2 NGS read mapping

This exposition has been developed by Knut Reinert. It is based on the following sources, which are all recommended reading:

1. Li, H, and Homer, N. (2010) *A survey of sequence alignment algorithms for next-generation sequencing*. Briefings in Bioinformatics 11 (5) (September 21): 473-483.
2. Holtgrewe, M., Emde A.-K., Weese D., Reinert K. (2011) *A Novel And Well-Defined Benchmarking Method For Second Generation Read Mapping.* BMC Bioinformatics 12 (1): 210.

The term *read mapping* has itself established since a couple of years for another, well studied problem, namely *approximate string matching* with certain application driven constraints.

The constraints are:

- usually DNA (or RNA) is considered (which means a *small alphabet* size).
- we have to map *short* strings (about 50 to 3000 bases) to a large string (billions of bases).
- there are relatively *few errors* allowed (usually around 3-4%, some application might go up to 10%).
- the problem sizes are very large (*billions* of small strings map to a string of size up to several billion characters).

2.1 Second-generation sequencing technologies

	454 FLX/Roche	Solexa/Illumina	SOLiD/ABI
Sequencing approach	pyrophosphate release	bridge amplification	ligation
Read lengths	400–500bp	36bp	35bp or 25bp (MP)
Mate pairs	yes	yes	yes
Output/Run	400–600Mbp in 10h	$>$ 1.5Gbp in 2.5d	3–4Gbp in 6d
Accuracy depends on	homopolymer length ($>$ 6 problematic)	nucleotide position in the read	nucleotide position in the read

GS FLX Titanium Series

Genome Analyzer 2

SOLiD System 2.0 Analyzer

The last slide was "old".

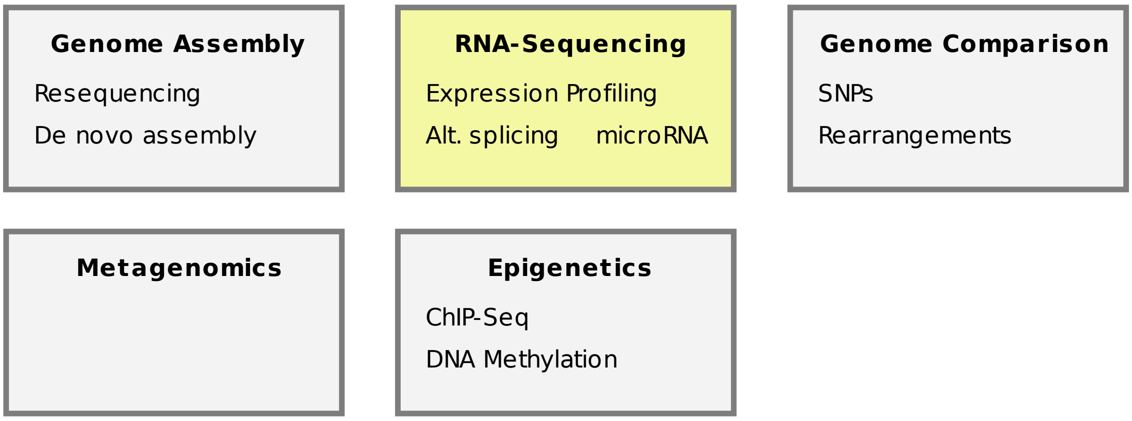
Illuminas HiSeq 2500 now produces at least 600 Gbp in about 12 days. That is about one billion reads, of length 100-150 bp in mate pairs. In addition, the end of higher throughput does not seem to be reached.

Also, new technologies allow the sequencing of *single molecules*.

What are the applications for which these technologies are currently used?

2.2 Second-generation sequencing applications

Here are some of them:



Genome Assembly

Resequencing

De novo assembly

RNA-Sequencing

Expression Profiling

Alt. splicing microRNA

Genome Comparison

SNPs

Rearrangements

Metagenomics

Epigenetics

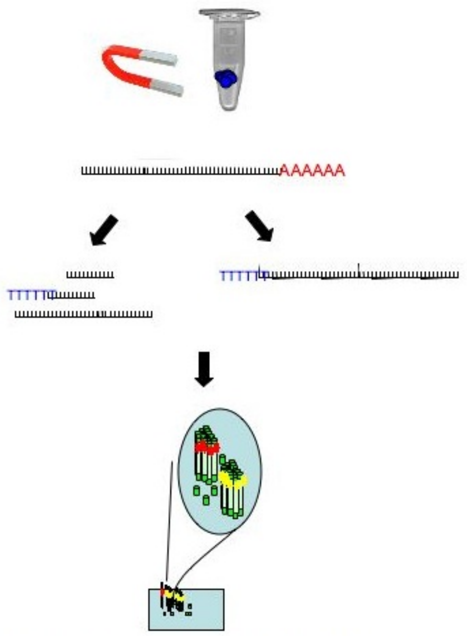
ChIP-Seq

DNA Methylation

2.3 RNA-Sequencing

How RNA-Seq works:

- RNA isolatation
- Reverse transcription to cDNA
- Fragmentation
- (Size selection)
- Sequencing



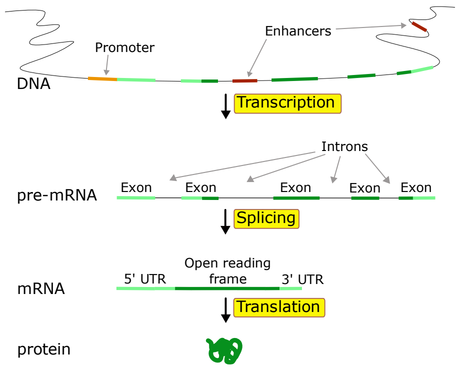
Illumina Solexa, Roche 454, or ABI SOLiD
Graphic shown here is Illumina

