QIAGEN

White Paper

White paper on de novo assembly in CLC Assembly Cell *4.0*

February 6, 2012

Sample to Insight

QIAGEN Aarhus · Silkeborgvej 2 · Prismet · 8000 Aarhus C · Denmark
Telephone: +45 70 22 32 44 · www.qiagenbioinformatics.com · support-clcbio@qiagen.com

Contents

1 Introduction

This is a white paper on the de novo assembler in CLC Assembly Cell *4.0*. Note that the same algorithm is used by *CLC Genomics Workbench* and *CLC Genomics Server*, and except for the performance benchmarks (speed and memory), this white paper applies to these products as well.

Sequencing technologies are in continuous development with new technologies emerging, improvements in the sequencing quality and rapidly increasing amounts of sequencing data. Combined with the fact that we for the last couple of years have seen a significant increase in the number of researchers involved in next generation sequencing projects, this creates a need for de novo assemblers which can handle extremely large amounts of data and solve the complex task of constructing de novo assemblies from short read data. Thus, a continuous development of de novo assemblers is crucial in order to benefit from the improved quality and quantity of sequencing data which are becoming available today.

We are continuously developing the CLC de novo assembler and CLC Assembly Cell version 4.0 is an improvement over previous versions with better support for large data sets and integrated scaffolding for joining contigs based on paired reads information. It is designed to accept a combination of data from Illumina, 454, SOLiD, Ion Torrent and Sanger sequencing as a mix of paired and unpaired reads which allows the qualities of different sequencing technologies to be exploited.

The first part of this paper introduces the basics of de novo assembly as it is done in the CLC de novo assembler while the second part presents the results of performance benchmarks. The benchmarks cover de novo assembly of the following three genomes: *E. coli*, *Arabidopsis thaliana* and *Homo sapiens*. The results of these demonstrate the CLC assembler's ability to produce accurate assemblies extremely fast using both standard consumer hardware and server systems. For example, the assembly of a high coverage dataset for the *A. thaliana* genome is completed in just half an hour on a quad-core laptop, while an assembly of the human genome using a 43x coverage dataset takes just seven hours on a quad-processor system.

2 The CLC de novo assembler

CLC bio's de novo assembly algorithm utilize *de Bruijn graph*s to represent overlapping reads which is a common approach for short read de novo assembly that allows a large number of reads to be handled efficiently [Zerbino and Birney, 2008, Li et al., 2010, Gnerre et al., 2011].

The first step in the CLC assembler is to construct a *word table* containing all unique subsequences (*words*) of a certain length (*word size*) found in the reads. Given a word in the word table, all potential neighboring words can then be looked up as shown in Fig. 1.

Backward neighbors	**Starting word**	**Forward neighbors**
AACGTAGCTAGCGCAT		CGTAGCTAGCGCATGA
CACGTAGCTAGCGCAT	ACGTAGCTAGCGCATG	CGTAGCTAGCGCATGC
GACGTAGCTAGCGCAT		CGTAGCTAGCGCATGG
TACGTAGCTAGCGCAT		CGTAGCTAGCGCATGT

Figure 1: *The word in the middle is 16 bases long, and it shares the 15 first bases with the backward neighboring word and the last 15 bases with the forward neighboring words.*

Typically, only one of the backward neighbors and one of the forward neighbors will be present in the table. A graph can then be made where nodes represent words in the table and edges connect neighboring nodes. This is called a de Bruijn graph and building this graph is the second step in the CLC assembler.

For genomic regions without *repeats* or sequencing errors, the graph contains long linear stretches of connected nodes where each node has one backward and one forward neighbor. Such nodes can be reduced to a single node representing subsequences longer than the initial words which reduces the size of the graph. Figure 2 shows an example where the graph contains one node with two forward neighbors.

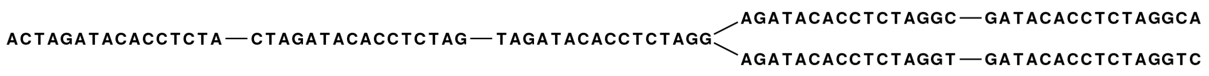

Figure 2: *Three nodes connected, each sharing 15 bases with its neighboring node and ending with two forward neighbors.*

A reduction of this graph reduces the number of nodes from seven to three as shown in Fig. 3, where the first node now represents 18 bases while the following two nodes each represent 17 bases. Note that neighboring nodes still have an overlap of 15 nucleotides since the word length is 16.

