many mapping algorithms discard reads from repeat regions and do not utilize the quality scores once the base has been ‘called’. GNUMAP provides a probabilistic approach that utilizes this additional information to provide more accurate results from fewer costly sequencing runs.

2.1 Unique mapping position

Accurately mapping reads to repetitive genomic elements is essential if next-generation sequencing is to be used to draw valid biological conclusions. For example, a ChIP-seq experiment attempts to accurately identify small DNA regions interacting with a protein of interest. Binding motifs often appear in or near repeat regions (Park et al., 2002; van Helden, 2004), reducing the ability to identify motifs in these regions. Other applications such as transcription mapping, alternative splicing analysis and miRNA identification may also suffer from inaccuracies using such a mapping method. Several programs (such as RMAP, SeqMap and ELAND) have attempted to significantly speed up this mapping process through creating a hash map to efficiently map reads to the genome. Reads are broken into short, 9–15 bp segments and assigned a numerical value in the hash map according to their sequence. The genome is then scanned and the hashing function is used to find corresponding locations for the genomic sequences in the read hash table. The reads at these locations are then aligned with the genome until either a match is found or the alignment is deemed too insignificant to continue. This strategy of hashing the reads poses a problem when there are multiple regions in the reference genome that produce the same alignment score. This approach does not identify all matching regions at the same time so that the read can be fairly allocated to all of these regions. This is a significant problem because genomic sequences contain a large number of repeat regions.

In the human reference genome, <85% of 30 bp sequences are unique (Butler et al., 2008). On a smaller scale, there are no unique 9 bp sequences. Due to this redundancy, when many programs find a sequence that matches multiple locations, they erroneously either discard the sequence or report all matching locations. For example, assume that an organism’s DNA were sonicated and sequenced, causing each sequence from the genome to appear exactly once in the final set of reads. If a read had originated from a repeat region which occurred three times in the genome, it would also appear three times in the set of reads. Traditional mapping procedures have attempted to score these regions in one of three ways: (i) discard all repeat regions, (ii) record only one position (first or random) for each read or (iii) record all positions as receiving a hit for each read. Discarding these reads results in the loss of up to half of the data (HarrisMendy et al., 2009). The second method would cause unequal mapping to some of the repeat regions. The third method would result in each location having three times the correct score. Since several algorithms (RMAP, ELAND and SeqMap) hash all the reads and then make a single pass through the genome, there is no way to know how many genomic locations will match in order to add one third of a read to each of these locations. The GNUMAP algorithm described below overcomes this problem by hashing the genome and referencing the reads one at a time to the genome. This approach allows for the simultaneous identification of all genomic matches for each read. GNUMAP then accounts for repetitive elements by assigning a proportion of the read to relevant genomic matches based on the relative likelihood that the read maps to each location. It should be noted that MAQ and SOAP also use an approach that hashes the genome; however, they do not proportionally assign multi-hit reads to the genome.

2.2 Information from probability and intensity files

The second problem with current mapping methods for next-generation sequencing data is the disregard for base-calling variability and the frequent discard of lower quality data. Several algorithms (such as MAQ, Bowtie and SOAP) use a single called base at each position in the read, thus ignoring all uncertainty and allowing for increased mapping error rates. This means that reads with a few low-quality bases can lead to an incorrect mapping of a read. When algorithms apply a quality filter to remove these reads, as many as half of the reads may be discarded (HarrisMendy et al., 2009). The GNUMAP algorithm effectively incorporates the base uncertainty of the reads into mapping analysis using a Probabilistic Needleman–Wunsch algorithm. The Probabilistic Needleman–Wunsch was developed to improve upon the common dynamic programming algorithm used for sequence alignment to accurately use reads with lower confidence values. The algorithm is discussed further in Section 3 of this article.

3 METHODS

Care must be taken to develop an algorithm that can accurately map millions of reads to the genome in a reasonable amount of time. In the GNUMAP algorithm, the genome is first hashed and then stored in a lookup table rather than hashing the reads. This allows reads to be accounted for in all of the duplicate genome sites. Next, the reads are efficiently stored as a position–weight matrix (PWM) so that quality scores can be used when aligning the read with genomic data. A Needleman-Wunsch alignment algorithm is modified to use these matrices to score and probabilistically align a read with the reference genome. Figure 1 is a flow chart which shows the major steps of the algorithm.
Table 1 for an example).