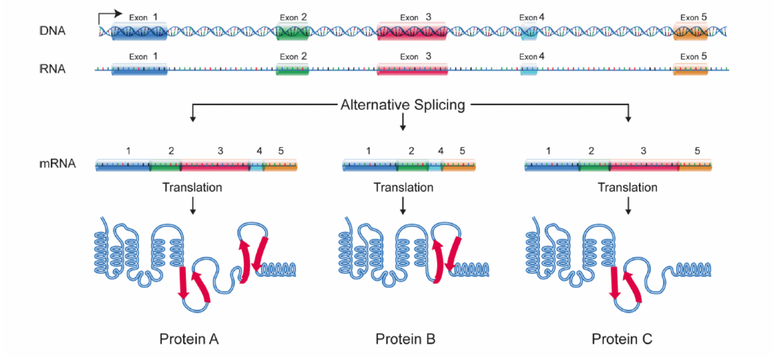
RNA-Seq applications:

- **Expression profiling:** Quantify gene expression levels
- **Alternative splicing:** Which mRNAs are generated from the same gene?
- **microRNA:** Where is the genomic source, which genes are regulated?

The first two applications share the problem, that the NGS reads do not constitute genomic DNA. Because of splicing, substrings of the read should occur consecutively on the genome divided by introns (or spliced out exons). Hence the algorithmic problem changes.

2.4 RNA-Seq - Alternative Splicing





Two approaches to determine splice variants:

1. Cut the genome at known splice sites and map mRNA reads onto combinations of merged genome fragments
2. Map as many mRNA reads as possible onto the genome and use coverage and known introns to detect new splice sites. Proceed as above.

Second Approach[a]:

- Map reads
- Assemble uniquely mapped reads
- Generate possible splices
- Try to map the non-uniquely mapped reads onto splices

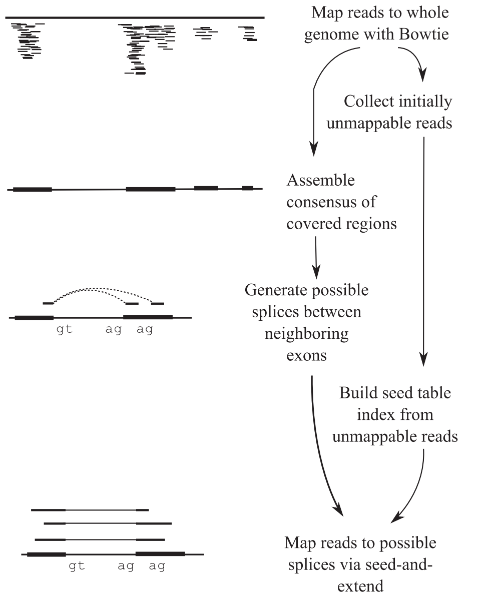
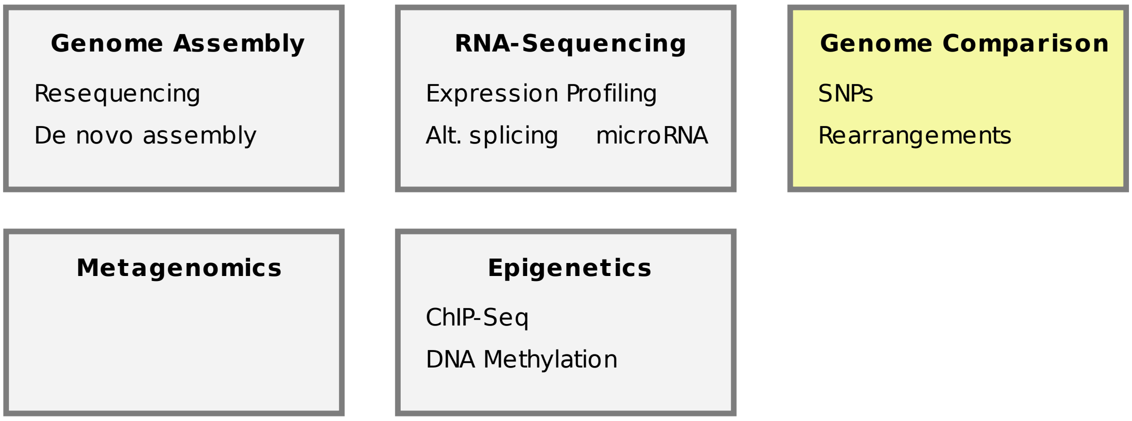

Map reads to whole genome with Bowtie
Collect initially unmappable reads
Assemble consensus of covered regions
Generate possible splices between neighboring exons
gt ag ag
Build seed table index from unmappable reads
Map reads to possible splices via seed-and-extend
gt ag ag

Fig. 1. The TopHat pipeline. RNA-Seq reads are mapped against the whole reference genome, and those reads that do not map are set aside. An initial consensus of mapped regions is computed by Maq. Sequences flanking potential donor/acceptor splice sites within neighboring regions are joined to form potential splice junctions. The IUM reads are indexed and aligned to these splice junction sequences.

