BARNACLE: An Assembly Algorithm for Clone-based Sequences of Whole Genomes

Vicky Choi[1,2], Martin Farach-Colton[1]
[1]Department of Computer Science, Rutgers University, Piscataway, New Jersey 08854

June 1, 2003

Abstract

We propose an assembly algorithm BARNACLE for sequences generated by the clone-based approach. We illustrate our approach by assembling the human genome. Our novel method abandons the original physical-mapping-first framework. As we show, BARNACLE more effectively resolves conflicts due to repeated sequences which is the main difficulty of the sequence assembly problem. In addition, we are able to detect inconsistencies in the underlying data. We present and compare our results on the December 2001 freeze of the public working draft of the human genome with NCBI's assembly (Build 28).

The assembly of December 2001 freeze of the public working draft generated by BARNACLE and the source code of BARNACLE are available at (http://www.cs.rutgers.edu/~vchoi).

Keywords: clone-based sequencing; sequence assembly algorithm

1 Introduction

Much of the attention and debate within the Human Genome Project has focussed on two sequencing strategies: *whole-genome shotgun sequencing*, adopted by Celera Genomics (Venter et al. 2001); vs. *hierarchical shotgun sequencing*, also referred to as *clone-based* or *BAC-by-BAC*, used by the International Human Genome Sequencing Consortium (IHGSC) (International Human Genome Sequencing Consortium 2001). The data set generated by the whole-genome shotgun approach consists of a set of pairs of sequence reads (mates). Each read has average length of around 500bp governed by current sequencing technology. The data set generated by the clone-based approach consists of a set of Bacterial Artificial Chromosome (BAC) clones, typically 100-200Kb. We will use 'BAC', 'clone' and 'BAC clone' interchangeably. Each BAC clone is individually sequenced and preassembled. In general, a clone consists of a set of mostly non-overlapping

[2]Corresponding author. Present Address: Duke University, Department of Computer Science, PO Box 90129, Durham NC 27708, Email: vchoi@cs.duke.edu, Fax: (919)660-6519.

fragments, which are the preassemblies of the shotgun reads of each BAC clone produced by some assemblers, such as PHRAP (Green, unpublished). A clone is called a *finished clone* if it consists of only one fragment which has accuracy at least 99.99%, otherwise it is a *draft clone*.

The computational problem of reconstructing the genome sequence by assembling the sequences of the data set is called the *sequence assembly problem*. The challenge of the problem lies in the repeat-rich nature of the genome. For example, the human genome is more than half repeated sequences, which include large 50-500Kb duplicated segments with high sequence identity ($\geq 98\%$) (International Human Genome Sequencing Consortium 2001). Computationally, the sequence assembly problems for the two sequencing strategies are completely different because the structure of the data sets are different.

In this paper, we focus on the assembly problem of sequences generated by the clone-based approach, with application to the human genome. In addition to the repeat problem mentioned above, the assembly problem of the clone-based approach is further complicated by laboratory errors, such as chimeras and contaminations; by the draft quality of the sequences, which are sometimes inaccurate; and by polymorphism.

The set of BAC clones, finished and draft, produced by IHGSC is called the *public working draft of the human genome*. It is worthwhile to point out that the clone-based strategy used is not the originally proposed "clone by clone" strategy, in which a physical map giving the clone ordering is first produced, and then a minimum tiling set of clones in the map is selected and individually subjected to shotgun sequencing. In practice, the physical map of the clones based on fingerprint overlaps is constructed concurrently with the sequencing (International Human Genome Sequencing Consortium 2001). The ordering of the sequenced clones is either unknown or not accurately known.

There are two other algorithms for assembling the public working draft of the human genome. One is GigAssembler (Kent and Haussler 2001) from UC Santa Cruz and the other is an unpublished NCBI's algorithm (http://www.ncbi.nlm.nih.gov/genome/guide/human/). The fingerprint-based physical map, which was constructed by manually editting the initial automated fingerprint assembly (The International Human Genome Mapping Consortium 2001), was employed by GigAssembler to assist in the sequence assembly, although the clone ordering was only roughly determined in the map (Kent and Haussler 2001). NCBI does not make use of the fingerprint map (http://www.ncbi.nlm.nih.gov/genome/guide/HsFAQ.html#diffassemblies).(Remark: NCBI's algorithm was said to be modified to use the hand-curated Tiling Path Files since Build 29.) Instead, several STS marker maps were used for chromosome assignments. Both algorithms are based on greedy approaches to assemble sequences in which the best overlap is assembled first, where *best* is defined by some score functions used to assign weights for overlaps. Since the genome is highly repetitive, these score functions are unlikely

to yield the true assembly (Bailey et al. 2001). Without the fingerprint clone contig to have BACs restricted to, the NCBI algorithm first formulates a Maximal Interval Subgraph (MIS) problem to obtain a BAC ordering and then assembles the sequences of overlapping BACs. This approach is natural given the history of the Human Genome Project, with its original physical-mapping-first approach. However, this top-down approach suffers from the fact that if a BAC is misplaced, the mistake in the assembly is unrecoverable.

In this paper, we propose an assembler BARNACLE for the clone-based sequences augmented with chromosome-assignment, thus we use the same input data as NCBI. Part of the novelty of our approach is to assemble the genome bottom-up, without making use of fingerprint-based physical map of the clones. That is, we use extensive sequence-level overlaps to suggest BAC overlaps. As we will explain in the discussion section, the clone-overlaps inferred from the sequences are more informative than from the fingerprint data. In other words, this approach allows us to order BACs with higher confidence. Also, unlike the use of score functions to attempt to resolve the repeat problem, our way of resolving the repeat problem by considering the consistency of overlaps and the interval graph formalism is mathematically justifiable. In addition, this enables us to detect inconsistencies in the underlying data. We remark that this error detection feature does not exist in the other algorithms. Finally, we remark that the idea of our algorithm can be naturally extended when additional information, such as low-copy repeated sequences are available. A separate work (Choi et al. 2002a) of modifying BARNACLE to incorporate a segmental duplication database (Bailey et al. 2002) is in preparation (Choi et al. 2002b).

2 Materials and methods

2.1 Details of Input

In this section, we describe the input for BARNACLE, which includes the sequence information of BACs and fragments, overlap information and orientation information.

2.1.1 BACs and fragments

A *BAC* consists of a contiguous stretch of DNA from a chromosome. The associated data with each BAC consists of the estimated length of the BAC, the phase of the BAC, the number of fragments, each fragment's sequence and the chromosome of the BAC, if assigned.

