GENOME TECHNOLOGY

Assembly and Alignment Algorithms for Next-Gen Sequence Data

A TROUBLESHOOTING GUIDE:
Experts share their advice on assembly and alignment algorithms for next-gen sequence data

METHODS

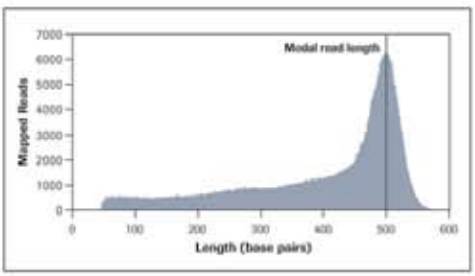
Q&A

www.roche-applied-science.com

Genome Sequencer FLX System

Introducing the GS FLX Titanium Reagents

Length Really Matters



Example Read Length Distribution of 629,643 reads from *E. coli* K-12 (Genome size ~4.5 Mb) with a modal read length of 504 bases.

- Obtain sequencing read lengths of 400 to 500 bases.
- Generate more than 1 million sequencing reads per 10-hour instrument run.
- Improve performance by using GS FLX Titanium series reagents – without instrument upgrades.
- Accelerate the pace of discovery with easy-to-use analysis tools for straightforward interpretation of data and biologically meaningful results.

Performance, Results, Impact

Learn more at ***www.genome-sequencing.com***

454 SEQUENCING

Roche Diagnostics GmbH
Roche Applied Science
68298 Mannheim, Germany

Roche

Table of Contents

Genotyping | Sequencing | Gene Expression | Biologics Evaluation & Safety Testing

Sharpen your Focus

With our global presence, strong reputation for quality and customer service across the broadest range of genomic services in the industry, let Cogenics be your genomics solution partner so you can focus on what you do best.

Visit www.cogenics.com/focus for more information

Cogenics sets the standard for delivering expert genomics solutions to life science and healthcare businesses and academic institutions worldwide.

Whether you are researching bacterial or plant genomes or preparing an NDA submission for the FDA, we are the right choice to design your studies, to guide and perform analysis on a wide range of platforms and apply our bioinformatics expertise to turn data into the solutions that you are looking for.

With twenty years of experience and a clientele that includes all top twenty pharmaceutical companies, you can rely on Cogenics to deliver world class results on time and on budget.

CO:GENICS™
The Genomics Services Company

US: 1 877 226-4364
UK: +44 (0) 1279-873837
Email: sales@cogenics.com

France: +33 (0) 456-381102
Germany: +49 (0) 8158-9985 0
www.cogenics.com

Letter from the editor

As next-generation sequencing pushes the era of personal genome sequencing from dream to reality — last month, two papers in *Nature* reported having used Illumina technology to sequence the first complete genomes of an African and an Asian — the tools to align and assemble read data are scrambling to keep up. While there is a slew of available software and programs out there, deciding which ones to use and how best to use them is still challenging. As alignment and assembly algorithms continue to be optimized for short-read data and *de novo* assembly, existing or out-of-the-box solutions are not always the best answer.

In this issue, we bring you a technical guide about these algorithms, with a focus on both users and developers. As the list of next-gen sequencing users continues to grow, so does the need for both speed and accuracy when dealing with read data. While not all users are developers and vice versa, we tailored the questions to both, hoping to cull from the broadest expert pool. Our questions, too, are wide-reaching and cover everything from balancing speed and accuracy to optimizing choice of algorithm and discovering the best ways to customize analysis to your specific sequencing run. We've also compiled a handy list of resources at the back of the guide. Happy aligning!

— Jeanene Swanson

Index of experts

Genome Technology would like to thank the following contributors for taking the time to respond to the questions in this tech guide.

Michael Brudno
UNIVERSITY OF TORONTO

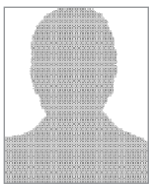
John Pearson
TRANSLATIONAL GENOMICS RESEARCH INSTITUTE

Andreas Sundquist
STANFORD UNIVERSITY

David Craig
TRANSLATIONAL GENOMICS RESEARCH INSTITUTE

Bruce Roe
UNIVERSITY OF OKLAHOMA

Haixu Tang
INDIANA UNIVERSITY



Heng Li
WELLCOME TRUST SANGER INSTITUTE

Andrew Smith
UNIVERSITY OF SOUTHERN CALIFORNIA

René Warren
BRITISH COLUMBIA GENOME SCIENCES CENTER

"The SOLiD™ System provides me the accuracy and throughput to expand the application of whole genome sequencing, which is so important because its direct application will be finding ways to solve human genetic diseases. This technology is really revolutionary because it allows us to tackle projects that we hadn't imagined we would be able to do just a few years ago."

