A Probabilistic Approach to Sequence Assembly Validation

Sun Kim*
School of Informatics
Indiana University
Sycamore Hall 339
Bloomington, IN 47405-7005
sun.kim@acm.org

Li Liao and Jean-Francois Tomb
Central Research and Development
DuPont Experimental Station, E328/240
Wilmington, DE 19880-0328
{Li.Liao, Jean-Francois.Tomb}@usa.dupont.com

ABSTRACT

Sequence assembly is an essential requirement for determining the complete sequence of long DNA. However, sequence assembly programs often generate misassembled contigs by either joining different repeat copies, resulting in joining non contiguous DNA regions (inverted or swapped) or by including many fragments from different repeat copies resulting in errors in the consensus sequence (noisy regions). Usually, sequence assemblies are experimentally validated. While this is the most reliable approach, it is time consuming and labor intensive. In this paper, we propose a probabilistic approach to identify possible misassembled regions in shotgun sequence assemblies. Based on the statistics using a set of randomly sampled patterns from shotgun data, a probability model that measures each fragment's contribution to misassembly is proposed. From the probability model, we compute entropy at each base position in contig assembly. Our approach correctly identified all misassembled regions in the assembly of the *Mycoplasma genitalium* genome from real shotgun sequence data. Furthermore, using this approach we identified many putative misassembled regions in the assemblies of bacterial genomes we are currently sequencing.

1. INTRODUCTION

Large scale sequencing projects employing a shotgun sequencing strategy consist of three phases: (1) sequencing short DNA fragments, (2) sequence assembly, and (3) post assembly processing. The sequence assembly phase generates *contigs*, a set of aligned DNA fragments. Although there have been substantial advances in the development of shotgun sequence assembly algorithms [5; 8; 9], sequence assembly programs often generate misassembled contigs by either joining different repeat copies, resulting in joining non contiguous DNA regions (inverted or swapped) or by including many fragments from different repeat copies resulting in errors in the consensus sequence (noisy regions).

Contig ordering and gap closure (between contigs) are achieved during post assembly processing phase. This phase is time consuming, labor intensive, and constitutes the bottle neck in genome projects which rely on the shotgun sequencing approach.

Thus, it is essential to validate the integrity of contigs prior to post assembly processing. Usually, sequence assemblies are experimentally validated. Experimental validation techniques include high clone coverage maps, multiple complete digest mapping, optical restriction mapping, and ordered shotgun sequencing [3]. Experimental sequence assembly validation can reliably check the integrity of contigs. Furthermore, experimental sequence assembly validation combined with a computer program can correctly identify positions of misassembled regions. Recently, Rouchka and States [3] proposed an approach to identifying misassembled regions by multiple restriction digest fragment coverage analysis. Their approach accurately identified misassembled regions by comparing restriction fragments deduced (computed) from the assembled contig to the experimentally determined restriction fragment pattern. While experimental sequence assembly validation is reliable, it requires extensive human intervention and has some limitations. For example, the multiple restriction digest fragment coverage analysis may have difficulty in misassembled regions for long inserts like BACs or YACs [3].

In this paper, we propose a probabilistic approach to identify possible misassembled regions in shotgun sequence assemblies without requiring any additional experimental intervention. Our approach is generally applicable to any sequence assembler that relies on overlapping regions between fragments – the most common approach to sequence assembly.

2. SHOTGUN SEQUENCING STRATEGY

In this section, we will briefly describe *the shotgun sequencing strategy*, which is being used for almost all large scale sequencing projects. The goal is to determine the sequence of a DNA that consists of *nucleotides* selected from four chemical bases, { `a, t, g, c` }. As DNA consists of two strands, the length of DNA is usually measured in base pairs, *bp* in short. The most common laboratory mechanism for reading DNA sequences, called *gel electrophoresis*, can determine sequence of up to approximately 10^3 nucleotides at a time. However, the sizes of genomes are much larger; for example, a human genome consists of approximately 10^9 nucleotides. To circumvent this limitation, the shotgun sequencing strategy is widely used.

