EAST: Expression Fragment Assembly from Spanning Trees

Yuan Zhang[1] , Dhananjai M Rao[1] , Mufit Ozden[1] , James C Moler[1] , Chun Liang*[1,2] and John E. Karro*[1,3]

[1]Department of Computer Science and Software Engineering, Miami University, Oxford OHIO, USA
[2]Department of Botanty, Miami University, Oxford OHIO, USA
[3]Department of Microbiology, Miami University, Oxford OHIO, USA

Email: Yuan Zhang - zhangy9@muohio.edu; Dhananjai Rao - raodm@muohio.edu; Mufit Ozden - ozdenm@muohio.edu; James Moler - molerjc@muohio.edu; Chun Liang*- liangc@muohio.edu; John E. Karro*- karroje@muohio.edu;

*Corresponding author

Abstract

Introduction

The *de novo* assembly of RNA transcripts is an important and computationally challenging problem, made more difficult by the advent of Next Generation Sequencing technologies such as 454 or Illumina [1, 2]. Given the completion of a large-scale sequencing run of multiple mature transcripts of a given organism, the investigator is left with a large set of sequence fragments, but no ordering or even transcript association information; these fragments must be clustered by transcript then assembled before the sequencing output can be truly useful. If the genome has not been sequenced, this must be done *de novo* – using only the sequence information extrapolated from the sequences themselves. Even when the genomic sequence is known, there are a number of advantages to ignoring the known sequence information and starting from scratch. Using an existing sequence as a template for sequencing in population studies has the potential to introduce bias. And when looking specifically for genes, a known sequence may be of limited use: the target transcripts may be undiscovered, and the splicing of introns before sequencing can complicate the approach of mapping fragments to the genomic sequence. In short, there is significant value in tools that

can reassemble fragment data into the original assembly without requiring access to genomic information. The problem of *de novo* transcript assembly has been addressed a number of times in the literature, with initial approaches concentrating on the longer reads produced by traditional Sanger Sequencing techniques, and newer approaches looking at produces of Next Generation Sequencing (NGS) technologies. A list of the tools most successfully addressing these problems include Cap3, TGICL, Velvet, and Mira3, each excelling within their own context [3–6]. Cap3 and TGICL use alignment-based strategies that produce high quality results when applied to the output of Sanger sequencing technologies. Velvet, based on the modeling of the problem with a *de bruijn* graphs, was designed to handle short read sequences. Mira3 is a multi-pass DNA assembler based on an overlap-layout-consensus strategy. [STILL NEED TO SAY SOMETHING ABOUT MIRA AND HYBRID SETS.]

In this paper we present EAST, a tool for the assembly of both Sanger, NGS and hybrid data with the capacity to incorporate quality scores when available. EAST is based on a novel minimum spanning tree (MST) approach to compute assembly. The tool works in tandem with the PEACE clustering tool [2]: PEACE clusters fragments by transcript and produces the initial MSTs that will serve as the basis for the EAST algorithm. In the following we present a comparison of EAST against those of Cap3, TGICL, Velvet, and Mira3on both Sanger and NGS sequences, showing that PEACE+EAST generates the best quality results in the fastest runtime for Sanger sequences, and that while certain tools out-preform PEACE+EAST under specific conditions, it is the only tool able to produce constant quality results for Sanger, 454, Illumina, *and* hybrid data sets.

Results and Discussion

In assessing result quality, we must view the PEACE clustering tool [2] and EAST as a combined process. While most assembly tools integrate these two jobs, EAST assumes clustering has been preformed and requires the PEACE MST output. While other cluster tools could be substituted in to preform the actual clustering (e.g. WCDor EasyCluster [7, 8]), only PEACE provides the requires MST output. Hence for purposes of the following section, we consider a single tool PEACE+EAST, and assess the quality of these results.

In order to assess the quality of the (PEACE +) EAST results, we compared against four prominent assembly tools: Cap3, TGICL, Mira3, and Velvet [3–6]. Tools were tested on data from three sequencing technologies (Sanger, 454 and Illumina), using both simulated output obtained from the ESTSim and MetaSim tools [9, 10] and benchmark sets obtained from GenBank [11].

Measuring Quality

Our primary measure of quality was the **normalized a-score**, based on the *assembly score* (a-score) used in *Liang et al.* [12]. For an assembly of some original sequence, the a-score is defined as:

$$(2\times\text{sequence length}) - (5\times\text{number of mismatches}) - (15\times\text{number of indels})$$

with a "perfect" score equal to twice the sequence length. Our normalized a-score modifies this definition as:

$$\frac{(2\times\text{assembly length}) - (5\times\text{number of mismatches}) - (15\times\text{number of indels})}{(2\times\text{sequence length})}$$

with a perfect score equal to 1. This normalization allows for the comparison of assembly quality between sequences of different lengths. Further, using this score we can more accurately quantify the quality of a partial reconstruction: if we are able to reconstruct only a portion of a fragment, the normalized a-score will take full consideration of the quality of the reconstruction, but then bound the score by the relative size of the reconstruction. (e.g. Perfectly reconstruction one fourth of the original sequence will lead to an a score of 0.25.) In addition to a-score we also look at: the number of contigs produced by a tool (which, in the perfect solution, will be exactly equal to the number of transcripts represented by the data set), normalized by the number of transcripts; the number of singleton ESTs in the solution (i.e. ESTs unassigned to any contig and thus, most cases, effectively thrown away), and the runtime of the tool.