Donna Muzny –
Human Genome Sequencing Center, Baylor College of Medicine

"The longer read lengths of the SOLiD™ 3 System are particularly exciting for enabling me to detect novel splice forms in RNA expression experiments. The strand specificity of the SOLID™ Small RNA Kit enables me to distinguish between sense and antisense transcription."

Jesse Grey –
Children's Hospital Boston, Harvard Medical School

"The SOLiD™ System enables a massive expansion of the application space - the open slide format and sample multiplexing capability provides me the flexibility to support my current customers as well as opens doors for clinical researchers."

Bill Farmerie –
ICBR, University of Florida

SOLID PROOF

Whether you're trying to decipher human genetic variation, discover novel splice forms, or identify strand specific expression patterns, the SOLiD™ 3 System is designed to enable experiments you never thought possible. Proven in labs across the world, the SOLiD™ 3 System provides the flexibility, throughput and accuracy to support your research today and the scalability to grow with you into tomorrow. When you do the math, it all adds up to SOLID Proof.

For more SOLID PROOF, visit solid.appliedbiosystems.com

Applied Biosystems

How do you choose which alignment algorithm to use?

Currently there are a plethora of methods that have been developed for mapping short reads to a genome. Some of the more popular ones for letter space data include MAQ and ELAND. For AB SOLiD color-space data, AB's CORONA pipeline and the SHRiMP tool made in our group are widely used. When choosing an alignment algorithm it is very important to consider not just the speed and sensitivity, but also the exact problem at hand. If you are doing a ChIP-seq, or RNA-seq analysis where insertions/deletions are expected to be very rare and most of the mismatches are due to sequencing errors, you can afford to use an extremely fast mapping method that allows for a fixed number of errors in the read — even if a fraction of the reads are not mapped due to some biological variation, the high coverage from even a single lane/quadrant for most of these experiments may allow you to ignore a good fraction of the data. If, on the other hand, you are interested in biological variation, you have to use a slower and more sensitive method, as otherwise you will miss the reads with the most variation — the very thing you want to find!

— Michael Brudno

It is important to recognize alignment algorithms are well researched. The primary difference with next-generation sequencing is that the programs using the algorithms are not fully developed and optimized to take advantage of all that next-generation sequencing data offers.

Essentially, there are two steps in the alignment process: candidate lookup and local alignment. Local alignment is solved using the Smith-Waterman and largely consistent between algorithms (around since 1970). Conversely, current alignment algorithms optimize the candidate lookup, which reduces the search space of the local alignment from the entire genome to a short list of possible alignment locations.

In our decision process, we balance speed and accuracy towards the reference we are using for alignment. We ask: How big is the genome? Do I wish to align indels? Typically for us accuracy defines speed.

For choosing algorithms, a well-maintained list exists at *http://en.wikipedia.org/wiki/Sequence_alignment_software#Short-Read_Sequence_Alignment*.

We currently are using BFAST, due to its speed and ability to align indels. A substantial portion of our data is from targeted sequencing, and frequently we have sequenced ~100 kb from multiple individuals. This particular program functions on both ABI and Illumina, allowing for indel detections in both platforms. Thus far, BFAST is the only algorithm we have found that aligns indels with ABI SOLiD's color space data.

Sequence alignment is hard, and there is no out-of-the-box solution. Just as any experimental design, many decisions need to be made which will affect your results.

Running your algorithm in a "default" mode, or without knowing your *a priori* accuracy robustness, is a sure way to publish in the journal of negative results.

— David Craig & John Pearson

The ideal alignment algorithm should be fast, light-weighted, sensitive, and flexible, but in practice, we can hardly achieve all these goals. For short read alignment, I think the baseline for a good algorithm is: 1) capable of aligning ~100 reads to the human genome per second; 2) consuming no more than 1 GB memory for each CPU core (this is important for parallelizing across a large computing cluster); 3) able to find most hits given maximum 5% error rate. In the future when reads are routinely longer, allowing long reads and gaps will be essential. My other alignment program BWA aims to achieve this goal, although it is still at its early stage.