Table 1 shows an example of BAC $AC002092.1$ with estimated length 95456bp has four fragments. Fragment $AC002092.1\tilde{}1$ is the sequence from 1 to 888 in the GenBank record of $AC002092.1$. Notice that we do not know the actual sequence of each BAC but we know the sequence of each of its fragments, which

are the preassemblies of the shotgun reads of the BAC produced by some assemblers, such as PHRAP (Green, unpublished).

Depending on the raw data coverage, BACs are categorized into three phases. Phases 1 and 2 are the draft sequences; phase 3 comprises the finished sequences. For a phase 1 BAC, the fragments of the BAC are not necessarily disjoint, and the order and orientation of fragments are in general unknown. For a phase 2 BAC, the fragments are disjoint and the order of fragments is known. A phase 3 BAC is a finished sequence, i.e. only one fragment. In addition, for some phase 1 BACs, some partial fragment order information and end fragment information are also available.

2.1.2 Overlap Information

Based on local alignments of all fragment sequences against all fragment sequences, we further process two types of valid overlaps between fragments: dovetail overlap and complete containment (see Fig. 1). The sequence identity of the overlaps has to be at least 97% and end-allowed-error (as shown in Fig. 1) is taken to be 350bp for finished sequences and min{10% of the fragment length, 1000}bp otherwise.

Also, according to the annotation of some finished sequences, some overlaps are generated. These are called *nt-pairs*. These overlaps include 0bp overlaps, i.e., the fragments do not overlap but are consecutive to each other. The nt-pairs are supposed to be true overlaps if the annotations are correct and they are generated accordingly.

2.1.3 Orientation Information

There are additional sequences from paired-end plasmid reads. These sequences are aligned against all fragment sequences. Based on these alignments, the relative orientation of fragment pairs is obtained as input.

2.2 Methods

Before we describe the idea behind our algorithm, we introduce some terminology.

We say an overlap is *true* if the fragments are from the overlapping segments of the genome, otherwise the overlap is *repeat-induced* (Fig. 2) (Myers et al. 2000). The overlaps which are not true are called *False Positives* (FPs). In other words, repeat-induced overlaps are FPs. On the other hand, the overlaps which are true but not detected are called *False Negatives* (FNs). FPs and FNs are called *noise*.

The assembly problem would be straightforward if we could divine all true overlaps, i.e., if the data of overlaps were noise-free. The key objective is thus to clean up the noise as much as possible and assemble

the fragments according to the true overlaps.

Observe that for an overlap to be true, it is necessary for the overlap to be "consistent" (Fig. 3). However, "consistency" is not sufficient for a true overlap. There might be some consistent but repeat-induced overlaps, which are due to either the lack of coverage or long repeats (Fig. 4). We call a consistent but repeat-induced overlap a *consistent repeat-induced overlap*, otherwise we call it an *inconsistent repeat-induced overlap*.

A *contig* (subcontig resp.) is a contiguous region that is covered by a set of overlapping BACs (fragments resp.).

The basic idea behind our algorithm is illustrated in Figure 5 and Table 6 shows the high level description of the algorithm. In following, we outline the idea of each step. The details of the implementation of the algorithm are described in (Choi 2001)(http://www.cs.rutgers.edu/~vchoi).

A. "Conservatively" assemble fragments into subcontigs. First, we assemble consistent overlapping fragments into subcontigs. Before we describe the algorithmic implementation, we introduce some terminology. A fragment is called a *subfragment* if the fragment is completely contained in another fragment, otherwise the fragment is *maximal*. Making use of the maximality property of the latter, we efficiently identify and assemble consistent overlapping maximal fragments (once again readers are invited to read (Choi 2001) for the simple algorithmic trick). Then consistent overlapping subfragments are assembled into the contig of their corresponding maximal fragments. With this implementation, 219,031 non-singleton fragments of the December 2001 freeze are assembled into 12,726 subcontigs in about 25 seconds on a Pentium III computer.

We then deduce clone-overlaps from these subcontigs: two clones overlap if and only if there is at least one fragment pair of the corresponding clones overlapping in a subcontig. Then the conflicting overlaps are resolved according to the clone-overlaps (Fig. 6). Unlike the use of score functions to resolve conflicts, we are making use of the BAC information of fragments and use the assembly obtained by consistent overlaps to resolve inconsistent overlaps. This approach is well justified because the consistent overlaps, which are necessary condition for true overlaps, give a good indication whether two BACs overlaps or not. Note that it is this approach that allow us to naturally make use of the segmental duplication database without substantial changing the algorithm.

As mentioned before, these subcontigs might still contain some consistent repeat-induced overlaps.

B. Detect and remove consistent repeat-induced overlaps and chimeric clones. We use the linear structure of the chromosome to detect the consistent repeat-induced overlaps. The linear structure of the sequence would be destroyed if the repeat-induced overlaps were used (Fig. 7**a**). Mathematically

speaking, the corresponding clone graph, whose vertices are clones and there is an edge between two vertices if and only if the two corresponding clones overlap, must be an interval graph. (Remark: This is true with the assumption that the length of gaps within a BAC is short enough such that there is at least one fragment pair overlapping if two BAC overlaps. However, this assumption is no longer true in the current working draft of the human genome because some BACs were heavily trimmed resulting some of the BACs only a few hundreds bp long (much shorter than a possible gap). Comparing April 2001 freeze with December 2001 freeze, we found that 1200 BACs whose length have been reduced more than half of its original length. Note that our algorithm is robust enough that with a slight modification of resolving non-interval graph procedure, one can heuristically take care of this possibility.) Intuitively, imagine the genome sequence as a line, then each BAC is an interval of the line such that two BAC sequences overlap if and only if the corresponding intervals overlap (Fig. 8**a**). In other words, if the corresponding clone graph is not interval, the assembly is incorrect. For example, BAC AC019248 in NCBI's Build 28 is misassembled (Fig. 8**b**). By resolving the non-interval connected components of the clone graph, we detect and remove suspicious repeat-induced overlaps and chimeric clones (Fig. 7**b**). Instead of resorting to the NP-hard Maximal Interval Subgraph problem (Goldberg et at. 1995), which does not characterize the real biology, to get an interval graph, we design an efficient algorithm for resolving the non-interval graph (see Table 7 for the idea). The interval representation (and hence the ordering) of clones is obtained from the resulting interval clone graph by a linear-time interval graph recognition algorithm (Corneil et al. 2001).