[a] Trapnell C, Pachter L, Salzberg SL. (2009) *TopHat: discovering splice junctions with RNA-Seq*, Bioinformatics



Genome Assembly

Resequencing

De novo assembly

RNA-Sequencing

Expression Profiling

Alt. splicing microRNA

Genome Comparison

SNPs

Rearrangements

Metagenomics

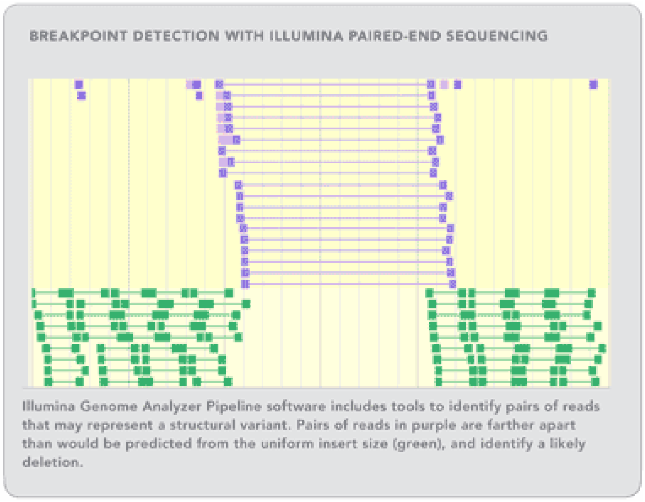
Epigenetics

ChIP-Seq

DNA Methylation

2.5 Genome Comparison

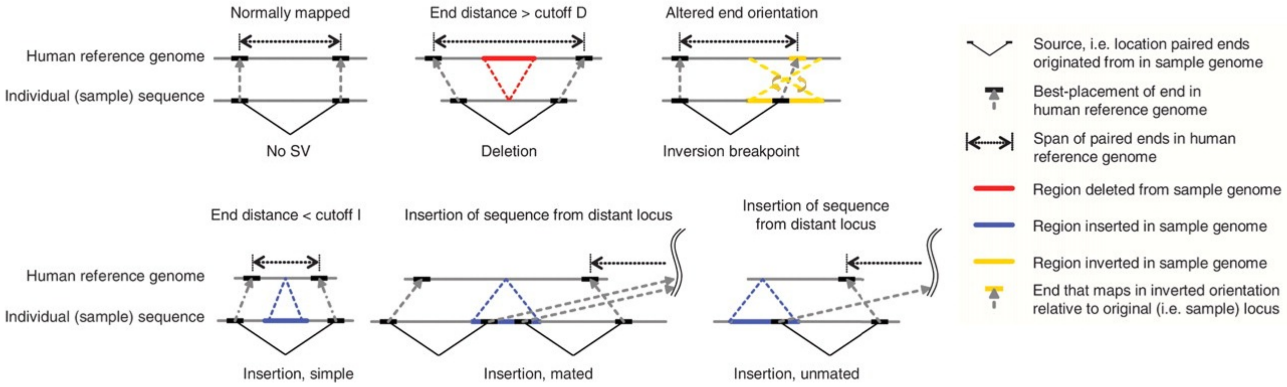
- Sequence paired-end reads of an unknown genome (sample)
- Map them onto a known reference genome (target)
- Search for small mutations (SNPs) or large structural variations (rearrangements) between them



BREAKPOINT DETECTION WITH ILLUMINA PAIRED-END SEQUENCING

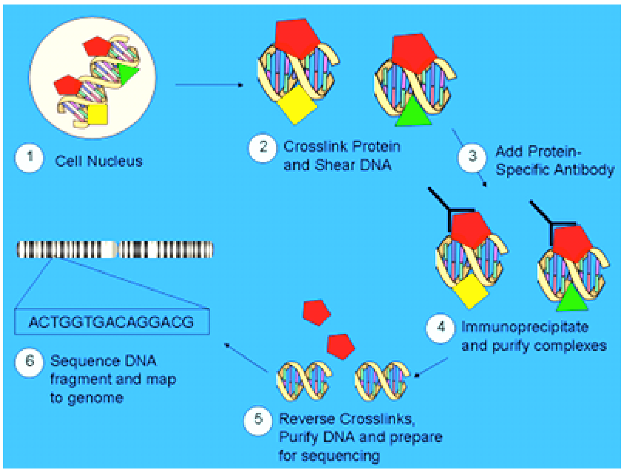
Illumina Genome Analyzer Pipeline software includes tools to identify pairs of reads that may represent a structural variant. Pairs of reads in purple are farther apart than would be predicted from the uniform insert size (green), and identify a likely deletion.

A deletion in the sample induces pairs of reads to be farther apart than predicted.

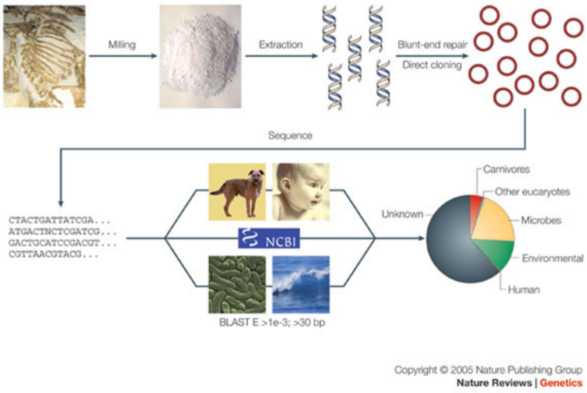

Normally mapped
End distance > cutoff D
Altered end orientation
Human reference genome
Individual (sample) sequence
No SV
Deletion
Inversion breakpoint
End distance < cutoff I
Insertion of sequence from distant locus
Insertion of sequence from distant locus
Human reference genome
Individual (sample) sequence
Insertion, simple
Insertion, mated
Insertion, unmated
Source, i.e. location paired ends originated from in sample genome
Best-placement of end in human reference genome
Span of paired ends in human reference genome
Region deleted from sample genome
Region inserted in sample genome
Region inverted in sample genome
End that maps in inverted orientation relative to original (i.e. sample) locus

Inversions, deletions, translocations can also be detected.[1,2]

2.6 Other applications


1 Cell Nucleus
2 Crosslink Protein and Shear DNA
3 Add Protein-Specific Antibody
4 Immunoprecipitate and purify complexes
5 Reverse Crosslinks, Purify DNA and prepare for sequencing
6 Sequence DNA fragment and map to genome
ACTGGTGACAGGACG

ChIP-Sequencing[3]



Copyright © 2005 Nature Publishing Group
Nature Reviews | Genetics

Metagenomics[4]

Fundamental to almost all of these applications is the following problem:

Problem 1 (Read Mapping Problem). Given a set of read sequences $\mathcal{R}$, a reference sequence G, and a distance $k \in \mathbb{N}$. Find all pairs (r, g) with $r \in \mathcal{R}$, g is substring of G and $\mathrm{dist}(r, g) \leq k$.