The shotgun sequencing strategy includes cutting multiple copies of the target DNA at random positions, cloning the resulting peices into a vector, and reading sequences of short

*This work was done while the first author was working at DuPont Central Research and Development.

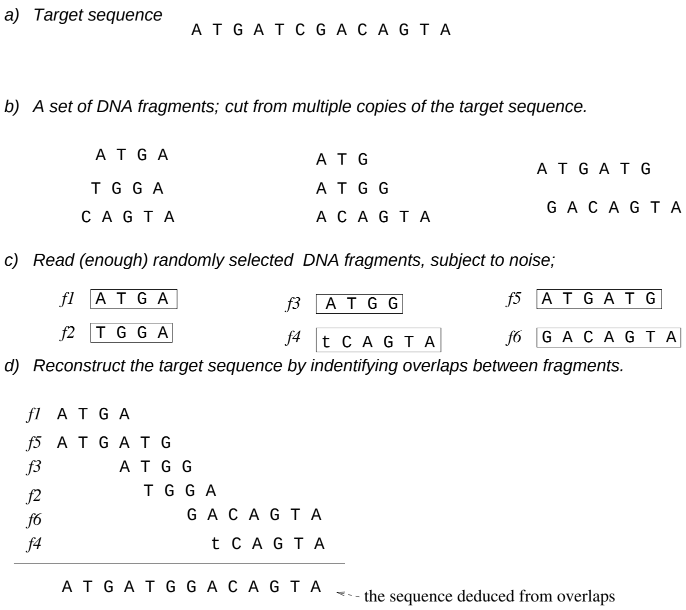

Figure 1: A sample shotgun sequencing. From a pool of shot DNA pieces (Step b), sequences of short DNA pieces are determined (Step c), which are input to a sequence assembly program. Then, a consensus sequence is deduced by aligning fragments based on overlaps between fragments (Step d).

DNA pieces from ends of randomly selected DNA pieces. The resulting data set is called *shotgun data* and short sequences reads are called *fragments*. To determine the sequence of a genome, it is necessary to assemble short fragments into long sequences based on sequence overlaps between fragments. This process is referred to as *sequence assembly* and the resulting assembled longer sequences are called *contigs*. Sequence assembly is perfromed by a computer program, called *sequence assembly program*. The complexity of genome sequencing is largely determined by the reliability of a sequence assembly program. Figure 1 illustrates a simplified shotgun sequencing and assembly procedure.

Sequence assembly is complex especially due to the existence of repetitive sequences, *repeats* in short, that occur multiple times at different positions of the DNA being sequenced. Repeats can be multiple occurrences of similar – not identical – sequences. For example, two substrings `ATGG` and `ATCGA` in the consensus sequence (Figure 1) are two similar sequences. Given that there are errors in fragment reads, it is difficult to distinguish true overlaps from false – repeat-induced – overlaps. For example, we may have to consider aligning $f1$ and $f3$ in Figure 1 since there is only one base difference in the overlapping region that may be due to an error in fragment reads. Misassemblies due to repeats frequently occur and are the main hurdle to achieve high throughput accurate sequencing. One intuitive way to check the correctness of sequence assembly is to look at the distribution of *coverage*. Coverage at a base position is defined as the number of fragments at that position. As fragments are randomly sampled, we expect that the distribution of coverage values centered around the average coverage. In Figure 1, Coverage values are either 2 or 3 where the average coverage is about 2.4. Several assemblers including TIGR Assembler [9] use coverage information to ensure correctness of assembly.

