Identifying and Sorting Tandem Duplications and an Inverted Repeat

Developed by Holly Kotkiewicz, Jennifer Hodges, and Wilson Leung

Prerequisites

A Complex Drosophila Fosmid
Drosophila Finishing Problem Set
GEP Misassembly Tools User Guide

Files for this Exercise

The tutorial files for this exercise are available for download at the GEP web site (under Curriculum ➔ Washington University ➔ Sequence Improvement ➔ Resolving Misassemblies).

Path to the Project Package

This walkthrough assumes the project package is located in your home directory (i.e. the path to the project is `~/1773K10/edit_dir`). If you placed the project package at a different location, you will need to change the path to the project. For example, if the project package is on your Desktop, the path to the project would become `~/Desktop/1773K10/edit_dir`.

Introduction

Many of the *Drosophila ananassae* Muller F element sequence improvement projects contain multiple misassemblies. This walkthrough will demonstrate the tools and techniques you could use to resolve these misassemblies.

Resolving misassemblies is generally not a linear process and will likely require you to experiment with different strategies. Hence it is essential to keep detailed notes so that you can backtrack when necessary. It may also be helpful to add descriptive words to the names of the ACE files as you save them. Projects with major misassemblies are often missing a substantial number of reads (e.g. because of collapsed repeats). We have created several tools to help you search for, download and incorporate new reads into your project. Please refer to the "GEP Misassembly Tools User Guide" for additional instructions on how to identify projects with missing reads and on how to retrieve missing reads from the NCBI Trace Archive.

Note: In this walkthrough, we will perform many tears and joins in order to resolve the misassemblies. Consed will assign a new number to each contig each time you perform a tear or a join. Consequently, depending on the order in which you perform these steps, the contig numbers shown in this exercise might not correspond exactly to the contig numbers in your project.

Examine Forward/Reverse Mate Pairs with Assembly View

Launch X11 and open a new `xterm`; then navigate to the `edit_dir` of the *D. ananassae* project 1773K10 (`cd ~/1773K10/edit_dir`). Enter `consed&` at the `xterm` prompt. The "&" will keep your terminal active in case you need to use it later. Open `1773K10.fasta.screen.ace.1`. Select "No" if a prompt appears which asks if you would like to apply edits from the edit history (`.wrk`) file.

Click on the "Assembly View" button on the Consed Main Window to assess the current state of the assembly (Figure 1). The large number of red lines under the grey contig boxes indicates that there are multiple misassemblies in this project (Figure 2).

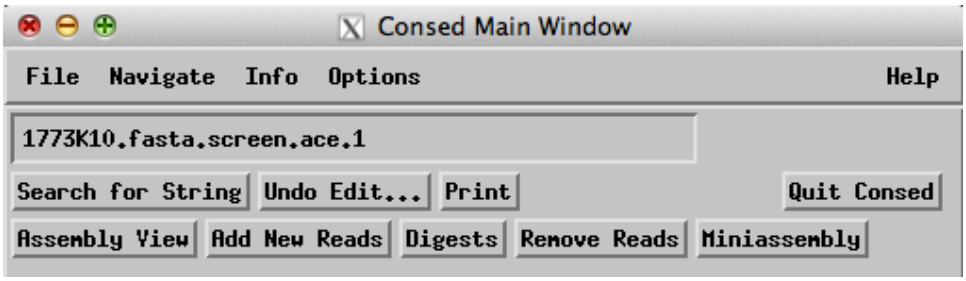


Figure 1 Click on the "Assembly View" button to get a general overview of the project.

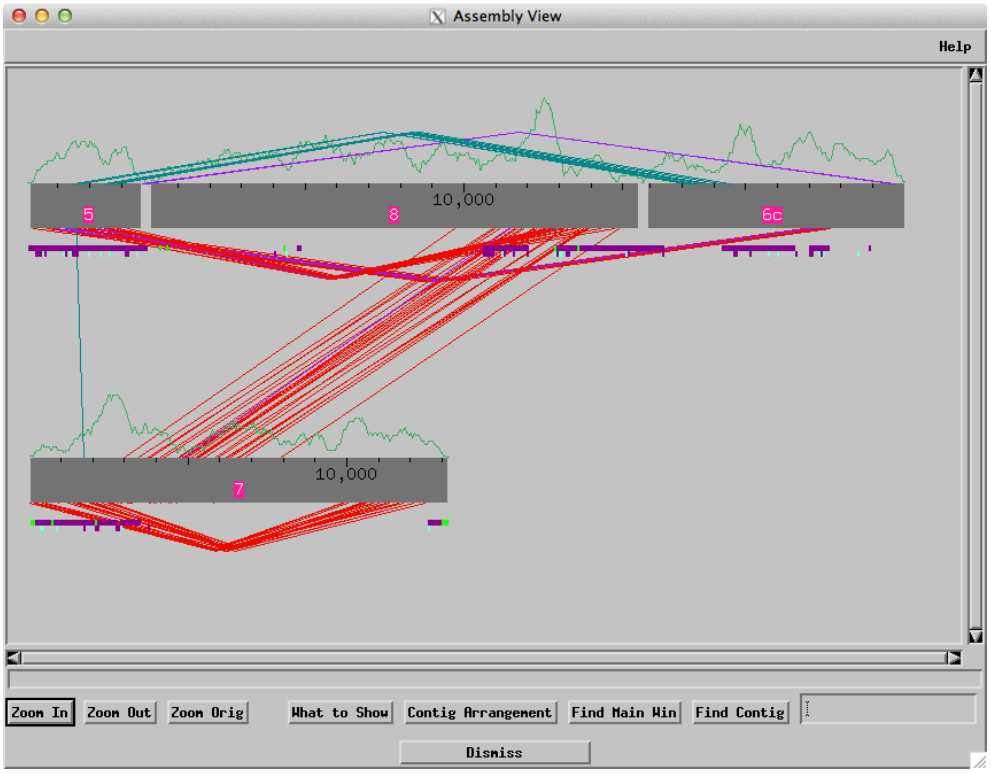


Figure 2 Initial Assembly View shows multiple sets of inconsistent forward-reverse mate pairs.

By default, Assembly View will only show inconsistent mate pairs and consistent mate pairs that span multiple contigs. To see all the consistent forward/reverse mate pairs, click on "What to Show" and then click on "Fwd/Rev Pairs". Toggle "on" the following options:

- show consistent fwd/rev pair depth
- show each consistent fwd/rev pair within contigs
- show gap-spanning fwd/rev pairs
- show consistent fwd/rev pairs between diff scaffolds
- show legs on squares for consistent fwd/rev pairs

The upper dark triangle on the top left corner of the small box indicates that the item is enabled. Click "Apply" once you have enabled all of the check boxes (Figure 3).

Figure 3 Enable all the options to show both consistent and inconsistent mate pairs in Assembly View

Dismiss the dialogue box and you should see many dark blue triangles above the grey contig boxes in Assembly View. These blue triangles denote consistent forward/reverse mate pairs (Figure 4). Note that we need to enable these options again in order to show the consistent mate pairs when we restart Assembly View.