Common distance measures are Hamming distance or (weighted) edit distance.
The pairs (r, g) are called **matches** of r.

However, depending on the application, we have to adapt the problem definition.

2.7 Objective functions

By now it should be clear to you, that the term read mapping subsumes a number of different objective functions. In the normal case we want to find the approximate occurrences of a complete read, that is, conduct a *semi-global* alignment.

If we have for example RNA reads, then the read may corresponds to several genomic loci that have been spliced together. In this case we speak of *split* alignment to distinguish it from local alignment. In split alignment we want to find the *complete* read, whereas this is not necessary in local alignment.

The problem of split alignment can be further subdivided depending on the decision whether we allow parts of the split read to be reverse complemented (e.g. assembly error), be missing, or out of order (e.g. genomics insertions or deletions).

Usually split read mapping is reduced to several subproblems of normal read mapping.

Finally, a distinction is made whether the approximate string matching supports (weighted) edit distance or only the Hamming distance.

While the edit distance is preferable, it makes the problem computationally harder. Often you will find in read mapping heuristics some "in-between" formulations (e.g. *supports mismatches and up to 2 insertions*).

Be aware of such limitations.

[1] Korbel JO, Urban AE, Affourtit JP, et al. (2007) *Paired-End Mapping Reveals Extensive Structural Variation in the Human Genome*, Science

[2] Bashir A, Volik S, Collins C, Bafna V, Raphael BJ. (2008) *Evaluation of paired-end sequencing strategies for detection of genome rearrangements in cancer*, PLoS computational biology

[4] Barski A, Cuddapah S, Cui K, Roh TY, Schones DE, Wang Z, Wei G, Chepelev I, Zhao K (2007) *High-resolution profiling of histone methylations in the human genome*, Cell

[4] Poinar HN, Schuster SC, et al. (2006) *Metagenomics to Paleogenomics: Large-Scale Sequencing of Mammoth DNA*, Science

If we have a fixed objective function for our special approximate string matching problem, we can still make distinctions about the set of matches we want to find. A reasonable distinction could be the tasks of finding:

1. *all* matches with up to *k* errors.
2. *all* best matches.
3. *any* best match.

Doing this of course implies to have a good definition, what we actually mean with a match?

Have a look at the following situation:

2.8 Benchmarking

```
 reference    C A G A C T C C C A A C T G T C A   · · ·   C A G A C T C C C C C C A A C T G

alignments              T C C C A A C                               T C C C - - - A A C
     ⋆                T - C C C A A C
     ⋆⋆                 T C C C A A - C
```

Different kind of approximate matches.

Say, we want to find the best two matches of the read in the reference sequence, with an edit distance of up to 3. Both locations in the reference sequence are shown. The row alignments shows two alignments of the read to the reference sequence that appear to be optimal. However, the alignments in the rows below have a lower edit distance than the right one.

Common sense would tell us that the alignments in the left column are not significantly different, though. Each alignment with distance k induces alignments with distance at most $k + 2$ by aligning the leftmost/rightmost base one more position to the left/right and introducing a gap.

Repeats are another issue. Consider the tandem repeats in the below figure.

```
reference   · · · C G A C C C A C C A C G A C C C A C C A C G A C C C A C

                  C G A C C C A C C A C G A C C C A C C A

                                      C G A C C C A C C A C G A C C C A C
```

Large period repeat.

Intuitively, we can identify the two distinct alignments in this situation. Now look at a tandem repeat with a shorter period:

```
reference   · · · C A A C A A C A A C A A C A A C A A C A A C A A C A A · · ·

                  C A A C A A C A A C A A
                        C A A C A A C A A C A A
                              C A A C A A C A A C A A
                                    C A A C A A C A A C A A
                                          C A A C A A C A A C A A
                                                C A A C A A C A A C A A
```

Small period repeat.

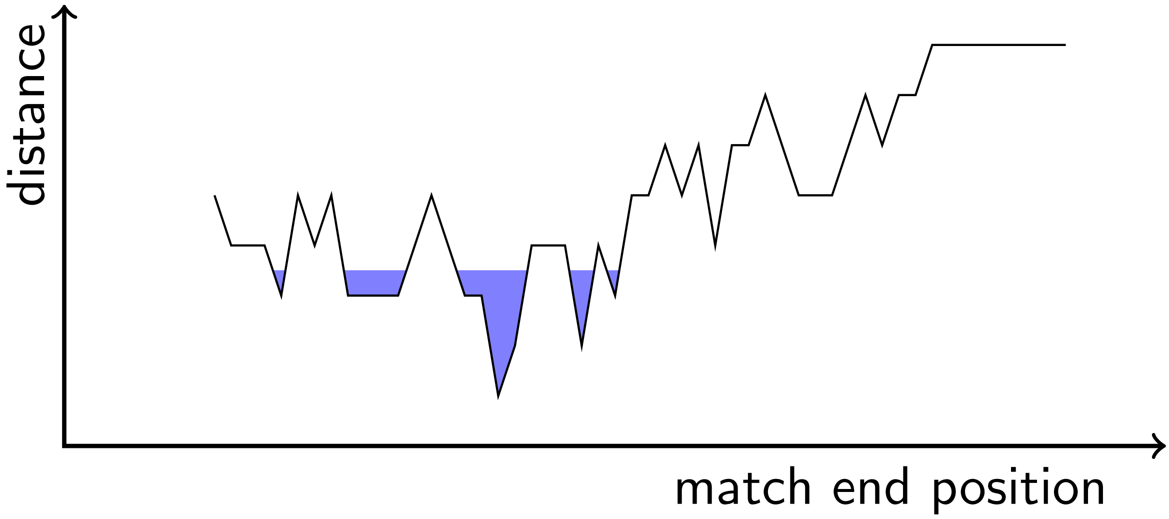
Do we really want to find all those alignments?

