Title
POLISHER: A tool for using ultra short reads in genome sequence improvement

Authors
Foster, Brian
LaButti, Kurt
Trong, Stephan
et al.

Publication Date
2009-08-07

POLISHER: a tool for using ultra short reads in genome sequence improvement

Brian Foster[1], Kurt LaButti[1], Stephan Trong[2], Cliff Han[3], Tom Brettin[3], and Alla Lapidus[1]

[1]Lawrence Berkeley National Laboratory

[2] Lawrence Livermore National Laboratory

[3] Los Alamos National Laboratory

DOE JOINT GENOME INSTITUTE
US DEPARTMENT OF ENERGY
OFFICE OF SCIENCE

STANDARD SEQUENCE IMPROVEMENT STRATEGIES

The current strategy for sequence improvement consists of repeat resolution, gap closure, and polishing. Polishing sequence in a 454 assembly seeks to resolve base calling errors introduced during the sequencing or assembly. Insertions and deletions typically occur in homo polymer base runs that the 454 platform has difficulties resolving. It is these errors In particular that are of interest due to potential introductions of frame shifts that impede correct annotation and are ideal candidates for the polisher application. It is the goal of the polisher to resolve any and all substitutions, insertions, and deletions with little manual intervention. It should also be emphasized that because the polisher relies only on illumina sequence and a reference sequence/acefile, the polishing process will suggest base corrections regardless of the source or sequencing platform of the assembly.

Our traditional sequence improvement process utilizing 454/Sanger hybrid assemblies was designed to bring the quality of the assembly up to a predefined standard:

•all bases >= Q30
•2x coverage
•<5% 454 only

This was historically accomplished through a cyclical process:

1. Substandard regions of the consensus tagged in ace (Figure 1)
   - Low quality consensus
   - <Q30/Single subclone
   - 454 only
2. Automated oligo/plasmid template picking and ordering
   - *Attempts* to design oligos around clustered tags
3. Sequencing
4. Data incorporation
   - Requires manual assessment and intervention
   - Not all are solved
5. Repeat process
   - Expect 1/3 of the reactions to fail
   - Typically need ~ 4 rounds

SEQUENCE IMPROVEMENT WITH THE POLISHER

In order to reduce cost, time, and increase capacity all while upholding our high quality sequence standard, we developed a tool that employs Illumina read data to polish substandard regions as well as fix consensus errors in our projects. The polisher tool works in several phases: filter, alignment, analysis, and polishing.

Filter

The Illumina reads are first filtered to remove low quality or low complexity reads. The reads must meet the default parameters for inclusion:

- average quality of greater than 15 across entire read.
- no poly base runs greater than 15 bases.
- 80% of the read must not be composedof a single base.

Align

The quality filtered and aligned to a lookup table of the assembly fasta sequence using Arachne's MakeLookupTable and QueryLookupTable with the following options:

MO=10 K=12 SMITH_WAT=True MAX_ERROR_PERCENT=10 WE=10 MC=0.01

Since we are aligning to unpolished draft-like fasta we found QueryLookupTable to be the most suitable aligner at the time because of its speed and ability to align reads with a large amount of discrepancies. An alignment for each flow cell is sent off in parallel and simultaneously parsed for best hit based on percent identity. Equal scores are placed at random.

Analyze

The alignment data is first screened to identify alignment stacks where identical alignments make up more than 10% of the alignment coverage starting at a particular base. The alignment coverage is reduced to 2x at these positions to prevent undue influence of artificial sequence duplication artifacts that may arise from the sequencing process.

The best hit information is then parsed for Illumina coverage per consensus base. Every discrepancy (substitution, deletion, insertion) is also tracked and this information is stored in a data structure (Figure 2.2). It then traverses the data structure and refines the information by calculating the fraction that agrees with the consensus base, and the largest fraction that disagrees. While traversing the refined data structure it looks for areas where the Illumina data suggests something is positively wrong and needs editing. These areas are kept in a list called AcefileEdits.list. An invitation to this list requires the following thresholds;

>= 10X Illumina coverage
70% of the Illumina coverage (majority discrepancy) disagrees with the consensus base

Polish

Substitutions identified in the previous step can be fixed via modification of the acefile consensus base and the quality is bumped up to Q99 so they will be ignored by subsequent polishing. Our normal substandard region identification tool (tagAceforPolishing) can then run to generate a list of polishing tags (polishingTags.list). These polishing tags specify the location and type (LowQualConsensus, SingleSubclone, 454Only) of every substandard region in the acefile. Each base of every polishing tag is then interrogated to see if the Illumina data suggests it is correct or not with the following thresholds;

>= 10X Illumina coverage
70% of the Illumina coverage agrees with the consensus base

If any base in a polishing tag meets the above criteria then the polishing tag over that base is changed to solexaSupported. If the information for the base does not meet the criteria then the original tag remains. The acePolisher tool then automatically designs primers targeting these areas for further sequencing. The resulting modified tags are then added to the acefile and deletions suggested in the AcefileEdits.list are fixed via modification of the acefile (Figure 1).

