Supplementary Figure 1

Correlation plot of estimated Jaccard similarity vs. directly computed fraction of matched k-mers and true Jaccard similarity

(A,B) Correlation plots for sketch size of 1256. (C,D) Correlation plots for sketch size of 512. Y-axes are the MinHash-estimated Jaccard similarity. X-axes are directly computed fraction of k-mer matching and true Jaccard similarity. The dots represent randomly selected 10K long reads extracted from the human genome, corrupted synthetically with PacBio simulated noise, as described in Supplementary Note 1. Each blue dot represents an independently selected pair of 20% overlapping sequences, while each red dot represents two non-overlapping reads. There are a total of 5,000 blue and 5,000 red dots in each plot. The dashed lines show the associated linear regression line.
Supplementary Figure 2

ROC plots for MHAP and BLASR with varying parameters for *E. coli* K12

All programs ran without any filtering for high-copy *k*-mers or simple repeats. The sequences were aligned to the reference genome to obtain the true sequence locations. Varied MHAP parameters were: *k*-mer size (12–16), sketch-size (256–1512), minimum matching min-mers (2–4), and minimum Jaccard threshold (0.02–0.04). Varied BLASR parameters were: *k*-mer size (10–16) and bestn (1C–100C, where C is the sequencing depth). Two reads whose mapping overlapped on the reference for more than 2,000bp were considered overlapping. Any reads whose mapping did not overlap, but were identified as overlapping were considered true when an overlap greater than 70% identity was confirmed by a Smith-Waterman alignment of the region (Supplementary Note 9). Therefore, high-quality, repeat-induced overlaps were not considered false positives. Predicted overlaps were evaluated against this standard to compute positive predictive value (PPV) and sensitivity. BLASR data points are represented by black diamonds, with the largest indicating default parameters. Colored circles are MHAP data points, with the circled points indicate the default MHAP parameter settings (fast or sensitive). Speedups for MHAP are given relative to the fastest BLASR run that achieved a minimum of 80% sensitivity. Results are given for three PacBio SMRT chemistries: (A, D) C2 chemistry, (B, E) P4C2 chemistry, and (C, F) P5C3 chemistry. DALIGNER performance is represented by a star on the P5C3 chemistry plots (C, F).
Supplementary Figure 3

ROC plots for MHAP and BLASR with varying parameters for A. thaliana

Same plot as Supplementary Figure 2 but for A. thaliana Ler-0 sequencing using (A, C) P4C2 chemistry and (B, D) P5C3 chemistry. For this dataset, only BLASR parameters ($k=10-16$, $bestn=250C$) achieved $>80\%$ sensitivity. When sequences and overlaps matching the plastid and mitochondrial genomes were removed, BLASR sensitivity increased from $\sim2\%$ to $\sim17\%$, suggesting repetitive sequences are responsible for the low overlap sensitivity.
Contig NG50 (A) and NG10 (B) lengths are given for assemblies of *S. cerevisiae* at various sequencing coverage (15–120X) using BLASR, MHAP fast, MHAP sensitive in CA 8.2, and a beta version of CA 8.3 PBcR-MHAP. For this example, there is benefit from increasing overlap sensitivity at lower coverage. The MHAP fast parameters mirror the BLASR assembly, while the MHAP sensitive parameters outperform the other assemblies at lower coverage. CA 8.3 PBcR-MHAP automatically switches to MHAP sensitive below 50X and fast above. The dashed line is an Illumina Miseq 2x300 @450bp assembly from Lee et. al. The ECTools result is also from Lee et. al. using the Illumina MiSeq assembly for PacBio correction.
Supplementary Figure 5

Mummerplot of the PBCr-MHAP assembly and *E. coli* reference

An alignment dotplot shows the relationship between the assembled contig of *E. coli* K12 (y-axis) and the *E. coli* K12 reference genome (x-axis). The single-contig assembly matches the reference over the entire length of the genome. The origin of the assembly is arbitrarily shifted due to the chromosome being circular, and does not represent an assembly error.
An alignment dotplot shows the relationship between the assembled contigs of *S. cerevisiae* W303 (y-axis) and the *S. cerevisiae* S288C reference genome (x-axis). Contig and chromosome boundaries are displayed as dotted lines (horizontal and vertical, respectively), and contigs are ordered and oriented to match the reference using mummerplot’s --fat option. The majority of chromosomes are assembled into a single contig.
An alignment dotplot shows the relationship between the assembled contigs of *A. thaliana* Ler-0 (y-axis) and the *A. thaliana* Col-0 reference genome (x-axis). Contig and chromosome boundaries are displayed as dotted lines (horizontal and vertical, respectively), and contigs are ordered and oriented to match the reference using mummerplot’s --fat option. The reference genome has been scaffolded with N’s, thus appearing more continuous in this plot than in reality. In terms of contigs, the assembly has better continuity than the reference assembly, with larger N25, N75, and N95 contig sizes. A small, potentially strain-specific inversion is visible towards
the top-right of the plot, in blue.
Supplementary Figure 8

Mummerplot of the PBCr-MHAP assembly and *D. melanogaster* reference

An alignment dotplot shows the relationship between the assembled contigs of *D. melanogaster* ISO1 (y-axis) and the *D. melanogaster* ISO1 reference genome (x-axis). Contig and chromosome boundaries are displayed as dotted lines (horizontal and vertical, respectively), and contigs are ordered and oriented to match the reference using mummerplot’s --fat option. Several contigs represent full chromosome arms, stretching into the telomeric and pericentromeric repeats. The more fragmented chromosome towards the middle of the plot is X, which was present in this sample at only half coverage (only male flies were sequenced). The heavily
fragmented alignments at the top-right are “unplaced” reference sequences, which are largely repetitive and not associated with a particular chromosome.
Assembled contigs of CHM1 are overlaid on human GRCh38 chromosomes. Alternating shades indicate adjacent contig mappings, so each transition from gray to black represents a contig boundary or alignment breakpoint. White indicates gaps in the tiling, the majority of which are uncharacterized reference sequence. Most chromosomes are covered by a few, large contigs that contiguously map to the...
reference.
Supplementary Figure 10

Contig tiling for the Illumina CHM1 assembly

Assembled Illumina contigs of CHM1 are overlaid on human GRCh38 chromosomes. Alternating shades indicate adjacent contig mappings, so each transition from gray to black represents a contig boundary or alignment breakpoint. White indicates gaps in the tiling, the majority of which are uncharacterized reference sequence. Most contigs are short, breaking each chromosome into
thousands of contigs.
Contig tiling for the Ref28/hg10 human assembly

Contigs from an early BAC-based Sanger assembly of human are overlaid on human GRCh38 chromosomes. Alternating shades indicate adjacent contig mappings, so each transition from gray to black represents a contig boundary or alignment breakpoint. There is
a mix of long and short contigs, depending on the chromosome.
Supplementary Figure 12

Mummerplot of the PBcR-MHAP assembly and human reference

An alignment dotplot shows the relationship between the assembled contigs of CHM1 (y-axis) and the GRCh38 reference genome (x-axis). Contig and chromosome boundaries are displayed as dotted lines (horizontal and vertical, respectively), and contigs are ordered and oriented to match the reference using mummerplot’s --layout option. The missing reference chromosome is Y, which is absent from the CHM1 genome.
Supplementary Figure 13

An example of the FalconSense algorithm

(a) Four reads and one template as the input of the algorithm shown. The assigned $p$ and $d$ are shown along with the alignments between the reads and the template. (b) The sorted tag list $B$ and the counts for each tag are shown. In this case, if a tag count is greater than 2, a consensus base will be generated. The final consensus sequence, ATATACGGC, is just a simple concatenation of the non-empty bases.
### Supplementary Table 1. MHAP sensitivity across varying sequence lengths.