Counting alignments in this way would require a read mapper to find lots of positions in repeat regions. This is not desirable since reads from long tandem repeat regions would get a higher weight with this counting scheme than reads from short tandem repeat regions or reads from non-repeat regions.

Only weighting each found match with $1/n$ (where n is the number of positions the read aligns at) is also deficient (why?).

Hence it is more desirable to define when two matches are considered the same and when they should be counted separately.

Without giving the details, one can define an equivalence relation on the set of matches, which can be depicted as follows as an *error landscape*.



Error landscape.

Flooding this landscape to the respective error level gives a number of intervals, which in turn can be used to define the specificity and sensitivity of read mappers. In the benchmark it is sufficient to return one endposition within the interval.

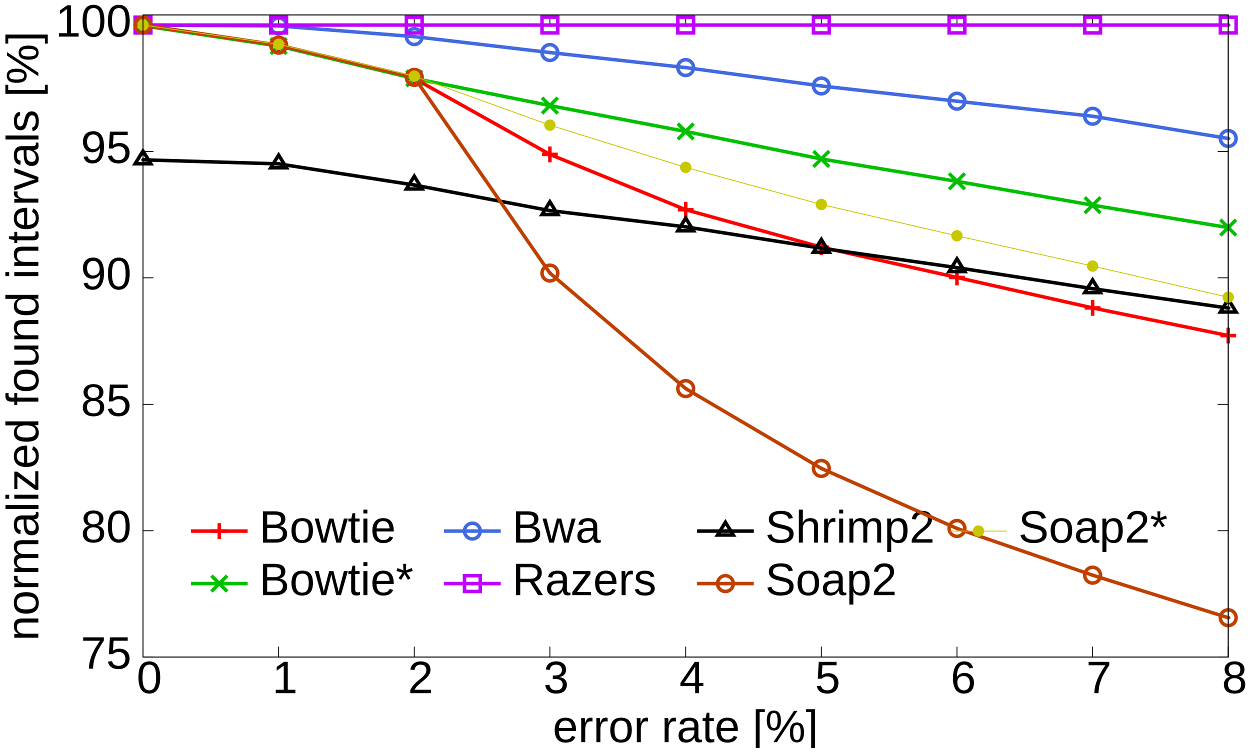
Some of the intervals will be labelled *optimal*, if they contain a matching position with the minimal distance (e.g. edit or hamming). If we benchmark read mapping application with the goal *find any best*, then it should return an alignment ending in one of those intervals. If we have the goal *find all best*, the read mapper should return all optimal intervals, etc.

This can now be used to make comprehensive comparisons between different methods to compare their performance.