The number of bases in the genome, \( n \), and the size of the hash, \( k \), can be computed as follows:

\[
B = 4 \times (2^k + 1) - 1
\]

For example, for a genome of 100,000 bp and a hash size of 9, the total memory used (\( B \)) will be \( 4 \times (2^9 + 200000) \approx 2 \text{ Mb of RAM} \).

Step 2: Processing the reads

One of the novel approaches implemented by GNUMAP lies in the data structure used for storing the reads. Instead of storing the reads as simple sequences, or even sequences with an attached probability as in the FASTQ format, each sequence is stored as a position weight matrix (PWM) (see Table 1 for an example).

Raw data from the Solexa/Illumina platform are obtained as either an intensity file or a probability file. From either of these files, it is possible to compute a likelihood score for any nucleotide of any position on any given sequence. Converting these bases to a single probability score will result in the loss of information (Fig. 5c).

Step 3: Score individual matches

In order to match the reads to the reference genome, the reads first undergo a quality filter, removing reads with too many unknown bases. The amount of memory, \( B \), required based on the number of bases in the genome, \( n \), and the size of the hash, \( k \), can be computed as follows:

\[
B = 4 \times (2^k + 1) - 1
\]

Step 1: Hashing and storing the genome

Hashing a large portion of the data allows for quick data retrieval while still maintaining a reasonable amount of memory use. GNUMAP creates a hash table from the genome instead of the reads, allowing for the computation of a probabilistic scoring scheme.

The entire genome is hashed based upon either a user-supplied hash size or the default hash size of nine. A larger hash size will tolerate fewer mismatches. Larger hashes will require more memory, but will also reduce the search space. The amount of memory, \( B \), required based on the number of bases in the genome, \( n \), and the size of the hash, \( k \), can be computed as follows:

\[
B = 4 \times (2^k + 1) - 1
\]

For example, for a genome of 100,000 bp and a hash size of 9, the total memory used (\( B \)) will be \( 4 \times (2^9 + 200000) \approx 2 \text{ Mb of RAM} \).

Step 2: Processing the reads

One of the novel approaches implemented by GNUMAP lies in the data structure used for storing the reads. Instead of storing the reads as simple sequences, or even sequences with an attached probability as in the FASTQ format, each sequence is stored as a position weight matrix (PWM) (see Table 1 for an example).

Raw data from the Solexa/Illumina platform are obtained as either an intensity file or a probability file. From either of these files, it is possible to compute a likelihood score for any nucleotide of any position on any given sequence. Converting these bases to a single probability score will result in the loss of information (Fig. 5c).

Step 3: Score individual matches

In order to match the reads to the reference genome, the reads first undergo a quality filter, removing reads with too many unknown bases. The amount of memory, \( B \), required based on the number of bases in the genome, \( n \), and the size of the hash, \( k \), can be computed as follows:

\[
B = 4 \times (2^k + 1) - 1
\]

For example, for a genome of 100,000 bp and a hash size of 9, the total memory used (\( B \)) will be \( 4 \times (2^9 + 200000) \approx 2 \text{ Mb of RAM} \).

Table 1. Dynamic Programming (DP) matrix for probabilistic Needleman-Wunsch

<table>
<thead>
<tr>
<th>j</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.526</td>
<td>0.578</td>
<td>0.375</td>
<td>0.356</td>
<td>0.325</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0.305</td>
<td>0.317</td>
<td>0.317</td>
<td>0.164</td>
<td>0.045</td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>0.108</td>
<td>0.320</td>
<td>0.136</td>
<td>0.209</td>
<td>0.330</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>0.059</td>
<td>0.000</td>
<td>0.172</td>
<td>0.271</td>
<td>0.300</td>
<td></td>
</tr>
<tr>
<td>NW</td>
<td>-2</td>
<td>-4</td>
<td>-6</td>
<td>-8</td>
<td>-10</td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>-1.844</td>
<td>-1.792</td>
<td>-0.520</td>
<td>-2.448</td>
<td>-4.448</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>-3.844</td>
<td>-1.792</td>
<td>-0.520</td>
<td>-2.448</td>
<td>-4.448</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>-5.844</td>
<td>-3.792</td>
<td>-2.374</td>
<td>-0.978</td>
<td>-2.978</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>-10</td>
<td>-1000</td>
<td>-5.792</td>
<td>-4.131</td>
<td>-2.774</td>
<td>-1.318</td>
</tr>
</tbody>
</table>

Aligning the genomic sequence TTTCAC and read TTTCAC, with the optimal alignment shown in bold. Also notice the PWM for the sequence, with several fairly ambiguous positions (especially the final position, probably representing a stop codon).