— Heng Li

Choice of alignment algorithm depends first on the sequencing technology used, which determines read length and expected accuracy of base calls. For a given technology, algorithm choice then depends on the sequencing application. We can expect sophisticated alignment algorithms to soon be introduced for specific sequencing applications. Scientists should select an algorithm with features that are important to the goals of their project. For example, in a project to identify genome variation, the algorithm should be sufficiently sensitive to the kind of variation deemed important — even more sensitivity might be required when aligning reads from an unassembled genome to a reference from a related assembled species. For heterogeneous samples, such as those seen in some metagenomics applications, greater specificity might be important. ChIP-seq is a more established application, and even basic algorithms like ELAND have been found to work very well. For RNA-seq where alignments might be to a reference transcriptome rather than a reference genome, it might be desirable to identify all alignments for each read and do some form of post-processing. The most popular algorithms do not always have the most important features for each application.

> **"There is no out-of-the-box solution."**
>
> **— David Craig & John Pearson**

Bioinformaticians working at sequencing facilities may have different criteria for selecting an alignment algorithm to use as part of their default pipeline. Many sequencing centers benchmark novel and updated alignment algorithms regularly to determine which are most appropriate for their most frequent applications and which work best with the computing infrastructure available. Regular benchmarking on reads from different sequencing applications also enables sequencing centers to help their clients select the most appropriate algorithms.

— Andrew Smith

Which alignment algorithm you should use depends on many factors: 1) the sequencing technology, 2) read length, 3) number of reads and available compute resources, 4) sensitivity/scoring requirements. Unfortunately, I am not aware of any algorithm that would do well in every situation. Certain factors will make the choice easy, however. For example, if you are sequencing with ABI's SOLiD platform, you should use an algorithm that does its computations in color space. On the other hand, 454 sequence reads are prone to homopolymer run count errors, so your method had better be able to align

with insertions and deletions. Read length will usually also constrain your choice of algorithms, as some methods are optimized for extremely short, fixed-length reads, while others depend on longer reads that have longer exact seeds. If you don't have very many reads to align, or you have access to a large cluster, you may be able to afford to do full Smith-Waterman alignment, while for extremely large datasets, you may be forced to use a fast, heuristic method. This leads to the last factor, sensitivity/specificity requirements. Many algorithms trade off sensitivity for speed, or do not properly calculate the likelihood of the alignments. Whether this is important for the analysis at hand is up to the researcher to determine.

— Andreas Sundquist

We will consider a balance between the speed and the low error rate depending on needs of the specific application. For the common purpose of aligning the ultra-short reads (e.g. from Illumina/Solexa sequencer) to the reference genome, speed is the first factor we consider. In particular, we will use ELAND or SOAP. For the purpose of similarity search using short reads (e.g. from Roche FLX), we consider the accuracy (sensitivity and specificity) as the most important factor. In particular, BLAST or Mega-BLAST will often be our choice, although they are relatively slower. In some specific applications, we may also choose to combine the results from a few alignment algorithms.

— Haixu Tang

"Many algorithms trade off sensitivity for speed, or do not properly calculate the likelihood of the alignments."

—Andreas Sundquist

For my work, the choice of an aligner usually comes down to two or three factors (in order of importance): speed, accuracy, and output formats. Depending on the nature of the work, more emphasis may be put on speed at the expense of accuracy. For instance, you may want to analyze a human genomic data set for contaminating DNA sequences. In this case, speedy alignments against a human reference genome may be used as a filtering step. A robust, fast and generic tool for this is Exonerate (Slater and Birney, 2005). Exonerate is simple to set up and use and parallelizes well; Exonerate alignment jobs can be farmed on a compute cluster without much fuss. Exonerate supports a number of standard or customizable output formats, which is handy if you don't want to spend too much time writing a parser to sift through the results. When accuracy is of the essence, exhaustive aligners such as MAQ (Li *et al*, 2008) or Novoalign (Novocraft) are guaranteed to return optimal alignments with low false-mapping rates for both single- or paired-end data sets. Both are designed to take quality scores into account, which is a must for any serious high-throughput SNP surveys. The latter aligners, especially MAQ, are feature-rich and fine-tuning for your needs requires intermediate to advanced skills.