C. Orient and order subcontigs. According to the interval representation of clones, subcontigs are ordered and the orientation of "long" subcontigs is determined. According to the ordering of clones, we orient subcontigs by flipping such that the ranks of BACs in each subcontig are in non-decreasing order, and assign coordinates to subcontigs so that they can be ordered by sorting according to these coordinates lexicographically. Note that our interval representation of clones is quite informative in that most of the subcontigs can be ordered unambiguously according to this reliable information. Thus we do not need to employ the quite noisy and uncertain information from plasmid reads, ESTs and mRNAs to order the subcontigs, as does GigAssembler, which resorted to the Bellman-Ford algorithm for testing the feasibilities of the information. Also, the interval representation allows us to detect FNs while ordering subcontigs. First observe that the corresponding end BACs of the adjacent subcontigs must be either the same or overlapping (Fig. 9**a**). We call this necessary condition the *adjacency condition*. Accordingly, when some subcontigs cannot be ordered such that they satisfy the adjacency condition, it indicates that there might be FNs (Fig. 9**b**). To further verify the identification of the

FNs, we aligned the involved fragments with their overlapping clones. Examining these alignments reveals the several possible causes, which include the consequences of repeat-masking, low accuracy of some draft sequences, chimeric fragments or fragment misassemblies and polymorphism. On the other hand, some subcontigs might be *ties* in which the adjacency condition is always satisfied for any permutation of them. In other words, the information is not sufficient to determine their order. In this case, we employ some additional information to break these ties.

D. Adjust the ordering and correct the orientation of the subcontigs using additional information. The additional information includes the identification of the end fragments of BACs, as well as, the partial order of some fragments. For each BAC which has end fragment information, i.e., one or two fragments, first according to their current position in the contig, we determine which one is the left end fragment and which one is the right end fragment. Then we orient the fragments so that they are the extreme fragments of the BAC. To ensure the reliability of the information, these adjustments (order and orientation) are always subject to the adjacency condition, i.e., whether we can adjust the order and orientation according to the information such that the adjacency condition is still satisfied. Similarly, the adjustments are being done for each group of the ordered fragments.

Finally, the relative orientation of fragment pairs generated from plasmid reads are used to orient the subcontigs which are not "long" enough to have been determined. According to the interval representation of clones, one can determine how confident the orientation of subcontigs is. For example, for the subcontigs which are long enough so that the smallest BAC and the largest BAC are not overlapping in the subcontig, the correctness of the orientation is sure. The relative orientation of fragment pairs generated from plasmid reads are used to orient the unsure subcontigs. For each unsure subcontig, we organize all its relative orientations into a list of agreeing subcontigs and a list of disagreeing subcontigs. We then progressively change the orientation of unsure subcontigs such that the number of disagreeing subcontigs is minimized.

3 Results and the comparison with the public assembly

In this section, we present the result of our algorithm applied to the input of the December 2001 freeze of the public working draft of the human genome. First, we describe the details of the input of the December 2001 freeze. Then we present and compare the results with the NCBI's assembly (Build 28).

3.1 Input of the December 2001 freeze

The sequence information with chromosome assignments, local alignments of all-against-all fragment sequences, local alignments of all fragment sequences against plasmid reads sequences were provided by NCBI. We further processed these local alignments to generate valid fragment pair overlaps and relative orientation of fragment pairs. The details of the December 2001 freeze are shown in Table 2.

3.2 Results on the December 2001 freeze

We assemble 219,031 non-singleton fragments into 12726 subcontigs, with length 2.51 Gbp. From these subcontigs, we obtain a clone graph of 33929 BACs in 2195 connected components, which consists of 23145 BACs in 2052 interval connected components and 1078 BACs in 143 non-interval connected components. Upon resolving non-interval components, 139 problematic BACs, which are either suspicious chimeras or contains unidentified repeats, are removed. The result is 33722 non-singleton BACs in 2443 interval components, as shown in Table 3. We remove 8189 ($= 259230 - 251041$) fragments based on our FN detection which indicates the fragments should be contained in some subcontigs.

A graphical display of the assembly of December 2001 freeze of the working draft of the human genome is available at (http://www.cs.rutgers.edu/~vchoi).

3.3 Comparison with NCBI's assembly

We compare our results with NCBI's assembly on December 2001 freeze (see Table 4) (Note GigAssembler was no longer in use. See (Choi 2001) for a comparison with both NCBI's assembly and GigAssembler on April 2000 freeze). The 29537 ($= 251928 - 222391$) fragments were absent from NCBI's build for reasons we do not know. To further measure the quality of the assembly, we introduce the definition of *warp* $= \frac{\text{assembled BAC length}}{\text{estimated BAC length}}$ (as used by NCBI's contig group), which is the ratio of the assembled BAC length over the estimated BAC length. See Table 5 for the comparison. There are 1016 BACs in our assembly with assembled BAC length more than 250 Kbp (the longest one is 732,527 bp), while there are 3786 such BACs (with the longest one 19,436,525 bp) in NCBI's assembly. We conjecture that the number would be even worse if they had not thrown away the 29K fragments. Note that while it is true that the warped BACs are misassembled, it is not necessary that the non-warped BACs are correctly assembled. Indeed, the suspicious chimeric BACs we detected are usually non-warped in NCBI's assembly because of the way their assembly is done. To measure the accuracy of the assembly, it is important to understand the method or the algorithm of the assembly.

4 Discussion

The original proposal of the Human Genome Project is "map first, then sequence". However, various formulation of physical mapping problem, which is to reconstruct the relative positions of the clones based on their fingerprint overlap, were proved to be NP-hard (Goldberg et at. 1995)(which are conjectured to be computationally intractable). By fingerprint overlap it means that two clones overlap if they share common fingerprints. Note that the clone overlap derived from fingerprints only tell whether two clones intersect but not where they overlap. Unlike physical mapping problem, there are additional information from sequences to infer clone overlap in the sequence assembly problem. Namely, clone overlaps can be derived from sequences overlaps which tell not only whether two sequence overlap but where they overlap. In other words, sequence overlaps have more information than fingerprint overlaps. See Figure 10 for this important distinction. Unfortunately, led by physical-mapping first approach, NCBI treats the sequence overlaps in fingerprint overlap fashion to infer clone overlaps. In particular, NCBI formulates a Maximal Interval Subgraph problem (which is NP-hard), as in (Goldberg et at. 1995) for solving physical mapping problem, to obtain a clone ordering, where the weight of a clone-overlap is the summation of weights of all the corresponding fragment pairs, which are assigned by some score functions.