<table>
<thead>
<tr>
<th>Sequence Length</th>
<th># Sequences</th>
<th>Overlap Length</th>
<th>Sensitivity (512)</th>
<th>Sensitivity (1256)</th>
<th>Specificity (both)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5000</td>
<td>124,073</td>
<td>1000</td>
<td>50%</td>
<td>77%</td>
<td>100%</td>
</tr>
<tr>
<td>10000</td>
<td>62,037</td>
<td>2000</td>
<td>52%</td>
<td>79%</td>
<td>100%</td>
</tr>
<tr>
<td>15000</td>
<td>41,358</td>
<td>3000</td>
<td>51%</td>
<td>76%</td>
<td>100%</td>
</tr>
<tr>
<td>20000</td>
<td>31,019</td>
<td>4000</td>
<td>52%</td>
<td>76%</td>
<td>100%</td>
</tr>
<tr>
<td>25000</td>
<td>24,815</td>
<td>5000</td>
<td>53%</td>
<td>79%</td>
<td>100%</td>
</tr>
<tr>
<td>30000</td>
<td>20,679</td>
<td>6000</td>
<td>54%</td>
<td>80%</td>
<td>100%</td>
</tr>
<tr>
<td>35000</td>
<td>17,725</td>
<td>7000</td>
<td>54%</td>
<td>80%</td>
<td>100%</td>
</tr>
<tr>
<td>40000</td>
<td>15,510</td>
<td>8000</td>
<td>55%</td>
<td>80%</td>
<td>100%</td>
</tr>
<tr>
<td>45000</td>
<td>13,786</td>
<td>9000</td>
<td>57%</td>
<td>84%</td>
<td>100%</td>
</tr>
<tr>
<td>50000</td>
<td>12,408</td>
<td>10000</td>
<td>53%</td>
<td>70%</td>
<td>100%</td>
</tr>
<tr>
<td>55000</td>
<td>11,280</td>
<td>11000</td>
<td>53%</td>
<td>77%</td>
<td>100%</td>
</tr>
<tr>
<td>60000</td>
<td>10,340</td>
<td>12000</td>
<td>53%</td>
<td>85%</td>
<td>100%</td>
</tr>
<tr>
<td>65000</td>
<td>9,545</td>
<td>13000</td>
<td>53%</td>
<td>84%</td>
<td>100%</td>
</tr>
<tr>
<td>70000</td>
<td>8,863</td>
<td>14000</td>
<td>53%</td>
<td>83%</td>
<td>100%</td>
</tr>
<tr>
<td>75000</td>
<td>8,272</td>
<td>15000</td>
<td>52%</td>
<td>83%</td>
<td>100%</td>
</tr>
<tr>
<td>80000</td>
<td>7,755</td>
<td>16000</td>
<td>53%</td>
<td>82%</td>
<td>100%</td>
</tr>
<tr>
<td>85000</td>
<td>7,299</td>
<td>17000</td>
<td>52%</td>
<td>81%</td>
<td>100%</td>
</tr>
<tr>
<td>90000</td>
<td>6,893</td>
<td>18000</td>
<td>53%</td>
<td>80%</td>
<td>100%</td>
</tr>
<tr>
<td>95000</td>
<td>6,531</td>
<td>19000</td>
<td>53%</td>
<td>80%</td>
<td>100%</td>
</tr>
<tr>
<td>100000</td>
<td>6,204</td>
<td>20000</td>
<td>53%</td>
<td>78%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Sequences were simulated from the human genome varying in size from 5Kbp to 100Kbp (by 5Kbp) and sensitivity evaluated given a fixed overlap percentage (20%). Sensitivity is given for sketch sizes of 512 and 1256.
Supplementary Table 3. Program performance on increasing sequence lengths.

<table>
<thead>
<tr>
<th>Organism</th>
<th>Program</th>
<th>Sensitivity</th>
<th>ΔSensitivity</th>
<th>PPV</th>
<th>Time (CPU h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E. coli K12</td>
<td>DALIGNER</td>
<td>81%</td>
<td>-5%</td>
<td>100%</td>
<td>0.4</td>
</tr>
<tr>
<td></td>
<td>MHAP fast</td>
<td>69%</td>
<td>+4%</td>
<td>99%</td>
<td>1.06</td>
</tr>
<tr>
<td></td>
<td>MHAP sensitive</td>
<td>87%</td>
<td>+2%</td>
<td>99%</td>
<td>1.52</td>
</tr>
<tr>
<td>S. cerevisiae W303</td>
<td>DALIGNER</td>
<td>71%</td>
<td>-2%</td>
<td>100%</td>
<td>1.95</td>
</tr>
<tr>
<td></td>
<td>MHAP fast</td>
<td>72%</td>
<td>+2%</td>
<td>99%</td>
<td>2.80</td>
</tr>
<tr>
<td></td>
<td>MHAP sensitive</td>
<td>78%</td>
<td>-5%</td>
<td>96%</td>
<td>4.13</td>
</tr>
<tr>
<td>A. thaliana Ler-0</td>
<td>DALIGNER</td>
<td>79%</td>
<td>-7%</td>
<td>100%</td>
<td>3.50</td>
</tr>
<tr>
<td></td>
<td>MHAP fast</td>
<td>74%</td>
<td>+5%</td>
<td>100%</td>
<td>4.28</td>
</tr>
<tr>
<td></td>
<td>MHAP sensitive</td>
<td>90%</td>
<td>+2%</td>
<td>100%</td>
<td>6.05</td>
</tr>
<tr>
<td>D. melanogaster ISOT</td>
<td>DALIGNER</td>
<td>80%</td>
<td>-5%</td>
<td>100%</td>
<td>4.36</td>
</tr>
<tr>
<td></td>
<td>MHAP fast</td>
<td>67%</td>
<td>+5%</td>
<td>99%</td>
<td>3.52</td>
</tr>
<tr>
<td></td>
<td>MHAP sensitive</td>
<td>85%</td>
<td>+2%</td>
<td>99%</td>
<td>5.34</td>
</tr>
<tr>
<td>Human CHM1</td>
<td>DALIGNER</td>
<td>73%</td>
<td>-9%</td>
<td>99%</td>
<td>2.66</td>
</tr>
<tr>
<td></td>
<td>MHAP fast</td>
<td>68%</td>
<td>+4%</td>
<td>98%</td>
<td>2.07</td>
</tr>
<tr>
<td></td>
<td>MHAP sensitive</td>
<td>82%</td>
<td>+5%</td>
<td>98%</td>
<td>3.17</td>
</tr>
</tbody>
</table>