A sliding window of size \( k \) is used to create a hash value which can be used to find matching positions in the reference genome. The matching genomic sequence is then aligned to the read using the probabilistic Needleman-Wunsch algorithm (Table 1).

The probabilistic Needleman-Wunsch score (PNWScore) for read \( r \) and genomic sequence \( S \) at position \( j \) in the dynamic programming matrix \( NW \) can be calculated as:

\[
NW_{ij} = \max \left\{ \begin{array}{l}
NW_{i-1,j-1} + \sum_{k=0}^{4} P_{k,c} \times P_{k,S} \times \text{cost}_{k,j}
\end{array} \right\}
\]

given that \( \text{cost}_{k,j} \) is the cost of aligning the character at position \( j \) with the character \( k \). For example, using the PWM in Table 1, the calculation of the score for position 3, 3 in the dynamic programming matrix would be:

\[
\text{cost}_{3,3} = 0.208 \times 1 + 0.136 \times 1 + 0.317 \times 0 + 0.375 \times 0 + 0.356 \times 1 - 1.948 - 2
\]

\[
\text{cost}_{6,3} = 0.578 \times 1 + 0.375 \times 1 + 0.356 \times 0 + 0.325 \times 0 - 2.774 - 2
\]

\[
\text{score}_{3,3} = 0.136 \times 1 + 0.317 \times 0 + 0.375 \times 0 + 0.356 \times 0 - 1.948 - 2
\]

\[
\text{score}_{6,3} = 0.578 \times 1 + 0.375 \times 1 + 0.356 \times 0 + 0.325 \times 0 - 2.774 - 2
\]
After all the reads have been matched and scored on the genome, two output files are created. The first file identifies the highest scoring match for each read, and the second file contains the genome in .sgr format for viewing in the UCSC Genome Browser (http://genome.ucsc.edu/) or Affymetrix’s Integrated Genome Browser (IGB) (http://www.affymetrix.com).

4 RESULTS

GNUMAP was tested using four datasets—two real and two simulated. The first dataset is a human ChIP-seq experiment attempting to identify the in vivo binding sites of the ETS1 transcription factor. The second dataset is a human small RNA sequencing experiment. In these two examples, we illustrate the ability for GNUMAP to find biologically relevant features while sequencing experiment. For genomic DNA, the 5′ adapter sequence is 33 bases long, and for small RNA experiments the adapter is 20 bases long. These adapters are constructed so they do not perfectly match any genomic fragment. For example, the best human match and the small RNA adapter have only 10 bases in common. For this reason, we hypothesize that if we see a read that contains a close match to the adapter sequence, then the read is directly sequencing the adapter.

For our error analysis, we searched for any read (or part of the read) whose called sequence matched the adapter with three or less mismatches. We fit a logistic regression model to estimate the sequencing error rate for each read position, regardless of the unequal uncertainty associated with the positions, potentially leading to less accurate results.

4.2 Sequencing error analysis

In a Solexa/Illumina sequencing experiment, 5′ and 3′ adapters are ligated to DNA/RNA fragments as part of the sequencing procedure. We have observed that multiple 5′ and 3′ adapters often join to each other without a DNA/RNA fragment between them. In this case, the sequencing reactions will return the 5′ adapter. A typical sequencing run will contain hundreds of thousands of such reads. These reads are ideal for evaluating the sequencing error rate for the experiment. For genomic DNA, the 5′ adapter sequence is 33 bases long, and for small RNA experiments the adapter is 20 bases long. These adapters are constructed so they do not perfectly match any genomic fragment. For example, the best human match and the small RNA adapter have only 10 bases in common. For this reason, we hypothesize that if we see a read that contains a close match to the adapter sequence, then the read is directly sequencing the adapter.

For our error analysis, we searched for any read (or part of the read) whose called sequence matched the adapter with three or less mismatches. We fit a logistic regression model to estimate the sequencing error rate for each read position, regardless of the unequal uncertainty associated with the positions, potentially leading to less accurate results.
Table 2. Sequencing error rates for the ChIP and RNA experiments

