Supplementary Material for “A Novel And Well-Defined Benchmarking Method For Second Generation Read Mapping”

Manuel Holtgrewe, Anne-Katrin Emde, David Weese, and Knut Reinert
March 2010

S1 Technicalities
In this section, we describe some more involved details of the benchmark.

S1.1 Ghost Matches And Landscape Smoothing
We note that each match with distance $\leq k - 2$ implies at least one match on both sides of it. Figure 1 in the main article shows an example. This can also be seen in Figure 2 in the main article. For $k = 5$, the third end position of the third lower branch in the left tree implies feasible matches left and right of it. This problem is partially solved by the definition of neighbour equivalence in Section 2.4 of the main article.

However, there is a problem when merging matches in this way: For $k = 4$, the end position marked with $\star$ (at the fourth lower branch of the left tree) separates the matches left and right of it. However, in Section 2.4 of the main article, we explained that alignments sharing their trace are basically the same. Thus, it is desirable to merge the matches left and right of such separating positions. This is the reasoning behind defining trace equivalence and combining it into $\equiv$.

S1.2 Matches On The Reverse Strand
We observe that there is a peculiarity with this definition when searching for reverse-complemented reads or reads on the reverse strand of the reference sequence. An example for this is shown in Figure S2 for $k = 5$. Because of the asymmetry of the search (forcing the alignment to end at each position while chosing the best begin position), there are two lakes in the first case but only one lake in the second case. Note that there is a truly separating match in Figure S2a, the match with distance 6. No two matches in the lake left and right of it are in relation $\equiv$. 
Figure S1: This figure shows a perfect match of a read in the reference sequence and two ghost matches left and right of it.

When building the gold standard, we can search for the reverse-complemented reads in the original reference sequence or search for the original reads in the reverse-complemented reference sequence. Depending on this choice, there might be more or less equivalence classes. In practice, we observed very few differences, mainly in low-complexity repeat regions and with very long (e.g. 454) reads only.

We build the gold standard using Razers. Razers reverse-complements the reference sequence because this is computationally more efficient for its approach. The situation from Figure S2a might occur at some place. A read mapper that reverse-complements the reads (index based read mappers such as Bowtie and bwa do so) will only see one lake, as depicted in Figure S2b.

However, since this almost only occurs in low-complexity regions and the number of occurrences is small, the resulting difference of results will be negligible. Furthermore, such read mappers can easily implement a special benchmark mode that reverse-complements the reference sequence instead of the reads. This might come at a higher computational cost but can then be used to assess the sensitivity and verify that the bias against the approach of reverse-complementing the contigs is small. For measuring computational costs, the normal mode can be used.

Note that one has to choose whether to reverse-complement the reads or the reference sequence. Each decision would slightly bias towards the chosen approach. We chose to reverse-complement the reads since this is what Razers does. When another read mapper with full sensitivity for Hamming and edit distance is released, an alternative decision could be made.
Figure S2: This figure shows a peculiarity with matches on the reverse strand. Depending on whether the reference sequence or the reads are complemented, there might be an infeasible match that separates two match classes. Thick lines correspond to matches whereas thin lines correspond to mismatches or indels.
S2 Read Simulation Details

S2.1 Illumina Reads.

For Illumina reads, the simulator follows a simple statistical approach. In a precomputation step, we aligned a set of reads against the reference genome with Razors using full sensitivity and allowing an error rate of up to 8%. The best matches for each read are returned. The resulting SAM alignment file is then analyzed for position specific substitution and indel rates as well as position and error dependent quality distributions. For more information on this procedure, see [1].

From this, we derive the following, simple error model. (We are not aware of any other read simulator that is able to simulate qualities position- and error-dependent.) Insert and deletion rates are not position dependent and set to 0.1%. Error rates are described by a piecewise linear function. It rises from 0.2% at position 0 to 0.26% at position 0.66 × read length. From there, it rises to 1.2% for the last position. The qualities are modeled by a normal distribution, one for substituted and one for non-substituted bases. The means and standard deviations are linear functions of the position. Means fall linearly from 40 to 39.5/from 39.4 to 30, standard deviations rise linearly from 0.05 to 10/from 3 to 15 from the first to the last position for substituted/non-substituted bases.