Sequences at least 10Kbp in length were extracted from the subsets in Table 1 and the same parameters from Table 1 run for DALIGNER and MHAP. ΔSensitivity gives the change in sensitivity for sequences >10Kbp versus the results of Table 1. While MHAP maintains sensitivity on long sequences, DALIGNER sensitivity drops by an average of 6%.
We tested the PBCR-MHAP algorithm on five genomes sequenced using PacBio SMRT sequencing. These datasets were generated and publically released by Pacific Biosciences\textsuperscript{1,2} and can be downloaded from: https://github.com/PacificBiosciences/DevNet/wiki/Datasets or NCBI SRA. For *E. coli* and *A. thaliana*, only the P5C3 datasets were used to generate the assemblies reported. The raw sequence lengths are reported before correction.

**Supplementary Table 4. Sequence data used in this study.**

<table>
<thead>
<tr>
<th>Genome</th>
<th>Download</th>
<th>Reference</th>
<th>Coverage</th>
<th># Sequences</th>
<th>Chemistry</th>
<th>Mean (bp)</th>
<th>Max (bp)</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Escherichia coli</em> K12</td>
<td>SRX255228</td>
<td>NC_000913</td>
<td>86X</td>
<td>161,791</td>
<td>C2</td>
<td>2,472.91</td>
<td>14,632</td>
</tr>
<tr>
<td><em>Escherichia coli</em> K12</td>
<td>SRX669475</td>
<td>NC_000913</td>
<td>94X</td>
<td>82,590</td>
<td>P4C2</td>
<td>5291.09</td>
<td>22,609</td>
</tr>
<tr>
<td><em>Escherichia coli</em> K12</td>
<td>SRX533603</td>
<td>NC_000913</td>
<td>85X</td>
<td>47,910</td>
<td>P5C3</td>
<td>8282.76</td>
<td>28,647</td>
</tr>
<tr>
<td><em>Saccharomyces cerevisiae</em> W303</td>
<td>SRX533604</td>
<td>NC_001133:NC_001148</td>
<td>117X</td>
<td>232,230</td>
<td>P4C2</td>
<td>6039.55</td>
<td>30,164</td>
</tr>
<tr>
<td><em>Arabidopsis thaliana</em> Ler-0</td>
<td>SRX533606</td>
<td>TAIR10</td>
<td>110X</td>
<td>3,444,146</td>
<td>P4C2</td>
<td>4,142.16</td>
<td>41,753</td>
</tr>
<tr>
<td><em>Arabidopsis thaliana</em> Ler-0</td>
<td>SRX533607</td>
<td>TAIR10</td>
<td>144X</td>
<td>2,021,431</td>
<td>P5C3</td>
<td>8,572.91</td>
<td>47,445</td>
</tr>
<tr>
<td><em>Drosophila melanogaster</em> ISO1</td>
<td>SRX499318</td>
<td>Ref v6</td>
<td>90X</td>
<td>1,889,814</td>
<td>P5C3</td>
<td>9,317.04</td>
<td>44,766</td>
</tr>
<tr>
<td>Human CHM1</td>
<td>SRX533609</td>
<td>Rev v38</td>
<td>54X</td>
<td>22,565,609</td>
<td>P5C3</td>
<td>7,447.12</td>
<td>42,774</td>
</tr>
</tbody>
</table>
### Supplementary Table 5. GAGE metrics for *E. coli* and *D. melanogaster*.

<table>
<thead>
<tr>
<th>Genome</th>
<th>Reference</th>
<th>CN50</th>
<th># SNPs</th>
<th># Indels &lt;5bp</th>
<th>QV (SNP+Indel)</th>
<th># Indels &gt;5bp</th>
<th>Inversions</th>
<th>Relocations</th>
<th>Translocations</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>E. coli</em> K12 BLASR pre-quiver</td>
<td>NC_000913</td>
<td>656,882</td>
<td>12</td>
<td>1,386</td>
<td>35.21</td>
<td>8</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><em>E. coli</em> K12 BLASR post-quiver</td>
<td>NC_000913</td>
<td>2,333,020</td>
<td>3</td>
<td>13</td>
<td>54.62</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><em>E. coli</em> K12 MHAP pre-quiver</td>
<td>NC_000913</td>
<td>838,903</td>
<td>6</td>
<td>551</td>
<td>39.20</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><em>E. coli</em> K12 MHAP post-quiver</td>
<td>NC_000913</td>
<td>838,907</td>
<td>3</td>
<td>8</td>
<td>56.25</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><em>E. coli</em> K12 SPAdes Illumina</td>
<td>NC_000913</td>
<td>117,642</td>
<td>129</td>
<td>18</td>
<td>44.99</td>
<td>7</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td><em>E. coli</em> K12 SPAdes Illumina/SMRT hybrid</td>
<td>NC_000913</td>
<td>408,306</td>
<td>266</td>
<td>17</td>
<td>42.15</td>
<td>6</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td><em>D. melanogaster</em> BLASR pre-quiver</td>
<td>Ref v5</td>
<td>216,453</td>
<td>8,577</td>
<td>75,290</td>
<td>31.64</td>
<td>2,144</td>
<td>20</td>
<td>79</td>
<td>10</td>
</tr>
<tr>
<td><em>D. melanogaster</em> BLASR post-quiver</td>
<td>Ref v5</td>
<td>470,986</td>
<td>4,443</td>
<td>24,720</td>
<td>36.24</td>
<td>400</td>
<td>18</td>
<td>75</td>
<td>8</td>
</tr>
<tr>
<td><em>D. melanogaster</em> MHAP pre-quiver</td>
<td>Ref v5</td>
<td>303,294</td>
<td>14,985</td>
<td>141,773</td>
<td>28.98</td>
<td>3,152</td>
<td>17</td>
<td>85</td>
<td>20</td>
</tr>
<tr>
<td><em>D. melanogaster</em> MHAP post-quiver</td>
<td>Ref v5</td>
<td>645,364</td>
<td>3,141</td>
<td>8,338</td>
<td>40.42</td>
<td>357</td>
<td>17</td>
<td>63</td>
<td>7</td>
</tr>
</tbody>
</table>

The assemblies were aligned to the reference genome as above and GAGE assembly correctness metrics were calculated. Perceived errors due to circular chromosomes were manually excluded. The *D. melanogaster* ArmU and ArmUextra were excluded from validation as they contain unordered sequences which cannot be used for continuity validation.
Supplementary Note 1: Overlap analysis

In order to determine the sketch size and k-mer size required to effectively discriminate between true and false matches, we developed a simulator to test Jaccard similarity. Sequences were simulated from a given reference genome and Jaccard similarity was measured for true and random matches. Repetitive k-mers can optionally be ignored when calculating similarity. The simulator is included in the MHAP source code and can be executed as:

```
java edu.umd.marbl.mhap.main.KmerStatsSimulator 50000 <10 or 16> 10000 2000 0.10 0.02 0.01 false <reference.fasta>
```

to simulate pairwise overlap error rates, or as:

```
java edu.umd.marbl.mhap.main.KmerStatsSimulator 50000 <10 or 16> 10000 2000 0.10 0.02 0.01 true <reference.fasta>
```

to simulate mapping error rates. For Figure 2 in the main manuscript, GRCh37 was used to select random sequences with no filtering for repetitive k-mers.