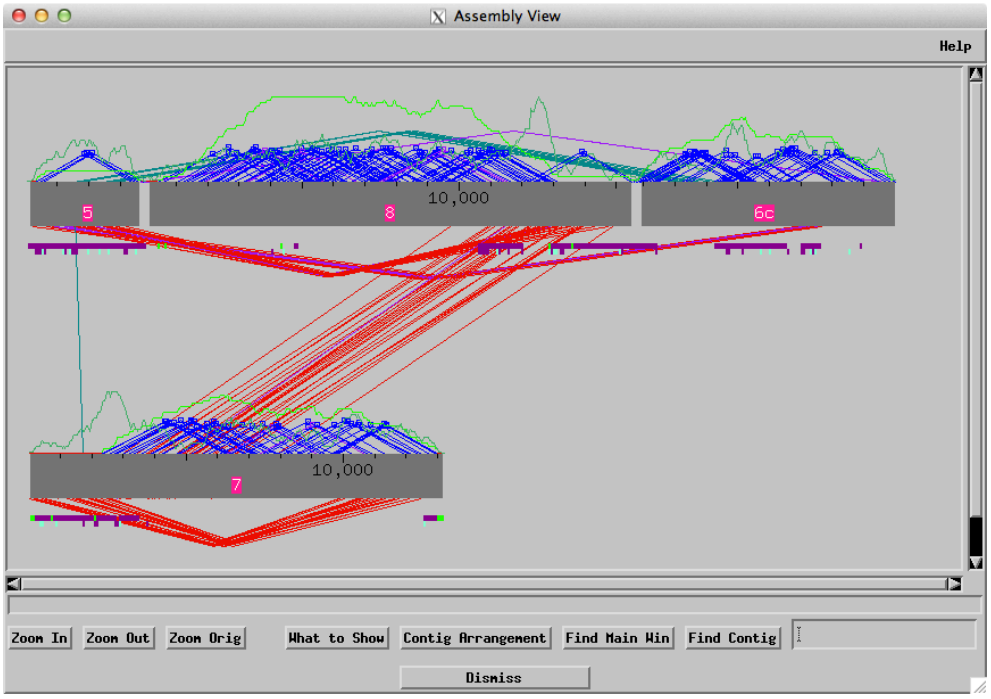


Figure 4 Blue triangles above the grey contig boxes indicate consistent forward/reverse pairs.

Using High Quality Discrepancies to Identify Misassembled Regions

A typical first step when dealing with misassemblies is to look for regions with multiple high quality discrepancies. These discrepancies can usually be attributed to either polymorphisms or misassemblies. If they are judged to indicate misassemblies, we can tag the discrepant reads to tell `phrap` not to overlap the reads if they differ at that position.

By default, `consed` considers discrepant bases with a quality score of 40 or above to be a high quality discrepancy. While this default threshold works well in general, many misassemblies are caused by repetitive sequences that are more difficult to sequence and are therefore more likely to be low quality. Consequently, for projects with major misassemblies, we might want to lower the high quality discrepancy threshold in order to increase the sensitivity in detecting potentially misassembled regions. Because we consider bases with a quality score of less than or equal to 30 to be low quality, we will change the high quality discrepancy threshold from 40 to 31.

To change the high quality discrepancy threshold, click on the "Options" button on the Consed Main Window and select "General Preferences". Change the value of the "Threshold for High Quality Discrepancy (lowest high)" field to **31** and then click on the "Apply & Dismiss" button.

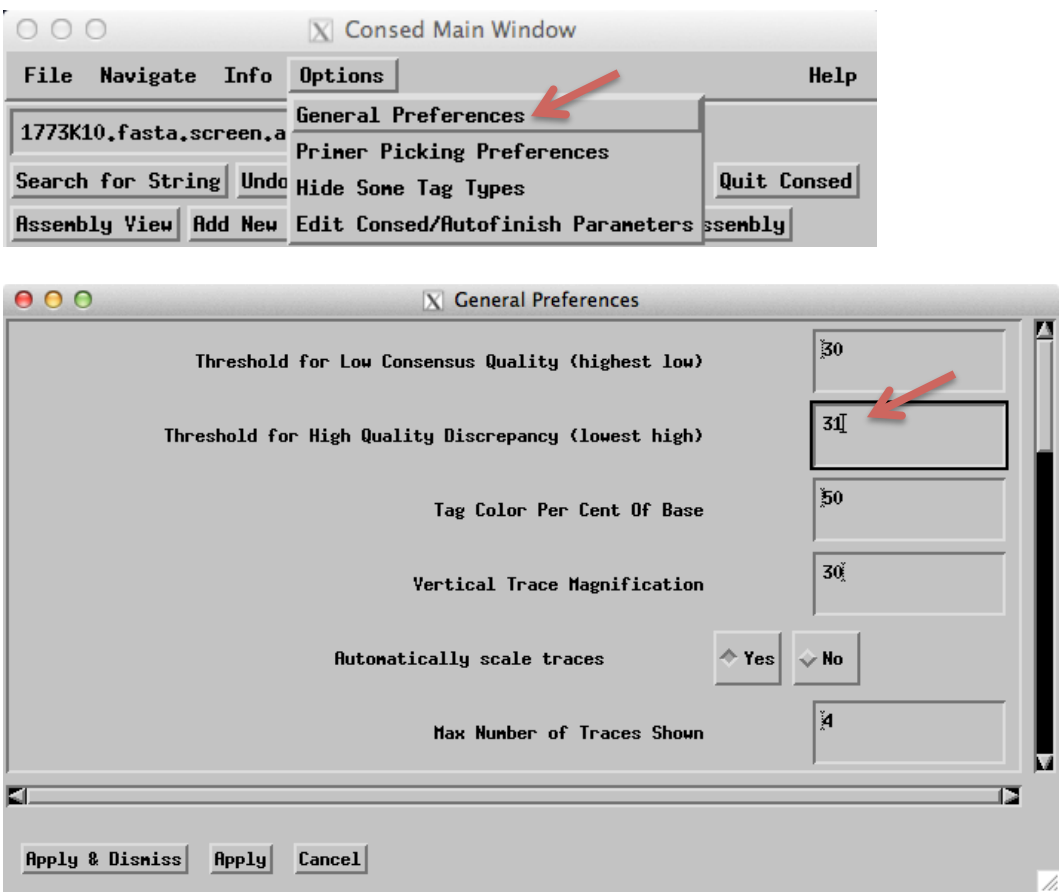


Figure 5 Change the threshold for high quality discrepancy from 40 to 31.

To see all the regions with multiple high quality discrepancies using this new high quality discrepancy threshold, click on the "Navigate" button on the Consed Main Window and select "Multiple High Quality Discrepancies" (Figure 6).