— René Warren

"For my work, the choice of an aligner usually comes down to two or three factors: speed, accuracy, and output formats."

—René Warren

How do you optimize your alignment algorithm for both high speed and low error rate?

This very much depends on the alignment tool. Some tools allow a setting up to some number of mismatches per read — optimizing the parameters for these tools is straightforward. Others allow you to vary a large set of parameters. For example, in the SHRiMP tool you can set the seed size (this has the biggest effect, as using bigger seeds is faster but less sensitive), a number of seeds to start a rigorous alignment (again, more so faster), and a threshold score which determines the number of alignments that are output. Finding a good compromise between these depends on the expected amount of variation (the settings for human, which is less polymorphic should be different from the settings for highly polymorphic organisms such as *Ciona*) and the length of the read, as longer reads allow one to use longer seeds.

—Michael Brudno

Two factors confound alignment, affecting speed and error rate: inability to map reads due to inadequate searching and mapping reads inaccurately.

The inability to map reads comes from the fact that most alignment tools cannot map reads with moderate error rates (>two mismatches) or moderate variant rates (>two SNPs in a read). Speed in this case comes from not enumerating all possible errors/variants in a read.

Additionally, using a small number of bases (<18 bp) to perform the alignment lookup (in the human genome) gives us the expectation of multiple possible alignment locations. This causes a speed decrease since now we have to filter through a list of possible alignments using the local alignment method, which is the most expensive step in the two-step process.

This leads us to mapping reads inaccurately. Short reads cause local ambiguities in the alignment, for example, where to place an insertion or deletion in a stretch of poly-As, or an indel anywhere to be correct. This relies on the local alignment method, which already has an agreed standard (Smith-Waterman). We are looking for the most likely alignment to a reference genome, so if a read is not mapped accurately because it maps to location X (false) that has fewer edits in relation to the read than the correct location Y, it is more likely that it comes from the X. This can occur with errors in your reads.

With the current alignment scheme we are utilizing in BFAST, we define accuracy at the first level and then speed at the second level.

— David Craig & John Pearson

There is always ambiguity in short read alignment due to repeats and sequencing errors, but if we know how likely an alignment can be wrong, we can use alignment more effectively. Mapping quality directly measures the alignment accuracy and we find it quite helpful in practice. Sometimes, we may like to trade speed for accuracy if we can get to ~100 reads per second. Allowing longer reads, qual-

ity awareness, and paired-end alignment all make MAQ slower, but we think these features are more important than speed. BWA achieves speed mainly by using Burrows-Wheeler Transform. It still trades speed for accuracy in that it also searches for good suboptimal hits. Finding uniquely mapped reads only would make BWA times faster.

— Heng Li

We're not really optimizing the alignment algorithms per se because we don't have the code for doing that. What we're doing is optimizing conditions for making the libraries that we're loading onto the [454] machines. So we've done a lot of tweaking of the protocols for making the libraries, for doing the emPCR, for clean-up afterwards, and tweaking some of the reagents so we're actually getting close to 300 bases per read on average.

— Bruce Roe

In general there is a trade-off between high speed and low error rate for alignment algorithms. But more precisely the tradeoff is between speed and sensitivity, which is only part of the error rate. Alignments where the reads match the reference genome very closely can usually be found quickly, but those alignments will have low tolerance to sequencing errors or natural genomic variation. Algorithms I have developed can identify very distant matches, but doing so takes much more time. This tradeoff has diminishing returns, however: allowing too many differences in the alignments lowers specificity. In other words, if too many differences are allowed, reads that should not align will begin to have sufficient similarity with the reference genome purely by chance.

Again the proper balance depends on the application. Drawing on experience with different algorithms and sequencing applications, bioinformaticians at sequencing facilities can usually provide some guidance on how to adjust alignment parameters to strike the proper balance for a specific project. When I analyze my own datasets, algorithm speed is a secondary concern. I would rather wait a little longer for alignment results if I expected to gain a few percent in terms of number of reads aligned and accuracy of those alignments. It is important to be confident in the quality of the alignments before drawing any conclusions from the experiment.