In the simulation step, an infix of the haplotype is sampled uniformly at random. Then, an edit string is simulated, depending on the error probabilities for each read position as computed in the precomputation step. Possibly, the infix size is updated in this step if the number of inserts and deletes in the edit string is not the same, such that the resulting simulated reads have the same length. This is followed by applying the edit string to the haplotype infix, yielding the simulated read. Finally, the quality value for each read position is generated, also based on the model described above.

S2.2 454 Pyrosequencing Reads.

For 454 reads, the simulator follows the approach also taken in [2,4]: It samples uniformly distributed infixes of normally distributed length from the haplotype. Then, it does a simple flow cell simulation. Light intensities are simulated with normal distributions, noise with lognormal distribution. The quality values are log-odds scaled error probabilities yielded by a simple Bayes base caller.

The means for the read lengths were 200 and 400, the standard deviations were 10% of the read lengths. For the pyrosequencing simulation, the parameters (see [4] for their meaning) were: Proportionality factor \( k = 0.15 \), error calculation was done with square root, background noise mean was 0.2, background noise standard deviation was 0.1.
S3  Future Work Details

S3.1  Incorporation Of Mate Pairs.

Mate pair information can simply be seen as a constraint on the resulting set of matches: Only matches satisfying the mate pair constraints should be considered. In extension, one could also consider all possible combination of matches for each read pair.

S3.2  Incorporation Of Quality Values.

Quality values could be incorporated by reordering matches as follows: For a sufficiently high value of $k$, all biologically interesting matches will be in the set of all feasible matches. Instead of using unit weight for mismatches and indels (as in the case of edit distance), each error is weighted. In the semi-global alignment of the read against the reference, each mismatching and inserted nucleotide is weighted by its quality value. Each deleted characters is weighted by the mean of its neighbouring bases.

Note, however, that this does not correspond to how Maq [3] and Bowtie handle qualities. Bowtie, for example, ignores any mismatches as long as the scores are below some threshold. Neither method handles qualities for indels.

Including quality values in the methodology is left as future work: There is no coherent handling of quality values in the literature and exploring different ways of including them would constitute work outside the scope of this manuscript.

S3.3  Incorporation of Smith-Waterman Scores.

Other scores like Smith-Waterman (SW) scores can be incorporated in a similar way. Again, all biologically interesting matches under SW score will be contained in the set of all feasible edit distance matches for a sufficiently high value for $k$. We can compute the corresponding equivalence classes for these matches as proposed in this article and then re-weight them with the best SW score of all alignments in each class. Note that one would possibly have to chose an error rate larger than the 8% used in this manuscript in order to guarantee to find all biologically relevant matches.
S4  Read Mapper Parametrization

This appendix describes the parametrization of the read mappers. Generally, we mapped the reads with a maximum of 8% errors. The value was given as an error rate if supported, otherwise as an error count of $0.08 \times \text{length of longest read}$, if possible.

Each read mapper was parametrized to output up to 100 alignments per read on real-world data and 1 alignment on simulated reads, if possible. In the evaluation, the best 100 alignments by edit distance were considered for each read for real-world data; ties were broken randomly. The limitation to one alignment is possible with all used read mappers, but not to arbitrary values with all read mappers.

Simulated paired-end reads were simulated with a normally distributed library size, $\mu = 1000$, $\sigma = 100$. All read mapper’s library size error was set to $3 \cdot \sigma = 300$.

S4.1  Bowtie

Bowtie version 0.12.5 was used. First, the index is built:

```
bowtie-build GENOME.fasta GENOME.fasta
```

**Default Parametrization: Bowtie**  Single-end mapping was performed without any special parameters. `--sam` makes Bowtie output SAM format, `--q` makes it read FASTQ files, `--k 100` allows up to 100 alignments per read.

```bash
bowtie --sam -q -k 100 GENOME.fasta READS.fastq OUTPUT.sam
```

For paired-end mapping on simulated data, the insert size was also given using `--minins` and `--maxins`:

```bash
bowtie --sam -q GENOME.fasta -1 LEFT.fastq -2 RIGHT.fastq \
OUTPUT.sam --minins 700 --maxins 1300 -k 1
```

**Parametrization With Improved Sensitivity: Bowtie**  Following the suggestions by the authors of Bowtie, we tried various parameters to yield an optimized version, called `bowtie*` in the evaluation. For short reads, changes from the default parameters lowered the result quality in terms of normalized intervals. For longer reads (> 50 bp), the following parametrization was performed:

We are setting the number of backtracking steps with `--maxbts 800`, the number of mismatches in the seeds with `--seedmms 3`, enable the *try hard mode* with `--y`, and set the sum of mismatching error qualities to 400 with `--maqerr`, such that at least 10 mismatches are ignored (40 is the cutoff for quality values). The additional parameters on the command line were:

```bash
--sam -q -k 100 --maxbts 800 --seedmms 3 -y --best --maqerr 400
```
We tried different values for these parameters. The change of the parameters
from the ones mentioned above either increased the running time significantly
but did not yield significantly better results or yielded significantly worse results
(at a lower running time, of course), except for -k which we kept fixed to 100
so read mappers cannot benefit from creating arbitrarily many hits.

S4.2 Bwa

Bwa was used in version 0.5.8a. First, the index is built.

```
bwa index GENOME.fasta
```

454 reads were aligned as follows. The parameter -n DISTANCE sets the
maximal allowed edit distance in alignments.

```
bwa bwasw -f OUTPUT.sam -n DISTANCE GENOME.fasta READS.fastq
```

For short reads, first SAI files have to be constructed, for paired-end reads
one for the left and one for the right mates.

```
bwa aln -f TMP.sai -n DISTANCE GENOME.fasta READS.fastq
```

Then, single-end reads were processed as follows.

```
bwa samse -n 99 -f OUTPUT.sam -n DISTANCE GENOME.fasta TMP.sai \ 
READS.fastq
```

Simulated mate-pair reads were processed as follows.

```
bwa sampe -n 99 -f OUTPUT.sam -a 1300 -n DISTANCE GENOME.fasta \ 
LEFT.sai RIGHT.sai LEFT.fastq RIGHT.fastq
```

The authors of Bwa suggested that default parameters should work well for
all kinds data sets. We tried to adjust the parameters -N, -o, -e, -l and -k for
aln but the result quality did only improve marginally, if at all, while running
times increased, sometimes greatly.

Bwa prints only one line per read in the SAM file. Alternative alignments
are stored in the extra XA field, which we expanded to additional lines for the
evaluation.

S4.3 Shrimp2

Shrimp2 was run in version 2.1.1b. The general usage is as follows, the param-
eter -E enables SAM output, -o 100 allows up to 100 matches per read. For
Hamming, we are using the parameter -U.

The command line for single-end reads was:

```
gmapper -E -o 100 [-U] READS.fasta GENOME.fasta > OUTPUT.sam
```
The command line for paired-end, simulated reads was as follows. The parameter \(-I\ 700,1300\) sets the insert size, \(-\text{opp-in}\) sets the mate protocol, \(-o\ 1\) sets the number of alignments per read to 1.

\[
gmapper \ -E \ -o \ 1 \ -p \ -p \ -\text{opp-in} \ -I \ 700,1300 \ \text{READS.fasta} \ \text{\backslash} \ \text{GENOME.fasta} > \text{OUTPUT.sam}
\]

The authors also suggested the non-standard setting \(-H\ -s \ \text{w16}\) which enables a weighted seed for longer reads to improve running time (in our tests, it also slightly improved normalized found intervals). For any sets of reads with reads of length > 50, we are also using this parameter. Furthermore, the authors suggested to tune the window generation threshold, but in our tests, this did not improve the result in terms of normalized found intervals.

We also tried to run Shrimp using the \(--\text{strata}\) option, however the program yielded worse results than with the parameters from above. Also, we tried to parametrize Shrimp to use edit distance (match score 0, mismatch score -1, gap open/extend score -1), but the program crashed with a floating point exception. Consequently, we ran it with the scores \(--\text{match} \ 1 \ --\text{mismatch} \ -1000 \ --\text{open-r} \ -1000 \ --\text{open-q} \ -1000 \ --\text{ext-r} \ -1000 \ --\text{ext-q} \ -1000\) but it also yielded worse scores than with default scores.