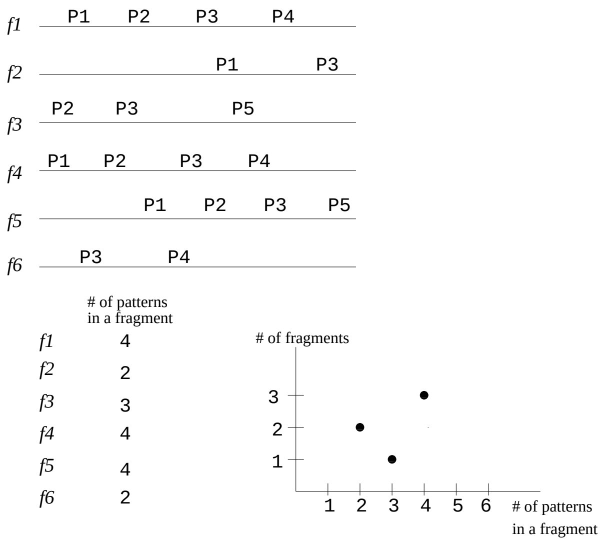

Figure 2: A sample fragment distribution

3. AN ENTROPY BASED SEQUENCE ASSEMBLY VALIDATION

We present a sequence assembly validation method by computing entropy of fragments in contigs. To compute entropy of fragments, we need to construct a probability model that measures how much each aligned fragment contributes to misassembly. The probability model is built using the *fragment distribution*, a measure used for repeat handling in a sequence assembler called AMASS[8]. We first describe how to collect the fragment distribution from a shotgun data.

3.1 Fragment distribution

A fixed number of non-overlapping patterns from each fragment are selected at random positions. Once all patterns are selected from all fragments, occurrences of the selected patterns are found in the entire shotgun data using a fast multiple string pattern matching algorithms[6]. A plot of the number of patterns in a fragment *v.s* the number of fragments is then generated. Figure 2 shows how to calculate the fragment distribution. Note that the fragment distribution is a fragment-centric distribution that counts multiple pattern occurrences within a fragment.

The rationale behind using this distribution for repeats handling is as follows. We select patterns that are long enough and therefore unlikely to occur by chance. For instance, if we use patterns of 16 bp, then patterns of that size occur only once by chance in a sequence of 4,294,967,296 bp ($= 4^{16}$) as there are four characters in DNA. Thus, the common occurrences of a pattern reflect with high probability true overlapping fragments. What we are collecting is a distribution of the number of such patterns within a fragment. Because we consider multiple patterns and their positions (within a fragment), the distribution should reflect the characteristic of the shotgun sequencing strategy, *i.e.,* random selection of fragments. Thus, the distribution is expected to look like a normal distribution centered around a mean value. Repeats,

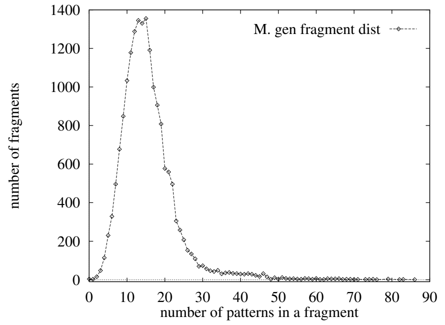

Figure 3: A fragment distribution from gmg shotgun data with 2 patterns of 16 bp per fragment.

of course, will distort the distribution since any pattern that belongs to a repeat will exhibit an increased number of pattern occurrences. To test this hypothesis, we performed an experiment with two sequences (Figure 5). One sequence is randomly generated and is expected to contain no significant number of repeats. The other sequence is obtained by inserting, in the randomly generated sequence, 5 copies of a 500 bp string at randomly selected positions, *i.e.,* 5 copies of a repeat. After generating a simulated shotgun data using a package called GenFrag [2], we collected fragment distributions using 8 bp patterns. As expected, the fragment distribution without repeats looks like a normal distribution centered around 140 while the fragment distribution with repeats look like the same distribution but with a long tail to the right (Figure 5). Fragments in the tail region are found to be from the 500 bp repeat copies. Thus, abnormally high number of pattern occurrences in a fragment with respect to the peak value of the fragment distribution indicates possible repeats. Consequently, the higher the number of patterns occurrences in a fragment, the more likely this fragment will contribute to misassembly. The probability model simply reflects this observation.