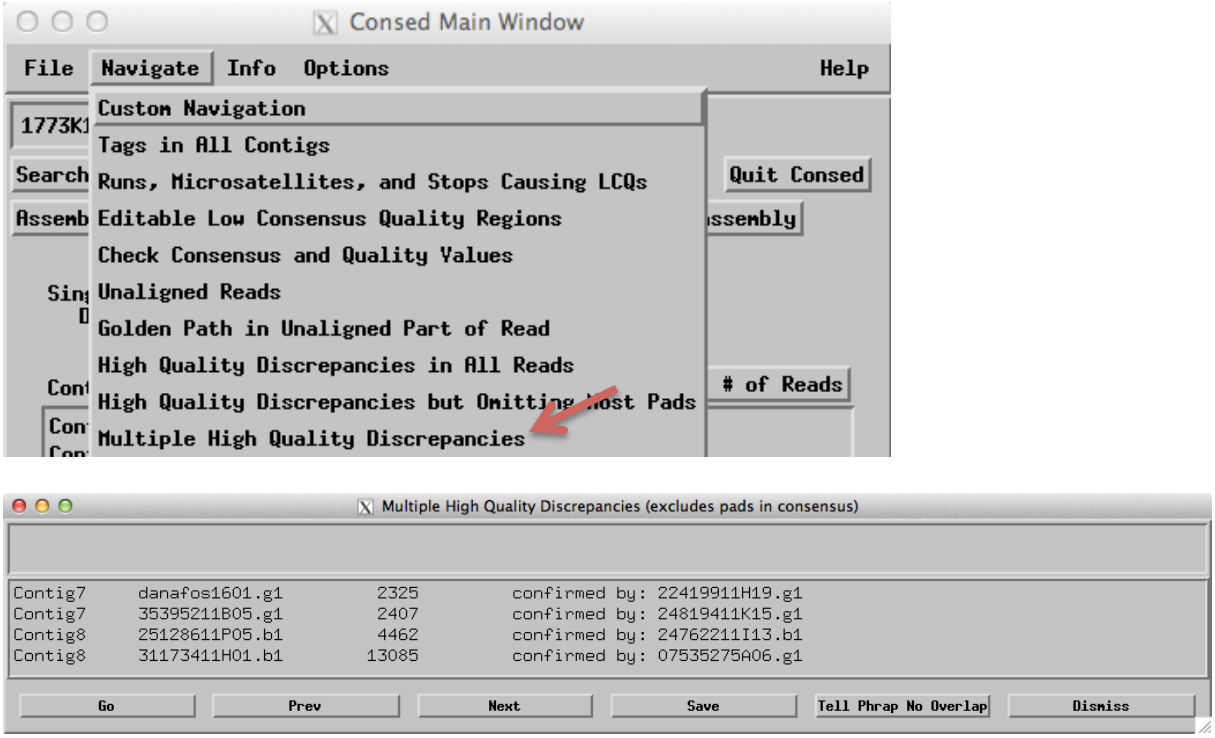


Figure 6 Four regions in this project with multiple high quality discrepancies

A new dialogue box will appear which shows that there are four positions in the assembly with multiple high quality discrepancies. To navigate to these discrepant regions, highlight the first discrepancy (Contig7 at position 2,325) in the navigation dialogue box and click "Go". You should tag these regions to not overlap unless you have good evidence for polymorphism. (Given the high repeat density, we suspect that the reads with high quality discrepancies are derived from different copies of the same repeat. However, be sure to inspect the traces to confirm that the discrepancies are high quality.)

To tag each of these positions, go back to the "Multiple High Quality Discrepancies" dialogue box and click on "Tell Phrap No Overlap". This will add an orange "markedHighQuality" tag to all the reads at this position (Figure 7). Repeat this step for the other three regions with multiple high quality discrepancies in contigs 7 and 8. Save the assembly (`1773K10.fasta.screen.ace.sorting`).

Note: For teaching purposes, you will find the ACE files described in this walkthrough in the `edit_dir` of the project directory. If you would like to save the assembly without overwriting these ACE files, you could append your initial) to the name of the ACE file described in this walkthrough when you save the assembly (e.g. `1773K10.fasta.screen.ace.sorting.WL`).

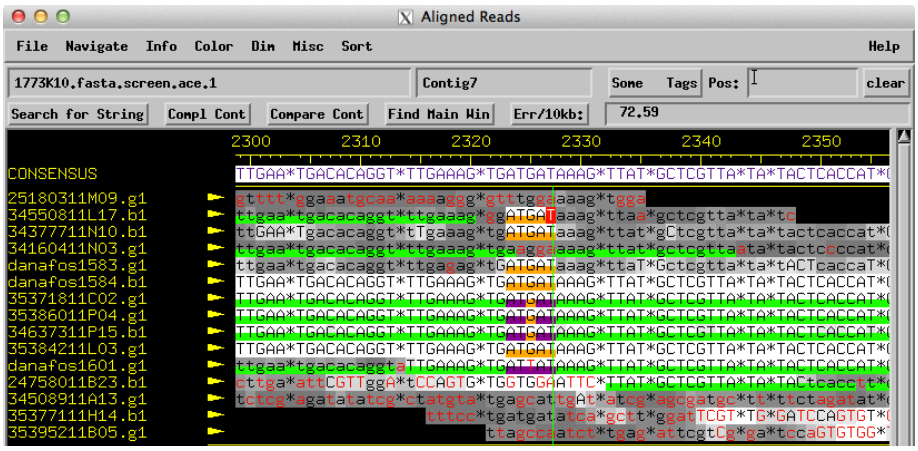


Figure 7 The orange "markedHighQuality" tags tell `phrap` not to overlap discrepant reads at this position.

Using Miniassembly to Assemble a Subset of Contigs

Click on the "Miniassembly" button in the Consed Main Window. This will bring up the "Reassemble Some Contigs" dialogue box. Select Contig 5 under the "All Contigs" section and then hold down the shift key to also select contigs 6, 7, and 8. Click on "Move Highlighted to Right". Once these contigs appear under the "Contigs to Reassemble" section, click on the "Reassemble" button to assemble these four contigs (Figure 8).

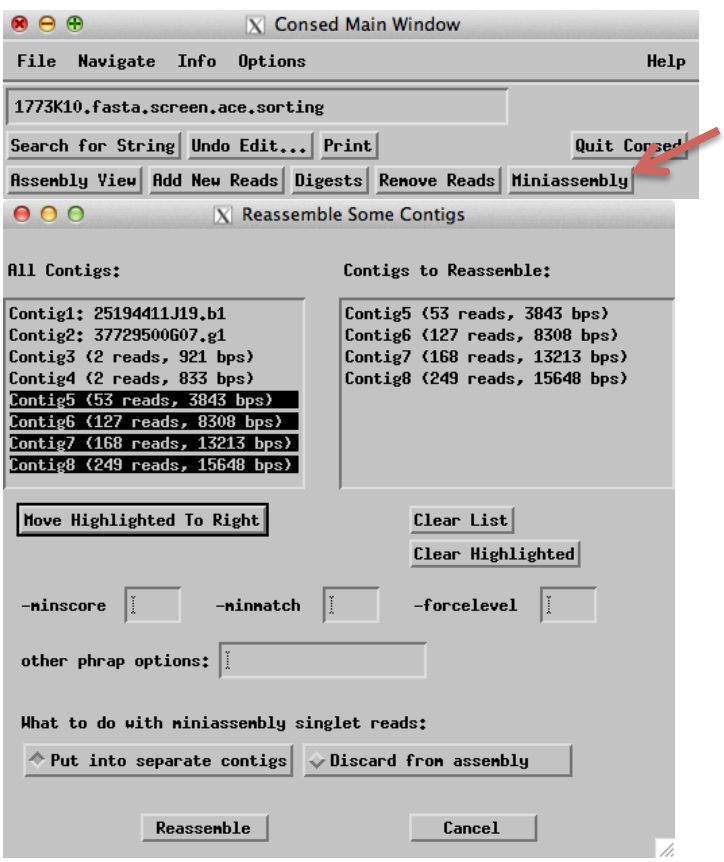


Figure 8 Reassemble contigs 5, 6, 7, and 8 using Miniassembly.

