There is an increasing need to rapidly and accurately align databases is the BLAST algorithm (Altschul et al., 1990). Further optimized versions have been presented to speed searches for large numbers of sequences. The BLAST family of algorithms implements a non-deterministic automata matching algorithm on a keyword tree of the search strings. Both queries with and without ambiguity codes can be searched. Search time is short for perfect matches, and retrieval time rises exponentially with the number of edits allowed.

Availability: The C++ source code for PatMaN is distributed under the GNU General Public License and has been tested on the GNU/Linux operating system. It is available from http://bioinf.eva.mpg.de/patman.

Contact: pruefer@eva.mpg.de

Supplementary information: Supplementary data are available at Bioinformatics online.

1 INTRODUCTION

There is an increasing need to rapidly and accurately align short sequences to genomic or other biological sequences. Short sequence motifs, including restriction enzyme sites, microarray probe sequences, transcription factor binding motifs and miRNA sequence motifs, including restriction enzyme sites, microarray trigger full alignment, thereby missing good hits. query sequences because these may not contain a seed match to prolong alignment time. This tradeoff is especially severe for short of the algorithm. A search with longer seeds may miss some good general optimization of search parameters to large databases. The BLAST algorithm (Altschul et al., 1990). Further optimized versions have been presented to speed searches for large numbers of sequences. The BLAST family of algorithms search for good alignments only where short, perfect seed matches between the query and target sequence exist. This heuristic vastly improves the overall speed by restricting the expensive alignment process to regions containing these short exact matches. There is a tradeoff between an extensive search and the speed performance of the algorithm. A search with longer seeds may miss some good alignments that contain mismatches or gaps, while shorter seeds will prolong alignment time. This tradeoff is especially severe for short query sequences because these may not contain a seed match to trigger full alignment, thereby missing good hits.

© 2008 The Author(s)
This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/2.0/uk/) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

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.

ABSTRACT

Summary: We present a tool suited for searching for many short nucleotide sequences in large databases, allowing for a predefined number of gaps and mismatches. The command-line-driven program implements a non-deterministic automata matching algorithm on a keyword tree of the search strings. Both queries with and without ambiguity codes can be searched. Search time is short for perfect matches, and retrieval time rises exponentially with the number of edits allowed.

Availability: The C++ source code for PatMaN is distributed under the GNU General Public License and has been tested on the GNU/Linux operating system. It is available from http://bioinf.eva.mpg.de/patman.

Contact: pruefer@eva.mpg.de

Supplementary information: Supplementary data are available at Bioinformatics online.

2 METHODS

2.1 Usage

The program accepts several parameters to specify a search. The user ca
We used PatMaN to match 201,807 Affymetrix HGU95-A micro-
array 25mer probes to the chimpanzee genome (panTro2). The 
parameters chosen for this evaluation allowed up to one mismatch,
but no gaps. The program spent ~2.5 h searching through all 
chimpanzee chromosomes and found 15.9 million hits (including 
14.4 million hits to ALU repeat sequences). A table containing all 
unique hits to the chimpanzee genome is available on our website.

Table 1. HGU95-A probes and Bonobo Reads against Chromosome 22

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Edits</th>
<th>Gaps</th>
<th>Run time</th>
<th>Hits</th>
</tr>
</thead>
<tbody>
<tr>
<td>HGU95-A probes</td>
<td>0</td>
<td>0</td>
<td>0m13.31s</td>
<td>93,225</td>
</tr>
<tr>
<td>HGU95-A probesb</td>
<td>1</td>
<td>1</td>
<td>1m51.87s</td>
<td>327,028</td>
</tr>
<tr>
<td>HGU95-A probes</td>
<td>1</td>
<td>1</td>
<td>3m36.92s</td>
<td>496,296</td>
</tr>
<tr>
<td>HGU95-A probesb</td>
<td>2</td>
<td>2</td>
<td>1h21m59s</td>
<td>1,843,008</td>
</tr>
<tr>
<td>Bonobo Solexa GII data</td>
<td>2</td>
<td>2</td>
<td>12h5m50s</td>
<td>14.3 x 10^9</td>
</tr>
</tbody>
</table>

aBenchmarking was performed on a 2.2 GHz workstation. Independently of the chosen 
parameters ~260 MB RAM were used. bBenchmarking was performed on a 1.8 GHz 
workstation and 8.6 GB of RAM was used during execution. The dataset contains 2.8 
million reads of 38 by length of genomic sequence from a Bonobo individual sequenced 
on the Solexa GII platform.

Fig. 1. Keyword tree with suffix links after adding the sequences ‘CCC’, ‘GA’ and ‘GT’. The keyword tree (represented as bold lines) encodes the 
probe sequence as a path leading from the root node on the left side to the 
leaves on the right side. Suffix links are shown as arrows, but have been 
omitted at leaf nodes for brevity.

Aho–Corasick algorithm [for a complete discussion see Navarro and Raffinot 
(2002)]. Figure 1 depicts the resulting data structure for a small input 
example.

Once the tree is constructed, each sequence in the target database is 
evaluated base by base and compared to a list of partial matches. Each 
partial match consists of a node together with the number of mismatches 
and gaps accumulated. The list is initialized with one element containing 
the root node of the tree and an edit count of zero. In each iteration of 
the algorithm, all partial matches are advanced along a perfectly matching 
outgoing edge. Additional elements are stored for following mismatched 
edges and for producing all possible gaps, as long as the number of edits 
remains below the threshold given. If the outgoing edge is a suffix link, the 
resulting partial match is only included if no mismatch or gap occurred in the 
part before the suffix. The number of edits needed to align the suffix is stored 
in the partial match when following a suffix link. Matches are reported when 
a partial match reaches a leaf node before exceeding the predefined number 
of allowed edits. The sequence identifier, match coordinates and number of 
edits are printed.

2.3 Complexity

When ambiguity codes are not interpreted and the query sequences contain no 
‘N’ character, the keyword tree can be constructed in O(L) time and requires 
O(L) space, where L represents the total length of all query sequences 
(Navarro and Raffinot, 2002). When ambiguity is enabled, both time and 
spacerequirements increase exponentially in the number of ambiguity codes 
used in the patterns.

The time efficiency of the search algorithm is linear in the size of the 
target database, but depends heavily on the maximum edit distance as well as 
the average length and number of query sequences. For each additional 
edit operation, an exponentially increasing number of partial matches must 
be considered, since neighboring mismatched nodes and all possible gapped 
alignments are searched along with the perfect matching path through the 
tree. However, if only perfect matches are searched, the algorithm acts like 
the Aho–Corasick algorithm, and search time depends solely on the length of 
the target sequence. Time constraints therefore mean that PatMaN is 
only suitable for searching short sequences with a limited number of edit 
operations.

3 RESULTS

We used PatMaN to match 201,807 Affymetrix HGU95-A micro-
array 25mer probes to the chimpanzee genome (panTro2). The 
parameters chosen for this evaluation allowed up to one mismatch,