3.2 The probabilistic model

From the fragment distribution, we construct a probability function that models how much each fragment contributes to misassembly. As we discussed in the previous section, we expect the fragment distribution to reflect the characteristic of the shotgun data, *i.e.,* randomness of fragment selection, when there are no significant repeats. Thus the contribution of a fragment to misassembly can be seen by how much the fragment is deviated in terms of randomness of fragment selection. One way to measure the degree of deviation is to compare the fragment distribution to the fragment coverage distribution. Unfortunately, the coverage distribution is not known. However, it could be approximated by sampling coverage distributions from many short assembled regions.

For now, we use an *ad hoc* probability model based on the observation that a fragment with more pattern occurrences is likely to lead to misassembly. The probability that a fragment f contributes to misassembly is computed as below:

$$prof(f_i) = \begin{cases} 0.001 & \text{if number of patterns in } f_i < 2 \times p_v \\ 0.01 & \text{if } 2 \times p_v \leq \text{number of patterns in } f_i < 3 \times p_v \\ 0.1 & \text{if } 3 \times p_v \leq \text{number of patterns in } f_i < 4 \times p_v \\ 0.889 & \text{if } 4 \times p_v \leq \text{number of patterns in } f_i \end{cases}$$

where p_v denotes the peak value in the fragment distribution.

3.3 Computing entropy

From the probability model, we compute entropy at base position p in a contig as below:

$$entropy(p) = \sum_{p-\delta \leq pos(f_i) \leq p+\delta} -prob(f_i) \times log(prob(f_i))$$

where $pos(f_i)$ denotes the left end position of f_i in the contig and δ is a user input parameter (by default, it is the same as the window size used for the fragment distribution calculation).

4. EXPERIMENT WITH ASSEMBLY OF *MYCOPLASMA GENITALIUM*

4.1 Phrap Assembly Result

We tested the effectiveness of our analyses with the *de novo* sequence assembly of *Mycoplasma genitalium* genome [4] generated using Phrap [5] with default parameters. Among sequence assemblers available in public, Phrap was chosen since it is the most widely used sequence assembler and it is known to generate longer contigs [1]. The detection of misassembled regions in Phrap assembly should not be interpreted to mean that Phrap generates contigs of low fidelity. Sequence assembly is very difficult and still unresolved. No existing assemblers can correctly handle long repeats. Chen and Skiena [1] showed that all three assemblers they tested, Phrap, Stroll, and TIGR Assembler, failed to seperate two copies of long ribosomal RNA of 3,140 bp.

Seven contigs were assembled from shotgun data obtained from *The Institute for Genomic Research* (TIGR)[1]: C1, C3, C4, C5, C6, C7, and C8. [2]

Since the target sequence is already known, we were able to identify misassembled regions in contigs by comparing Phrap-generated contigs to the published consensus sequence using cross_match. Six misassembled regions were identified (Figure 4).

4.2 Sequence assembly validation: entropy plot *v.s* fragment coverage plot

There are two main problems with the fragment coverage approach; (1) it is difficult to distinguish randomly occurring high coverage regions from repeat-induced high coverage regions, and (2) misassembly can occur in low coverage regions.

Our entropy based approach is not sensitive to coverage. Entropy calculated from the fragment distribution was successful in identifying all misassembled regions from the Phrap-generated assembly. Less than twenty peaks, representing possible misassembled regions, were found in the assembly of *Mycoplasma genitalium.* In fact, in all contigs, regions with entropy over 3 are correctly identified as misassembled regions (see Figures 6, 7, 8 and 9). Compared to the fragment coverage plots, entropy plots clearly distinguished misassembled regions. For instance, among the peaks in the fragment coverage (Figure 6), only one peak corresponds to misassembly while all other peaks are simply high coverage regions.