Miniassembly will construct a new assembly using all the reads found in contigs 5, 6, 7 and 8, and it will not overlap reads that are discrepant at the positions we have tagged previously. After Miniassembly is complete, you should see a new dialogue box that asks if we are finished with Miniassembly (Figure 9). Click on "Yes" to dismiss the "Reassemble Some Contigs" window. Save the new assembly (`1773K10.fasta.screen.ace.sorting.1`).

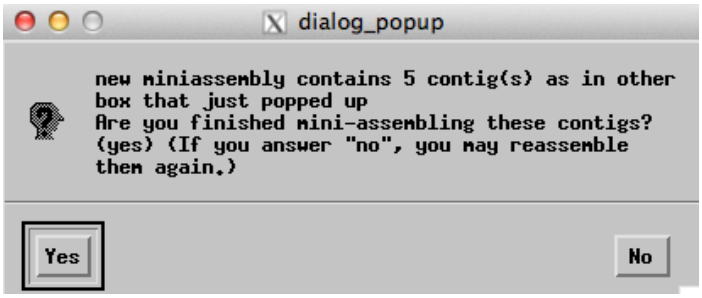


Figure 9 Click on "Yes" to dismiss the "Reassemble Some Contigs" window after Miniassembly is complete.

Open Assembly View so that we can examine the state of the project after Miniassembly. Assembly View shows that our project still consists of four major contigs with multiple clusters of inconsistent forward reverse mate pairs (Figure 10). In addition to the four major contigs, Miniassembly also created a small (3 reads, 877bps) Contig5, which is not shown in Assembly View under the default settings. We will defer incorporating Contig5 into the main assembly until we have resolved the misassemblies among the major contigs.

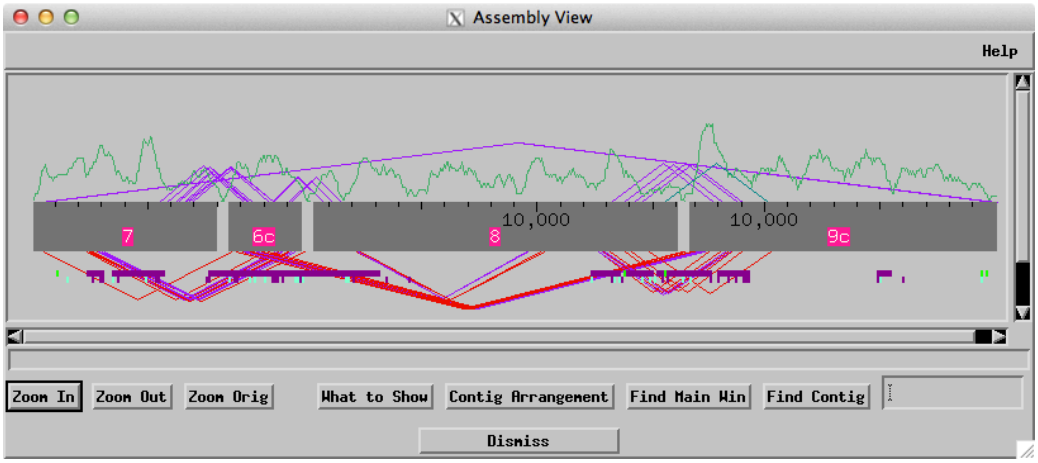


Figure 10 New Assembly View shows contigs 6 and 9 are in the complemented orientation

Notice Contig6 and Contig9 have a small "c" after their contig numbers. This indicates that there are forward/reverse mate pairs that anchor these contigs in the complemented orientation relative to the scaffold. Consequently, we need to complement contigs 6 and 9 so that these contigs are in the correct orientation with respect to the other contigs.

Click on "Contig Arrangement" and select "Reorient Contigs". Select "7-6c-8-9c" under the "Select a scaffold" section and select "Make all contigs the same orientation as the scaffold (remove each 'c') and retain the scaffold orientation and the new orientations of all its contigs" (Figure 11). Click on "Apply and Restart Assembly View". This will put contigs 6 and 9 in the correct orientation. Save the assembly (`1773K10.fasta.screen.ace.sorting.2`).

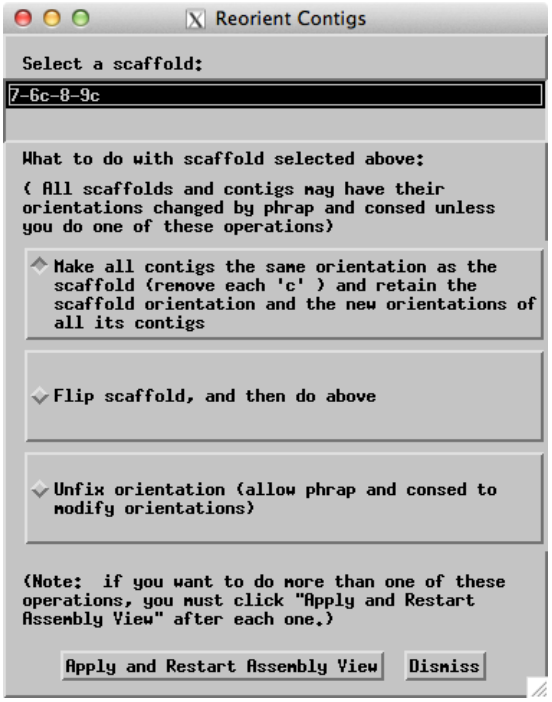


Figure 11 Complement contigs 6 and 9 so that they are in the orientation suggested by Assembly View.

Identifying the Fosmid Ends

Another step in resolving the misassembly is to identify the fosmid ends and mask reads (i.e. change to X's) that extend beyond the insert/fosmid boundary. This will help us elucidate the relative order and orientation of the contigs and demarcate the misassembled regions in our current assembly. In the Consed Main Window, type "end" in the text box next to the "Find reads containing (*'s allowed)" field (Figure 12). A new dialogue box will appear with a list of the fosmid end reads that are found in this project (Figure 13).