— Andrew Smith

It's always a trade-off between high speed and accuracy. Unfortunately, with the large amounts of data being produced by some of the next-gen sequencers, I suspect many researchers will be forced to opt for high speed in order to get results. Although it's possible that a majority of the reads will still be aligned correctly using the fastest algorithms, there will be many reads that are missed altogether, or misaligned. For example, some of the fastest methods for aligning microreads do not handle insertions and deletions, which means that these reads will not be mapped, or worse, may be mapped falsely to another location. A similar problem exists for repetitive genomes, where a read may align to many locations. The fastest algorithms look for the "best" match first, and ignore other alignments. But how good is this "best" match really? We have no idea whether or not to trust this placement.

—Andreas Sundquist

Typically, we will perform some experiments by using the simulated reads with similar discrepancies with the reference sequences for a specific application. Our objective is to select the best alignment algorithm with the optimal parameter settings for specific applications.

— Haixu Tang

There's almost always a compromise to make between high speed and accuracy. In

continued on page 18

What approach do you use to handle mismatches or alignment gaps?

For mismatches, AB SOLiD's color space technology has an advantage over regular letter space. Sequencing errors look completely different in the read output from SNPs or other polymorphisms. A tool that is aware of the color space encoding can leverage it to provide confident SNP and indel calls even from a single read, though two reads that support each other are, of course, better. For handling alignment gaps the use of mate pairs is critical: if the alignment gap is supported by mate-pair data where one of the pair-ends does not map, it is possible to do *de novo* assembly on just the small set of reads to recover the polymorphic region.

— Michael Brudno

Mismatches and indels can be handled by avoiding them when performing your candidate lookup. We use generalized masks (choosing a subset of possibly non-contiguous bases) to perform the lookup. Effective aligners avoid indels by assuming that we cannot use the indel sequence, and must map using bases on either side of the indel. SHRiMP and BFAST are programs that utilize these types of approaches.

Some algorithms try to use bases that span the indel. In this case, the algorithm must perform enumeration, which scales exponentially. Because of this, we do not utilize algorithms based on enumeration. Splitting the reads into smaller pieces also has the same effect. In general, we tend to avoid alignment algorithms that use these approaches in order to be robust in alignment to variants.

— David Craig & John Pearson

Like ELAND, MAQ uses a non-contiguous seed template to find mismatches. MAQ finds short indels only for paired-end reads. If one end is mapped but the other end not, we try Smith-Waterman alignment for the unmapped read in the region defined by the mapped end. BWA is completely different. It uses branch-and-bound search to try all possible mismatching/gapped alignments.

— Heng Li

Mismatches — if there are high quality discrepancies, then there are high quality discrepancies, that's what nature's giving you. As far as gaps, that's a really frustrating point. We don't understand how Newbler doesn't connect things. What we end up doing is going back and performing PCR across those gaps.

— Bruce Roe

My colleagues and I developed the RMAP algorithm for aligning Solexa/Illumina reads and in the original version mismatches were weighed according to the base call quality scores. Recent versions make more extensive use of quality score information. When a read aligns to non-consensus bases, we can penalize each base very precisely. We have observed this approach to both increase the number of aligning reads and decrease the error rate on several sequencing runs.

I have also implemented alignment algorithms for short reads that allow a small number of gaps in the alignments.

We know a great deal about how to model gaps in alignments of functional sequences, for example, using structural or evolutionary information about proteins, but these models are not as helpful in aligning reads. More research is required to understand how gaps can be caused by different sequencing technologies, and also to understand small gaps in genomic regions that are not under strong selective pressure.

— Andrew Smith

For most researchers who aren't interested in implementing their own methods, we are really at the mercy of whatever the tool we choose provides. Some extremely fast methods that use clever indexing schemes simply will not do alignment gaps, and no amount of post-processing will fix that. Ideally, mismatches and gaps should be handled using the traditional probabilistic alignment scores. For example, Smith-Waterman methods succeed at this, while many heuristic methods compute approximations of this.

— Andreas Sundquist

For ultra-short reads, we allow very few gaps and mismatches (typically up to two). For short reads (e.g. Roche FLX), we will evaluate the alignment based on a specific pair-HMM-based error model.

— Haixu Tang

Base mismatches and alignment gaps are inevitable and may be very informative. There are a couple of things to look at to convince yourself that what you observe may be real: 1) the base quality of the base mismatch/bases in the gap and 2) the read depth in the region spanning the mismatch/gap. From this observation, the significance of seeing these events by random chance can be easily calculated and reported.