[1]The first author obtained the shotgun data from TIGR while writing Amass[8] at the University of Iowa. We are grateful to Granger Sutton for providing the data.

[2]Contigs are numbered as Phrap generated.

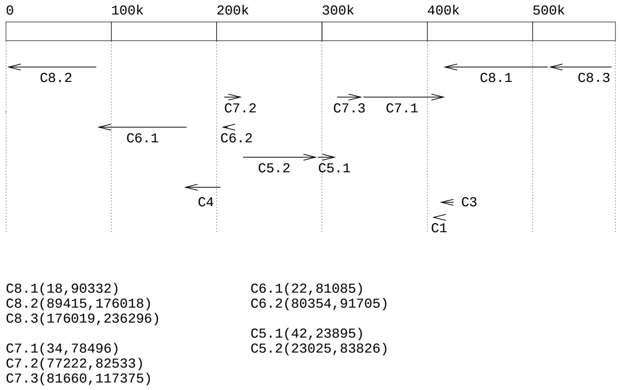

Figure 4: Identification of misassembled regions in the PHRAP-generated assembly of *Mycoplasma genitalium*. Misassembled regions are identified by comparing contig sequences to the published sequence: C8.1 should be placed after C8.3, C7.1 and C7.3 should be joined and C7.2 be separated, C6.1 and C6.2 should be separated, and C5.1 and C5.2 should be swapped. In addition, C1 is to be embedded in C7.1. C3 connects C7.1 and C8.1.

5. CONCLUSION

We presented a probabilistic approach to sequence assembly validation. Compared to a widely used fragment coverage method, our approach clearly identified misassembled regions. This approach is also computationally very efficient. Fragment distribution can be computed using an efficient multiple string matching algorithms[6]. Probability and entropy are easy to compute.

While the current probability assignment is effective, it is *ad hoc*. We are currently developing a method to assign probability based on predicted coverage distribution and fragment distribution as discussed in Section 3.2, *i.e,* approximate coverage distribution and statistical methods to measure the degree of deviation. We are also developing several alternative methods for sequence assembly validation, including clone coverage analysis [7].

6. REFERENCES

[1] Ting Chen and Steven S. Skiena, "A Case Study in Genome-level Fragment Assembly," *Bioinformatics*, 16 (6), 2000, pp 494-500.

[2] M.L. Engle and C. Burks, "GenFrag 2.1: NewFeatures for More Robust Fragment Assembly Benchmarks," *Genomics* 10 (1994), pp. 567-568.

[3] Eric C. Rouchka and David J. States, "Sequence Assembly Validation by Multiple Restriction Digest Fragment Coverage Analysis" *Proc. of Intelligent Systems for Molecular Biology (ISMB)*, 1998, pp 140 - 147, AAAI Press

[4] Claire M. Fraser, Jeannine D. Gocayne, Owen White, Mark D. Adams, Rebecca A. Clayton, Robert D. Fleischmann, Carol J. Bult, Anthony R. Kerlavage, Granger Sutton, Jenny M. Kelley, Janice L. Fritchman, Janice F. Weidman, Keith V. Small, Mina Sandusky, Joyce Fuhrmann, David Nguyen, Teresa R. Utterback, Deborah M. Saudek, Cheryl A. Phillips, Joseph M. Merrick, Jean-Francois Tomb, Brian A. Dougherty, Kenneth F. Bott, Ping-Chuan Hu, and Thomas S. Lucier, "The Minimal Gene Complement of Mycoplasma genitalium," *Science* 1995 October 20; 270: 397-404.