Simulation Results

Using simulated fragment sets allows us to both compare assemblies to a known correct solution and control parameters in their model (e.g. base-call error rate) to test the effect of variation. All sets were generated from a set of 100 zebra fish transcripts obtained from *Hazelhurst et al.* [7]. For simulated Sanger sequences we used the ESTSim tool, using its default model of EST generation when not otherwise specified. For simulated 454 and Illumina sets we used the MetaSim tool [10], again using their default model to simulate the errors associated with those technologies.

Sanger Sequencing: We first compare the assembly quality of the tools on Sanger Sequencing, using EAST, Cap3, and TGICL (the tools designed to work with Sanger sequences), looking at the a-score as a function of error rate for a range of coverage depths. For each parameter combination we applied each tool to 30 independently randomly generated simulated sets, derived from 15 zebra fish genes, reporting the average normalized a-score. In Figure 1 we look at normalized a-score as a function of error rate, fixing coverage at 50, 30 and 10 and varying base-read error rate from 0% to 6% for each of the three tools

designed for Sanger Sequencing. While all three tools preform well when applied to error free (i.e. impossible) sequencing technology, we find EAST to be considerably more robust to base-call errors than the other tools. Taking the mid-range coverage value of 30 as an example, at a 1% error rate we see EAST, Cap3, and TGICL all achieving an essentially perfect normalize a-score . At a 3% error rate EAST maintains its almost perfect normalized a-score, showing a 2% improvement over TGICL and a 21% improvement over Cap3. At a 4% error rate EAST still maintains its near-perfect score (showing a 54% and 85% improvement over Cap3 and TGICL respectively), and even at a 6% error rate is still achieving an normalized a-score of 0.95 (a 110% improvement over the next closest tool, TGICL), meaning that at most 5% of the bases were left out of their contigs or were incorrectly reconstructed.

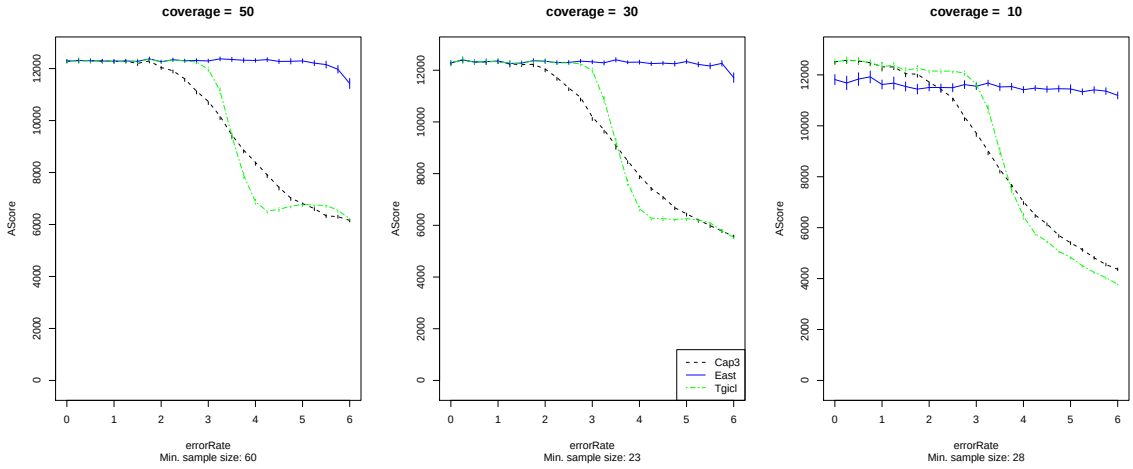

Figure 1: Average normalized a-score as a function of base error rate for simulated Sanger sequencing. EAST=blue, Cap3=Black, TGICL=green, Velvet=red, Mira3=purple.

Another useful metric is the number of contigs and number of singleton fragments in the final result. In Figure 2 is a plot of the average number of contigs per gene sequence (left column), and the number of singletons (right column). For the first we observe that all tools have comparable results for lower error rates, but only EAST is able to maintain the low score with increasing error rates – generating an perfect score in almost every case. In minimizing the number of singletons, we find that EAST does a better job than Cap3 for error rates larger than 2%, while TGICLdoes a better job that EAST for error rates larger than 4% to 5%.

Next Generation Sequences: In Figure 3 we compare the normalized a-score for simulated 454 and Illumina outputs (with sequence lengths averaging 250 and 62 bp respectively). For these tests we varied coverage levels but otherwise took the default Metasim model. On 454 data we see significant improvement

in results quality in EAST for all levels of coverage. On Illumina data we see that EAST is competitive with Velvet at higher coverages, though is unable to match that tool when coverage is sparse.

Hybrid: In Figure 4 we look at the quality of results derived from applying each tool to a hybrid data set of Sanger (40%), 454 (40%) and Illumina (20%) sequences, again showing normalized a-score as a function of base call error rate in the Sanger portion. Here we observe EASTmatching or outperforming Cap3 and Mira3 at moderate to high coverages. At a coverage level of 50, EAST maintains a normalized a-score hovering around 0.85 at all error levels (showing a 54% improvement over Mira3 at a 3% base-call error rate). EAST actually improves its performance at a coverage level of 30 (hovering around a normalized a-score of 0.95, showing a 142% improvement of Mira at the 3% base-call error rate), an oddity we attribute to the decreased absolute number of NGS sequences with the smaller coverage. At a coverage level of 10 EAST's normalized a-score is around 0.37 – doing considerably worse that Mira3 at low error rates (e.g. 0.94 v. 0.40 for error-free bases), but showing a 52% improvement over Mira3 at the 3% error rate mark and in general showing much more robustness to error. Remarkably, it is only in this situation that Cap3 is competitive, with that tool achieving a normalized a-score of 0.62 (v. EAST's 0.37) for the 3% base call error rate, but loosing out to EASTat greater than a 4.5% base call error rate – a result that we attribute to EAST's problems with the 62 bp Illumina sequences (see below).