Motivated by the fact that sequence overlaps are more informative than fingerprint overlaps and the hardness result of the physical mapping problem, we abandon the physical map first framework and assemble the genome bottom-up. With this approach, the clone-overlaps are derived from consistent fragment sequence overlaps which is the necessary condition for the true overlaps. The clone graph thus constructed is not too noisy: Most of the components are interval and the non-interval component permits an efficient and effective heuristic resolution. Note with this approach, we do not make use of fingerprint map (the physical map built based only on fingerprint data), but any fingerprint data would be very useful as additional information. Indeed, it is theoretically impossible to assemble the true assembly based only on the sequence overlap information, although the way we resolve the repeat problem is justified by considering the consistency of overlaps and the interval graph formalism. As a matter of fact, the assemblies are far from perfect. For example, the assemblies of those "warped" BACs whose assembled length is greater than 250Kb are obviously incorrect. These misassemblies might be largely due to undetected FNs, or they might be due to over-collapse of repetitive regions, which do not destroy the interval graph property; for example, repeats which occur at one end of contigs due to the lack of coverage or repeats which are longer than an entire BAC. More biological information (e.g. fingerprint data) is needed to resolve these cases. For instance, the 500Kb inverted repeat on chromosome Y is resolved by using the annotation of sequences in the GenBank. Furthermore, our algorithm is able to detect the inconsistency underlying the data which is consequential characteristic of

a good assembler. The detected suspicious error include chimeras, chromosome misassignments, wrong annotation of some finished sequences and fragment misassemblies (see (Choi 2001) for a list of suspicious errors detected from April 2001 freeze).

Our algorithm is not only conceptually simple and easy to implement (The source code of BARNACLE in C language is freely available at (http://www.cs.rutgers.edu/~vchoi), but also it is very efficient that it takes 3 minutes on a Pentium III (933 MHz) computer to assemble the public working draft of the human genome after preprocessing to the input format as described. For the same step, GIGASSEMBLER takes about two hours on a cluster of 100 Pentium III CPUs. This huge difference is partly due to GigAssembler does not make use of the underlying data structure for the algorithm design. For example, from assembling fragments (build "rafts" in GigAssembler's language), by making use of property of maximal fragments and subfragments, to ordering the subcontigs (build"barges" in GigAssembler's language), by making use of the interval ordering of the clone graph.

The International Human Genome Sequencing Consortium recently announced the completion of the Human Genome Project (http://www.genome.gov/11006929) and that "The finished sequence produced by the Human Genome Project covers about 99 percent of the human genome's gene-containing regions, and it has been sequenced to an accuracy of 99.99 percent." The assembly was done by NCBI based on curated Tiling Path Files (TPF) provided by the international sequencing consortium. But how were these TPF curated? How accurate were these curation? Would the nt-pair (which was derived from TPF) errors as in Figure 11, which were detected and corrected early by our algorithm because of the conflicts with the redundant data, be corrected by the curation? As we have demonstrated, with the appropriate algorithmic approach the clone-based sequences of the human genome is by all means manageable and we believe that incorporating the information for curating TPF as additional information into BARNACLE would have improved the accuracy of the "finished" sequence. Since the Human Genome Project, new sequencing strategies have been developed, e.g., hybrid of clone-based and whole-genome shotgun. Nevertheless, we believe researchers can benefit from the algorithmic skills to develop the new assemblers.

References

Bailey, J.A., Gu, Z., Clark, R.A., Reinert, K., Samonte, R.V., Schwartz, S., Adams, M.D., Myers, E.W., Li, P.W. and Eichler, E.E. 2002. Recent segmental duplications in the human genome. *Science* **297**:1003-1007.

Bailey, J.A., Yavor, A.M., Massa,H.F., Trask, B.J. and Eichler, E.E. 2001. Segmental Duplications: Organi-

zation and Impact Within the Current Human Genome Project Assembly. *Genome Research* **11**:1005-1017.

Choi, V. 2001. BARNACLE: An Assembly Algorithm for Clone-based Sequences of Whole Genomes. PhD Thesis, Rutgers University, Department of Computer Science.

Choi, V., Bailey, J.A., Schuler, G., Gu, Z., Li, P., Farach-Colton, M. and Eichler., E.E. 2002. The sequence and assembly of highly duplicated regions in the human genome. *Genome Sequencing & Biology meeting at Cold Spring Harbor Laboratory 2002.*

Choi, V., Bailey, J.A., Farach-Colton, M. and Eichler, E.E. 2002. ROPE: Empowered BARNACLE by a segmental duplication database. In preparation.

Corneil, D.G., Olariu S. and Stewart, L. 2001. The LBFS structure and recognition of interval graphs. Submitted for publication.

Goldberg, P.W., Golumbic, M.C., Kaplan, H., Shamir, R. 1995. Four strikes against physical mapping of DNA. *Journal of Computational Biology* 2:1, 139-152.

International Human Genome Sequencing Consortium. 2001. Initial sequencing and analysis of the human genome. *Nature* **409**:860-921.

Kent, K.J. and Haussler, D. 2001. Assembly of the working draft of the human genome with GigAssembler. *Genome Research* **11**:1541-1548.

Myers, E.W., Sutton, G.G., Delcher, A.L., Dew, I.M., Fasulo, D.P., Flanigan, M.J., Kravitz, S.A., Mobarry, C.M., Reinert, K.H.J. and Remington, K.A. 2000. A whole-genome assembly of Drosophila. *Science* **287**:868-877.

The International Human Genome Mapping Consortium. 2001. A physical map of the human genome. *Nature* **409**:934-941.

Venter, J.C., Adams, M.D., Myers, E.W., Li, P.W., Mural, R.J., Sutton, G.G., Smith, H.O., Yandell, M., Evans, C.A., Holt, R.A., et al. 2001. The sequence of the human genome. *Science* **291**:1304-1351.

Table 1: An example of a BAC.

accession #	estimated length	phase	chrm	number of fragments
AC002092.1	95456	1	17	4

fragment acc#	start	end	length
AC002092.1~1	1	888	888
AC002092.1~2	889	46200	45312
AC002092.1~3	46201	84925	38725
AC002092.1~4.1	84926	95170	10245

Table 2: The details of the input of the December 2001 freeze.