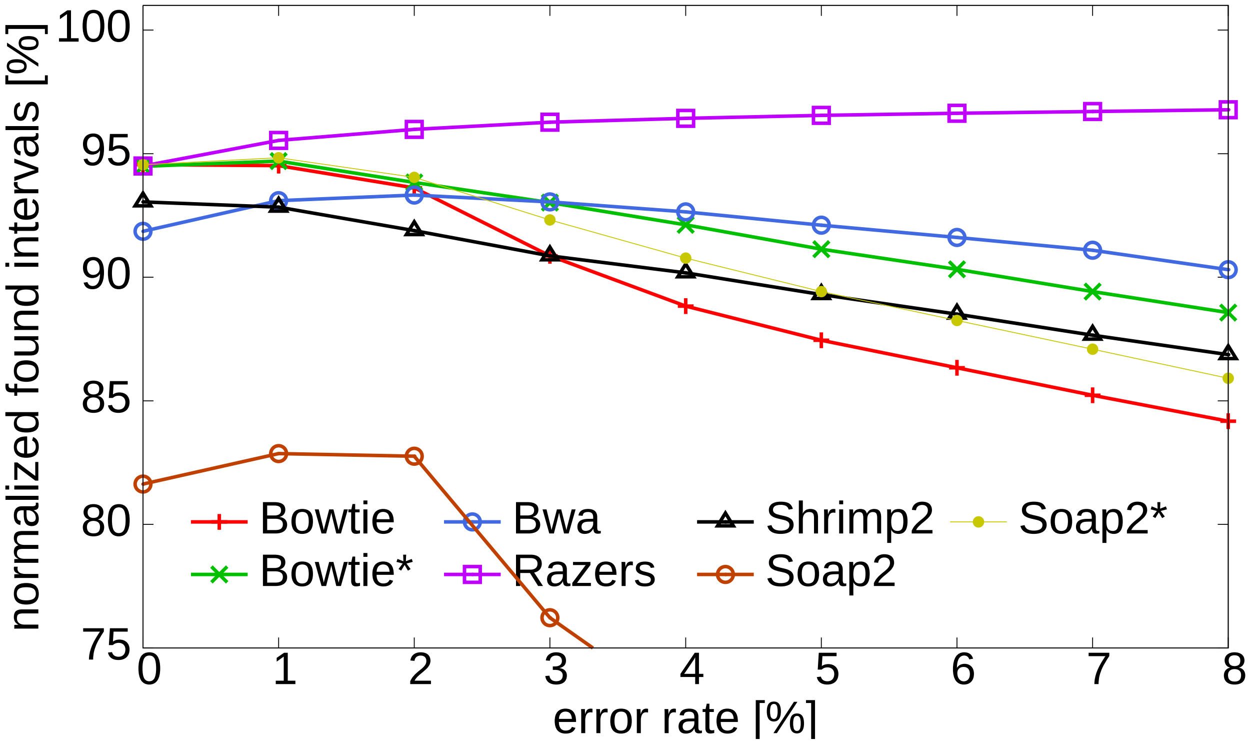
As an evaluation metric we use the number of *normalized found intervals*.

This is defined as follows: Each read gives at most one point. If a read matches at n locations (i.e. intervals), each found location gives $1/n$ point. To get percentages, the number of achieved points is divided by the number of reads and multiplied by 100.

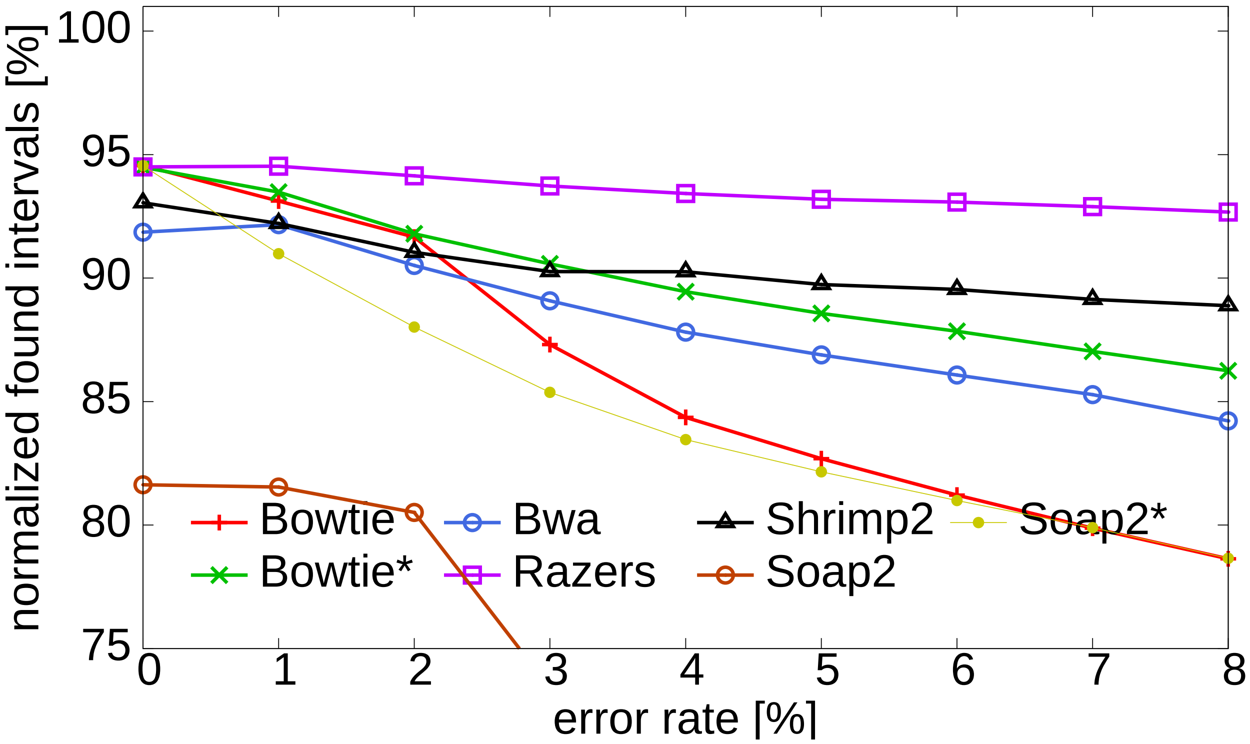
Have a look at results of recent read mappers (2011) for the three different categories (Illumina reads of *Drosophila Melanogaster*, 100 bp length), but mind that those plots do not give the run times.



Benchmark *any best*.



Benchmark *all best*.



Benchmark *all*.

If read mappers are benchmarked, you should be aware whether a program is made or configured to find only one (or a few) mapping locations, or whether it is made (or configured) to find *all* mapping locations.

In the first case we talk of a *best mapper*, in the second case we talk of an *all mapper*. Depending on the algorithms and heuristics used, a program which is a good best mapper might not perform well as an all mapper, or vice versa. Also, the run times might change significantly.

Some read mappers take quality values into account, either already in the filtering phase, or in the verification phase. If a difference between reference genome and read is caused by a sequencing error, this strategy should help in finding the correct location more often.

However, if the difference is caused by a real genomics difference (e.g. a SNP) the quality score does not help.