Tool Runtime: As EAST assembly requires the PEACE cluster (as opposed to the competing tools that do both jobs), for EAST runtime we actually include the PEACE runtime. In Figures 5 we see the average runtime for each of the experiments reflected in Figures Figures 1-4 for a coverage of 30, while in Figure 6 we look at the average runtime for each tool when applied to simulated data sets generated from a single zebra fish gene at varying levels of coverage – effectively plotting the runtime as a function of fragment set size. For Sanger sequences, we see marked improvement of (PEACE+)EAST over all other tools for almost all coverage levels: for the smallest set (coverage = 10) EAST has 1.9 speedup over Cap3 and a 3.9 speedup over TGICL, while at a more moderate size (coverage=50) we see speedups of 4.7 and 8.4. This returns back down to 3.0 and 5.2 at the largest set tested (coverage = 100), as can be seen in the bump around a coverage of 85 in Figure 6. We are unable to superficially explain this bump, but have noticed the same effect in data sets derived from other genes as well.

EAST is less competitive on shorter sequence (an effect of the d^2 bias of our algorithm, as discussed below), reducing its runtime improvement for the 250 bp 454 sequences, severely reducing it for the 62 bp Illumina sequences, and also affecting the hybrid data sets (also due to the presence of the Illumina sequence). We argue that the high quality of our results still makes EAST an appealing option for 454 and hybrid data

(while acknowledging that cannot complete with Velvet on pure Illumina data), and also observe that while it has not been implemented, the basic EASTâlgorithm should easily admit paralilization – thus increasing the practical limit on data set size.

Conclusions

In this manuscript we have described the EAST algorithm and run a comparison of the PEACE+EASTcombination against the best preforming tools in the literature. When compared against the fine benchmark tools on normalized a-score and runtime metrics, the PEACE+EAST combination of tools prove to be both the best tool for Sanger sequencing, the best tool overall, and by far the most robust against sequencing error. While certain tools may do better than it in specific circumstances, there is no tool that does better than it on multiple sequencing technology platforms. While its speed is slower when applied to short sequence fragments, the quality of its results more then compensate.

As observed in the results section, when applying EAST, Cap3, and TGICL to Sanger sequenced transcripts with normal to high error rates, EAST consistently out preforms all other tools in terms of the normalized a-score metric when applied to transcript fragments resulting from Sanger sequencing (Figures 1, 5, and 6). We also find that the number of contigs produced is almost exactly equal to the number of transcripts represented in the set, regardless of error rate, while the number of singleton ESTs remains low up to about a 5% error rate (Figures 2). The high normalized a-score indicates that EAST is both reconstructing larger portions of each transcript and doing so more faithfully, while the low number of contigs means that we are getting complete transcripts. Finally, runtimes for EAST are considerably faster than other tools for larger sets (Figures 5 and 6, with (PEACE+)EAST running a minimum of two to three times as fast as the other tools, with speedups over Cap3 (the faster tool) ranging from 1.9 to 6.0, depending on the coverage level. In short, when used for Sanger Sequences EAST is a faster tool that produces the high quality reconstructions versus comparing tools and is significantly more robust to error than other tools.

When involving shorter sequences, the runtime of EAST takes a significant hit. As explained in our methods section, EAST is dependent on the d^2 sequence distance function [13], whose parameters must be adjusted when applied to smaller sequences to preserve quality, at the cost of runtime. We describe below our *adoptive* d^2 algorithm that allows us to target more specific parameters on a case-by-case basis, but large numbers of short sequences will still lead to slower runtimes. In our tests our simulated 454 sequences averaged 250 bp in length, while our Illumina sequences were uniformly 62 bp, accounting for the slightly reduced improvement of EAST, as compared to other tool runtimes, in the 454 tests and the significant

reduction for the Illumina and Hybrid test sets containing the short Illumina sequences.

Despite the slower runtime, it can still be argued that the EAST is competitive, given the quality of its output, excepting when compared to Velvet on Illumina data. For 454 sequences, no tool comes even close to matching the EAST results. EASTis able to match Velvetfor high-coverage Illumina data, though loses out at lower coverages (and is not even remotely competitive in terms of runtime). Looking at hybrid sets, which the Mira3 tool is specifically geared to address, we again find EAST to produce significantly superior results at moderate to high levels of coverage. At low coverage levels EAST does suffer (another effect of the short Illumina sequences), but its robustness then leads to better results when the Sanger sequence portion of the set is subject to elevated error rates.

Methods

The algorithm underlying EAST is based on three components: minimum spanning trees (MSTs) [14], the d^2 sequence distance measure [13], and standard sequence alignment [15, 16]. While we would ideally be using sequence alignment to identify overlapping ends and reassembling on that basis, the quadratic run time of alignment algorithms coupled with the large number of sequences makes such an approach infeasible. To avoid excessive applications of such algorithms, we make use MSTs to provide a *guide* through the EST set, dictating which pairs we aligned. A modified version of the d^2 measure provides the edge weights for the underlying graph, allowing us to define the MST and also providing some speedup to the alignment algorithm.