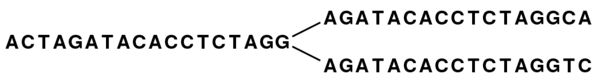

Figure 3: *The result of merging five nodes into three nodes.*

Given this way of representing the de Bruijn graph for a set of reads, we can consider some different situations: If reads contain polymorphisms (e.g. heterozygote SNPs) or sequencing errors, the resulting de Bruijn graph may contain *bubbles* as shown in Fig. 4. In this example the *bubble size* is 6 and contains two ambiguities. If the bubble was caused by sequencing errors occurring only once, one of the two paths through the bubble would originate from words observed only once in the reads. Conversely, if the bubble was caused by one or more polymorphisms, the words in both paths are likely to occur with similar frequencies in the reads. Thus, by storing the frequency of each word in the word table, it becomes possible to distinguish between sequencing errors and polymorphisms and create more accurate assemblies.

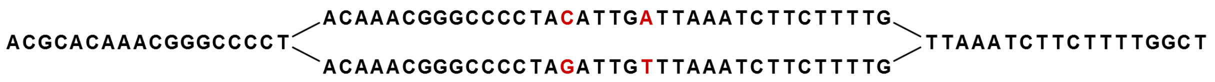
ACGCACAAACGGGCCCCT
ACAAACGGGCCCCTACATTGATTAAATCTTCTTTTG
ACAAACGGGCCCCTAGATTGTTTAAATCTTCTTTTG
TTAAATCTTCTTTTGGCT

Figure 4: *Bubble caused by SNPs or sequencing errors.*

If a repeat sequence occurs twice in a genome, the resulting graph will contain a node representing the repeat region with the flanking regions as neighbors. An example with a 28b repeat is shown in Fig. 5 where the length of the repeat is measured as the length of the central node plus the identical regions of the neighboring nodes. If the repeat had been shorter than 15b, it would not have shown up as a repeat since the word length is 16. This is an argument for using a large word size, since longer words are able to span longer repeat regions.

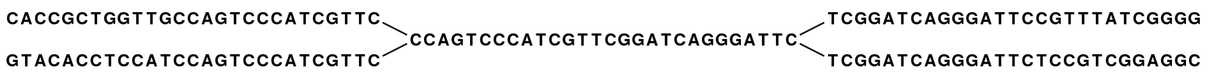
CACCGCTGGTTGCCAGTCCCATCGTTC
GTACACCTCCATCCAGTCCCATCGTTC
CCAGTCCCATCGTTCGGATCAGGGATTC
TCGGATCAGGGATTCCGTTTATCGGGG
TCGGATCAGGGATTCTCCGTCGGAGGC

Figure 5: *A de Bruijn graph where the central node represents a repeat region which occurs twice in a genome.*

The CLC assembler supports word sizes in the interval $[12, 64]$ ($[12, 24]$ on 32 bit machines) and a suitable word size will be chosen automatically based on the amount of input data: the more data, the larger the word size. It is also possible to manually set the word size which can help increase the assembly accuracy in some cases as discussed in Sec. 4.

A simple de novo assembly result can be constructed by outputting the sequence of each reduced node in the de bruijn graph. However, bubbles from sequencing errors and SNPs as well as repeat regions cause fragmentation of the result, i.e. the output will contain many short *contigs* (continuous genomic sequences) instead of a few large contigs. To improve the assembly result, the third step in the CLC assembler use reads to further reduce the graph by resolving sections that represents e.g. bubbles or repeats. Bubbles where one path has a significantly higher read coverage compared to other possible paths are resolved by choosing the path with the highest coverage. However, only bubbles where the bubble size is below a user-defined threshold will be resolved (the default threshold is 50b). Repeat regions can often be resolved if they are spanned by reads that have a significant overlap with two neighboring nodes. Thus, inclusion of long reads in a de novo assembly is essential for creating an accurate assembly with long contigs. If paired reads are available the CLC assembler will also attempt to resolve repeat regions that are spanned by paired reads. This is done by searching for a unique path through the repeat region containing paired reads where the distance between each read pair is within the expected distance interval. If a section cannot be resolved, contigs are created for each node in the section, thus the result of an assembly can contain small contigs with very similar sequences.

The fourth step of the CLC assembler performs *scaffolding* where paired reads spanning two contigs are used to both estimate the distance between the contigs and determine their orientation. Scaffolding is done using a greedy approach where the shortest gaps between contigs are closed first. By iteratively closing the shortest gaps, long gaps may be closed using multiple short gaps as shown in Fig. 6. After the scaffolding is completed, contigs in the same scaffold are output as one large contig with Ns inserted in between.