— René Warren

How do you choose which assembly algorithm to use?

We utilize Velvet. In the future, we hope to identify or develop programs that combine both alignment and assembly. Essentially this may allow for discovery of large insertions, such as foreign DNA in a microorganism.

— David Craig & John Pearson

We've defaulted to Newbler and then alternatively try Phred and Phrap. The problem with these algorithms is ... how do they deal with the repeats? Newbler is getting better at it because we're getting longer reads.

— Bruce Roe

Unlike alignment, no applications of *de novo* assembly from second-generation technology have become routine. If the only data available is from a short-read technology, with read lengths under 50bp, then *de novo* assembly of larger genomes will be exceptionally difficult. Hybrid approaches that use a related genome as a reference or approaches that mix long and short reads have been successful. It remains unclear which combination of sequencing technologies and algorithmic strategies will work best together. *De novo* assembly has once again become a very active research area, and new algorithms will open the door to exciting new applications of sequencing.

— Andrew Smith

Similar to the choice of alignment algorithm, in choosing an assembly algorithm we must consider the underlying sequencing technology, its read lengths, whether we have paired reads, the overall sequencing protocol, and the size of the genome being assembled. Whole-genome paired-end shotgun appears to be the sequencing protocol of choice for many, and there are many programs available for this. However, these programs differ tremendously in the size of assemblies they can handle. An assembler for paired microreads on small bacterial genomes will not be appropriate for mammalian genomes. Similarly, a whole-genome mammalian assembler will not produce good results when used on paired microread data. Assembly algorithms are usually designed with a particular read length in mind, and will likely fail if you give them something different than what they expect. One way to choose an assembly algorithm is to look at what was used successfully in the past to assemble genomes similar to your own.

— Andreas Sundquist

Our first objective is to get less and longer contigs and scaffolds. Speed is not our primary concern. However, it turns out the performance of the assembly algorithms on Sanger reads, short reads (e.g. Roche FLX), and ultra-short reads (e.g. Illumina/Solexa) are very different. In general, we use Arachne for assembling Sanger reads; Newbler for assemblying FLX reads; and Velvet for assembling Solexa reads. Sometimes, we use EULER/EULER-SR to help us to resolve repeats. We have not tried any hybrid assembly yet.

— Haixu Tang

I just use my own. SSAKE (Warren *et al.*, 2006, 2007) is one of the first tested and true micro-

continued on page 18

Do you use mate-paired reads for *de novo* assembly? How?

I do not currently use mate-paired reads for *de novo* assembly, but some of my colleagues do. In some ways the pairing information effectively lengthens the reads and can help very much in algorithms that combine assembly with alignment to an existing reference genome. The pairing information also tells us about relative orientations of reads, and this really helps to understand the final assembly structure.

— Andrew Smith

We make constraints that if this sequence and that sequence are roughly 1,000 to 2,000 bases apart, then we have to try and get them to be that way when we do the assembly. We're letting the computer do the assembly, but we use the read pairs to ensure that the assembly gave the right answer. In instances where those reads are 10,000 bases apart, you know that something's screwed up. So read pairs actually play two functions: one function is to give you more data for the assembly, [and] the second thing is they help you figure out if it's assembled correctly.

— Bruce Roe

Mate-paired reads should always be used for assembly if it is reasonable to sequence in that way. The additional long-range information for a *de novo* assembler is invaluable in avoiding misassemblies in repetitive regions. Algorithms that use mate-pairs will need to know the insert sizes of the read pairs, and oftentimes it's beneficial to have several different insert sizes: smaller ones to help assemble local regions, and larger ones to help disambiguate large, repetitive structures.

— Andreas Sundquist

Yes. We only tried to use mate-paired reads from Roche FLX for *de novo* assembly using Newbler. In many cases, they are very helpful. In a few cases, we also use EULER-SR for the resolution of repeats using mate-pairs.

— Haixu Tang

I do. SSAKE supports the use of paired-end reads to build scaffolds, an ordered and oriented arrangement of contigs. It will also give stats on how well each contig assembled by analyzing the logical placement of mate pairs in the assembly.

— René Warren

"Read pairs actually play two functions: one function is to give you more data for the assembly, [and] the second thing is they help you figure out if it's assembled correctly."