EAST is technically an assembly tool, and require clustering be done as a pre-processing step, as well as requiring that d^2-based MSTs be provided for each cluster. Conveniently, the PEACE clustering tool [2] provides exactly this information. While PEACE an EAST are technically two tools, EAST was designed specifically for PEACE output. For purposes of this paper, it makes sense to consider them as one. In the following we briefly outline the PEACE algorithm for clustering (as is relevant to EAST), and then discuss the steps taken by EAST for assembly of the PEACE clusters.

Clustering

Given a set of fragments from two or more transcripts, we must first cluster the fragments by transcript association – leaving us with a single cluster of fragments for each transcript. Details of the PEACE algorithm can be found in *Rao et al.* [2], but the general idea is as follows: we model the problem as a weighted, undirected graph, representing each fragment with a node and assigning edge weights based on a

d^2 comparison of the incident fragments (explained in more detail below). By employing the u/v and t/v filtering heuristics of *Hazelhurst et al.* [7], we can quickly dismiss most of the edges as irrelevant (i.e. connecting nodes that belong in different clusters), then use Prim's algorithm to compute an MST [14]. We then remove all edges exceeding a specified threshold value, and take each component of the resulting forest as a cluster. As a result PEACE can pass to EAST a set of clusters, an MST for each cluster, and the orientation of each fragment relative to the cluster. Hence EAST has not only the clusters, but the PEACE MST and may quickly adjust the sequences to ensure all are on the same strand.

Overlap Detection

Given the PEACE generated MST for a cluster, the challenge is to identify and align the ends of overlapping fragments without taking the significant run time hit needed to align every pair. We accomplish this using two methods: we reduce the number of times an alignment is calculated aligning only those sequences adjacent in the MST, and we reduce the amount of time spent per alignment by aligning only those portions indicated by an application of a modified form of the d^2 sequence distance measure. The presence of an edge in the tree is a heuristic indication that the incident sequences overlap; were there not sufficient sequence similarity to indicate an overlap, the edge would have been assigned a high d^2 value and would have been removed by PEACE. To determine how the nodes overlap, we calculate a modified d^2 score as follows. The standard calculation of d^2, as defined by *Hide et al.* [13], requires looking at every pair of length w sub-strings (windows) from the two sequences, calculating a distance between these two windows based on the number of shared words (6-mers), and taking the score of the lowest-scoring window pairs as the d^2 distance. However, when looking for overlap of fragments s_1 and s_2, we are looking only for end-overlap. That is, we wish to determine of the left end of s_1 overlaps the right end of s_2, or the reverse (recalling that we know them to be on the same strand). Hence we need only consider the left-most and right-most windows of s_1 against each window of s_2, reducing the number of window comparisons from quadratic to linear in the size of s_2.

In Figure 7 we illustrate the use of modified d^2, using it to take two sequences (a), find the window on s_2 most resembling the right-most window of $s1$ (red section of (b)), extending to the end of s_2 (green section of (c)) and preforming a *global* alignment on the overlapping portion. We then compute the *overlap distance* ($1 - s/l$, where s is the alignment score and l is the length of the overlap segment), a value that is inversely proportional to the probability that this is a legitimate overlap.

We note that both PEACE and EAST make use of an *adaptive* d^2 strategy to handle hybrid data sets. The

d^2 measure (and our modification) is flexible enough to handle a range of sequence sizes, requiring only a modification of the parameters (i.e. window size and threshold values). However, it runs into problems when comparing sequences of significantly different sizes. Comparing a 62 base Illumina read against a 1000 base Sanger read requires the use of parameters appropriate to the shorter fragment – which are considerably less accurate when applied to the longer read. Our solution is to partition fragments by sizes (e.g. into groups of small, medium and large fragments), and compute distances only between segments in the name or neighboring groups. By skipping the comparisons between sequences of significantly different sizes we avoid the pitfalls of such comparisons, while still picking up the connections through transitivity.

Ordering and Reconstruction

Having defined overlap distance, ordering the fragments is straight-forward. We pick an arbitrary node on the cluster's MST and transverse the tree, using the overlap distance measure to check each node against its two neighbors. Those nodes with no identified left-overlap are checked against nodes further out in the tree, and any remaining with no identified left-overlap are designated as a contig left end.
Having identified the distance of each node from its two closest overlapping sequences, we now construct a *directed* graph, with each edge pointing from a sequence to its right neighbor and weighted with the overlap distance. Using this graph we calculate a new minimum spanning tree, the starting from the root of the tree (necessarily a left-end of the contig), we transverse the tree and assign the first base of each sequence a position relative to its parent. Such an assignment gives us an implicit ordering of the fragments, by scanning through them in order and aligning each to the next with the Smith-Waterman alignment we end up with a multiple alignment of the fragments that allows us to derive a consensus sequence.

Authors contributions

Text for this section ...

Acknowledgements

Text for this section ...

References

1. Nagaraj S, Deshpande N, Gasser R: **ESTExplorer: an expressed sequence tag (EST) assembly and annotation platform**. *Nucleic Acids Res* 2007, [http://nar.oxfordjournals.org/cgi/content/abstract/gkm378v1].