Figure 12 Search for the fosmid end reads using the "Find reads containing" field in the Consed Main Window.

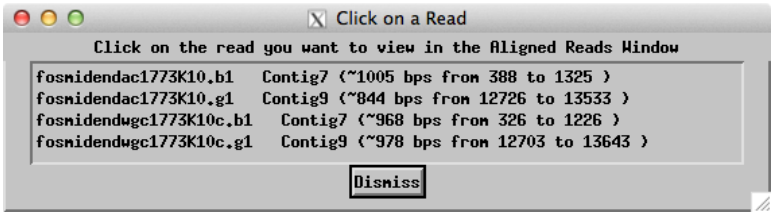


Figure 13 The fosmid end reads in the project 1773K10.

Click on the read `fosmidendac1773K10.b1`. This read is placed at Contig7 from 388 to 1,325. Notice that `fosmidendwgc1773K10c.b1` is also in Contig7 at approximately the same location. However the beginning of that read starts with X's (i.e. clipped vector sequences). This is the left end of the fosmid 1773K10. The high quality data to the left of position 383 are extra data from the whole genome reads that extends beyond the end of the fosmid. Consequently, we will change all of the bases to the left of consensus position 383 to X's.

Left click on the consensus position at 382 in the Aligned Reads window and then select "Change to X's to Left in All Reads" under "Misc" (Figure 14). Consed might, as it does in this case, change the consensus coordinates and the first unmasked base is now at consensus position 389. Click on base 389 of the consensus and select "Add Clone End Tag with Insert to the Right" under "Misc" (Figure 15). Use the "Click on A Read" dialogue box to navigate to the end of Contig9 and add a "cloneEnd" tag to the other end of the fosmid clone by selecting the option "Add Clone End Tag with Insert to the Left" under "Misc". Save the new assembly (`1773K10.fasta.screen.ace.sorting.3`).

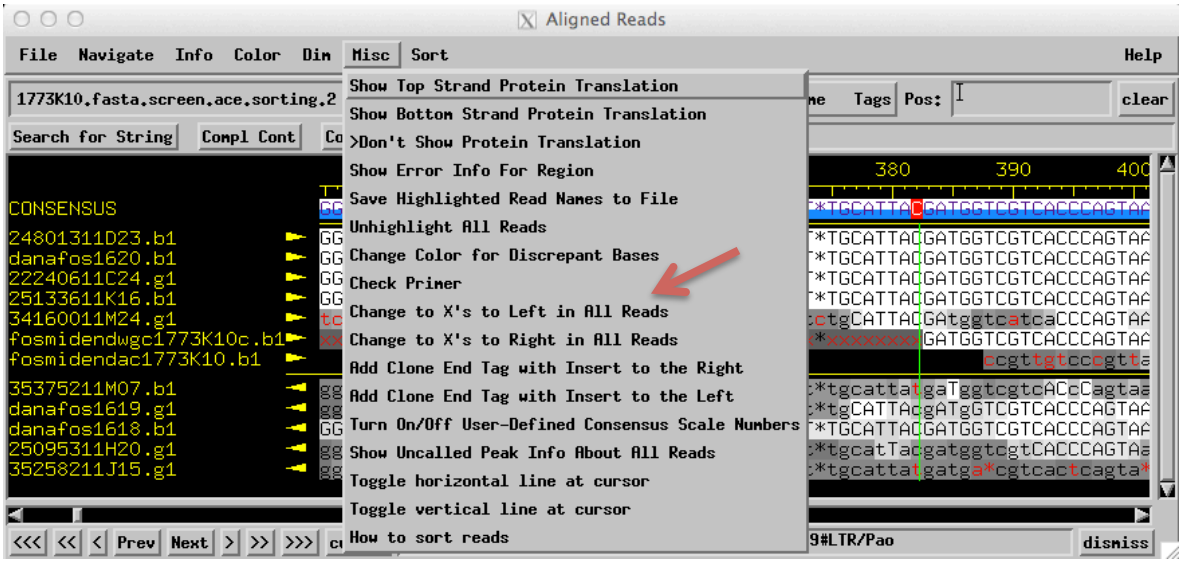


Figure 14 Change bases to the left of base 383 in the consensus to X's.

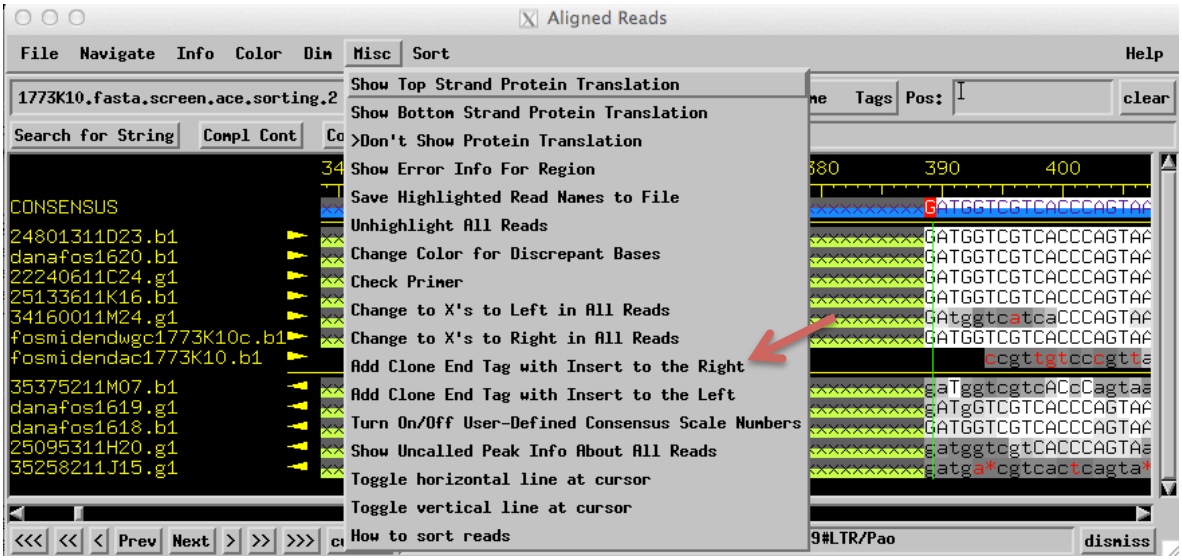


Figure 15 Add clone end tag with insert to the right at position 389 of the consensus.

When you restart Assembly View, the clone end tags will appear as salmon colored arrows that point toward the insert (Figure 16).