[5] P. Green, `http://www.phrap.org`

[6] Sun Kim and Yanggon Kim, "A Fast Multiple String-Pattern Matching Algorithm," *Proc. of 17th AoM/IAoM Conference on Computer Science*, August 1999

[7] Sun Kim, Li Liao, Michael P. Perry, Shiping Zhang, and Jean-Francois Tomb, "Clone Coverage Analysis for Shotgun Sequence Assembly Validation," In preparation

[8] Sun Kim and Alberto Maria Segre, "AMASS: A Structured Pattern Matching Approach to Shotgun Sequence Assembly," *Journal of Computational Biology*, 6(2), 1999, pp 163-186, Mary Ann Liebert, Inc

[9] G. Sutton, O. White, M. Adams, and A. Kerlavage, "TIGR Assembler: A New Tool for Assembling Large Shotgun Sequencing Projects," *Genome Science and Technology* 1(1), 1995, pp. 9-19.

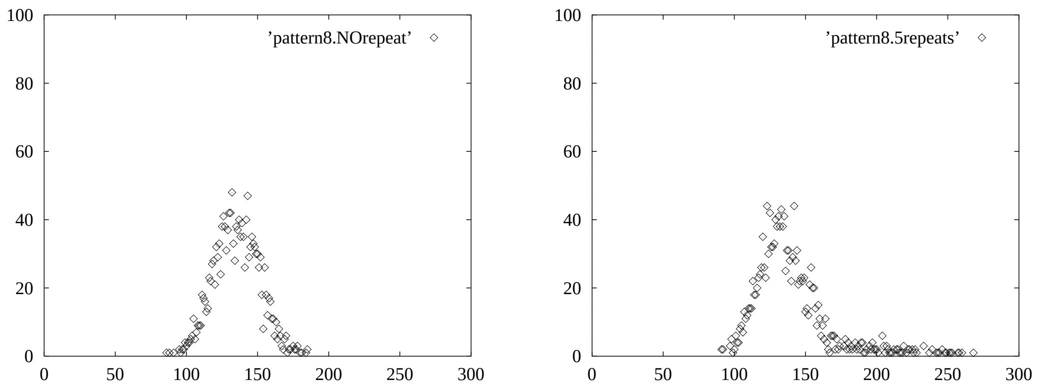

Figure 5: Two fragments distributions collected from two shotgun data; one from random 50k bp sequence and the other from 5×500 bp inserted random 50k bp sequence.