<table>
<thead>
<tr>
<th>Read mapping method (Algorithms)</th>
<th>ChIP error rate (% data used)</th>
<th>RNA error rate (% data used)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highest intensity base (None)</td>
<td>2.26</td>
<td>4.17</td>
</tr>
<tr>
<td>Most probable base (MPB)</td>
<td>2.29 (100)</td>
<td>4.17 (100)</td>
</tr>
<tr>
<td>MPB + chastity filtering (Solexa Pipeline)</td>
<td>4.45 (97.8)</td>
<td></td>
</tr>
<tr>
<td>MPB + quality filtering (RMAP)</td>
<td>0.83 (69.8)</td>
<td>2.44 (64.2)</td>
</tr>
<tr>
<td>Probability mapping (GNUMAP, Novo, Slider)</td>
<td>0.67 (100)</td>
<td>1.64 (100)</td>
</tr>
<tr>
<td>Probability + quality (GNUMAP)</td>
<td>0.27 (69.8)</td>
<td>1.11 (64.2)</td>
</tr>
</tbody>
</table>

Notice that the probabilistic mapping (GNUMAP) error rates are lower than those for other methods. In particular, the error rates for GNUMAP (using all the data) are lower than those that incorporate base-calling and filtering (deleting 30–35% of the data).

4.3 ChIP data

A useful test of a mapping algorithm occurs when real data are processed with unknown spike regions. For this example we use the ChIP-seq experiment attempting to identify the in vivo binding sites of the ETS1 transcription factor. We processed the reads using many of the algorithms to determine the sensitivity on real data. Figure 3 shows a spike found by GNUMAP that was not found by any of the other programs. This spike lies in the RALGPS2 gene promoter. The spike occurs in a highly repetitive region that is rich in ‘GGAA’ motifs that can potentially be bound by ETS1. This is an example of a repeat region that holds biological significance that would be missed by most methods.

4.4 Spike-in data

One of the most important end results of the mapping process is to identify ‘spike’ regions in the reference genome with a significant number of read matches. To compare the accuracy of GNUMAP and other methods, we created a test dataset with a large number of known spikes and evaluated the ability of the applications to map the sequences to these spiked-in regions. Promoter regions (which often

---

**Notice** that the probabilistic mapping (GNUMAP) error rates are lower than those for other methods. In particular, the error rates for GNUMAP (using all the data) are lower than those that incorporate base-calling and filtering (deleting 30–35% of the data).
To correct for errors in the benchmark dataset, these regions were
was used here. This may not give an accurate representation of the method, but does show the noise that would occur if each matching location was reported.

region and has a ‘GGAA’ motif, indicating that this region may bound by ETS1.

Note: The option for SOAP to report every match to a particular location
contain repetitive elements) from the 15 Mb C.elegans genome were
used in the comparison dataset. Fifty base-pair windows from these
promoter regions were selected as spiked-in regions, and a random
35 bp sequence within them was inserted into the benchmark dataset.
To correct for errors in the benchmark dataset, these regions were
mapped back to the genome, adding all repeat regions to the list of
expected spikes. Probability and intensity files were created with the
probability of 70% for each called base, simulating quality scores
commonly seen in Solexa/Illumina data.

4.4.1 Visual analysis The sequences were matched with several
different applications available for comparison (RMAP, SOAP, 
SeqMap and Novocraft). After the sequences were matched, the
resulting output files were parsed and converted into .xgr format
capable for viewing and comparison using the Affymetrix IGB. The
.xgr files were further processed, creating bins 100 bp in size to show
density as well as magnitude. These files were visually compared
with that of the original and that of GNUMAP (Fig. 4).

Because the majority of these reads came from promoter regions
with a significant number of repetitive regions, differences are
apparent among the several algorithms. As discussed previously,
the two classical options for aligning sequences from repetitive
sequences is to either discard them or count them multiple times. As
shown in Figure 4c, when discarding these sequences (default for
most programs, and included with RMAP’s run), repetitive regions
are missed. In the test dataset, this region consists of two highly
similar sequences which cannot be identified when discarding reads
from repetitive regions.

The second option is to report all locations that show a significant
alignment. As shown in Figure 4b, the abnormal spike in the data
occurs as a result of SeqMap reporting all these reads. The mapping
process of GNUMAP has also identified these repetitive locations,
but because of the proportional nature of the scoring algorithm
(step 4 of the algorithm), the spike is significantly reduced. As can
be seen in the figure, reporting all the locations adds significant noise
to the final output.

Instead of discarding repetitive reads or adding them to all
matching locations in the genome, the posterior probability of
the read matching a reference genome location should be proportionally
added to all hit locations (the method employed by both Novocraft
and GNUMAP). Using a probabilistic method allows for important
regions to be expressed while not confusing the analysis with the
overexpression of insignificant regions. Using this method,
there is a relation between the amplitude at a given location
and the number of reads in the original data that matched that
location.