—Bruce Roe

What impact does the quality of raw read data have on alignment or assembly? How do your algorithms enhance this?

The quality of the raw read data is important for short read alignment — if the reads were perfect, calling SNPs and other variation would be easy. Unfortunately nucleotide calls are far from perfect, and must be taken into account for SNP calling when working with letter-space data. For color-space this is not as critical, as adjacent color-calls support each other, and a single error will never lead to a SNP call.

— Michael Brudno

Read quality is generally only to be taken into account during local alignment (step 2), not the candidate lookup step (step 1). Any alignment algorithm that uses quality scores (MAQ, etc.) will only take them into account in step 2, or will actually ignore certain bases in (step 1). Nevertheless, the local alignment algorithm should handle this, and the local alignment algorithm should be standard across all algorithms (Smith-Waterman).

— David Craig & John Pearson

Low quality means two things: 1) the base error rate is high and 2) qualities are inaccurate. High error rate leads to more mismatches and fewer mapped reads. As in Illumina and 454 errors tend to accumulate at the end of a read, we can use the first tens of base pairs to find seed hits and extend the hits to the whole reads. This in principle allows unlimited number of mismatches at the tail of a read. However, in practice we may not like to do so as alignments with many high-quality mismatches are likely wrong. Inaccurate qualities are more related to base callers. We can calibrate qualities before giving reads to alignment programs.

— Heng Li

Data quality has a significant impact on both alignment and assembly. Most programs for aligning reads, including those I have developed, have a preprocessing stage to identify and remove low-quality reads. If the data from a sequencing run seems poor in general it might be wise to discard the entire dataset, rather than try to make use of the portion that appears to map well. Each technology keeps improving, and the research community keeps gaining experience with the technologies, but we are still learning how to make sure each sequencing run produces the highest-quality data.

In the case of alignment it may also be possible to remove bad reads after the alignment has been done. Information from the alignments, such as the proportion of aligned reads, uniquely aligned reads, or the average number of mismatches can be used diagnostically if similar sequencing runs have been done before and we know what to expect. If certain features of the alignment fall below the level considered normal, it might be an indication of a problem in some part of the experiment. Assembly is different: including low-quality reads can have a greater impact because the nature of *de novo* assembly is to identify relationships between. So a small proportion of bad reads can bias the entire assembly.

— Andrew Smith

continued on page 18

Q2: *continued from page 12*

order to find the right balance between these factors, it is usually a good idea to set parameters that favor accuracy (preferentially by running exhaustive aligners or set heuristic aligners to run in exhaustive mode over regions where a heuristic alignment is found, if applicable) and do the following to maximize the speed: split your input [sequence read] file and farm your jobs on a computer cluster, keep your data on local disks to minimize heavy IO over your network, set the aligners such that they return only the highest scoring alignments and, last but not least, filter your data sets. The same way you don't cook your vegetables without cleaning, peeling, and chopping them, you don't want to use your sequences without weeding out the crappy ones. Your reads might be comprised of adapter-dimers, homopolymeric stretches, and extremely redundant sequences. Removing all the garbage will most certainly speed things up.

— René Warren

Q4: *continued from page 15*

read assemblers released. One of the strengths of SSAKE is its scalability. I have successfully run it on 80-M reads, requesting at most 49 GB RAM, a feat difficult to achieve with most recent assemblers without getting the dreaded "out of memory" error. SSAKE also performs well in lower-coverage areas; it was built to explore all seed/contig extension possibilities and reports the assembly of all seed sequences. Low sequence coverage is usually not an issue with high-throughput next-generation sequencing of bacterial genomes, but can be limiting when attempting metagenomics or whole-transcriptome reconstructions. SSAKE is written in a scripted language and requires the Perl or Python (up to v2.0) interpreters. This makes SSAKE very portable and capable of using the full extent [64-bit] of the compute resources at its disposal, especially RAM which it may need a lot of depending on the size of the data set. SSAKE is very simple to run, to understand and modify if need be; the development and publication of VCAKE (Jeck *et al.*, 2007) attests to this.

— René Warren

Q6: *continued from page 17*

The lower the error rate, the easier the assembly! Every effort should be made to produce as high quality reads as possible. However, once we reach the typical 1% error rate, I don't believe the assembly will improve tremendously. Large-scale misassemblies are more a result of repetitive, ambiguous regions of DNA than low-quality read data.