To examine the performance for increasing sequence lengths, we simulated 620Mbp (0.2X) of sequences from 5Kbp to 100Kbp in length. The sequences can be simulated using the command:

```
java edu.umd.marbl.mhap.main.KmerStatsSimulator <# sequences to generate> <sequence length> 0.10 0.02 0.01 <reference.fasta>
```

Finally, we ran the “direct” implementation with output disabled as:

```
java edu.umd.marbl.mhap.main.DirectAlignMain -k 16 --num-hashes 0 --num-min-matches 100000 --threshold 0.03 --subsequence-size 100000 --min-store-length 2000 --num-threads 32 -s <simulated_sequences.fasta>
```

and our MinHash implementation as:

```
java edu.umd.marbl.mhap.main.FastAlignMain -k 16 --num-hashes <512/1256> --num-min-matches 3 --threshold 0.04 --subsequence-size 100000 --min-store-length 2000 --num-threads 32 -s <simulated_sequences.fasta> > <mhap.ovls>
```

As these were simulated sequences, the truth was already known and sensitivity was evaluated as described in the methods section of manuscript:

```
java edu.umd.marbl.mhap.main.EstimateROC <truth.m4> <mhap.ovls> <simulated_sequences.fasta> <min overlap> 10000 true false
```

The minimum overlap size was held constant at 20% of sequence length. Sensitivity of MHAP averaged 53% for sketch-size=512 and 80% for sketch size=1256.
Supplementary Note 2: MHAP Parameter Selection

Supplementary Figure 1 demonstrates the relation between k-mer counting, Jaccard similarity, and approximating the Jaccard similarity using MinHash. Increasing $H$ decreases the variance in the MinHash Jaccard similarity estimate, which has a direct monotonic relationship with k-mer counts (see Eq. 9). Supplementary Figure 1 B,D show that the slope of the regression line between actual Jaccard similarity and MinHash estimated similarity is near or below 1, demonstrating that random hash collisions for 16-mers have a negligible impact the Jaccard estimate. If collisions were frequent, the estimate would be expected to overestimate the true value. For smaller $H$, the slope for of the randomly overlapping reads (red line in S1D) is lower than expected, since the lack of MinHash collisions for such small Jaccard values tends to underestimate the Jaccard similarity. This bias in the estimate disappears with increasing $H$.

When selecting parameters for MHAP one is inherently trying to optimize the tradeoff between three observables, sensitivity, specificity, and speed. These observables are controlled primarily by three parameters, the sketch size $H$, the k-mer size, and the cutoff similarity value of the second filter. Given that the desired sensitivity vs. specificity vs. speed tradeoffs are application specific, there are no globally optimal settings even for a specific read size and error rate. However, there are general relationships between the observables and the input parameters that can help guide users in adjusting for specific applications, if the default settings are inadequate.

The optimal first stage filter settings depend primarily on the size of the genome being assembled and the coverage. Since the number of false positive scales quadratically with genome size, it is important to minimize false positives for larger genomes. In addition, for very long reads the number of k-mers shared by two random reads increases exponentially, which also increases the false positive rate (decreases PPV). Both of these effects can be suppressed by increasing the k-mer size. However, there is a limit to how big $k$ can be, since for large $k$ no k-mers will be shared between two reads, thus decreasing sensitivity. $k$ is set to 16 by default, the largest value found to work well empirically for large genomes and current PacBio chemistries. For smaller genomes this value can be reduced to 14, which will increase the number of k-mers shared between overlapping sequences and allow a smaller sketch size $H$. In turn, a lower $H$ accelerates the computation and decreases memory usage, but will allow more false positives through the first stage filter. Thus, $H$ is linked to k-mer size, and also affects sensitivity vs. speed tradeoff, with a larger value of $H$ improving the accuracy of the Jaccard estimate (Supplementary Figure 1).

The second stage filter has only one primary parameter, the similarity cutoff for the overlapping region. In most cases, for a given read error rate, it should be held constant. This parameter controls the sensitivity vs. specificity tradeoff of the final output and has no effect on runtime. Note that the limit of sensitivity is bounded by the first stage filter, and so setting an unnecessarily small similarity threshold would result in a lower PPV without increasing sensitivity.
Optionally, MHAP can accept a list of repetitive k-mers that cannot be included in a sketch for a sequence. For assembly, a conservative threshold is set to exclude k-mers, which represent more than 0.001% of all k-mers in the sequences. However, MHAP can operate efficiently without filtering of k-mers, as shown in Supplementary Figures 2, 3 and Supplementary Table 2. In tests on the genomes from Table 1, the average speed increase with this conservative filtering was <25%. Increasing the filtered k-mers reduces the false positive rate of the first-stage MHAP filter, thus allowing a decreased k-mer length and H, improving performance. However, this can also decrease sensitivity, especially for repetitive regions.

To select the optimal set of parameters on real datasets, we ran BLASR and MHAP with a varying parameter space. The sensitivity and accuracy of overlaps was measured on *E. coli* using three available chemistry datasets (C2, P4C2, and P5C3) (Supplementary Table 2, Supplementary Figure 2). MHAP was run with varying k-mer size (10-16), # hashes (64-1512), and minimum # matches (2-4). BLASR was run with varying k-mer size (10-16) and # best matches (1C, 5C, 10C, 100C), where C represents approximate genome coverage (100X in this case). The same analysis was repeated on a random subset of *A. thaliana*. A random subset of sequences adding up to 500Mbp was selected and parameters evaluated (Supplementary Table 2, Supplementary Figure 3). We selected two sets of parameters based on these results, *fast* (H=512, k=16, # matches=3) and *sensitive* (H=1256, k=16, #matches =3).
Supplementary Note 3: Comparison of overlapping tools

The latest versions as of July 2014 of BWA-MEM (0.7.9a), RazerS (3.1.1), SNAP (1.0beta.10), and BLASR (SMRTanalysis 2.2.0) and DALIGNER as of November 20th, 2014 were downloaded. A simulated dataset of 543 sequences from lambda phage was used to evaluate both mapping to reference and overlap detection.