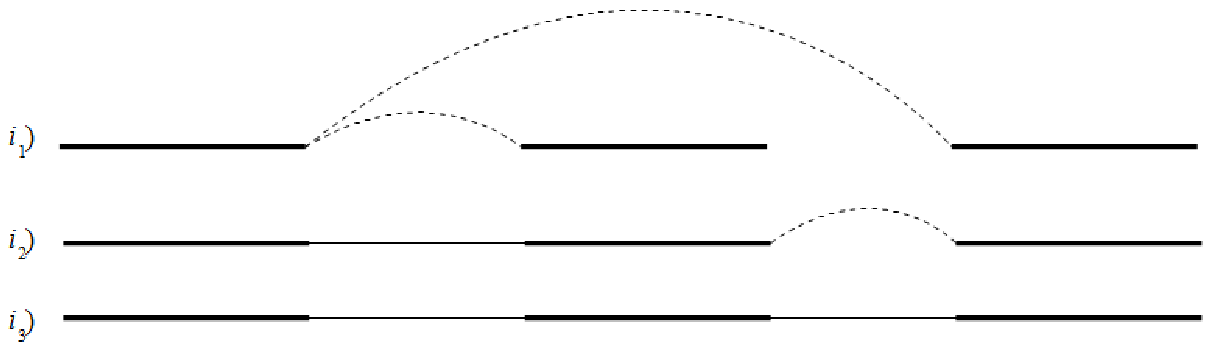

Figure 6: *Iterative scaffolding of the shortest gaps allows long pairs to be optimally used.* i_1) *shows three contigs with dashed arches indicating potential scaffolding.* i_2) *is after first iteration where the shortest gap has been closed and the long potential scaffolding has been updated.* i_3) *is the final result with three contigs in one scaffold.*

Repeat regions in large genomes are often very complex. One specific repeat may be found thousands of times and part of one repeat can also be part of another repeat. If a repeat is longer than both the read length and the distance between paired reads it becomes impossible to resolve the repeat region of the graph. This means that even with a high coverage of error free reads, the result of an assembly will still be fragmented if long repeat regions are present.

In summary, the de novo assembly algorithm performs the following steps:

- Build a table of the words seen in the reads.
- Create the de Bruijn graph using the word table.
- Use reads and paired read information to resolve the graph.
- Perform scaffolding.
- Output the resulting contigs and scaffolds.

3 SOLiD data support in de novo assembly

SOLiD sequencing is done in color space. When viewed in nucleotide space this means that a single sequencing error changes the remainder of the read. An example read is shown in figure 7.

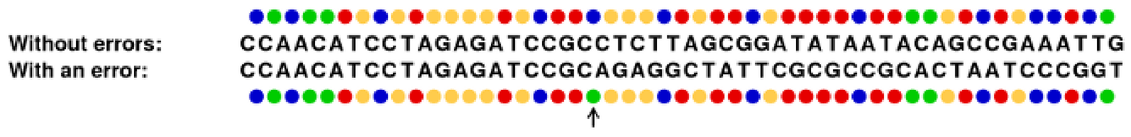
Without errors:
CCAACATCCTAGAGATCCGCCTCTTAGCGGATATAATACAGCCGAAATTG
With an error:
CCAACATCCTAGAGATCCGCAGAGGCTATTCGCGCCGCACTAATCCCGGT

Figure 7: *How an error in color space leads to a phase shift and subsequent problems for the rest of the read sequence*

Basically, this color error means that C's become A's and A's become C's. Likewise for G's and T's. For the three different types of errors, we get three different ends of the read. Along with the correct reads, we may get four different versions of the original genome due to errors. So if SOLiD reads are just regarded in nucleotide space, we get four different contig sequences with jumps from one to another every time there is a sequencing error.

Thus, to fully accommodate SOLiD sequencing data, the special nature of the technology has to be considered in every step of the assembly algorithm. Furthermore, SOLiD reads are fairly short and often quite error prone. Due to these issues, we have chosen not to include SOLiD support in the first algorithm steps, but only use the SOLiD data where they have a large positive effect on the assembly process: when applying paired information.

4 Quality trimming and parameter settings

Although the default parameter settings of the CLC assembler often result in a fairly accurate assembly, the accuracy can sometimes be improved by changing the default word size and bubble size parameters. Choosing optimal values for these two parameters require some experimentation, but the average coverage, length and quality of the input reads can often be used to guide such experiments as discussed below. Furthermore, to achieve a high accuracy assembly, it is also important to make use of paired read information (if available) and to reduce the number of sequencing errors in the reads using read quality trimming.