2. Rao DM, Moler JC, Ozden M, Zhang Y, Liang C, Karro JE: **PEACE: Parallel Environment for Assembly and Clustering of Gene Expression**. *Nucleic Acids Res* 2010, **38 Suppl**:W737–42, [http://nar.oxfordjournals.org/cgi/content/full/38/suppl_2/W737?view=long&pmid=20522511].

3. Huang X, Madan A: **CAP3: A DNA sequence assembly program**. *Genome Res* 1999, **9**(9):868–877.

4. Pertea G, Huang X, Liang F, Antonescu V: **... Indices clustering tools (TGICL): a software system for fast clustering of large EST datasets**. *Bioinformatics* 2003, [http://bioinformatics.oxfordjournals.org/cgi/content/abstract/19/5/651].

5. Chevreux B, Pfisterer T, Drescher B, Driesel AJ, Müller WEG, Wetter T, Suhai S: **Using the miraEST assembler for reliable and automated mRNA transcript assembly and SNP detection in sequenced ESTs**. *Genome Res* 2004, **14**(6):1147–59.

6. Zerbino DR, Birney E: **Velvet: algorithms for de novo short read assembly using de Bruijn graphs**. *Genome Res* 2008, **18**(5):821–9, [http://genome.cshlp.org/content/18/5/821.long].

7. Hazelhurst S: **Algorithms for clustering expressed sequence tags: the wcd tool**. *South African Computer Journal* 2008, **40**:51–62, [http://www.cs.wits.ac.za/~scott/papers/sacj527.pdf].

8. Picardi E, Mignone F, Pesole G: **EasyCluster: a fast and efficient gene-oriented clustering tool for large-scale transcriptome ...**. *BMC Bioinformatics* 2009, [http://www.biomedcentral.com/1471-2105/10/S6/S10].

9. Hazelhurst S, Bergheim A: **ESTSim: A tool for creating benchmarks for EST clustering algorithms**. *Dept. of Computer Science, Univ. of Witwatersrand (South Africa), Tech. Rep. CS-2003-1* 2003.

10. Richter DC, Ott F, Auch AF, Schmid R, Huson DH: **MetaSim: a sequencing simulator for genomics and metagenomics**. *PLoS ONE* 2008, **3**(10):e3373.

11. Benson DA, Karsch-Mizrachi I, Lipman DJ, Ostell J, Sayers EW: **GenBank**. *Nucleic Acids Res* 2010, **38**(Database issue):D46–51, [http://nar.oxfordjournals.org/content/38/suppl_1/D46.long].

12. Liang F, Holt I, Pertea G, Karamycheva S: **An optimized protocol for analysis of EST sequences**. *Nucleic Acids Res* 2000, [http://nar.oxfordjournals.org/cgi/content/abstract/28/18/3657].

13. Hide W, Burke J, Davison DB: **Biological Evaluation of d2, an Algorithm for High-Performance Sequence Comparison**. *Journal of Computational Biology* 1994, **1**(3):199–215.

14. Prim R: **Shortest connection networks and some generalizations**. *Bell System Technical Journal* 1957, [http://monet.skku.ac.kr/course_materials/undergraduate/al/lecture/2006/prim.pdf].

15. Needleman SB, Wunsch CD: **A general method applicable to the search for similarities in the amino acid sequence of two proteins**. *J Mol Biol* 1970, **48**(3):443–53, [http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B6WK7-4DN8W3K-7X&_user=2518055&_coverDate=03%252F28%252F1970&_rdoc=1&_fmt=high&_orig=search&_sort=d&_docanchor=&view=c&_acct=C000057738&_version=1&_urlVersion=0&_userid=2518055&md5=bb39f37fe275c090da23bb6f53ade603].