First, programs were run to map sequences to the reference to ensure they were compatible with noisy, long sequences:

BWA-MEM was run as:
```bash
bwa mem -t 16 -x pacbio lambda.fasta lambda.sim.varyLength.fasta > bwamem.sam 2> bwamem.out
```

BLASR was run as:
```bash
blasr -nproc 16 -maxLCPLength 16 -minMatch 12 -m 4 -bestn 1 lambda.sim.varyLength.fasta lambda.fasta > blasr.m4 2>blasr.err
```

RazerS was run as:
```bash
razers3 -i 85 -tc 16 -o ill.razers ../lambda.fasta lambda.sim.varyLength.fasta
```

SNAP was run as:
```bash
snapxl single lambda_snap/ lambda.sim.varyLength.fastq -o snap.sam -d 700 -t 16
```

BLASR and BWA-MEM performed acceptably. SNAP was unable to align a majority of the sequences to the reference genome due to a compile-time restriction of the edit distance to 63. After recompiling to increase this threshold, a total of 65% of sequences could be mapped to the reference. RazerS did not complete after 30 minutes and was excluded from further testing. Next, sequences were aligned to each other to identify pairwise overlaps:

BWA-MEM was run as:
```bash
bwa mem -t 16 -x pbread lambda.sim.varyLength.fasta lambda.sim.varyLength.fasta > bwamem_self.out 2> bwamem_self.err
```

BLASR was run as:
```bash
blasr -nproc 16 -maxLCPLength 16 -minMatch 12 -m 4 -bestn 50 lambda.sim.varyLength.fasta lambda.sim.varyLength.fasta > blasr.m4 2>blasr.err
```

SNAP was run as:
```bash
snapxl single output lambda.sim.varyLength.fastq -o snap.sam -d 2900 -t 16
```

BWA-MEM sensitivity was 64% and was excluded from further analysis due to low sensitivity on this dataset. Subsequently to our testing, BWA-MEM support for PacBio sequence overlapping was removed in version 0.7.11. SNAP only detected 221 overlaps out of a total of over 25K and was also excluded. Both BLASR and MHAP achieved over 90% sensitivity.
DALIGNER was run with the parameters:

```bash
fasta2DB DB <fasta input>
DBdust DB
DBsplit -x0 DB
HPCdalign -d -v -H2000 DB > script.sh
sh script.sh
```

With default parameters, DALIGNER loses ~6% in sensitivity when overlapping long sequences, whereas MHAP generally benefits from long sequences (Supplementary Table 3). This could be a result of DALIGNER’s local alignment strategy, in contrast to MHAP’s strategy of considering each overlap in whole.

DALIGNER supports two repeat filters, DUST³ filtering as well as high-count k-mer filtering. We attempted to isolate the effects of filtering by running DALIGNER without both filters. For *E. coli*, DALIGNER ran in 5.6 CPU hrs with all filtering disabled (versus 0.96 filtered). Thus, filtering is required to achieve DALIGNER’s fast runtimes. However, k-mer filtering (with DUST filtering enabled) can have an unpredictable effect on sensitivity, with almost no discernible reduction in average sensitivity for most genomes, but an overall decrease of 8% on *S. cerevisiae*. The effect of DUST filtering is unclear, as DALIGNER failed to run (segmentation fault) without it on all datasets except *E. coli*. 
Supplementary Note 4: Coverage Analysis

To evaluate the impact of varying coverage and sensitivity on genome assembly, the S. cerevisiae dataset was subsampled from 25–100X, with a step size of 5X. Subsets were generated using the command:

```
fastqSample -I filtered_subreads -U -O <coverage> -g 12000000 -c <coverage>
```

Two different MHAP parameters were run (fast and sensitive) as well as BLASR with defaults. A beta version of CA 8.3 was also included for comparison. A hybrid assembly of the same genome and Illumina-only assembly from Lee et. al. was included as a baseline. At low coverage, there is benefit from increasing sensitivity, and MHAP sensitive parameters are recommended. By higher coverage, the performance is approximately equal between sensitive and fast parameters, allowing the use of the faster parameter settings without sacrificing assembly quality (Supplementary Figure 4).
**Supplementary Note 5: Assembly validation**

PBr-MHAP was able to accurately resolve the entire genome of *E. coli* K12 using 85X of SMRT reads in 4.6 CPU hours, or 20 minutes using a modern 16-core desktop computer. The SPAdes assembler\(^5\) required 3.6 CPU hours to assemble 85X Illumina reads from the same genome, but left the genome fragmented in more than 100 contigs. A SPAdes hybrid assembly of all short and long read data required 4.9 CPU hours, and improved upon the short-read assembly. However, the hybrid assembly left the genome in 63 contigs, versus a single, complete contig from PBr-MHAP. In comparison to both SPAdes assemblies, the initial MHAP assembly does contain more single-base insertion/deletion (Indel) errors, but polishing this assembly with Quiver (requiring an additional 6.6 CPU hours) resulted in a consensus accuracy >99.999% and the lowest number of consensus errors (11 vs. 147 for SPAdes) (Supplementary Table 5).

All genomes, except human, were aligned to their reference genomes and plotted using Nucmer (Supplementary Figures 5-8, 12). The commands to generate alignments were:

```bash
nucmer -mumref -l 100 -c 1000 -d 10 -banded -D 5 <ref.fasta> <asm.fasta>
delta-filter -i 95 -o 95 out.delta > out.best.delta
dnadiff -d out.best.delta
```

Human alignments were generated with:

```bash
nucmer -mumref -l 100 -c 1000 <ref.fasta> <asm.fasta>
delta-filter -i 95 -o 95 out.delta > out.best.delta
dnadiff -d out.best.delta
```

We used GAGE metrics to evaluate the *E. coli* and *D. melanogaster* assemblies against the reference genomes constructed from the same strain (Supplementary Table 5). The following command was used to identify contig counts for each reference chromosome:

```bash
cat out.1delta |awk '{if ($7 > 99 && $5 > 200) print $NF "$(NF-1)"}' |sort |uniq |awk '{print $NF}' |sort |uniq -c
```

To identify resolved gaps in *D. melanogaster*, the v5 reference was split at ten or more contiguous Ns and each break considered a gap, for a total of 124 gaps. Based on the Nucmer alignment generated above, a contig with at least 500bp leading up to and 500bp after the gap was considered to potentially close the gap. Finally, the size of the patch in the contig was compared to the size of the reference gap. The patch was considered to match the reference gap length if the reference gap was exactly 100bp (an artificial gap length) or within 25% of the patch size.