1. **Sequence Information**

phase	BACs	fragments	total length in Gbp	average number of fragments
1	15298	246424	2.55	16.11
2	2154	8161	0.33	3.79
3	17624	17624	2.04	1.00
Total	35076	272209	4.992	7.76

2. **Chromosome Assignment**

   The scheme employed by NCBI to assign a chromosome to a BAC is based on:

   - STS: presence of at least 2 STS markers that have themselves been mapped to the same chromosome;
   - GenBank: annotation on the submitted GenBank record;

   Otherwise, the chromosome of the BAC is *Unknown*.

   The chromosome assignments are summarized as below:

STS	GenBank	Unknown
31543	2450	1083

3. **Overlap Information** There are $403,466$ fragment pairs of the same chromosome or at least one of them an unknown chromosome. There are $12,656$ nt-pairs.

4. **Orientation Information** Relative orientation fragment pairs are generated from paired-end plasmid reads. There are $321,751$ such fragment pairs.

Table 3: The results of BARNACLE on December 2001 freeze.

	BACs	Fragments Used/Fragments	Contigs	Length in *Gbp*
singletons	1215	9967/9967	1215	0.142
non-singletons	33722	251041/259230	2443	2.708
	34937	261008/269197	3658	2.850

Table 4: Build 28 - NCBI's assembly on December 2001 freeze.

	BACs	Fragments Used/Fragments	Contigs	Length in *Gbp*
singletons	836	9074/9074	836	0.112
non-singletons	32902	222391/251928	2292	2.745
	33738	231465/261002	3128	2.857

Table 5: Comparison of BARNACLE's assembly with NCBI's assembly on December 2001 freeze.

Warp	Our Assembly	NCBI's Public Assembly
$<= 1.5$	33474	29647
$1.5 - 1.8$	753	725
$1.8 - 2.0$	278	371
$2.0 - 5.0$	421	1813
$5.0 - 10.0$	10	612
> 10.0	1	570

Restricting warp > 1.5,

Assembled BAC Length	Our Assembly	NCBI's Public Assembly
$250K - 300K$	434	461
$300K - 500K$	549	1328
$500K - 800K$	33	798
$800K - 1M$	0	248
$1M - 2M$	0	496
$2M - 3M$	0	129
$3M - 10M$	0	259
$10M - 20M$	0	67
Total	1016	3786

Table 6: The high level description of the algorithm.

A. **"Conservatively" assemble fragments into subcontigs.**

1. Classify fragments: singleton, subfragment and maximal fragment.
2. Assemble consistent overlapping maximal fragments into subcontigs.
3. Put back *good* subfragments to subcontigs.
4. Detect and resolve conflicting chromosome assignments.
5. Construct a BAC graph from subcontigs.
6. Resolve inconsistent overlaps according to the connectivity of the BAC graph.

B. **Detect and remove consistent repeat-induced overlaps and chimeric clones.**

7. For each component G_i of the BAC graph, if G_i is not interval, resolve the component by removing repeat-induced overlaps or suspicious chimeric BACs.

C. **Orient and order subcontigs.**

8. Obtain the interval representation of BACs.
9. Orient subcontigs.
10. Assign coordinates to subcontigs and order subcontigs by sorting lexicographically.
11. Detect the potential false negatives and remove the involved fragments.
12. Detect false positives (consistent repeat-induced overlaps that do not destroy the interval property).

D. **Adjust the ordering and correct the orientation of the subcontigs using additional information.**

13. Adjust the order of the subcontigs according to the extra fragment information.
14. Orient subcontigs according to the relative orientation of fragment pairs generated from plasmid reads, ESTs and mRNAs data.
15. Derive a consensus sequence for each contig from the assembly of maximal fragments of the contig.

Table 7: The idea of resolving non-interval graph.

In the following, we sketch the idea of how the non-interval graphs are resolved. The interested reader is invited to read (Choi 2001) for the details.

Let $G = (V, E)$ be a non-interval graph. Without lose of generality, assume that G is connected. Define a vertex $v \in V$ to be *I-critical* if $G|_{V \setminus \{v\}}$ is interval.

Given a non-interval graph G, we first identify a forbidden subgraph of G, then check whether at least one of the vertices of the forbidden subgraph is I-critical, if so, we say G is *fixable*.

Based on the structure of the forbidden subgraph, a fixable graph G is resolved by

1. adding an FN edge; or
2. removing FP edges due to an identified repeat; or
3. removing a vertex which is either a suspicious chimera or contains an unidentified repeat.

For the non-fixable graphs, we employ a divide-and-conquer method by dividing the graph according to some articulation points such that each subcomponent is fixable. After fixing the subcomponents, the non-fixable graph would become fixable because the articulation point would become I-critical.

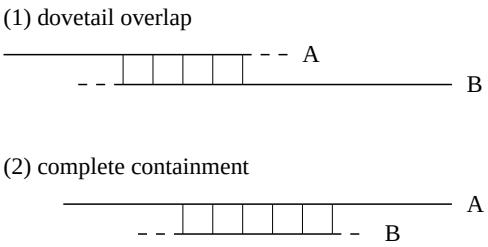


Figure 1:

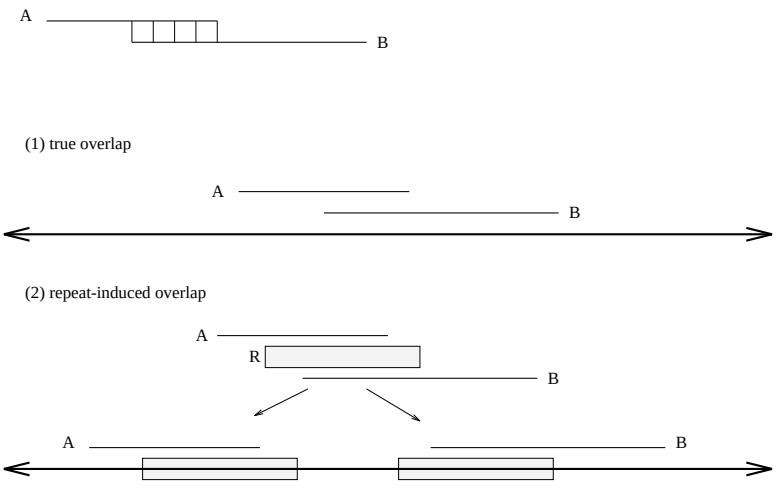


Figure 2:

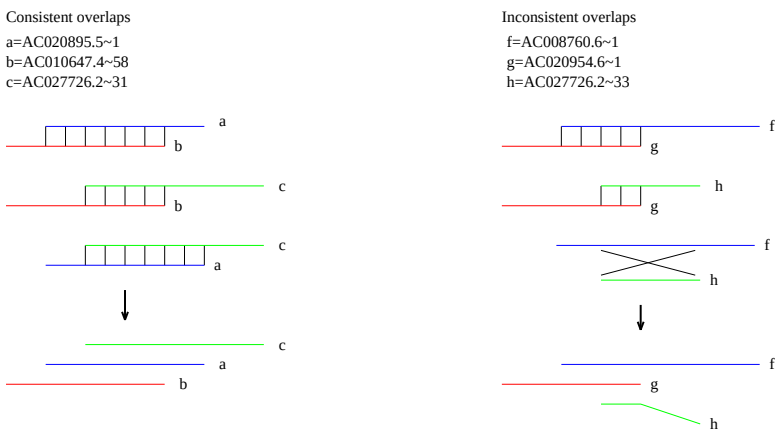


Figure 3:

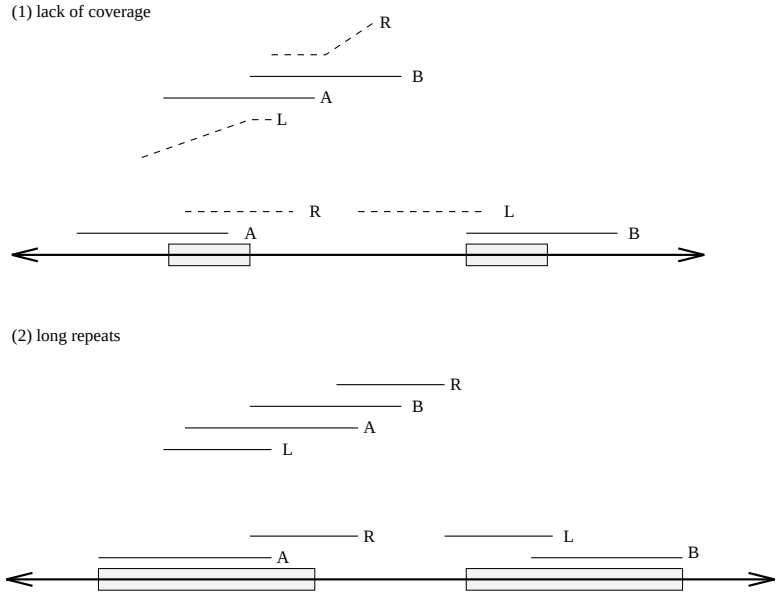


Figure 4:

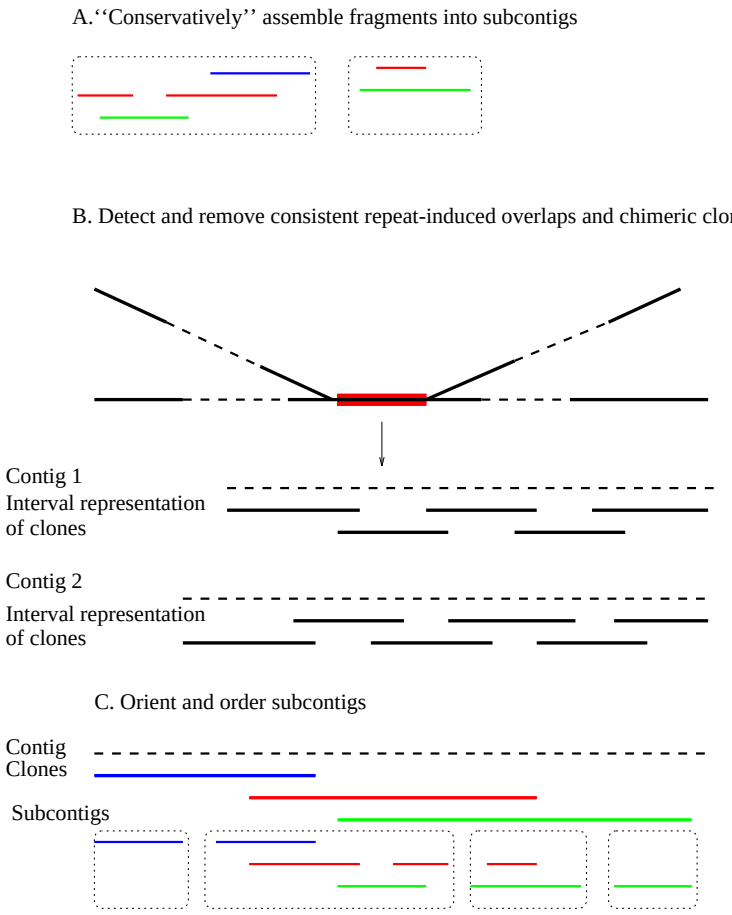

A.‘‘Conservatively’’ assemble fragments into subcontigs
B. Detect and remove consistent repeat-induced overlaps and chimeric clones
Contig 1
Interval representation
of clones
Contig 2
Interval representation
of clones
C. Orient and order subcontigs
Contig
Clones
Subcontigs

Figure 5:

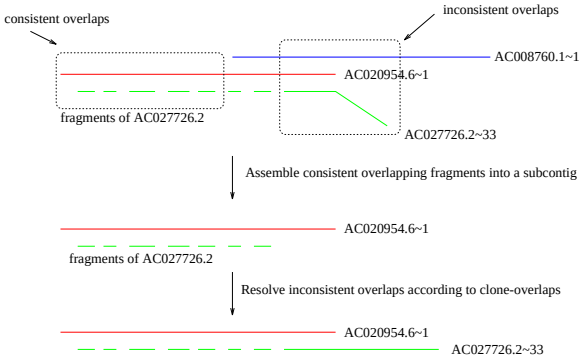


Figure 6:

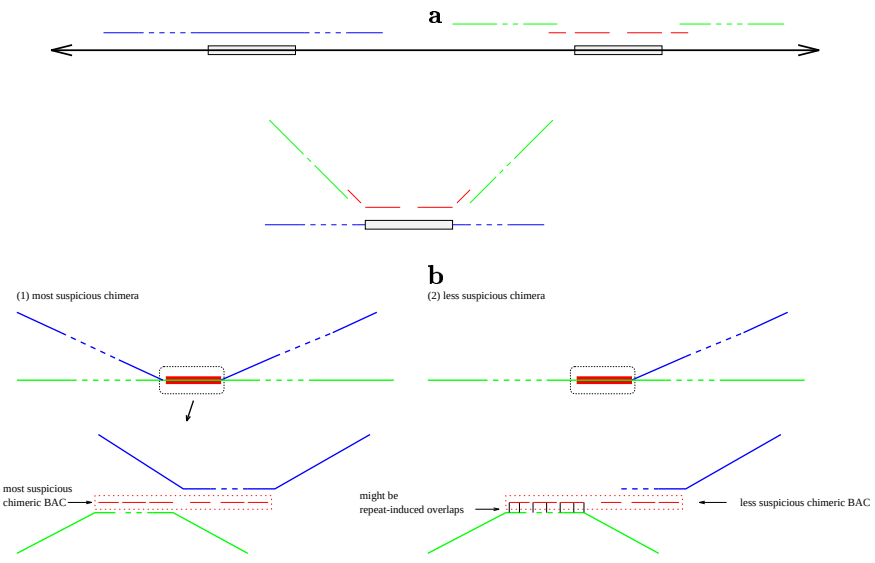
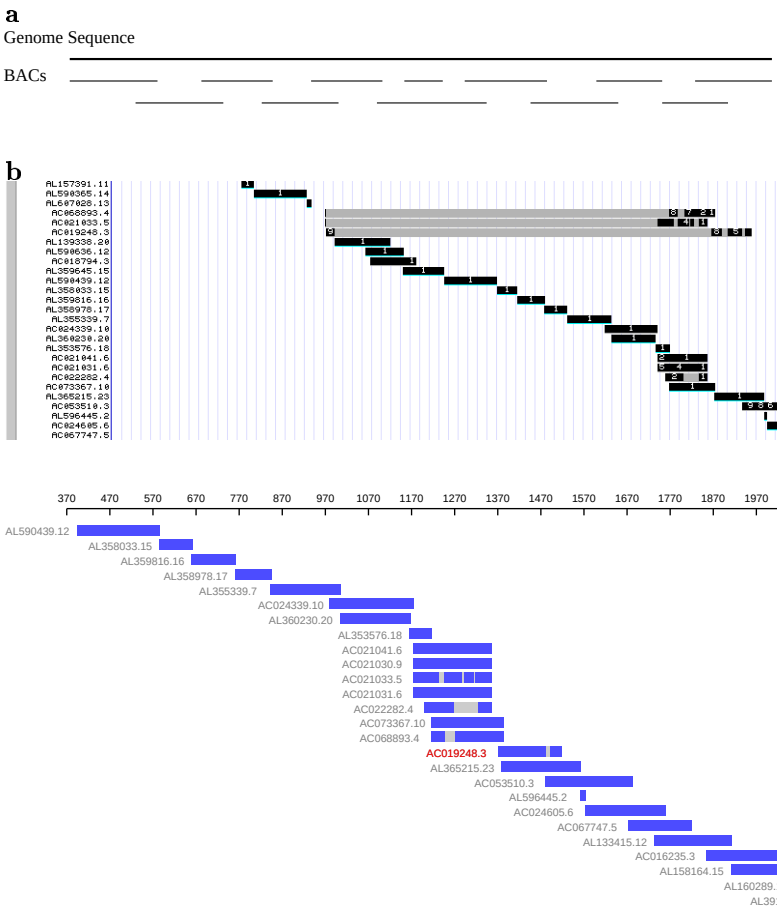


Figure 7:



Clone Information:
Accession: AC019248.3
Chromosome: 10
Laboratory: WIBR
Clone Name: RP11-115G24
Phase: 1
Estimated Length: 147259 bp
Assembled Length: 147725 bp

Figure 8:

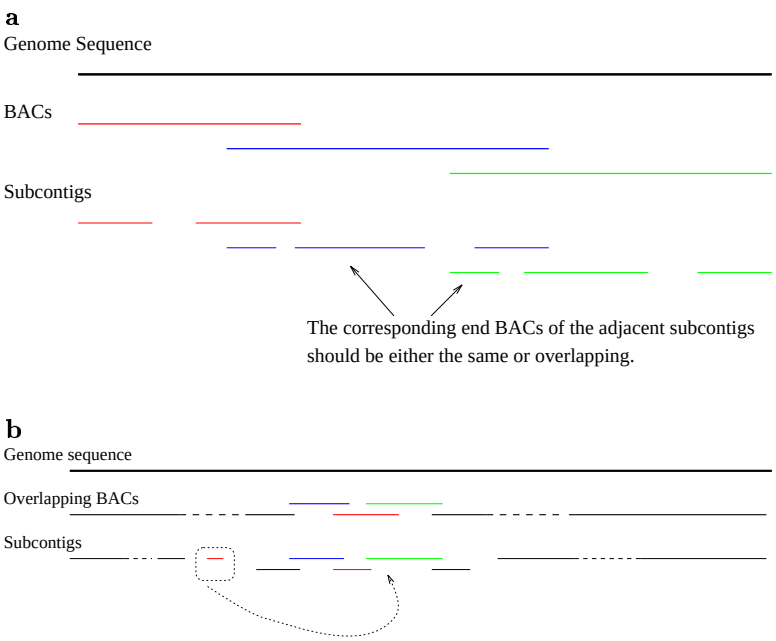


Figure 9:

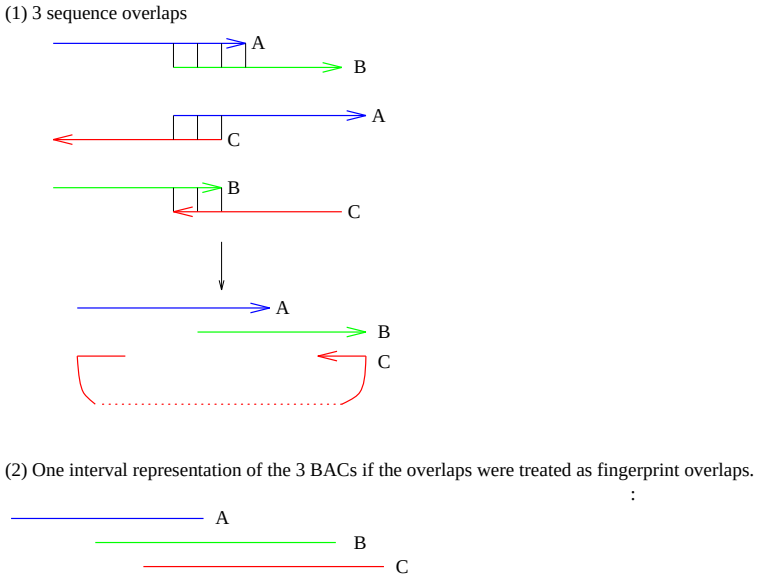
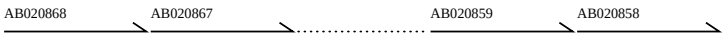
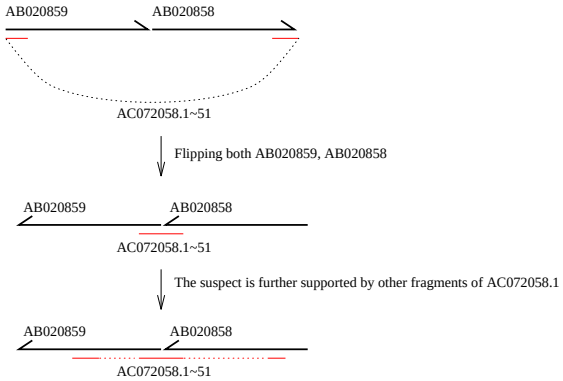


Figure 10:

(1) NCBI's "ngtable" for BACs (AB020858 ... AB020868):



(2) The overlaps of AC072058.1~51 with AB020858, AB020859 result in a very unusual cycle shown below if the ngtable were correct.



(3) Our corrected annotation:

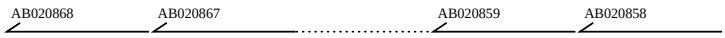


Figure 11:

Figure Legends

Figure 1: Two types of valid overlaps between a fragment pair. (1) dovetail overlap vs. (2) complete containment. Because of the draft quality of sequences, end errors are allowed (shown as dashed lines).

Figure 2: True vs. repeat-induced overlaps. Consider the overlap of two fragments A and B. There are two possible inferences: (1) The overlap is true, where the fragments are from the overlapping segments of the genome. (2) The overlap is induced by repeated sequence R.

Figure 3: Consistent vs. inconsistent overlaps. Left, fragment b overlaps with fragments a and c as shown, and a and c also overlap accordingly. The overlap of b and a is *consistent*. (For the rigorous definition of consistent overlaps, please see (Choi 2001).) Right, fragment g overlaps with fragments f and h as shown, but f and h do not overlap accordingly. The overlaps of g and f, g and h, are *inconsistent*. At least one of them is repeat-induced.

Figure 4: Overlaps that are consistent but not true. There are two possibilities: (1) lack of coverage: absence of both fragments L and R; (2) long repeats: the repeat is long enough such that the overlaps are consistent.

Figure 5: The idea behind our algorithm. Conceptually, the algorithm consists of three steps. (A) Fragments are "conservatively" assembled into subcontigs. The details of this step constitute steps 1-6 of the algorithm. (B) Consistent repeat-induced overlaps and chimeric clones will destroy the interval property of the clone graph. Resolving this step corresponds to step 7 of the algorithm. (C) According to the interval realization of clones obtained from the interval clone graph, subcontigs are oriented and ordered in steps 8-10 of the algorithm. Remark: Steps 11-15 of the algorithm, which use additional information to adjust the ordering and correct the orientation, are not shown in this figure.

Figure 6: Resolving inconsistent overlaps. Since $AC020954.6\tilde{}1$ has several consistent overlaps with fragments of BAC $AC027726.2$, BAC $AC020954.6$ overlaps with BAC $AC027726.2$. The inconsistent overlaps of $AC020954.6\tilde{}1$ and $AC008760.6\tilde{}1$, $AC020954.6\tilde{}1$ and $AC027726.2\tilde{}33$ are resolved according to the clone-overlap. In this case, the overlap of $AC020954.6\tilde{}1$ and $AC027726.2\tilde{}33$ is chosen, i.e. the overlap of $AC020954.6\tilde{}1$ and $AC008760.6\tilde{}1$ is repeat-induced.

Figure 7: **a** The result of collapsing a repeat region. The blue and green denote contigs covered by a set of overlapping BACs; while the red represents a BAC consisting of 4 fragments. For illustrative purposes, suppose the repeat region occurs in the two fragments of the red BAC. The linear structure of the sequence would be destroyed if the repeat region were to be collapsed. **b** Suspicious chimeric BACs. Left, the problematic region occurs in the middle of a pair of contigs, the BAC is the most suspicious chimera. Right, when the problematic region occurs at one end of one contig, it is difficult to tell that it is due to a repeat or whether the BAC is chimeric. To ensure the quality, the BAC is removed.

Figure 8: **a** Geometric view of the sequences. The genome sequence is a line; each BAC corresponds to an interval of the line. **b** Above, the contig of BAC $AC019248$ in NCBI's Build 28, which consists of 10 fragments (only 9 fragments were used in NCBI's Build 28), is non-interval. Biologically, the misassembly of BAC $AC019248$ is also evident by its assembled length of $1,556,292$bp. Below, BARNACLE's interval-graph assembly of the corresponding contig.

Figure 9: **a** The condition of adjacent subcontigs. The corresponding end BACs of the adjacent subcontigs must be either the same or overlapping. **b** FNs detection. No matter how we order the subcontigs, the subcontig in the box will violate the adjacency condition. This is due to a FN as the arrow shows.

Figure 10: Sequence overlap vs. fingerprint overlap. (1) There are 3 sequence overlaps, but they are incompatible. At least one of them is FP. (2) Treated as fingerprint overlaps (whether two fragments overlap) would result in an incorrect interval representation of the 3 BACs.

Figure 11: An example of wrong nt-pairs. According to the annotation of the 11 BACs of 8p21.3-p22 anti-oncogene of hepatocellular colorectal and non-small cell lung cancer ($AB020858 \ldots AB020868$), the NCBI's "ngtable", which is then used to generate nt-pairs, is generated for these BACs as shown in (1). However, our "conservative" assembly detects that all the BACs are in the wrong orientation/strand. For example, the wrong orientation of $AB020858.1$ and $AB020859.1$ (and hence the nt-pair) was detected by a draft BAC $AC072058.1$ (as shown in (2)). Similarly, we have five other draft BACs to support the suspicion of wrong orientation of the other BACs. (3) is the corrected annotation of these BACs.