16. Smith TF, Waterman MS: **Identification of common molecular subsequences**. *J Mol Biol* 1981, **147**:195–7.

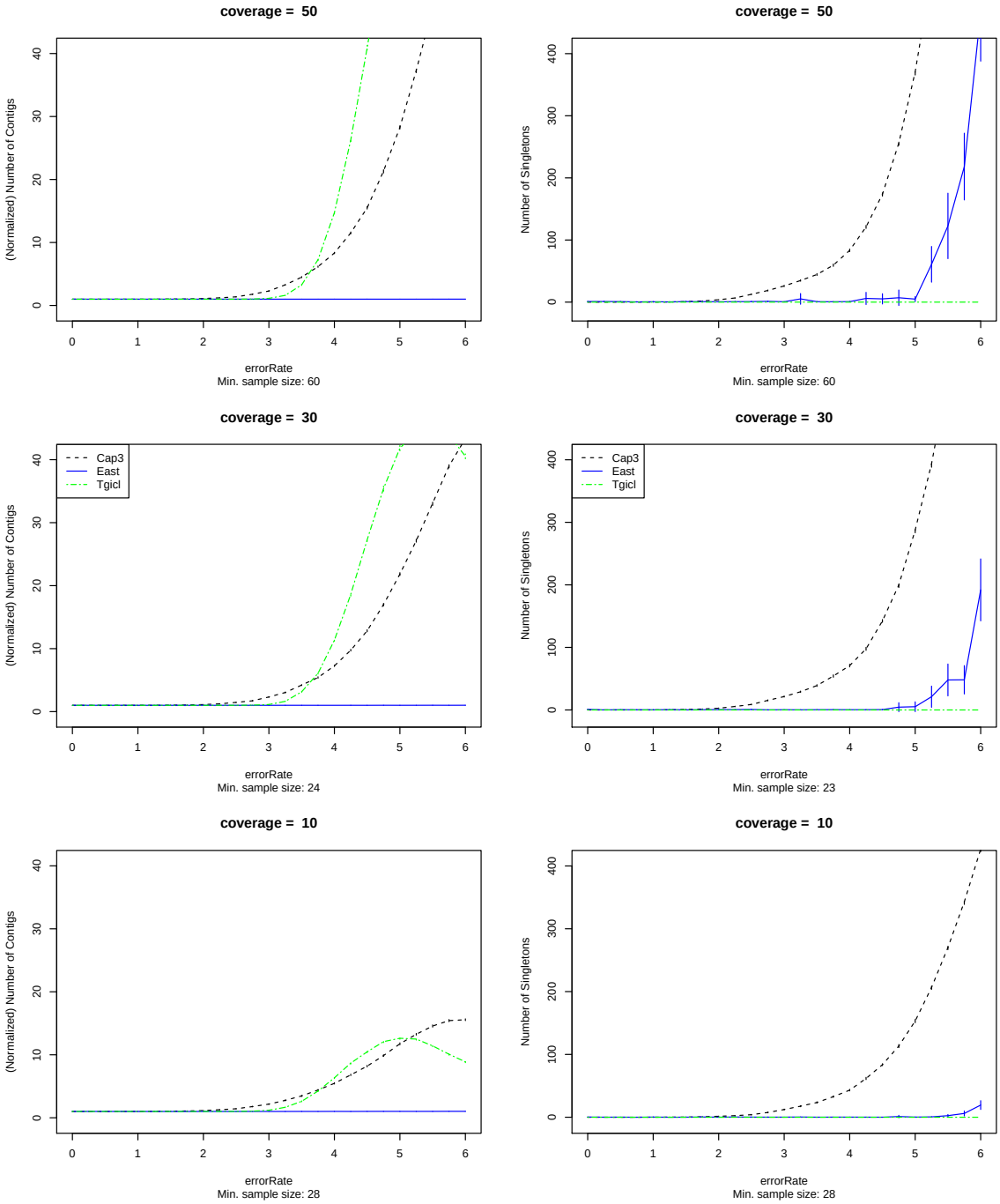

Figure 2: Number of contigs (normalized to the number of transcripts) and number of singletons as a function of base call error rate for simulated Sanger sequencing. EAST=blue, Cap3=Black, TGICL=green, Velvet=red, Mira3=purple. (Number of singles for Velvet and Mira3 not shown.)

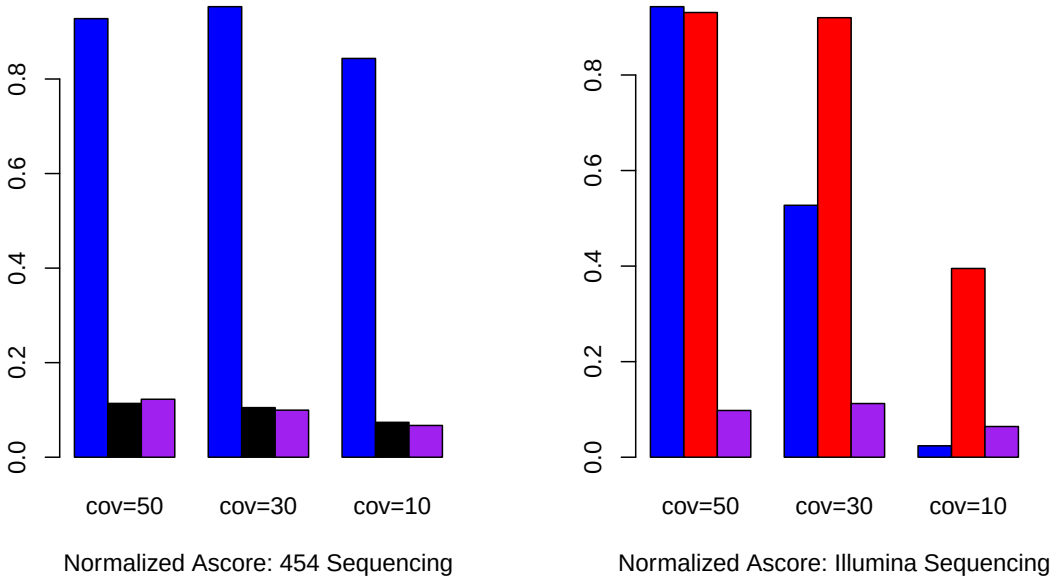

Figure 3: Comparative normalized a-scores for 454 and Illumina sequence output, using the MetaSimdefault model at different levels of coverage. EAST=blue, Cap3=Black, TGICL=green, Velvet=red, Mira3=purple.