Human gaps were restricted to the annotated gap table downloaded from UCSC genome browser [https://genome.ucsc.edu](https://genome.ucsc.edu) for Ref38 (a total of 819 gaps) and the same procedure followed. Further validation is required to determine if these patch sequences can be incorporated into the reference sequences.
Supplementary Note 6: Human assembly analysis

Genes falling within the MHC region (chr6:29655981-33193468) were downloaded from UCSC\(^6\) genome browser ([http://genome.ucsc.edu/cgi-bin/hgTables?hgsid=382525021_VQMxM60dyQG3vGeAVldr99Usmiu&boolshad.hgta_printCustomTrackHeaders=0&hgta_ctName=tb_refGene&hgta_ctDesc=table+browser+query+on+refGene&hgta_ctVis=pack&hgta_ctUrl=&fbQual=whole&fbUpBases=200&fbExonBases=0&fbIntronBases=0&fbDownBases=200&hgta_doGetBed=get+BED](http://genome.ucsc.edu/cgi-bin/hgTables?hgsid=382525021_VQMxM60dyQG3vGeAVldr99Usmiu&boolshad.hgta_printCustomTrackHeaders=0&hgta_ctName=tb_refGene&hgta_ctDesc=table+browser+query+on+refGene&hgta_ctVis=pack&hgta_ctUrl=&fbQual=whole&fbUpBases=200&fbExonBases=0&fbIntronBases=0&fbDownBases=200&hgta_doGetBed=get+BED)). Each individual gene sequence was extracted from the reference and mapped with Nucmer to the assembly contigs covering this region in the assembly. Split mappings were included and the gene coverage was computed as the sum of mapped bases, while identity computed as the average mapped identity for all bases.

To generate the tiling figures, sequences were aligned as in Supplementary Note 5 using Nucmer and a custom script to convert output to the format expected by coloredChromosomes.pl ([http://sourceforge.net/projects/cchrom/](http://sourceforge.net/projects/cchrom/)):

```
python makeMappings.py ref.1coords 10000 > ref1.tiling
perl convertToChr.pl human.chr.map ref1.tiling human.lanes human.chrPos > ref1.cfg
perl coloredChromosomes.pl --chromosomeSpec ref1.cfg -o ref1.ps
ps2pdf ref1.ps
```

Because Nucmer was run using unique match anchors (to accelerate the alignment), some repetitive sequence remained unaligned. To avoid showing these regions as gaps in the tiling, the conversion script chains together consecutive alignments from the same contig if the alignment gap in the reference is less than 10,000bp. Thus, breaks in the resulting tiling occur whenever a contig switch occurs or there is a >10,000bp gap between two alignments of the same contig. The entire process (alignment and figure generation) can be reproduced using the scripts available on the MHAP home page (assuming perl, python, and MUMmer are in the system path) by running the following to generate a ref.pdf:

```
sh makeHuman.sh ref.fasta asm.fasta
```

To validate consensus accuracy, 40-fold coverage Illumina data was downloaded (SRX652547, 3 runs) and mapped using Bowtie\(^8\) to the full MHAP assembly, both before and after Quiver polishing. FreeBayes\(^9\) was used to identify variants. Only bases with at least 3 mapped reads were considered both for variant calling and for total assembly size. Any variant supported by more than 50% of the mapped sequence was considered an error in the assembly and reported. The final QV computed using the formula:

\[-10 * \log_{10}\left(\frac{\#\text{variants}}{\#\text{bases}}\right)\]

The commands to align and call variants were:

```
bowtie -build asm.fasta asm-ctg
bowtie -I 200 -X 800 -p 32 -f -l 25 -e 140 --best --chunkmbs 256 -k 1 -S --un=unaligned.fasta -q asm-ctg -l <run>.1.fastq -2 <run>.2.fastq
samtools view -bS <run>.sam > <run>.bam
samtools sort <run>.bam <run>.sorted.bam
```

Nature Biotechnology: doi:10.1038/nbt.3238
Each run was aligned independently and the results merged:
samtools merge -h <header> asm.sorted.bam <run1>.sorted.bam <run2>.sorted.bam <run3>.sorted.bam

FreeBayes ran as:
freebayes -C 2 -O -q 20 -F 0.5 -z 0.02 -E 0 -X -u -p 1 -b asm.sorted.bam -v asm.vcf -f asm.fasta

In the unpolished assembly, a total of 3.01GB (92.6% of the bases) had >=3X coverage and the QV estimate was 33.9. In the polished assembly, a total of 3.06GB (94.1%) had >= 3X coverage and the QV estimate was 40.1.

Finally, the 16 BACs used for validation by Chaisson et al. (AC243499.2, AC243585.2, AC243586.3, AC243629.3, AC243650.3, AC243654.3, AC243734.3, AC243742.3, CH17-09106, CH17-144M16, CH17-150B4, CH17-285M6, CH17-390G16, CH17-63L4, CH17-68I14, CH17-9E4) were aligned to the assembly using the commands:
nucmer -mumref -b 5000 -c 1000 -l 100 -p <BAC name> asm.fasta
delta-filter -i 95 -o 95 <BAC name>.delta > <BAC name>.best.delta
dnadiff -d <BAC name>.best.delta -p <BAC name>

The default break length (-b) option was increased to avoid biasing the alignment towards well-aligned regions. All 16 BACs were fully contained in a single alignment to a single MHAP contig. All variants reported in out.snps were counted and QV calculated as above. An average QV was computed as the weighted mean (by length) for all mapped BACs. The estimated QV was 26.0 (99.69%) pre-quiver and 35.8 (99.97%) post-quiver.

In addition, all 102 BACs (totaling ~20Mbp, including the 16 above) from Chaisson et al. were mapped to the assembly as above. These BACs include 32 from the extremely complex 10q11.23 region. This 6.5Mb region of the genome includes segmentally duplicated sequence and contains 7 unfinished gaps and multiple mis-assemblies in the human GRCh37 reference. A weighted average identity (by alignment length) was computed. The average identity was 99.6563%. The unweighted average identity was 99.3409%. However, this figure may be artificially low due to mis-aligned repeats, unfinished BACs, and potential BAC errors. A majority of BACs (57 / 102) were contained in a single MHAP contig, including 14 currently unfinished BACs.
Supplementary Note 7: *D. melanogaster* assembly analysis

High-quality alignments were defined as those >1Kbp in length and >99% identity. Euchromatic regions of the genome were defined as reference sequence assigned to 2L, 2R, 3L, 3R, 4, and X, while heterochromatic regions were defined as reference sequence assigned to 2LHet, 2RHet, 3LHet, 3RHet, XHet, YHet. A total of 601 high-quality Nucmer alignments cover 126.8Mbp (97.8%) of the 129.7Mbp version 5 reference genome. Of these alignments, 118.7Mbp cover known euchromatic sequence and 8.1Mbp cover known heterochromatic sequence.