4.1 Paired read distance

Paired reads from the same library should ideally be separated by the same number of bases. However, in reality paired distances often have a substantial variance and it is therefore necessary to determine a distance interval in which the distances between the majority of the paired reads are contained. If the paired read distance is not properly chosen, paired read information may be ignored or even cause misassemblies.

4.2 Quality trimming of reads

Sequencing data are usually annotated with quality scores (phred scores in most cases) which indicate the probability of base-calling errors. Low quality reads, i.e. reads containing a large number of bases with low phred scores, often lead to misassemblies or ambiguous contigs and should be either trimmed or filtered out. The base quality will usually decrease towards the end of each read, and a common strategy is therefore to trim away low quality bases from read ends. The definition of a low quality base depends on the sequencing technology and it is often necessary to experiment with different quality thresholds to strike a balance between a high read quality and a high coverage.

4.3 Word size

A high read coverage combined with a low number of sequencing errors support a larger word size which will help resolve repeat regions. However, if reads contain a high number of sequencing errors, a large word size will increase the percentage of words that are affected by errors and hereby increase the complexity of the de Bruijn graph, thus a smaller word size is recommended in these cases. A small word size can also be an advantage if the read coverage is low or if the read length is short as it increases the size of the word table and hereby improves the chance of identifying overlapping words. Given a coverage of $\geq$30x and a read length of $\geq$50b the optimal word size will usually be in the range $[20, 30]$.

4.4 Bubble size

A high rate of systematic sequencing errors, which is for example often seen with 454 reads, may create large bubbles in the de Bruijn graph that stretch over hundreds of bases. The CLC assembler outputs a contig for each reduced node in bubbles larger than the bubble size, hence if a small bubble size is used for reads containing many systematic errors, the result will be a large number of small contigs. The bubble size can be increased manually which allows the assembler to resolve more bubbles, but a large bubble size can also increase the chance of misassemblies. For reads containing systematic errors, a bubble size equal to the average read length is a good starting point but some experimentation is required to find the optimal value.

5 Quality measures

In most cases, the ideal result of a de novo assembly is a few large contigs. A common way to measure assembly quality is therefore to compute statistics over the number and size of contigs which indicate how many reads the assembler was able to merge. We have used the following standard statistics to evaluate the results of the CLC de novo assembler.

#**contigs** The number of contigs produced. In most cases, a low number is preferred.

$\sum$ |**contig**| The total sum of bases in all contigs. Ideally, this number should be close to the expected size of the target sequences, i.e. the size of the target genome for whole genome sequencing.

N50 The N50 statistic is computed by first summing the sizes of contigs in the order largest to smallest until the sum is $\geq \frac{1}{2} \sum |\text{contig}|$. The N50 value is then the size of the contig that was last added to the sum, i.e. the smallest of the largest contigs covering 50% of the total size of all contigs. As N50 is fairly insensitive to a large number of small contigs it is considered a better measure of assembly quality than the average contig size.

When a reference genome is available contigs are - for the benchmarks in this white paper - aligned to the reference using NCBI BLAST 2.2.25 [Altschul et al., 1997] with default parameter settings. Following the approach in [Finotello et al., 2011] for evaluating de novo assembly results, we used the BLAST search results to compute the following quality measures:

HR Hypothetical Reconstruction: The ratio between the sum of bases in all contigs and the length of the reference.

RR Real Reconstruction: The ratio between the number of matching bases, and the reference length where bases in overlapping alignments are only counted once. For a perfect assembly the RR value should be 1, given that all reads map perfectly to the reference.

ER Erroneous Reconstruction: The difference between HR and RR. If all contigs align perfectly with the reference this value is 0.

Contigs of length $<$200b are ignored when computing the above quality measures and in cases where BLAST returned multiple hits, only the top hit for each contig was used.

6 De novo assembly of *Escherichia coli*

To assess the CLC assemblers performance on small datasets, we performed de novo assemblies for two strains of *E. coli* using high quality reads from three popular next generation sequencing platforms. Illumina [Illumina, 2011] and Ion Torrent reads [Technologies, 2011] from DH10B, a substrain of the K-12 strain with a 4.69Mb genome and 454 reads [Brzuszkiewicz et al., 2011] from O104:H4, an enterohemorrhagic *E. coli* strain from the German *E. coli* outbreak in 2011 with an estimated genome size of 5.31Mb.

Table 1 presents statistics for each of the three *E. coli* datasets. Both the Illumina and Ion Torrent datasets have a high read coverage, but where the Illumina reads are all paired, the Ion Torrent reads are both paired and unpaired. The 454 dataset consists entirely of long unpaired reads and has a relatively low read coverage compared to the Illumina and Ion Torrent datasets.