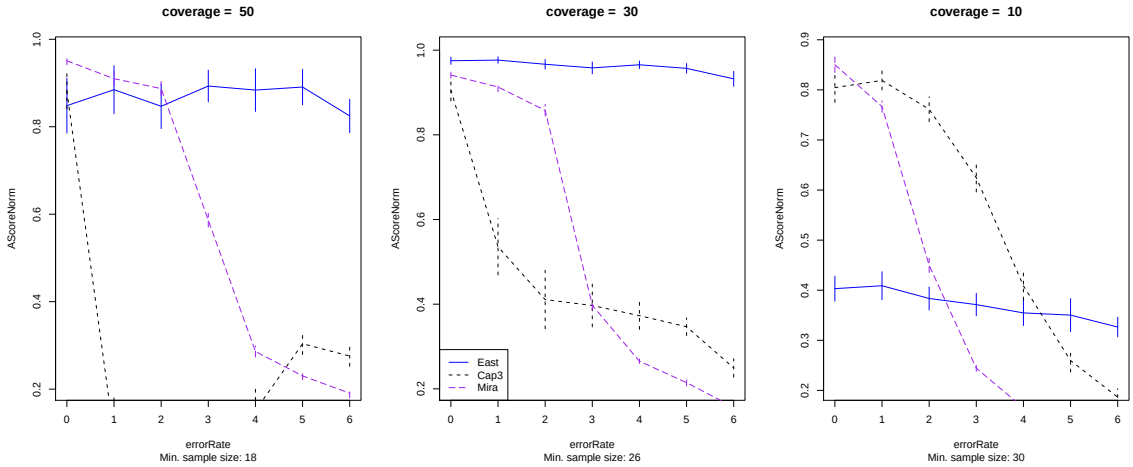

Figure 4: Comparative normalized a-scores a hybrid set of Sanger, 454 and Illumina sequence output, varying the base call error rate for Sanger sequences and using the MetaSim default model for the 454 and Illumina sequences. Each data set tested was composed of 40% Sanger Sequences, 40% 454 Sequences, and 20% Illumina sequences. EAST=blue, Cap3=Black, TGICL=green, Velvet=red,Mira3=purple.

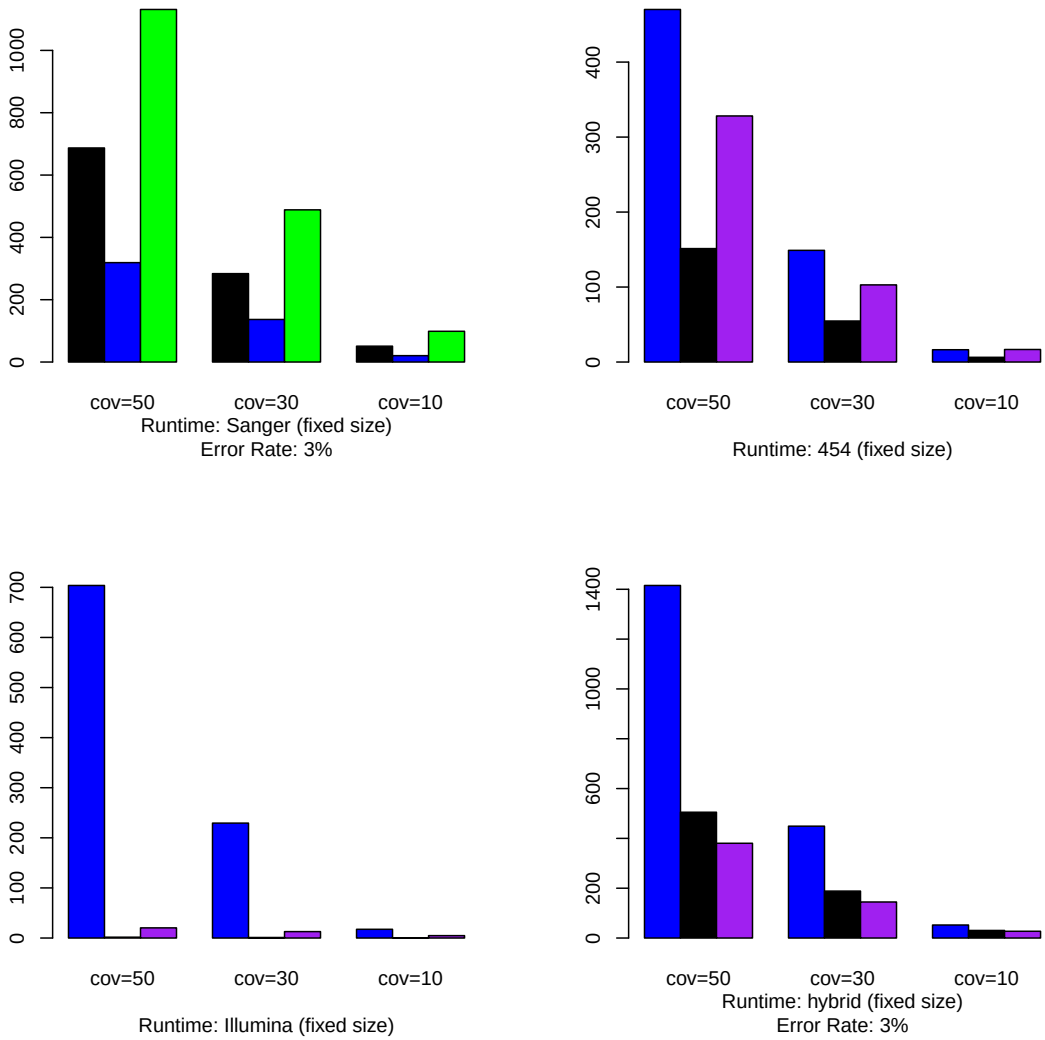

Figure 5: Average runtime per trial for when generating data for Figures 1-4 when coverage = 30). Note that the exceptionally high speed of Velvet on Illumina sequences make the red (center) bars in that plot essentially indistinguishable. (PEACE+)EAST=blue, Cap3=Black, TGICL=green, Velvet=red,Mira3=purple.