FlyBase 5.57_FB2014_03\(^{10}\) was downloaded from ftp://ftp.flybase.net/genomes/dmel/dmel_r5.57_FB2014_03/fasta/dmel-all-gene-r5.57.fasta.gz. The gene sequences were aligned to the assembled contigs with the commands:

```
nucmer --maxmatch asm.all.fasta dmel-all-gene-r5.57.fasta
delta-filter -q out.delta > out.qdelta
show-coords -lrcTH out.qdelta > out.qcoords
```

Genes contained within a single contig were counted as:
```
cat out.qcoords | awk '{if ($7 > 0 && $11 == 100) { print $NF} }' |wc -l
```
Genes contained within a single contig at >99 % identity:
```
cat out.qcoords | awk '{if ($7 > 99 && $11 == 100) { print $NF} }' |wc -l
```
Genes perfectly reconstructed:
```
cat out.qcoords | awk '{if ($7 >= 100 && $11 == 100) { print $NF} }' |wc -l
```

To analyze the TE repeat families, the analysis from McCoy *et. al.*\(^{11}\) was reproduced. The assembly features were downloaded from ftp://ftp.flybase.net/genomes/dmel/dmel_r5.57_FB2014_03/gff/dmel-all-r5.57.gff.gz. Assembled features matching “FlyBase transposable_element” were extracted and converted to bed, yielding 5,433 TE features. The assembled_feature_pipeline.sh was updated to speed up MUMmer computation by changing:
```
nucmer --maxmatch --prefix $WORKDIR/out $REFERENCE $ASSEMBLY
```
to:
```
nucmer -mumref -l 100 -c 1000 --prefix $WORKDIR/out $REFERENCE $ASSEMBLY
```
The pipeline was then executed as
```
assembled_feature_pipeline.sh -a asm.all.fasta -r reference.fasta -f dmel-all-trans.bed
```
The final output results/FINAL.REPORT was used to identify TEs with >= 100% Pct_length and corresponding Pct_ident. To identify repeat families roo and juan, TEs with name matching roo\{} and Juan, respectively were extracted from the final report.

The PBcR-BLASR and PBcR-MHAP assemblies (both before and after Quiver polishing) were evaluated using the above metrics.

The pre-quiver MHAP assembly contains 16,604 genes in a single alignment in a single contig (15,880 at > 99% identity and 7,299 at 100% identity). There are 5,201 (96%) TE elements in a single contig in a single alignment with the majority aligned perfectly to the reference (4,026). The assembly contains 132 of 138 copies in a single contig in a single
alignment of the highly-abundant roo TE family (37 with perfect alignment) and 7 of 11 copies of the juan family contained in a single perfect alignment to the reference. Post-quiver, the MHAP assembly contains 16,776 genes in a single contig in a single alignment (16,751 at > 99% identity, 14,824 at 100% identity). It contains 5,274 (97%) TE elements in a single contig in a single alignment (4,984 aligning perfectly to the reference). It has 134 copies of the roo TE family in a single alignment in a single contig (93 perfect alignment). For the juan family, 11 of 11 copies are contained in a single perfect alignment to the reference.

The post-quiver BLASR assembly contains 16,164 genes in a single alignment in a single contig (16,164 at > 99% identity, 12,591 at 100% identity). There are 5,158 TE elements in a single contig in a single alignment with the majority aligned perfectly to the reference (4,528). The assembly contains 128 of 138 copies in a single contig in a single alignment of the highly-abundant roo TE family (52 with perfect alignment) and 9 of 11 copies of the juan family contained in a single perfect alignment to the reference.
Supplementary Note 8: Telomere assembly analysis

The *S. cerevisiae* S288C other_features database was downloaded from http://downloads.yeastgenome.org/sequence/S288C_reference/other_features/other_features_genomic.fasta.gz. The features were aligned to the assembly using the commands:

```
nucmer --maxmatch asm.all.fasta features.fasta
show-coords -lrcTH out.delta |sort -nk12 |awk '{if ($7>85 && $11>50) print $0}'|grep TEL |sort -rnk8 > tels.coords
```

Finally, contigs were identified that contain telomeric features within 50Kbp of the contig ends.

The *D. melanogaster* repeat libraries from Koch *et al.*\(^{12}\) were downloaded and dmRepBase.fasta was used for all analysis. The repeats were mapped to the assembly with the commands:

```
nucmer --maxmatch asm.all.fasta dmRepBase.fasta
show-coords -lrcTH out.delta > out.coords
```

Any contigs with hits to *Het-A*, *TART*, and *Tahre* within 100Kbp of the contig ends were flagged as potentially containing telomeric sequences, 24 contigs passed the criteria. One contig contained *Het-A*, *TART*, and *Tahre* as well as *HetRp_DM*. 
Supplementary Note 9: Quantifying overlap detection accuracy

Quantifying the accuracy of an overlapping method is important, since it provides a more direct method for understanding performance. However, rather than exhaustively computing the sensitivity of an overlapper by checking if all possible overlaps were detected, we randomly sampled the overlap space that exists when sequences are mapped to their best position in the reference. This measures the ability of an overlapper to determine if two reads originated from the same genomic locus, rather than an overlapper’s ability to find “false” (repeat-induced) overlaps. In this case the estimated sensitivity can be expressed as the expected value, E[X], where X is the Bernoulli random variable with value 1 when the selected reference overlap is found in the overlapper’s output, and 0 otherwise. We can estimate E[X] by randomly subsampling X, such that

\[ E[X] \approx \bar{X} = \frac{1}{m} \sum_{i=1}^{m} X_i \]

for a large enough m. The 95% confidence intervals [-δ,+δ] of the estimates \( \bar{X} \) where derived using the Clopper-Pearson method. To determine if the reference mapping of two reads overlapped, all sequences were mapped using BLASR:

```
blasr <sequences.fasta> <ref.fasta> -minReadLength 200 -nproc 32 -bestn 10 -m 4 -out <ref.m4>
```

To compute PPV, a similar sampling strategy was employed. A random overlap was selected from the program output and evaluated. If detected based on the reference mapping, it was counted as a true positive. If not detected based on reference mapping, an alignment at >70% identity was confirmed via dynamic programming (Methods). Overlaps failing both of these tests were counted as false positives. If \( N \) = number of samples, the PPV is then computed as the \( TP/N \).

The sensitivity/specificity calculation is included in the MHAP distribution and can be run as:

```
java edu.umd.marbl.mhap.main.EstimateROC <mappingToReference.m4> <overlaps.m4 or overlaps.mhap> <fasta sequences> <min overlap> 10000 true false
```

**Sensitivity and specificity on all datasets**

Due to BLASR’s long runtime and DALIGNER’s instability and file system requirements, a maximum subset of 1Gbp (approximately 150K sequences) was randomly extracted from each of the five datasets. BLASR with \( k=12 \), \( \text{bestn}=1C \), \( 10C \), and \( 100C \) was run along with MHAP with default (\( k=16 \), \( h=512 \), \( \text{min}=3 \)), and sensitive (\( k=16 \), \( h=1256 \), \( \text{min}=3 \)). Sensitivity and specificity was evaluated as above. For BLASR the \( \text{bestn} \) corresponded to 100, 1000 for \( E. \ coli \); 85, 850, for \( S. \ cerevisiae \); 8, 80 for \( A. \ thaliana \) and \( D. \ melanogaster \); and 2, 20 for human.
Supplementary Note 10: Consensus algorithm for correcting noisy, long reads

Corrected consensus sequences can be constructed by: (i) building a multiple sequence alignment (MSA) and (ii) deciding the correct base from the MSA columns\textsuperscript{14,15}. The errors in each individual read will have no or few support from other reads, while the correct bases will have many aligned bases from other reads. Encoding the MSA as partial order alignment\textsuperscript{16} with a directed acyclic graph (DAG) is also an effective way for generating the consensus sequences. A consistent base-level alignment and the right partial order are essential to efficiently catch missing and extra bases correctly in the initial template sequence. A “tagging” and “sorting” approach described here uses the same principle to construct a consensus sequence. However, with this tagging and sorting approach, one does not need to construct a multiple sequence alignment or an alignment graph explicitly. The partial order relationship is maintained naturally in the process for generating the consensus sequence.