Platform	Illumina MiSeq	454 FLX	Ion Torrent
Strain	DH10B	O104:H4	DH10B
#Bases	2.18Gb	127Mb	990Mb
Avg. coverage	464x	23.9x	211X
Library type	Paired-end	Unpaired	Mate-pair (62%) Unpaired (38%)
Insert size	300b	N/A	10Kb
Avg. read length	151b	363b	140b

Table 1: E. coli *sequencing data statistics.*

6.1 Parameter settings and data preparation

To illustrate the effect of changing the default parameters of the CLC assembler as well as the use of read quality trimming in relation to assembly accuracy, we iteratively improve assemblies of the datasets described in Table 1 by optimising parameters. The results of these assemblies are presented in Table 2 where the results are grouped according to the parameter being optimized.

For the initial set of assemblies, we use default parameters and ignore paired read information and this results in numerous small contigs. HR values close to one for all datasets indicate that the produced contigs contains enough bases to match the size of the corresponding reference and for the Illumina and Ion Torrent datasets high RR values indicate that most of the reference has been correctly assembled. Furthermore, low ER values indicate a low number of contigs that are not mapped to the reference, only partly mapped or overlapping with another contig. However, the large number of contigs and the low N50 values show that the assemblies are fragmented due to unresolved sections in the de Bruijn graph.

Paired read information enables the assembler to resolve more sections in the de Bruijn graph, which results in fewer and longer contigs for both the Illumina and Ion Torrent dataset. Paired reads also makes it possible to perform scaffolding and in the case of the Ion Torrent dataset 716 contigs are reduced to 626 scaffolds with one scaffold covering 344Kb (scaffolding results are shown in parentheses). Because the Illumina data use a relatively short insert size, the reads can only span short gaps. Consequently, scaffolds for the Illumina dataset are relatively small compared to the size of the contigs produced.

In the next set of assemblies, low quality bases[1] have been trimmed from the read ends and reads where the trimming resulted in a read length of $<$20b have been removed. By trimming the Illumina reads, the assembler was able to find more overlapping reads and merge several contigs which results in larger N50 values. In case of the 454 dataset, quality trimming also enabled the assembler to merge several contigs which increases the N50 value by a factor of two. The total contig length of the 454 dataset is also reduced by 180Kb which could indicate that quality trimming eliminated a number of erroneous contigs. The Ion Torrent dataset contains a large number of reads where many bases have low quality scores. Quality trimming of the Ion Torrent reads therefore caused $>$40% of the bases to be removed even though a low quality threshold was used. However, after trimming this dataset, the assembly accuracy is improved considerably both in terms of N50 and ER. Notably, quality trimming allows the CLC assembler to produce a single scaffold which spans 2.95Mb and contains contigs with a total of 2.82Mb.

Adjusting the word size and the bubble size of the CLC assembler did not affect the assembly accuracy significantly for any of the datasets. However, because the 454 dataset contains systematic errors, adjusting the bubble size to 300 results in a significant increase of the N50 value and reduces the total contig length to 5.26Mb which is close to the expected length of 5.31Mb.

6.2 Combining datasets

It is possible to further improve the assembly of the DH10B genome by combining the Illumina and Ion Torrent reads in an assembly. The high quality Illumina reads are well suited for building an initial de Bruijn graph while the Ion Torrent reads are perfect for both resolving the graph and

[1]The quality threshold for each dataset was selected experimentally as the one which gave the best overall assembly quality.

Dataset	**#contigs**	$\sum$ |**contig**|, Mb	**N50**, Kb	**HR**, %	**RR**, %	**ER**, %
	Unpaired reads with default parameters					
Illumina	177	4.48	76.3	95.7	95.2	0.5
454 FLX	2,555	5.62	6.9	105.8	N/A	N/A
Ion Torrent	747	4.56	30.6	97.4	94.6	2.8
	Use of paired information					
Illumina	127 (112)	4.49 (4.49)	88.3 (107.6)	95.7	95.2	0.5
454 FLX	N/A	N/A	N/A	N/A	N/A	N/A
Ion Torrent	716 (626)	4.57 (4.69)	32.2 (129.7)	97.5	94.9	2.6
	Trimmed reads					
Illumina	113 (99)	4.48 (4.49)	97.0 (107.6)	95.7	95.1	0.6
454 FLX	1326	5.44	13.6	102.4	N/A	N/A
Ion Torrent	270 (136)	4.52 (4.66)	55.2 (2,950)	96.5	94.8	1.7
	Word size					
Illumina, 23⇒23	113 (99)	4.48 (4.49)	97.0 (107.6)	95.7	95.1	0.6
454 FLX, 20⇒22	1320	5.43	15.1	102.3	N/A	N/A
Ion Torrent, 21⇒21	270 (136)	4.52 (4.66)	55.2 (2,950)	96.5	94.8	1.7
	Bubble size					
Illumina, 50⇒50	113 (99)	4.48 (4.49)	97.0 (107.6)	95.7	95.1	0.6
454 FLX, 50⇒300	1002	5.26	25.8	99.1	N/A	N/A
Ion Torrent, 50⇒50	270 (136)	4.52 (4.66)	55.2 (2,950)	96.5	94.8	1.7