### S4.4 Soap2

Soap2 was run in version 2.20. The index was built as follows:

\[
\text{2bwt-builder} \ \text{GENOME.fasta} \ \text{GENOME.fasta}
\]

**Default Parametrization: Soap2** Single-end reads were mapped as follows. The parameter \(-v\) sets the number of mismatches that are allowed in one read.

\[
\text{soap} \ -v \ \text{DISTANCE} \ -a \ \text{READS.fastq} \ -D \ \text{GENOME.fasta.index} \ -o \ \text{TMP.soap}
\]

The Perl script \texttt{soap2sam_fastq.pl} is a modified version of the script \texttt{soap2sam.pl} that Soap2 ships with. If the character N appears in the input Soap2 will replace it by one of the characters C, G, A, T. The script \texttt{soap2sam_fastq.pl} fixes this by taking the read sequence from the FASTQ input file.

\[
\text{soap} \ -1 \ 60 \ -r \ 2 \ -v \ \text{DISTANCE} \ -a \ \text{LEFT.fastq} \ -b \ \text{RIGHT.fastq} \ \backslash
\]
\[
\ -D \ \text{GENOME.fasta.index} \ -o \ \text{MP.soap} \ -2 \ \text{UNPAIRED.soap} \ \backslash
\]
\[
\ -m \ 700 \ -x \ 1300
\]

\[
\text{soap2sam_fastq.pl} \ -p \ -a \ \text{LEFT.fastq} \ -b \ \text{RIGHT.fastq} \ \backslash
\]
\[
\ \text{MP.soap} > \text{OUTPUT.sam}
\]
**Parametrization With Improved Sensitivity: Soap2**  
The parameter `-r 2` makes Soap2 output all repeat hits, even if it finds more than 100. The parameter `-v DISTANCE` allows up to DISTANCE mismatches in one read. DISTANCE is set to 8% of the maximal read length. The parameter `-l 60` makes Soap2 first align the first 60 characters as a seed. This is recommended for long reads by the documentation and 60 appears to be a good value by our experiments. We tried out several other settings for `-r` and `-l` but there were only marginal improvements at substantially higher running times with other parametrizations.

Single-end reads were mapped as follows.

```
soap -l 60 -r 2 -v DISTANCE -a READS.fastq -D GENOME.fasta.index \ 
- o TMP.soap
soap2sam_fastq.pl -a READS.fastq MP.soap > OUTPUT.sam
```

Paired-end reads were mapped as follows. `-m 700` and `-x 1300` set the insert size.

```
soap -l 60 -r 2 -v DISTANCE -a LEFT.fastq -b RIGHT.fastq \ 
- D GENOME.fasta.index - o MP.soap -2 UNPAIRED.soap \ 
- m 700 - x 1300
soap2sam_fastq.pl - p - a LEFT.fastq - b RIGHT.fastq \ 
MP.soap > OUTPUT.sam
```
S5  Additional Figures

\[
S = \text{CG - AGCTTACTG} \\
X \\
R = \text{CCTAGC - TCCTG} \\
X \\
\text{read} = \text{TA - CTTCG}
\]

Figure S3: This figure shows the alignment of a read at its physical sampling position in \( R \). Additionally, \( R \) is aligned to the reference sequence \( S \).

\[
\text{reference} = \text{CAGCG - AGAT} \\
\text{distance} = 1 \\
X \\
\text{read} = \text{GC GCC}
\]

Figure S4: This figure shows the alignment of a read at a gap in the reference sequence.

\[
\text{reference} = \text{CAACAACAAACAACAA} \\
\text{reads} = \text{CAACAACAA} \\
\text{CAAACA - A} \\
\text{CAAACA - - A} \\
\text{CAAACA - - - A} \\
\text{CAAACA - - - - A}
\]

Figure S5: This figure shows an example of a read aligning against a repeat region in the reference sequence.
Figure S6: Normalized found intervals (percentage, category all) for different programs on reads ([EMBL:SRR000853], [EMBL:SRR003673], [EMBL:SRR038098]) from yeast. The gold standard was generated with an error rate of 8% and edit distance. The x axis shows the variation of error rate in the evaluation. The key is only shown once and is the same for all plots. Note the different axis scalings.
Figure S7: Normalized found intervals (percentage, category *any-best*) for different programs on reads ([EMBL:SRR000853], [EMBL:SRR003673], [EMBL:SRR038098]) from yeast. The gold standard was generated with an error rate of 8% and edit distance. The x axis shows the variation of error rate in the evaluation. The key is only shown once and is the same for all plots. Note the different axis scalings.
### Table S1: This table shows some popular read mappers with some key properties. The example evaluation was performed with these tools. The full source code is freely available for all tools. Note that while the authors of Soap2 mention gap support in the publication, we could find no option to switch on gapped alignments. Furthermore, the provided script for conversion to the SAM format outputs ungapped alignments only.