— Andreas Sundquist

A tremendous impact. Even though base error countermeasures have been implemented in SSAKE, ultimately, the assembler will give you an assembly as good as the data you feed into it. That's why it is usually a good idea to filter your reads (see my answer to Q2). In addition, I always quality-trim the reads before assembly, using probability scores provided by Illumina's Genome Analyzer. A small Python utility called TQS.py is included with the current release of SSAKE and will trim the reads using user-defined quality score thresholds. Some of the features affecting SSAKE's output and implemented in the current software release include: A) read coverage: the greater the depth, the more chances the right base will offset a base error at a given position; B) error-handling: uses a majority-rule approach for building consensus bases during a seed extension similar to VCAKE; C) removes bases that cause premature breaks during the fragment assembly. If set, end-trimming kicks in only when all possibilities have been exhausted for a seed/contig extension.

— René Warren

List of resources

Our panel of experts referred to a number of publications and online tools that may be able to help you get a handle on assembly and alignment tools for next-generation seuqencing technology.

PUBLICATIONS

Butler, J. *et al.* **ALLPATHS: *de novo* assembly of whole-genome shotgun microreads**. *Genome Res.* 18, 810–820 (2008).

Li, H., Ruan, J., Durbin, R. **Mapping short DNA sequencing reads and calling variants using mapping quality scores**. *Genome Res.* published online, August 19, 2008.

Li, R., Li, Y., Kristiansen, K., Wang, J. **SOAP: short oligonucleotide alignment program**. *Bioinformatics* 24, 713–714 (2008).

Ning, Z., Cox, A.J., Mullikin, J.C. **SSAHA: a fast search method for large DNA databases**. *Genome Res.* 11, 1725–1729 (2001).

Shendure, J., Hanlee J. **Next-generation DNA sequencing**. *Nature Biotechnology* 26, 1135–1145 (2008).

Sundquist, A., Ronaghi, M., Tang, H., Pevzner, P., Batzoglou, S. **Whole-genome sequencing and assembly with high-throughput, short-read technologies**. *PLoS One 2*, e484 (2007).

Warren, R.L., Sutton, G.G., Jones, S.J., Holt, R.A. **Assembling millions of short DNA sequences using SSAKE**. *Bioinformatics* 23, 500–501 (2007).

Zerbino, D.R., Birney, E. **Velvet: algorithms for *de novo* short read assembly using de Bruijn graphs**. *Genome Res.* 18, 821–829 (2008).

WEBSITES

http://en.wikipedia.org/wiki/Sequence_alignment_software#Short-Read_Sequence_Alignment

http://www.phrap.org/phredphrapconsed.html

http://www.ebi.ac.uk/~guy/exonerate

http://maq.sourceforge.net

http://bioinformatics.bc.edu/marthlab/Mosaik

http://rulai.cshl.edu/rmap

http://compbio.cs.toronto.edu/shrimp

http://soap.genomics.org.cn

http://www.sanger.ac.uk/Software/analysis/SSAHA2

http://synasite.mgrc.com.my:8080/sxog/NewSXOligoSearch.php

http://www.genomic.ch/edena

http://sharcgs.molgen.mpg.de

http://www.bcgsc.ca/platform/bioinfo/software/ssake

http://sourceforge.net/projects/vcake

http://www.ebi.ac.uk/%7Ezerbino/velvet

http://bioinformatics.bc.edu/marthlab/PyroBayes

http://bioinformatics.bc.edu/marthlab/PbShort

http://www.sanger.ac.uk/Software/analysis/ssahaSNP

"i can go where the biology takes me."

"In research, one discovery leads to another which leads...well, who knows where? The Illumina Genome Analyzer gives me the technology to follow almost any path. Only with this system could we create the most detailed and integrated epigenome map to date for any species."

Dr. Brian Gregory
Postdoctoral Fellow
The Salk Institute for Biological Studies

Study the genome. Epigenome. Transcriptome. All at single base-pair resolution. Do more, and do it better, with the Illumina Genome Analyzer.

~~Next~~-gen Sequencing now

www.illumina.com/sequencing?gt

SEQUENCING
GENOTYPING
GENE EXPRESSION

illumina®