Table 2: *Results of assemblies using different parameter settings for the CLC assembler. The first group of results are from assemblies produced with the default parameter settings of the CLC assembler. The results in the following groups are from assemblies where parameters are improved iteratively.* $x \Rightarrow y$ *indicate that a parameter was changed from* x *to* y*. Where paired-read information is available, results for scaffolding are shown in parentheses.*

performing scaffolding due to the large insert size. To avoid sequencing errors in the Ion Torrent reads affecting the initial construction of the de Bruijn graph, we used the *-g* option of the CLC assembler to ignore these reads when performing step 1 and 2 of the assembly process (see Sec. 2). Quality trimming was performed for all reads and the CLC assembler was configured to use a word size of 26 and the default bubble size of 50. An assembly with these parameter produced 83 contigs and 26 scaffolds which is a significant reduction compared to the results in Table 2. Accordingly, the N50 values with and without scaffolding are improved to 136Kb and 2.45Mb respectively while RR and ER values are 95.6% and 0.73% respectively. The scaffolding N50 value is lower than the 2.95Mb achieved with the Ion Torrent dataset alone, however the better RR and ER values for the combined assembly indicate that a number of misasemblies are made when only the Ion Torrent reads are used. Hence, by combining the two datasets, the CLC assembler is able to produce long contigs while the number of misassemblies is minimised.

6.3 Running time and memory consumption

Table 3 shows time and memory consumption of the CLC assembler when assembling reads from the Illumina dataset on three different systems, including one laptop. On all systems, the assembly was completed in a few minutes using less than 1GB of memory. To avoid exhausting the memory bandwidth, only 40 of the 80 cores were used on the E7-4870 system.

CPU	i7 2720QM (laptop)	2x Xeon X5550	4x Xeon E7-4870
Cores	8	16	80
Threads used	8	16	40
Time used	5m 12s	4m 8s	2m 25s
Peak memory	129MB	287MB	778MB

Table 3: *Time and memory used by the CLC assembler on three different systems when assembling Illumina MiSeq reads from* E. coli DH10B. *The numbers include scaffolding of the contigs.*

7 De novo assembly of *Arabidopsis thaliana*

The relatively small diplod genome of *A. thaliana* is one of the most well studied model organisms in plant biology and therefore well suited for benchmarking the CLC de novo assembler on a plant genome. In a recent study [Gan et al., 2011] 18 different wild types of *A. thaliana* were sequenced with the Illumina Genome Analyser (GA) and assembled using a hybrid assembly strategy. The assembly of the reads was done by first mapping reads to the *A. thaliana* Col-0 reference genome followed by a de novo assembly of the reads using SOAPdenovo [Li et al., 2010] (another de Bruijn graph based assembler) from the SOAP package to resolve e.g. regions containing large indels.

Platform	Illumina GAII	Illumina GAIIx	Illumina GAIIx
#Bases	4.97Gb	3.21Gb	1.85Gb
Avg. coverage	42.2x	27.2x	15.7x
Library type	Paired-end	Paired-end	Unpaired
Insert size	200b	400b	N/A
Avg. read length	36b	51b	51b

Table 4: A. thaliana Edi-0 *sequencing data statistics.*

Using the dataset from [Gan et al., 2011], we have performed a de novo assembly of the *A. thaliana* Edi-0 strain with the CLC de novo assembler. Table 4 shows some statistics for the dataset which contains both paired and unpaired reads and has a 85.1x average read coverage of the 117.9Mb Edi-0 reference genome published in [Gan et al., 2011].

Quality trimming of the reads did not affect the assembly accuracy, and we therefore used all available reads as input for the CLC assembler. The high read coverage allowed the word size to be increased from the automatically chosen value of 24 to 26 while the bubble size was reduced to 40 due to the short read length. Table 5 presents the results for the assembly produced by the CLC assembler together with results for the SOAP assembly used in [Gan et al., 2011]. An alignment of the SOAP contigs against the reference was not performed as these contigs were used to infer the Edi-0 reference genome, hence such an alignment would lead to an overestimate of the SOAP assemblers accuracy.