<table>
<thead>
<tr>
<th></th>
<th>Bowtie</th>
<th>Bwa</th>
<th>Soap2</th>
<th>Shrimp2</th>
</tr>
</thead>
<tbody>
<tr>
<td>algorithmic technique</td>
<td>FM-index based</td>
<td>FM-index based</td>
<td>FM-index based</td>
<td>q-gram counting</td>
</tr>
<tr>
<td></td>
<td>approximate string search</td>
<td>approximate string search</td>
<td>based seed search and verification</td>
<td>and verification</td>
</tr>
<tr>
<td>distance measures supported</td>
<td>Hamming</td>
<td>Hamming, Edit, Smith-Waterman</td>
<td>Hamming with gaps allowed</td>
<td>Smith-Waterman</td>
</tr>
<tr>
<td>supported read length</td>
<td>≤ 1024</td>
<td>arbitrary</td>
<td>≤ 1024</td>
<td>arbitrary</td>
</tr>
<tr>
<td>supports quality value</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>supports full sensitivity mode</td>
<td>Not for suboptimal hits</td>
<td>Not for suboptimal hits</td>
<td>No, only for up to two mismatches</td>
<td>Adjustable to be fully sensitive, but many parameters need to be set</td>
</tr>
</tbody>
</table>

**Table S2: For the combination of sequencing technology and species, this table gives the accession numbers of the reads used in the benchmark. The values in brackets are the lengths for Illumina reads and their means for 454 reads. The attributes “short” and “long” are relative to the length of the reads available in the SRA.**

<table>
<thead>
<tr>
<th>sequencing technology</th>
<th>S. cerevisiae</th>
<th>D. melanogaster</th>
</tr>
</thead>
<tbody>
<tr>
<td>454 LS</td>
<td>[EMBL:SRR000853] (267)</td>
<td>[EMBL:SRR034673] (273)</td>
</tr>
<tr>
<td>“short” Illumina</td>
<td>[EMBL:SRR038098] (20bp)</td>
<td>[EMBL:SRR026674] (36bp)</td>
</tr>
<tr>
<td>“long” Illumina</td>
<td>[EMBL:SRR003673] (36bp)</td>
<td>[EMBL:SRR049254] (100bp)</td>
</tr>
</tbody>
</table>
Table S3: This table shows the genomes used in the benchmark and information about them.

<table>
<thead>
<tr>
<th>Genome</th>
<th>Size</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>S. cerevisiae</td>
<td>12 Mbp</td>
<td>NC_001133-48, NC_001224</td>
</tr>
<tr>
<td>D. melanogaster</td>
<td>169 Mbp</td>
<td>flybase r5.29</td>
</tr>
</tbody>
</table>

Table S4: Running times and memory usage building the indices.

<table>
<thead>
<tr>
<th></th>
<th>S. cerevisiae</th>
<th>D. melanogaster</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bowtie time [s]</strong></td>
<td>17.0</td>
<td>454.9</td>
</tr>
<tr>
<td><strong>memory [MB]</strong></td>
<td>52.4</td>
<td>396.3</td>
</tr>
<tr>
<td><strong>Bwa time [s]</strong></td>
<td>8.6</td>
<td>200.8</td>
</tr>
<tr>
<td><strong>memory [MB]</strong></td>
<td>63.9</td>
<td>844.1</td>
</tr>
<tr>
<td><strong>Soap2 time [s]</strong></td>
<td>19.5</td>
<td>228.3</td>
</tr>
<tr>
<td><strong>memory [MB]</strong></td>
<td>273.6</td>
<td>440.4</td>
</tr>
</tbody>
</table>

Table S5: Running times and memory usage for read mappers on various read sets for D. melanogaster. Data is given for mapping in edit distance mode. Only Shrimp2 and Bwa were run on 454 reads. The times exclude any file conversion times. Missing entries in the memory rows are caused by too short running times hindering the memory usage measuring with `top`.