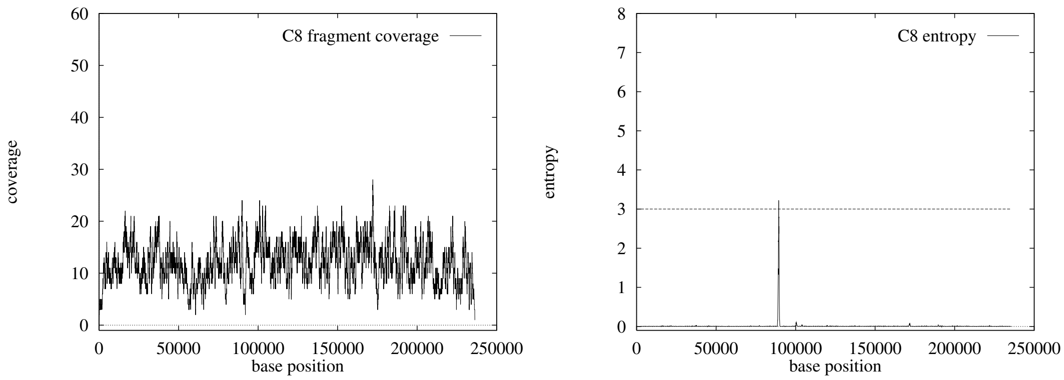

Figure 6: The fragment coverage and the entropy plot for contig 8

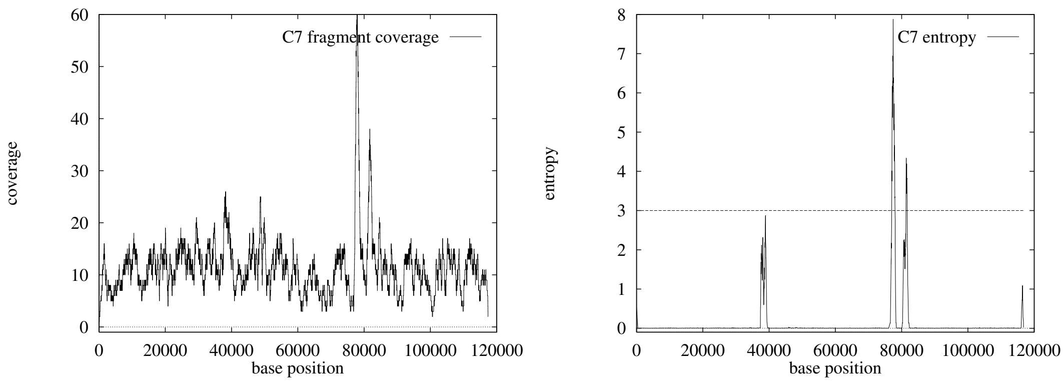

Figure 7: The fragment coverage and the entropy plot for contig 7

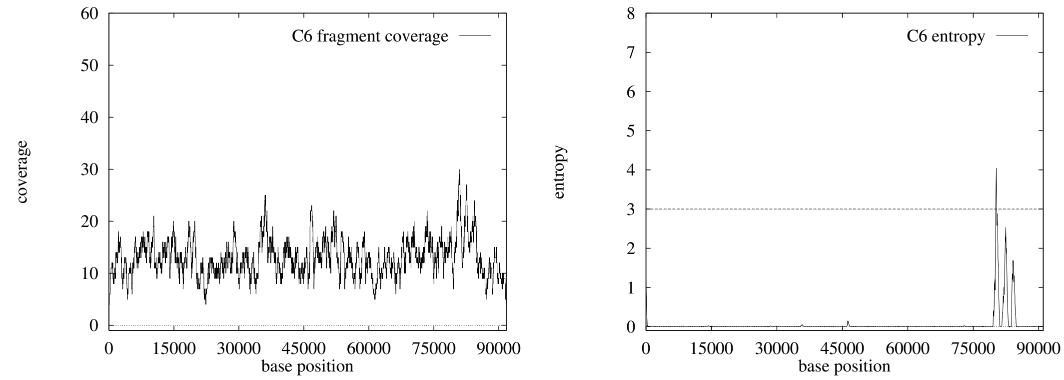

Figure 8: The fragment coverage and the entropy plot for contig 6

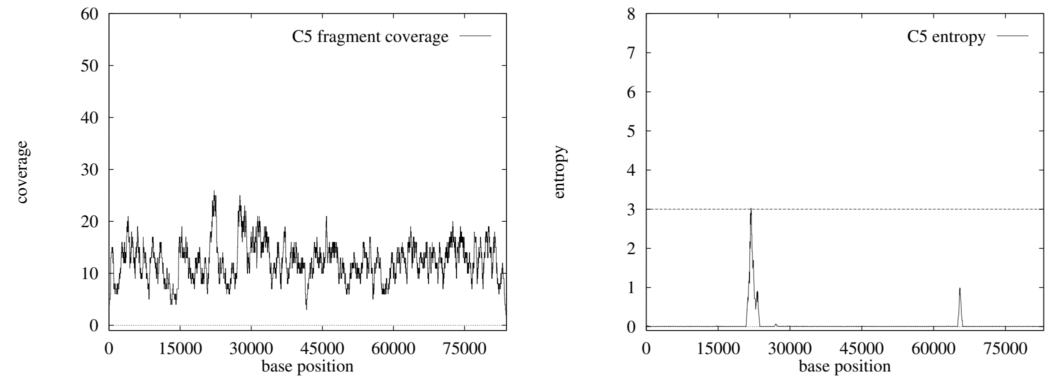

Figure 9: The fragment coverage and the entropy plot for contig 5