The CLC assembler produces fewer and longer contigs compared to the SOAP assembler which indicates that the CLC assembler was able to resolve more regions of the de Bruijn graph. The total length of the contigs produced by the CLC assembler and SOAP corresponds to 90% and 99% of the Edi-0 reference genome, respectively. However, by default the CLC assembler discards all contigs shorter than 200b while SOAP only discard contigs shorter than 50b. By setting the minimum contig length to 50b for the CLC assembler, the total contig length is increased to 120Mb at the cost of including a massive number of small contigs. An ER value of 8.69% indicates that misassemblies have been made by the CLC assembler. however, as the

Assembler	**CLC**	**SOAPdenovo**
#**contigs**	26,606 (13,164)	322,757
$\sum$ |**contig**|, Mb	106.2 (108.1)*	117.0*
N50, Kb	14.7 (35.0)	1.9
HR, %	90.3	N/A
RR, %	81.6	N/A
ER, %	8.69	N/A

Table 5: *Results for de novo assembly of* A. thaliana *using CLC and SOAP de novo assemblers. Scaffolding results are shown in parentheses.*
Minimum contig length for CLC and SOAP were 200b and 50b respectively.

Edi-0 reference genome is an initial draft genome it is likely to contain errors which affect the ER value of the CLC assembler negatively.

The time and memory used by the CLC assembler to create contigs and perform the scaffolding of these is shown in Table 6. The assembly time is measured in minutes even on the quad-core laptop (i7 2720QM), hereby demonstrating that the CLC assembler is able to assemble fairly large genomes on standard consumer systems efficiently.

CPU	i7 2720QM (laptop)	2x Xeon X5550	4x Xeon E7-4870
Cores	8	16	80
Threads used	8	16	40
Time used	30m 7s	17m 41s	13m 16s
Peak memory	2.8GB	2.8GB	3.2GB

Table 6: *Performance of the CLC assembler on three different systems when assembling the* A. thaliana Edi-0 *genome using 10Gb of Illumina reads. All numbers include scaffolding of the contigs.*

8 De novo assembly of *Homo sapiens*

To assess the performance of the CLC assembler on a large genome, we assembled the genome of Hapmap [Gibbs et al., 2003] individual NA18507 (African male) using reads obtained from ENA [Leinonen et al., 2011] (accession no. ERA015743). As shown in Table 7, the dataset contains 135Gb of paired Illumina reads which corresponds to an average read coverage of 43.7x for the GRCh37 reference genome. The actual sequencing was done by Illumina and the dataset has not previously been used in any published de novo assembly experiments to our knowledge.

Platform	#**Bases**	**Avg. coverage**	**Library type**	**Insert size**	**Avg. read length**
Illumina GAII	135.3Gb	43.7x	Paired-end	300b	101b

Table 7: H. sapiens *sequencing data statistics.*

#**contigs**	$\sum$ |**contig**|*, Gb	**N50**, b	**HR**, %	**RR**, %	**ER** %
1,079,610 (985,587)	2.51 (2.52)	5717 (6256)	87	N/A	N/A

Table 8: *Results for de novo assembly of* H. sapiens *using and the CLC de novo assembler. Scaffolding results are shown in parentheses.*

Quality trimming of the reads was not performed because experiments showed that it did not improve the assembly accuracy of the CLC assembler due to the high quality of the Illumina reads (phred score of $\geq$Q20 for most bases). The default word size and bubble size were increased from 26 to 32 and from 50 to 80 respectively which gave a significant increase in assembly accuracy. The results of an assembly with these parameters are presented in Table 8. A total of 10^6 contigs were produced with a total length of 2.51Gb which corresponds to 87% of the known bases in the reference genome and 81% of the total reference genome length. Values for RR and ER are not shown because of the large computational resources needed for aligning all contigs to the reference using BLAST.

CPU	2x Xeon X5550	4x Xeon E7-4870
Cores	16	80
Threads used	16	40
Time used	11h 49m	7h 1m
Peak memory	45.3GB	46.9GB

Table 9: *Performance of the CLC assembler on two different systems when assembling a human genome. The numbers include scaffolding of the contigs.*

Table 9 shows the time consumption and maximum memory consumption of the CLC assembler when assembling all 135Gb reads for the NA18507 individual on two different systems. In comparison, a de novo assembly of the same individual using 146Gb paired Illumina reads with an average length of 36b, took 40h and required 140GB of memory with SOAP [Li et al., 2010] on a cluster consisting of eight quad-core 2.3Ghz CPUs. The same dataset has also been assembled using ABySS on a 168 core cluster which required 16GB of memory on each of the 23 machines in the cluster and took 3d 15h to complete [Simpson et al., 2009].