<table>
<thead>
<tr>
<th></th>
<th>[EMBL:SRR034673]</th>
<th>[EMBL:SRR026674]</th>
<th>[EMBL:SRR049254]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>454, ⊕ 273 bp</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Bwa time [s]</strong></td>
<td>69.6</td>
<td>38.5</td>
<td></td>
</tr>
<tr>
<td><strong>memory [MB]</strong></td>
<td>246.4</td>
<td>130.0</td>
<td>235.9</td>
</tr>
<tr>
<td><strong>Shrimp2 time [s]</strong></td>
<td>848.9</td>
<td>368.0</td>
<td>349.0</td>
</tr>
<tr>
<td><strong>memory [MB]</strong></td>
<td>5,153.9</td>
<td>5,046.6</td>
<td>5,046.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>[EMBL:SRR001003]</th>
<th>[EMBL:SRR060114]</th>
<th>[EMBL:SRR049254]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Illumina, 36 bp</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Bwa time [s]</strong></td>
<td>1.2</td>
<td>38.5</td>
<td></td>
</tr>
<tr>
<td><strong>memory [MB]</strong></td>
<td>69.6</td>
<td>130.0</td>
<td>235.9</td>
</tr>
<tr>
<td><strong>Shrimp2 time [s]</strong></td>
<td>848.9</td>
<td>368.0</td>
<td>349.0</td>
</tr>
<tr>
<td><strong>memory [MB]</strong></td>
<td>5,153.9</td>
<td>5,046.6</td>
<td>5,046.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>[EMBL:SRR001003]</th>
<th>[EMBL:SRR060114]</th>
<th>[EMBL:SRR049254]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Illumina, 100 bp</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Bwa time [s]</strong></td>
<td>1.6</td>
<td>38.5</td>
<td></td>
</tr>
<tr>
<td><strong>memory [MB]</strong></td>
<td>69.6</td>
<td>130.0</td>
<td>235.9</td>
</tr>
<tr>
<td><strong>Shrimp2 time [s]</strong></td>
<td>848.9</td>
<td>368.0</td>
<td>349.0</td>
</tr>
<tr>
<td><strong>memory [MB]</strong></td>
<td>5,153.9</td>
<td>5,046.6</td>
<td>5,046.6</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------</td>
<td>---------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Bowtie memory [MB]</td>
<td>–</td>
<td>20.0</td>
<td>20.0</td>
</tr>
<tr>
<td>time [s]</td>
<td>–</td>
<td>1.7</td>
<td>0.9</td>
</tr>
<tr>
<td>Bowtie* memory [MB]</td>
<td>–</td>
<td>20.0</td>
<td>–</td>
</tr>
<tr>
<td>time [s]</td>
<td>–</td>
<td>1.7</td>
<td>0.9</td>
</tr>
<tr>
<td>Bwa memory [MB]</td>
<td>29.4</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>time [s]</td>
<td>15.1</td>
<td>0.3</td>
<td>0.5</td>
</tr>
<tr>
<td>Shrimp2 memory [MB]</td>
<td>1,932.7</td>
<td>1,718.0</td>
<td>1,718.0</td>
</tr>
<tr>
<td>time [s]</td>
<td>44.2</td>
<td>29.8</td>
<td>32.9</td>
</tr>
<tr>
<td>Soap2 memory [MB]</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>time [s]</td>
<td>–</td>
<td>0.4</td>
<td>0.5</td>
</tr>
<tr>
<td>Soap2* memory [MB]</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>time [s]</td>
<td>–</td>
<td>0.5</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Table S6: Running times and memory usage for read mappers on various read sets for S. cerevisiae. Data is given for mapping in edit distance mode. Missing entries in the memory rows are caused by too short running times hindering the memory usage measuring with `top`. The times exclude any file conversion times.

<table>
<thead>
<tr>
<th></th>
<th>Illumina 36 bp</th>
<th>Illumina 50 bp</th>
<th>Illumina 100 bp</th>
<th>454 ⊕ 200 bp</th>
<th>454 ⊕ 400 bp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bowtie</td>
<td>82.3</td>
<td>75.0</td>
<td>57.6</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Bowtie*</td>
<td>82.3</td>
<td>75.0</td>
<td>61.6</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Bwa</td>
<td>96.4</td>
<td>95.9</td>
<td>96.2</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Shrimp2</td>
<td>94.9</td>
<td>94.7</td>
<td>95.7</td>
<td>92.2</td>
<td>93.9</td>
</tr>
<tr>
<td>Soap2</td>
<td>82.4</td>
<td>74.8</td>
<td>57.1</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Soap2*</td>
<td>82.4</td>
<td>74.8</td>
<td>59.6</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

Table S7: Performance in found interval percentage for the read mappers on simulated S. cerevisiae reads. Data is given for mapping in edit distance mode. Only Shrimp2 was run on simulated 454 mate-paired reads.
References