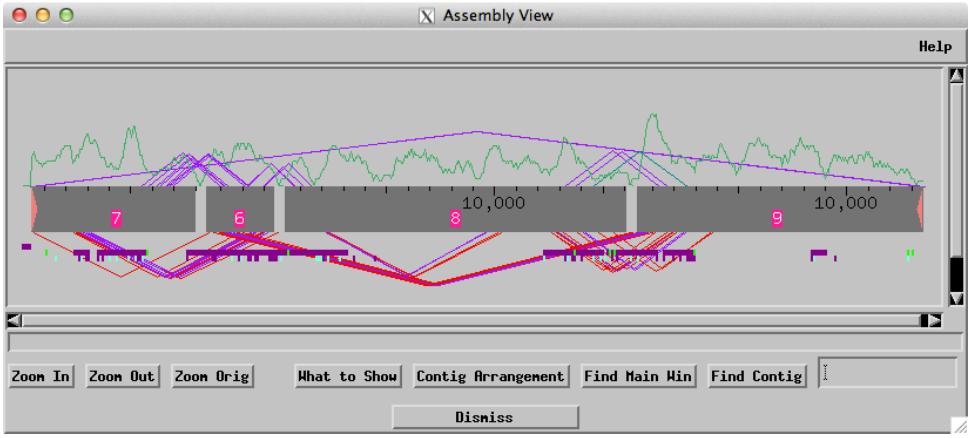


Figure 16 The salmon colored arrows in Assembly View mark the locations of the fosmid clone ends.

Analyzing a Duplication Using crossmatch

At this point, we have established a basic framework for this assembly by tagging the fosmid clone ends and breaking apart regions with obvious misassemblies as indicated by the presence of multiple high quality discrepancies. The next step is to identify the different repeat copies that are in this project and place the reads in the correct copy of the repeat.

We will start by analyzing the repeat structure in our current assembly using `crossmatch`. Select the "What to Show" button at the bottom of Assembly View and then click on "Sequence Matches." Click on "run crossmatch" (Figure 17).

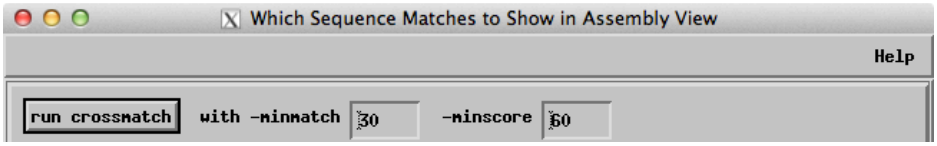


Figure 17 Run crossmatch to identify repetitive regions within the assembly.

Repetitive regions will be shown graphically in Assembly View after you ran `crossmatch`. The orange boxes and lines denote direct repeats and the black boxes and lines indicate inverted repeats (Figure 18).

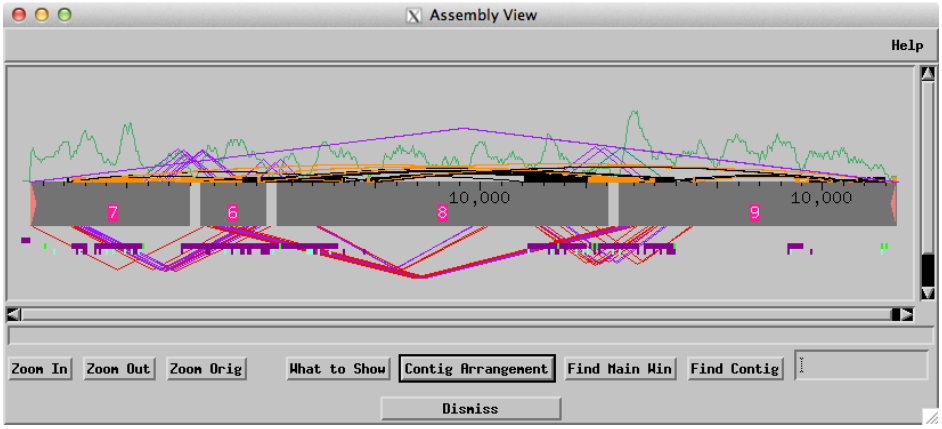


Figure 18 crossmatch found multiple direct (orange) and inverted (black) repeats in our project.

Left click on the repeats at the right end of Contig7. A new dialogue box will appear which shows four different repetitive regions (Figure 19).

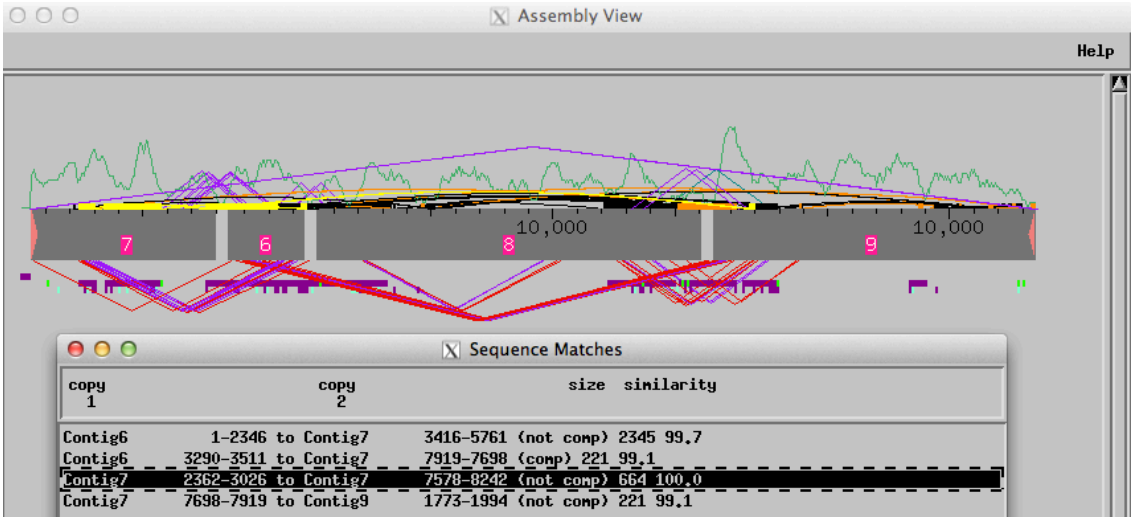


Figure 19 Sequence Matches dialogue box showing sequence similarity within and between contigs.

According to the Sequence Matches dialogue box, the right end of Contig7 (7,578-8,242) matches perfectly with itself near the beginning of Contig7 (2,362-3,026) for 664 bases. Hence this part of the contig contains a direct repeat (also known as a duplication). Double click on this match or highlight it and click on the "Show Alignment" button. This will bring up a "Compare Contigs" window with the alignment between these two repeat copies. We can see high quality discrepant bases just before the beginning of this repeat (Figure 20).