In the following I present you some benchmark results taking into account the classification of read mappers into best and all mappers, the presented Rabema benchmark, and a benchmark using simulated SNPs.

The following table shows a current comparison of read mappers using the Rabema classes. The small numbers show the percentages for 0%, 1%, 2 %, 3%, and 4% error rate.

	method	all		all-best		any-best		recall	
best-mappers	Bowtie 2	92.04	99.18 98.72 96.80 93.44 81.94 40.19	96.16	97.79 97.85 95.80 94.83 93.37 88.86	98.08	100.00 99.96 97.55 96.62 94.93 90.46	95.94	98.01 97.72 95.55 94.24 92.79 89.52
	BWA	92.18	99.18 98.72 97.81 94.25 80.92 37.65	96.81	97.79 97.87 97.88 96.59 92.63 83.47	98.81	100.00 99.95 99.81 98.55 94.28 85.37	96.41	97.93 97.69 97.25 95.77 91.98 84.61
	Soap 2	65.93	99.18 95.55 91.34 8.67 0.70 0.00	69.89	97.79 94.74 91.37 8.98 0.79 0.00	71.37	100.00 96.78 93.18 9.21 0.81 0.00	69.91	98.05 94.62 91.20 11.85 1.41 0.36
	R3-100	93.30	99.18 98.73 97.93 95.60 85.81 44.15	97.96	97.79 97.88 98.03 98.00 98.27 97.93	100.00	100.00 100.00 100.00 100.00 100.00 100.00	97.80	98.00 97.85 97.75 97.65 97.70 97.69
	R3-95	93.10	99.18 98.73 97.93 95.49 84.76 42.82	97.75	97.79 97.88 98.03 97.88 97.03 94.97	99.79	100.00 100.00 100.00 99.89 98.74 97.00	97.60	98.03 97.85 97.74 97.52 96.56 94.99
all-mappers	Bowtie 2	95.69	99.98 99.91 99.45 97.99 90.69 55.14	98.85	99.74 99.79 98.61 98.21 97.55 93.84	99.16	100.00 99.98 99.01 98.63 97.94 94.17	98.54	99.74 99.58 98.27 97.64 96.87 94.40
	BWA	95.89	99.96 99.88 99.49 97.13 87.79 64.11	97.98	98.81 99.01 99.02 97.83 93.95 85.20	98.82	100.00 99.95 99.82 98.56 94.34 85.37	97.80	99.03 98.96 98.75 97.35 93.43 86.36
	Hobbes	96.56	99.41 99.00 98.76 97.80 93.20 73.05	97.08	97.23 96.59 97.01 97.38 98.16 97.42	98.01	97.92 97.51 97.96 98.43 99.12 98.46	96.41	95.49 95.84 96.54 97.03 97.98 97.79
	mrFAST	99.97	100.00 100.00 100.00 100.00 99.99 99.53	99.97	100.00 100.00 100.00 100.00 100.00 99.10	99.97	100.00 100.00 100.00 100.00 100.00 99.13	99.97	100.00 100.00 100.00 99.99 100.00 99.18
	SHRiMP 2	96.53	99.87 99.82 99.53 98.37 92.58 64.63	99.50	99.34 99.50 99.60 99.64 99.65 98.32	99.85	99.87 99.90 99.91 99.89 99.84 98.57	99.25	99.35 99.30 99.24 99.30 99.09 98.48
	R3-100	100.00	100.00 100.00 100.00 100.00 100.00 100.00	100.00	100.00 100.00 100.00 100.00 100.00 100.00	100.00	100.00 100.00 100.00 100.00 100.00 100.00	100.00	100.00 100.00 100.00 100.00 100.00 100.00
	R3-95	99.54	100.00 100.00 100.00 99.89 98.67 95.11	99.79	100.00 100.00 100.00 99.89 98.71 96.96	99.79	100.00 100.00 100.00 99.89 98.74 97.00	99.79	100.00 100.00 100.00 99.89 98.77 97.17

The following table shows a current comparison of read mappers and their run times on real data sets (10 million Illumina reads) (left) and simulated long reads (right).