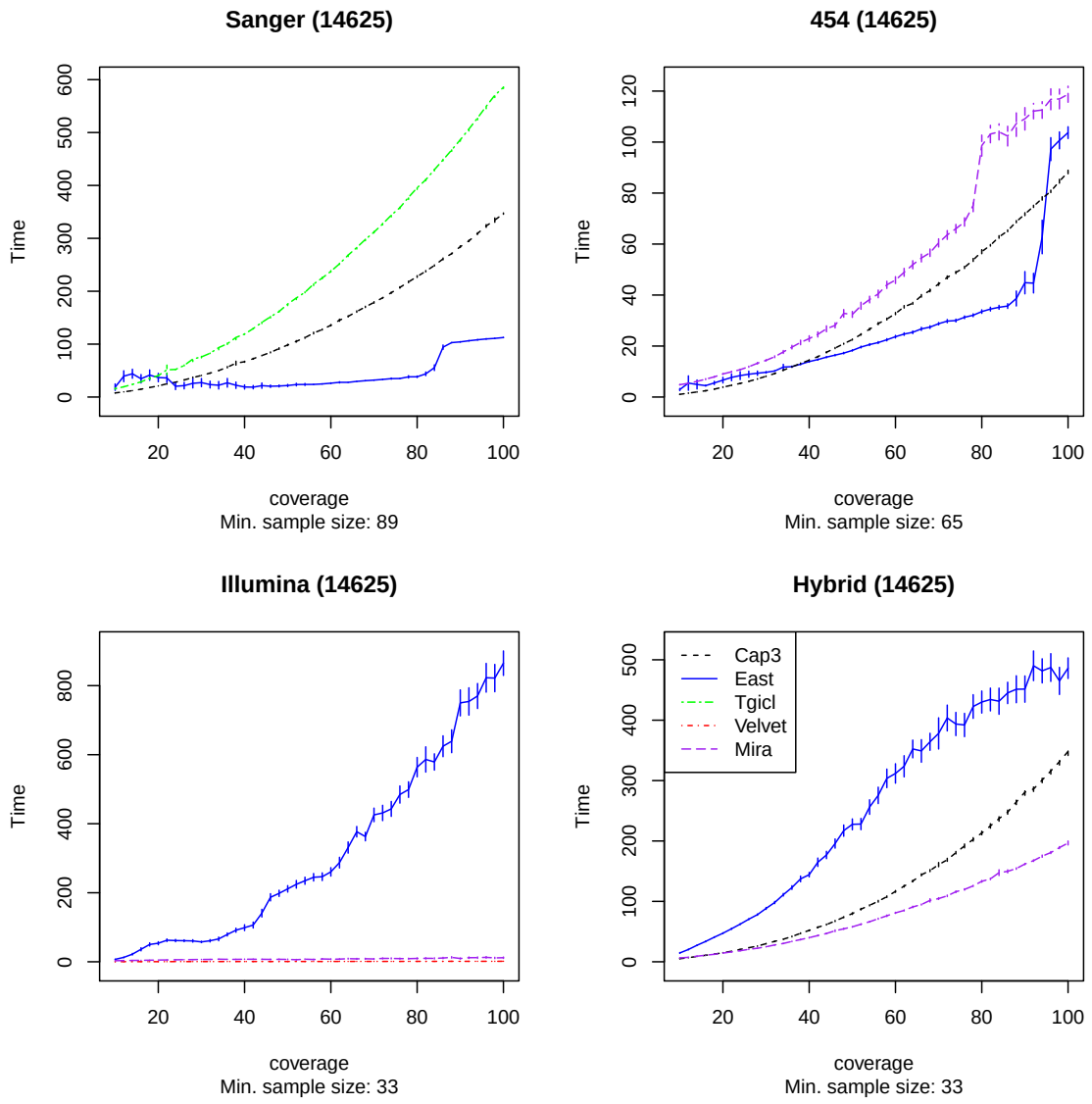

Figure 6: The average runtime when applying each tool to a simulated data set derived from a specific zebrafish transcript of length 14,625 bp. As we decrease coverage the number of fragments drops, hence illustrating the relationship between tool runtime and fragment set size. Each data point represents the average runtime from the application of the tool to at least 30 randomly generated simulated sets coming from the transcript at the given level of coverage. (PEACE+)EAST=blue, Cap3=Black, TGICL=green, Velvet=red,Mira3=purple.

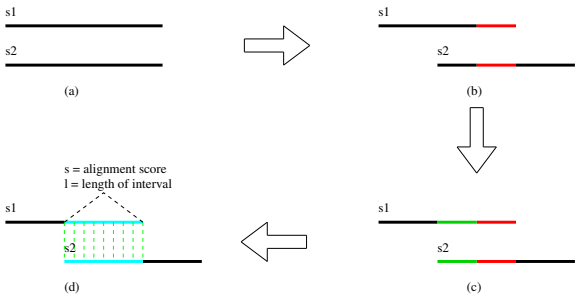

Figure 7: (a) Two sequences potentially overlapping at the ends. (b) The window of s_2 pairing with the rightmost window of s_1 to minimize the d^2 score. (c) An extension of the s_2 window to the end of the sequence, and the corresponding extension on s_1. (d) A global alignment of the overlapping areas determined from (c).