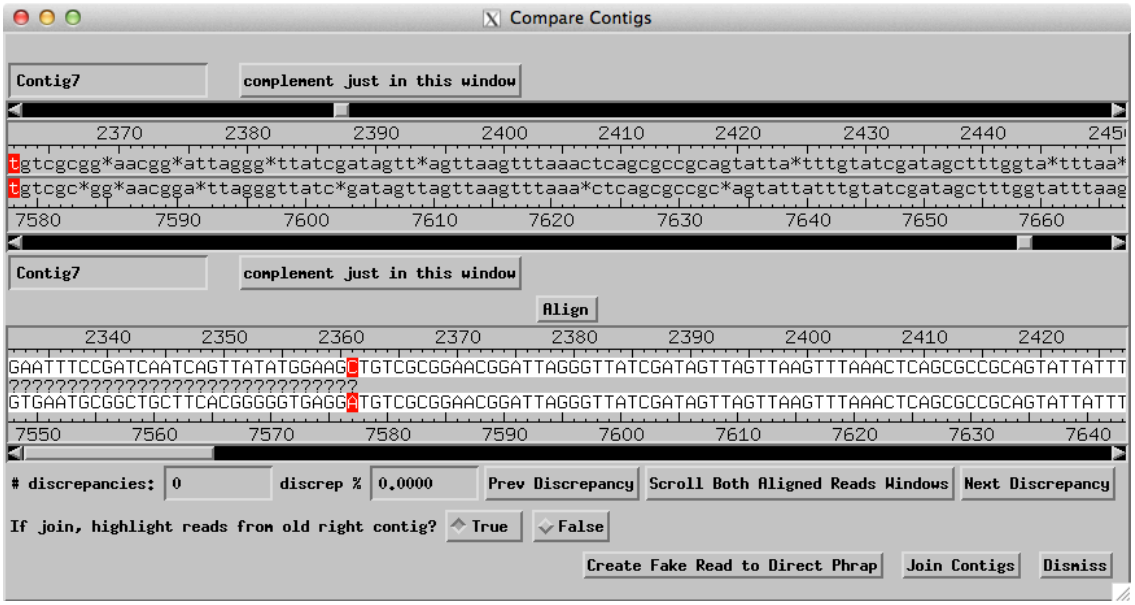


Figure 20 Alignment between the repeat copies at the beginning and at the end of Contig7.

To summarize our analysis thus far, the first copy of the duplication begins at 2,362 in Contig7 and ends at 3,026. The second copy of the duplication starts at 7,578 and terminates at the end (8,242) of Contig7. We will refer to this duplication as **Duplicate 1**.

Because the second copy of this duplication terminates at the end of Contig7, there could be additional data beyond the end of the first copy of the repeat at 3,026 that actually belongs to the second copy. This data might have been misassembled into the wrong copy of the repeat because of the high degree of similarity between the two repeat copies. Data misassembled like this is known as "buried data" because it is "hiding" in the wrong repeat copy. The next step is to ascertain if there is any evidence of buried data and to determine the extent of the duplication.

Looking at the Sequence Matches window, we noticed there is another duplication between the beginning (1-2,346) of Contig6 and the middle part (3,416-5,761) of Contig7, with 99.7% similarity. We will refer to this duplication as **Duplicate 2**. This is likely the second part of the duplication. Consequently, our analysis suggests the actual duplication could be much larger and it could encompass both Duplicates 1 and 2. The gap between contigs 7 and 6 might have interrupted the second copy of this much larger duplication (Figure 21).

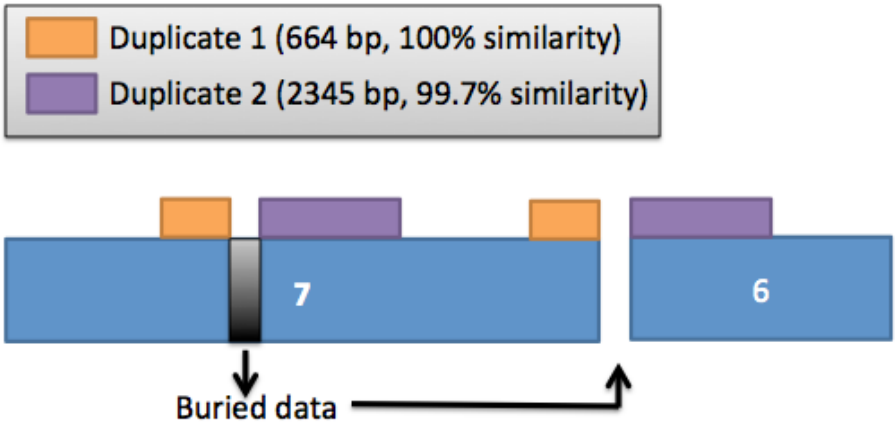


Figure 21 Proposed model of the large duplication between contigs 6 and 7.

Identifying Unclipped Vector Sequences

Because we will rely heavily on high quality discrepancies to distinguish different repeat copies from each other, we should be aware of the fact that some of the high quality discrepancies could be attributed to unclipped sequencing vectors at the beginning of the *D. ananassae* reads.

Unclipped vector sequences are typically found within the first 50 bases of the read at the vector/insert junction. For the *D. ananassae* whole genome plasmid library, this junction is an *Eco*RI restriction site (with the sequence GAATTC). (This sequence may be different for other libraries.) The *Eco*RI cut site is preceded by the sequence "GTGTGGTG" on the positive strand (Figure 22) and followed by the sequence "CACCACAC" on the minus strand (Figure 23). In the rest of this walkthrough, we will ignore these unclipped vector sequences because they are not genuine discrepancies. You may want to mask these vector sequences so they do not appear on any subsequent discrepancy lists. To change the vector sequences to X's, middle click on the edit (edt) line at the *Eco*RI cut site in the Trace Window and select "Change to x's to [left, right]" as appropriate.

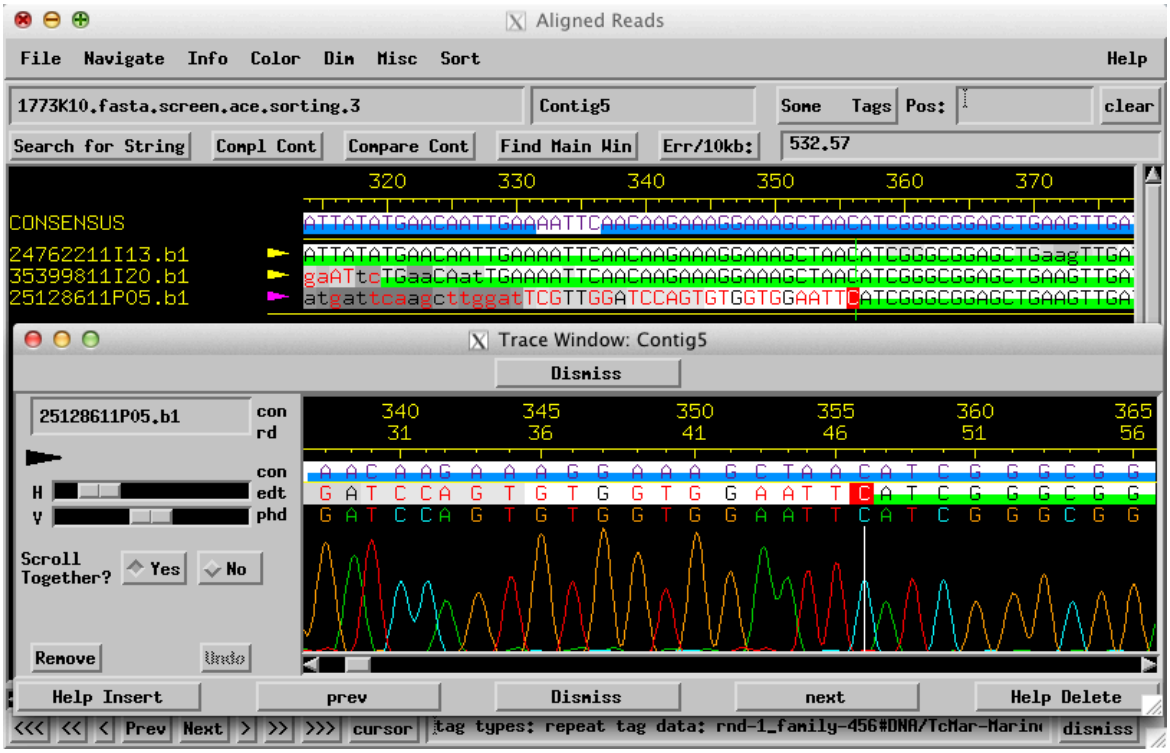


Figure 22 Unclipped vector sequence at the beginning of the read `25128611P05.b1`