	dataset	SRR497711 D. melanogaster			ERR012100 H. sapiens			simulated, $m = 800$ D. melanogaster		
	method	time [min:s]	correctly mapped reads [%]	mapped reads [%]	time [min:s]	correctly mapped reads [%]	mapped reads [%]	time [min:s]	correctly mapped reads [%]	mapped reads [%]
best-mappers	Bowtie 2	2:00	99.65 100.00 99.77 99.02 97.65 94.92	85.71 52.08 67.27 73.62 76.88 78.81	5:37	99.62 100.00 99.75 96.02 92.88 87.86	96.72 75.99 87.81 90.54 91.85 92.76	13:48	96.73 97.47 99.67 98.05 87.95 85.17	99.99 0.03 41.07 73.95 82.31 90.03
	BWA	5:35	98.96 100.00 99.57 98.40 90.72 82.08	79.37 52.08 67.24 73.57 76.62 78.31	13:45	99.66 100.00 99.50 98.01 93.39 88.92	93.53 75.99 87.78 90.59 91.91 92.82	5:38	74.96 97.47 98.62 82.47 0.00 0.00	68.09 0.03 40.61 68.09 68.09 68.09
	Soap 2	1:55	91.78 100.00 96.24 89.35 0.09 0.02	72.49 52.08 66.73 72.48 72.49 72.49	2:34	96.45 100.00 94.94 86.54 0.32 0.16	89.73 75.99 87.24 89.73 89.73 89.73	0:54	41.21 97.47 67.99 28.10 0.22 0.00	38.14 0.03 28.17 37.88 38.14 38.14
	R3-100	1:28	100.00 100.00 100.00 100.00 100.00 100.00	78.92 52.08 67.31 73.69 76.97 78.92	85:56	100.00 100.00 100.00 100.00 100.00 100.00	92.99 75.99 87.84 90.67 92.02 92.99	1:17	100.00 100.00 100.00 100.00 100.00 100.00	90.43 0.03 41.13 74.13 82.65 90.43
	R3-95	1:26	99.87 100.00 100.00 100.00 99.11 96.34	78.82 52.08 67.31 73.69 76.94 78.82	43:16	99.96 100.00 100.00 100.00 99.30 96.92	92.95 75.99 87.84 90.67 92.01 92.95	1:15	100.00 100.00 100.00 100.00 100.00 100.00	90.43 0.03 41.13 74.13 82.65 90.43
all-mappers	Hobbes	4:51	96.49 96.55 96.46 96.94 96.28 93.86	76.16 50.28 64.98 71.16 74.33 76.16	265:48	95.97 95.94 96.14 96.39 96.10 94.63	89.24 72.90 84.30 87.02 88.33 89.24	–	–	–
	mrFAST	4:01	100.00 100.00 100.00 100.00 100.00 100.00	78.92 52.08 67.31 73.69 76.97 78.92	413:40	100.00 100.00 100.00 100.00 100.00 100.00	92.99 75.99 87.84 90.67 92.02 92.99	5:16	65.25 93.14 95.65 59.59 0.00 0.00	69.32 0.03 39.34 69.32 69.32 69.32
	SHRiMP 2	23:40	99.83 99.99 99.99 99.74 98.71 96.33	89.91 52.07 67.30 73.66 76.92 78.83	1312:09	99.81 99.89 99.83 99.39 98.29 96.81	99.06 75.90 87.74 90.56 91.91 92.87	796:06	95.70 97.47 99.75 97.60 82.95 80.14	99.31 0.03 41.04 73.67 81.63 89.14
	R3-100	1:51	100.00 100.00 100.00 100.00 100.00 100.00	78.92 52.08 67.31 73.69 76.97 78.92	118:26	100.00 100.00 100.00 100.00 100.00 100.00	92.99 75.99 87.84 90.67 92.02 92.99	1:20	100.00 100.00 100.00 100.00 100.00 100.00	90.43 0.03 41.13 74.13 82.65 90.43
	R3-95	1:45	99.87 100.00 100.00 100.00 99.11 96.34	78.82 52.08 67.31 73.69 76.94 78.82	58:13	99.96 100.00 100.00 100.00 99.30 96.92	92.95 75.99 87.84 90.67 92.01 92.95	1:20	100.00 100.00 100.00 100.00 100.00 100.00	90.43 0.03 41.13 74.13 82.65 90.43

For single-end reads the table show the percentages of found origins (recall) and fraction of unique reads mapped to their origin (precision) grouped by reads with s SNPs and i indels (s, i).

In this table you can see that read mappers have a different behaviour depending on the kind of error they encounter (SNPs vs sequencing error).

		(0,0)		(1,0)		(2,0)		(3,0)		(4,0)		(1,1)		(1,2)		(0,3)		(0,4)	
	method	prec.	recl.	prec.	recl.	prec.	recl.	prec.	recl.	prec.	recl.	prec.	recl.	prec.	recl.	prec.	recl.	prec.	recl.
best-mappers	Bowtie 2	97.6	97.3	95.6	94.8	94.6	92.0	93.3	88.7	92.6	82.5	95.3	93.3	93.5	92.3	96.1	95.4	97.6	97.4
	BWA	98.2	97.9	97.1	96.4	97.6	95.3	96.5	90.2	94.9	85.1	97.4	90.9	97.1	80.3	96.3	66.5	97.5	67.1
	Soap 2	98.1	82.9	97.0	63.6	97.4	31.0	0.0	0.0	0.0	0.0	90.6	6.2	0.0	0.0	0.0	0.0	0.0	0.0
	R3-100	98.4	98.4	97.7	97.7	98.2	98.2	97.5	97.5	96.3	96.3	98.1	98.1	97.9	97.9	97.6	97.6	98.4	98.4
	R3-99	98.4	98.4	97.7	97.7	98.2	98.0	97.4	96.6	96.2	95.1	98.2	98.1	97.9	97.9	97.6	97.6	98.4	98.4
	R3-95	98.4	98.3	97.7	97.5	98.2	97.3	97.5	94.9	96.1	91.7	98.2	97.6	97.9	97.6	97.5	97.5	98.4	98.4
all-mappers	Hobbes	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	100.0	100.0	100.0	99.8	100.0	93.6	99.6	90.5	99.6	87.6
	mrFAST	100.0	99.9	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	99.3
	SHRiMP 2	100.0	99.4	100.0	99.5	100.0	99.7	100.0	99.9	100.0	99.7	100.0	99.5	100.0	99.2	100.0	99.6	100.0	99.6
	R3-100	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	R3-99	100.0	100.0	100.0	99.9	100.0	99.8	100.0	99.1	100.0	98.9	100.0	99.9	100.0	100.0	100.0	100.0	100.0	100.0
	R3-95	100.0	99.9	100.0	99.7	100.0	99.0	100.0	97.3	100.0	95.4	100.0	99.4	100.0	99.6	100.0	99.9	100.0	100.0

2.9 Computational paradigms

Lets go back to algorithmic paradigms used in read mapping algorithms.

Given the large data, obviously all algorithms use some *string indices* to preprocess the reads, the genome, or both. The indices can be used directly for searching as in the case of the enhanced suffix array or Burrows Wheeler transform (BWT), or they are used to filter out regions that do not contain matches (as in the case of (gapped) q-gram indices).

You have already encountered a simple filter that is based on a q-gram index and uses a simple version of the q-gram lemma. This paradigm is called *q-gram counting*.

In the following lectures we will talk about a q-gram based pigeonhole filter and hierarchical verification scheme introduced by Navarro and about a fast bit-vector based verification algorithm by Myers.