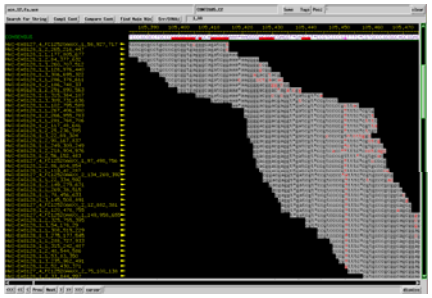


Figure 2. Mosaik visualization

Consed view of Illumina read data aligned to the draft consensus from Figure 1. Acefile and alignment were created with Mosaik for visual verification of the Polisher's performance. Note the agreement in read data below the polishing tags as well as under the extra T in the consensus (pink tag).

ABSTRACT

Polishing is one of the major steps of genome sequence improvement at the JGI (Joint Genome Institute). Along with repeat resolution and gap closure, it is required to produce a fully sequenced high quality genome. Polishing consists of consensus error correction and quality improvement such that the resulting consensus meets a pre-defined standard. This has traditionally been done through targeted Sanger clone based sequencing, which is both time consuming and resource intensive. Our process, conducted using Illumina data produced by the JGI, demonstrates that aligning ultra short reads against unpolished contigs helps to correct a significant amount of consensus errors and greatly reduce the amount of quality improvement necessary to produce a high quality assembly. Our tool named the "Polisher" was developed in order to automate this process. It facilitates polishing and error correction of a subject assembly (acefile), typically a draft or closed assembly, using Illumina read data.

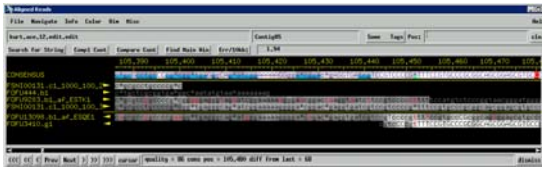
The Polisher analyzes Illumina alignment data to indicate which consensus errors to correct and supports correctly called bases that would normally be targeted for polishing due to their below standard quality. The tool then utilizes this information to automatically modify the consensus. The Illumina read data in Fasta format is aligned to the subject sequence and simultaneously parsed for the best hit based on percent identity. The best hit alignment results are then used to determine coverage and discrepancy information per base in the subject. A list of errors is generated where there is overwhelming evidence that the consensus base is wrong and needs to be changed. Such areas are determined as substitutions, deletions, or insertions and can be automatically corrected in the acefile. In addition to the previously described error correction, other areas of the genome that would normally be targeted for polishing are inspected. These areas currently represent low quality, single subclone, and 454 only regions and exist as tags in the acefile. If there is overwhelming evidence that a particular base targeted for traditional polishing is correct, then that base is termed Supported and retagged. If there is not enough evidence for support, then the original polishing tag remains for traditional manual polishing.

Figure 1. Draft consensus: pre and post polishing

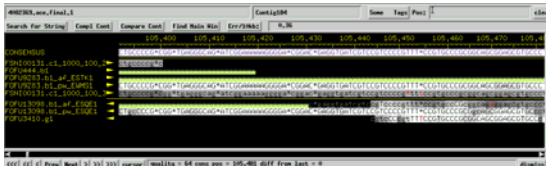
(a) Portion of a draft assembly project as viewed in the assembly editor Consed. The consensus is composed of a single 454 shred and several low quality sanger reads. The consensus regions that are <Q30 have been tagged with LowQualityConsensus tags (red) in the first panel. **(b)** The second panel shows the same region after polishing with the Polisher. The Illumina data aligned to this region suggests with a high amount of certainty that the consensus bases that were tagged for polishing were correct and therefore retagged as solexaSupported (blue). Also note the green solexaCorrected tag where the polisher deleted a base. **(c)** The third panel shows the same region finished using traditional methods (sanger sequencing off of plasmid templates). It required two rounds of polishing as well as manual editing.

a.

b.



c.



PERFORMANCE

Which thresholds work the best?

In order to determine which configurable thresholds perform the best the Polisher was tested on draft assemblies from 8 previously finished genomes of varying complexity and GC content using a variety of coverage and correction thresholds (Figure 3). Each project was assembled with Newbler and polished using the Polisher with default parameters to produce the alignment data files. The analysis portion of the code was then run on each project using a variety of conditions (required aligned Illumina coverage from 5 to 45x in increments of 5, disagreement threshold from 20% to 90% in increments of 10) resulting in 72 different conditions for each project. The polish portion of the code was run next to polish the draft acefile using each of the conditions results. The corrections made in the polished acefile were then checked against the finished project for correctness by pulling a chunk of sequence (25bp on either side of a correction) for each correction and aligning it using BLAT to the finished project. The results of the BLAT were parsed to determine whether the chunk of sequence containing the correction was correct or not. Substitutions and Indels were tracked separately and are referred to as correction type. Errors of each type in the polished acefile were also determined using Arachne's TruePoly. The statistics for each condition for each type were then averaged across the 8 projects and plotted.