The inputs of the “FalconSense” algorithm are (1) a template read $T$ and (2) a set of reads denoted as $R_1$ to $R_n$ that are used to correct the template read. The output of the algorithm is a consensus sequence based on $T$. The sketch of this simple algorithm is as follows (Supplementary Figure 13):

(1) For each read $R_i$, align it to the template $T$ with an alignment algorithm developed by Myers\textsuperscript{17}. Notice that there is no mismatch in the alignment output. Every alignment position is a single-base match, insertion, or deletion. Traditional mismatches become an insertion followed by a deletion.

(2) For each base $b$ in $R_i$, according to the alignment, we assign a tuple of number $(p, d)$. If $b$ is aligned as a match to $j$th base in $T$, $p$ is assigned as $j$ and $d$ is assigned as 0. The tuple $(p=j, d=0)$ is inserted into a aligned tag list $B$. If there is no base aligned to $j$th base in $T$, a tuple $(j, 0)$ is added into the aligned base list $B$. If $b$ is an insertion relative to $T$, and the previous aligned base is on $j$th base in $T$, then $p=j$ and $d$ is the number of bases from the previous aligned position. The tag $(j, d \neq 0)$ is added to $B$. See Supplementary Figure 13 for an example.

(3) After all reads are aligned to $T$. We sort the elements in the list $B$ by their numerical order for $p$ and $d$ followed by alphabetical order for $b$. To generate the consensus sequence, we count the number of for the same $(p, d, b)$. If the number is larger then half of the number of reads that are aligned across position $j$ in $T$ (= the number of element with $p=j$, $d=0$), we append the base $b$ to the consensus sequence.

A heuristic is developed to handle templates with long stretches of errors or chimeric fragments formed by random ligation during sample preparation. In these cases, one might want to break the template at chimeric junctions or low quality regions to avoid propagating errors for downstream assembly steps. A sliding window counting the number of matches is used to decide if a region of the alignment in $R_i$ should be used. A minimum
number of matches should be found with in the window. If the number is below a threshold then the alignment in the window will not be used. At each position in the template, if the number of sequences available for correction falls below a pre-specified threshold, the template is broken. By default, the consensus sequence will only be generated for the longest contiguous region.
Supplementary Note 11: Assembling corrected reads

All assemblies ran on a cluster composed of AMD 6136 2.4GHz CPUs. Overlap jobs were configured to use no more than 32GB of RAM, the maximum available per node. Total CPU hours were obtained using SGE’s qacct utility, with the time computed as the sum of user and UTIME and STIME divided by 3,600. Data for E. coli, S. cerevisiae, and D. melanogaster were filtered with default parameters in SMRTportal to generate input. Due to an import bug in SMRTportal, A. thaliana data was filtered using length=500, quality=0.8. Human data was downloaded in fastq format post-filtering from the PacBio DevNet website. All Quiver polishing was run via the SMRTportal interface, except for CHM1, which was run manually to manage splitting and merging of multiple intermediary files.

All PBCr-MHAP assemblies, except human, used the same Celera Assembler spec file, reproduced below:

```
merSize=16
mhap=-k 16 -num-hashes 512 -num-min-matches 3 -threshold 0.04 -subsequence-size 100000
useGrid=1
scriptOnGrid=1

ovlMemory=32
ovlStoreMemory=32000
threads=32
ovlConcurrency=1
cnsConcurrency=8
merylThreads=32
merylMemory=32000
ovlRefBlockSize=20000
frgCorrThreads = 16
frgCorrBatchSize = 100000
ovlCorrBatchSize = 100000

sgescript = -pe threads 1
sgeconsensus = -pe threads 8
sgescript = -pe threads 15 -l mem=2GB
sgecorrection = -pe threads 15 -l mem=2GB
sgefragmentcorrection = -pe threads 16 -l mem=2GB
sg eovertapcorrection = -pe threads 1 -l mem=16GB
```

The pipeline was run as:

```
PBCR -l <genomeName> -s pacbio.spec -pbCNS -fastq <fastq input sequences>
genomeSize=<approximate genome size> sgeName=<genome name> "sge=-A <genome Name>
```

Where <genomeName> was set to ecoli, yeast, dmel, athal, and human, respectively and genome size was set to 4,650,000; 12,000,000; 130,000,000; 130,000,000; 3,000,000,000, respectively.
For human, we observed that the data was lower identity (average 82.98%) than other datasets (e.g. *D. melanogaster* average = 85.48%) and increasing MHAP sensitivity improved assembly performance. Due to this and the lower coverage, the spec file was modified to be (changes to the original spec file in bold):

```
merSize=14
mhap=\-k 14 \-num\-hashes 1024 \-num\-min\-matches 3 \-threshold 0.04 \-subsequence\-size 100000
```

```
useGrid=1
scriptOnGrid=1
```

```
ovlMemory=32
ovlStoreMemory=32000
threads=32
ovlConcurrency=1
cnsConcurrency=8
merylThreads=32
merylMemory=32000
ovlRefBlockSize=20000
frgCorrThreads = 16
frgCorrBatchSize = 100000
ovlCorrBatchSize = 100000
```

```
sgeScript = \-pe threads 1
gseConsensus = \-pe threads 8
gseOverlap = \-pe threads 15 \-l mem=2GB
gseCorrection = \-pe threads 15 \-l mem=2GB
gseFragmentCorrection = \-pe threads 16 \-l mem=2GB
gseOverlapCorrection = \-pe threads 1 \-l mem=16GB
```

```
# relax overlap parameters
asmOvlErrorRate=0.10
asmUtgErrorRate=0.07
asmCgwErrorRate=0.10
asmCnsErrorRate=0.10
utgGraphErrorRate=0.07
utgGraphErrorLimit=3.25
utgMergeErrorRate=0.0825
utgMergeErrorLimit=5.25
asmOBT=0
```

and the pipeline run as:

```
PBCR \-l <genomeName> \-s pacbio.spec \-fastq <fastq input sequences>
genomeSize=<approximate genome size> sgeName=<genome name> \"sge=\-A <genome Name>"
```

Pre-compiled source code and datasets used for this publication:

http://www.cbcb.umd.edu/software/PBcR/MHAP

Celera Assembler source code and documentation:

http://wgs-assembler.sourceforge.net

PBCR-MHAP correction pipeline documentation:

References