9 Conclusion

In this white paper we have introduced the CLC de novo assembler which is a part of the CLC Assembly Cell *4.0*. Benchmarks show that the computationally efficient algorithms used in the CLC assembler enable fast de novo assembly of high coverage bacterial and eukaryotic genomes using a laptop. Furthermore, an efficient parallelization scheme ensures full utilization of multi-core processors, thus enabling de novo assemblies of the human genome to be completed in seven hours using a single computer. The benchmark results also indicate that the accuracy of the CLC assembler is similar to or better than other state of the art de novo assemblers which are capable of handling the massive amounts of short read data produced by modern sequencing technologies.

References

[Altschul et al., 1997] Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402.

[Brzuszkiewicz et al., 2011] Brzuszkiewicz, E., Thürmer, A., Schuldes, J., Leimbach, A., Liesegang, H., Meyer, F.-D., Boelter, J., Petersen, H., Gottschalk, G., and Daniel, R. (2011). Genome sequence analyses of two isolates from the recent Escherichia coli outbreak in Germany reveal the emergence of a new pathotype: Entero-Aggregative-Haemorrhagic Escherichia coli (EAHEC). *Archives of microbiology*, 193(12):883–891.

[Finotello et al., 2011] Finotello, F., Lavezzo, E., Fontana, P., Peruzzo, D., Albiero, A., Barzon, L., Falda, M., Di Camillo, B., and Toppo, S. (2011). Comparative analysis of algorithms for whole-genome assembly of pyrosequencing data. *Briefings in Bioinformatics*.

[Gan et al., 2011] Gan, X., Stegle, O., Behr, J., Steffen, J., Drewe, P., Hildebrand, K., Lyngsoe, R., Schultheiss, S., Osborne, E., Sreedharan, V., and Others (2011). Multiple reference genomes and transcriptomes for Arabidopsis thaliana. *Nature*, 477(7365):419–423.

[Gibbs et al., 2003] Gibbs, R., Belmont, J., Hardenbol, P., Willis, T., Yu, F., Yang, H., Ch'ang, L., Huang, W., Liu, B., Shen, Y., and Others (2003). The international HapMap project. *Nature*, 426(6968):789–796.

[Gnerre et al., 2011] Gnerre, S., Maccallum, I., Przybylski, D., Ribeiro, F. J., Burton, J. N., Walker, B. J., Sharpe, T., Hall, G., Shea, T. P., Sykes, S., Berlin, A. M., Aird, D., Costello, M., Daza, R., Williams, L., Nicol, R., Gnirke, A., Nusbaum, C., Lander, E. S., and Jaffe, D. B. (2011). High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proceedings of the National Academy of Sciences of the United States of America*, 108(4):1513–8.

[Illumina, 2011] Illumina (2011). High coverage MiSeq sequencing data for E. coli DH10B. http://www.illumina.com/systems/miseq/ecoli.ilmn.

[Leinonen et al., 2011] Leinonen, R., Akhtar, R., Birney, E., Bower, L., Cerdeno-Tárraga, A., Cheng, Y., Cleland, I., Faruque, N., Goodgame, N., Gibson, R., Hoad, G., Jang, M., Pakseresht, N., Plaister, S., Radhakrishnan, R., Reddy, K., Sobhany, S., Ten Hoopen, P., Vaughan, R., Zalunin, V., and Cochrane, G. (2011). The European Nucleotide Archive. *Nucleic acids research*, 39(Database issue):D28–31.

[Li et al., 2010] Li, R., Zhu, H., Ruan, J., Qian, W., Fang, X., Shi, Z., Li, Y., Li, S., Shan, G., Kristiansen, K., Li, S., Yang, H., Wang, J., and Wang, J. (2010). De novo assembly of human genomes with massively parallel short read sequencing. *Genome research*, 20(2):265–72.

[Simpson et al., 2009] Simpson, J. T., Wong, K., Jackman, S. D., Schein, J. E., Jones, S. J. M., and Birol, . (2009). Abyss: A parallel assembler for short read sequence data. *Genome Res*, 19(6):1117–1123.

[Technologies, 2011] Technologies, L. (2011). High coverage Ion Torrent sequencing data for E. coli DH10B. http://www.iontorrent.com/.

[Zerbino and Birney, 2008] Zerbino, D. R. and Birney, E. (2008). Velvet: algorithms for de novo short read assembly using de Bruijn graphs. *Genome Res*, 18(5):821–829.