4.4.2 Quantitative analysis In addition to a visual comparison,
a procedure was developed for quantitatively comparing the
differences in the mapping methods (Fig. 5). Each algorithm
produced a set of spikes that were ordered according to the number
of reads that mapped to that location. The top 50 spikes from each
application were compared with the known top 50 spikes (from
the benchmark dataset). For each ordered spike index, i, in an
application, the number of top i spikes occurring in the top 50 spikes
of the real dataset was plotted on the y-axis. If the plot followed
the diagonal line, the algorithm would have an accuracy of 100%,
identifying all correct spikes with no false positives. Falling too
far below the diagonal would imply the occurrence of too many
false positives which washed out the identification of true positives.
RMAP and SeqMap were capable of identifying the highest two
spikes (as evidenced by the fact that their lines follow the diagonal
for the first two points); however, GNUMAP outperformed all other
programs in correctly identifying fewer false positives after all 50
spiked-in regions were processed (Fig. 5a).

These results show that either recording or discarding all
ambiguous locations results in a drastic decrease in positive versus
false positive rates. For the SeqMap repeat-including algorithm, the
high number of false positives washed out nearly every significant
alignment, causing a very poor detection rate. RMAP’s repeat-
discarding algorithm reported so few locations that the true positive
rate was significantly low. The only other algorithm that was capable
of approaching the accuracy of statistical mapping software was
SOAP, with a detection rate nearly equal to that of Novocraft and
GNUMAP. However, when reviewing locations such as in Figure 4c,
it becomes clear that, while the method employed by the SOAP
algorithm is capable of recognizing true positive sequence spikes,
the amplitude at these locations is often incorrect.
Fig. 4. Benchmark real spikes (bottom) compared with SeqMap (Jiang and Wong, 2008), SOAP (Li et al., 2008), RMAP (Smith et al., 2008), Novocraft (unpublished data) and GNUMap. Benchmark data were constructed by sampling from 1000 promoter regions in the C.elegans genome. In (b) an enlargement of the first boxed region in (a), SeqMap incremented every location for reads from identical regions, producing a significant false positive spike. Attempting to remove false positives by discarding these reads, such as was done by RMAP, results in missing important information, as can be seen in (c) an enlargement of the second boxed region in (a). Note: The intention of this figure is not to discuss the relative mapping capabilities of all currently implemented programs specifically, but to show the trend that would occur if each of these read-placement techniques were used.

In this spike-in comparison, Novocraft and GNUMap performed similarly because they both utilize a posterior probabilistic scoring method (as described in step 4 in Section 3). For this reason, both algorithms are correctly able to discard spurious match locations and include genuine spikes. However, GNUMap’s implementation of a probabilistic Needleman–Wunsch alignment, incorporating the probabilities of every base at each position, is able to out-perform Novocraft in several occasions (see Supplementary Material for further analysis).

5 PERFORMANCE ANALYSIS

Although accuracy is probably the most important feature of a mapping program, the speed of an algorithm is also important.
Table 3. Performance comparison

<table>
<thead>
<tr>
<th>Program</th>
<th>Benchmark data</th>
<th>Human genome</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time</td>
<td># mapped</td>
</tr>
<tr>
<td>GNUMap</td>
<td>47.9 s</td>
<td>71 262</td>
</tr>
<tr>
<td>Bowtie</td>
<td>7.0 s</td>
<td>62 298</td>
</tr>
<tr>
<td>SOAP</td>
<td>11.7 s</td>
<td>62 208</td>
</tr>
<tr>
<td>MAQ</td>
<td>46.5 s</td>
<td>62 208</td>
</tr>
<tr>
<td>Slider</td>
<td>16 m 31 s*</td>
<td>58 551</td>
</tr>
<tr>
<td>SeqMap</td>
<td>81.2 s</td>
<td>56 326</td>
</tr>
<tr>
<td>Novocraft</td>
<td>24.4 s</td>
<td>56 238</td>
</tr>
<tr>
<td>RMAP</td>
<td>9.2 s</td>
<td>1202</td>
</tr>
</tbody>
</table>

Bold values show that GNUMap achieves the best performance.

6 CONCLUSION

Next-generation sequencing promises to revolutionize biological research by providing millions of bases of sequenced data per experiment. The more accurate approach used by GNUMap can create a more accurate identification of spike regions in the reference genome. The more accurate approach used by GNUMap can research by providing millions of bases of sequenced data per experiment. The more accurate approach used by GNUMap can create a more accurate identification of spike regions in the reference genome. It is

Conflict of Interest: none declared.

REFERENCES