How well can I expect it to work?

The performance of a two class classification system, with information about actual and predicted classifications, is commonly evaluated using a confusion matrix. The averaged results generated from the above thresholds experiments were used to formulate statistics using a confusion matrix with the following entries for substitutions and indels separately:

a is the number of **correct** predictions that an instance is **negative**.
b is the number of **incorrect** predictions that an instance is **positive**.
c is the number of **incorrect** of predictions that an instance **negative**.
d is the number of **correct** predictions that an instance is **positive**.

		Predicted	
		-	+
Actual	-	**a**	**b**
	+	**c**	**d**

RESULTS

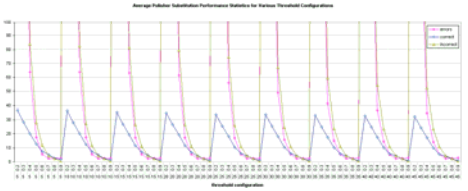
Varying the threshold configurations for the polisher resulted in noticeable trends in the number of correct corrections (bases that should have been changed and were), incorrect corrections (bases that were corrected but shouldn't have been), and overall errors in the polished projects (figure 3). Surprisingly, the amount of required aligned Illumina coverage did not seem to significantly affect the performance of the polisher with the exception of the errors introduced at extremely low correction thresholds (fraction of aligned Illumina coverage needed to invoke a correction) . As one would expect, however, the correction threshold had a significant impact on the polishers performance. Too low a threshold results in the greatest amount of correct corrections, however the amount of errors introduced with such a threshold is tremendous, and gets worse with low coverage thresholds as mentioned above. Too high a threshold introduces the least amount of errors but also the least amount of correct corrections, resulting in many errors remaining in the project.

The plots of the confusion matrix calculations also resulted in the same noticeable trends as plotting the correction statistics (Figure 4). Since the vast majority of errors in 454 only and hybrid assemblies are mono nucleotide runs, statistics on indels only were explored. Much more data on substitutions is necessary to accurately determine how well the polisher performs on that particular error type. The results of the confusion matrix on indels suggest that the polisher is performing with high accuracy, and precision. Of all the indel errors in the 8 projects, on average it identified 82% of them, correctly corrected 91% of them, missed 18% of them, and introduced an extremely low amount of errors. This is an acceptable performance for automated error correction and saves a large amount of finishing time and cost.

Figure 3. Threshold configuration results

Plots detailing a) substitution and b) indel correction statistics for polished draft assemblies using a variety of conditions. The ideal threshold configuration results in the most correct, least incorrect, and overall least amount of errors after polishing.

a.



b.

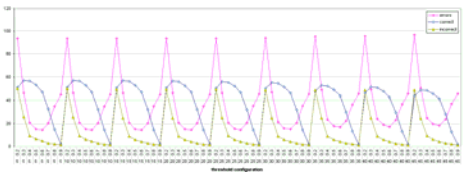
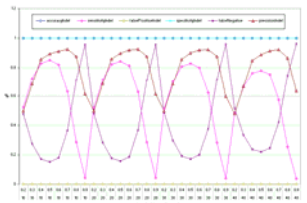


Figure 4. Confusion matrix results

Overall average confusion matrix performance statistics for a variety of threshold configurations for indels a). Best threshold configuration confusion matrix overall performance statistics for indels b).

a.



b.

	Indel 10x 60% threshold
Accuracy	0.999993
True Positive (sensitivity)	0.819634
False Positive	0.000001
True Negative (specificity)	0.999998
False Negative	0.180365
Precision	0.911176

CONCLUSIONS

The results from the above experiments suggest a 50-60% correction threshold for indels, and a 60-70% correction threshold for substitutions will result in the best overall performance of the polisher. These thresholds should result in the most corrections, least amount of introduced errors, and least amount of overall errors in a project. The experiments also suggest that coverage does not significantly impact the polishers performance at reasonable correction thresholds. Because indels are the most common errors in 454 only and hybrid assemblies the focus of the confusion matrix performance calculations were based on indels. More data for substitutions is needed to make any conclusions on the performance of the polisher on those error types. With an ideal threshold configuration the polisher identifies and fixes the majority of errors in the project, while introducing extremely few mistakes. Because each genome is inspected for completion at the end any errors introduced or missed by the polisher should be identified and fixed. In conclusion, the polisher is an automated tool that saves finishing time and cost. In conjunction with this analysis and others this translates roughly into an estimated 98.55% (average 81%) savings on traditional polishing reactions and a ~25% average savings in finishing per genome.

LLNL-POST-416091

This work was performed under the auspices of the US Department of Energy's Office of Science, Biological and Environmental Research Program, and by the University of California, Lawrence Berkeley National Laboratory under contract No. DE-AC02-05CH11231, Lawrence Livermore National Laboratory under Contract No. DE-AC52-07NA27344, and Los Alamos National Laboratory under contract No. DE-AC02-06NA25396.