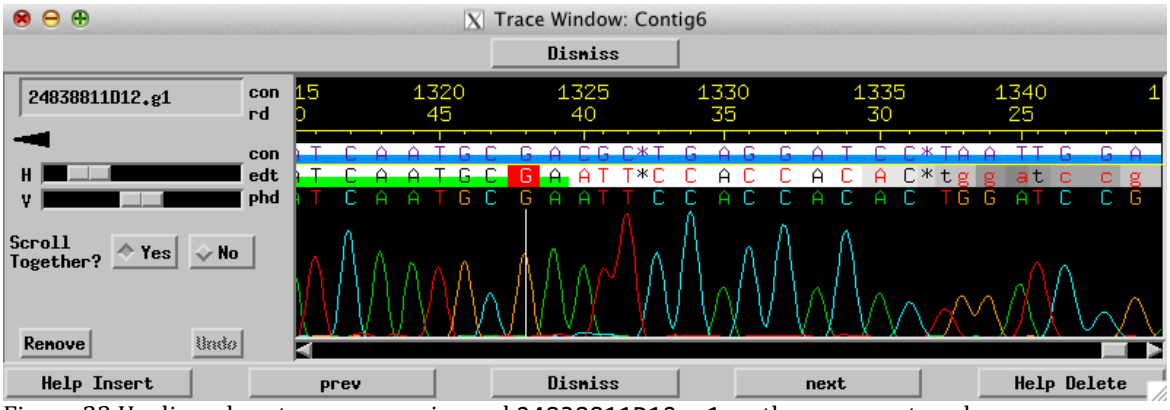


Figure 23 Unclipped vector sequence in read `24838811D12.g1` on the reverse strand.

Identifying Buried Reads in Contig7

To test the hypothesis that some of the reads that spanned contigs 6 and 7 might have been buried in Contig7, we will examine the alignment between these two regions more closely. Double click on the 2.3kb match (Duplicate 2) between contigs 6 and 7 or highlight it and click on the 'Show Alignment' button in the Sequence Matches dialogue box. Then click on the 'Next Discrepancy' button in the Compare Contigs window (Figure 24).

This will bring up the Aligned Reads windows for both Contig6 and Contig7 at the first discrepant position (i.e. base 20 in Contig6). The first two discrepancies that come up are in a low quality area on Contig6 so they should be ignored.

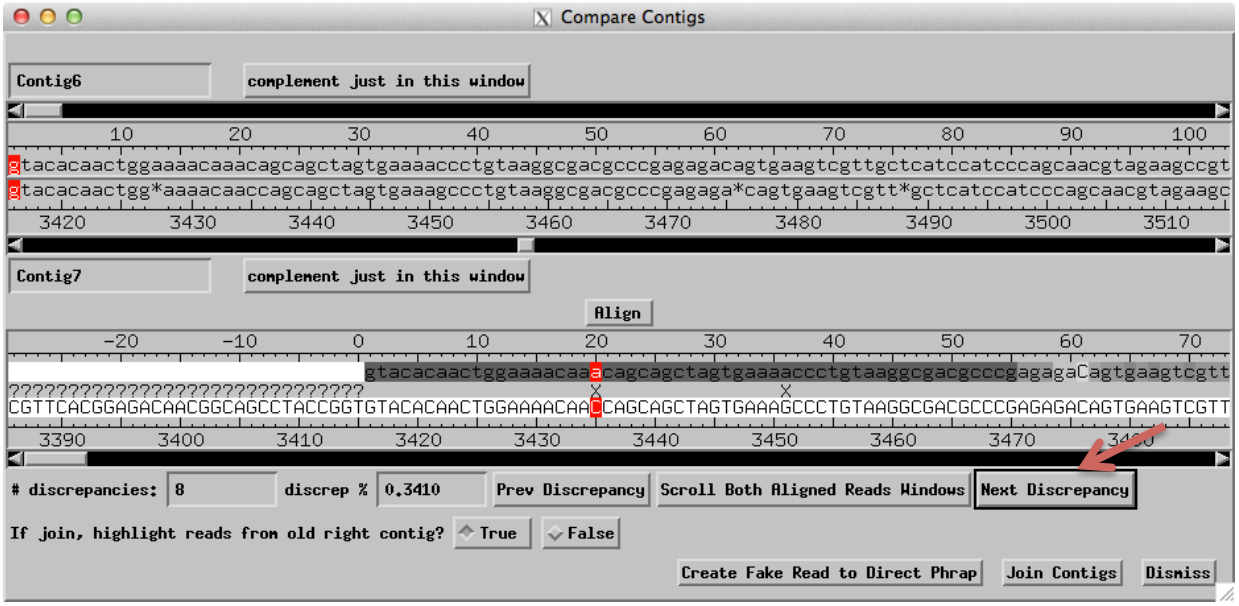


Figure 24 Click on the "Next Discrepancy" button in the Compared Contigs window to navigate to the next base discrepancy between the aligned region of contigs 6 and 7.

The third discrepancy at base position 122 of Contig6 is higher quality so we will take a closer look at this region. Open all the traces on Contig6 at this position by middle clicking on each trace. We can see that the consensus at this position is based on the C from the sequencing vector in the read `34372111L18.b1` instead of the real base (T) in the other two reads (Figure 25). Therefore, we need to change the consensus to the correct base. In the Trace Window, middle-click the "T" on the edit (edt) line for one of the traces (i.e. `25104011O21.b1` or `38176500P19.b1`) with the correct "T" base and select "Change Consensus".

Click on the "Next Discrepancy" button in the Compared Contigs window again. This should take you to a genuine base pair change between the repeat copies in Contig6 and Contig7 (T versus A, Figure 26). Add a comment tag to both positions to indicate that the discrepancy is real. Place the cursor on the Contig6 consensus at base 628. Middle click, select "comment" and enter the comment "base pair change" in the text box and then click "OK". Repeat the same steps to add the same comment to position 4,043 on Contig7. Then save the assembly (`1773K10.fasta.screen.ace.sorting.4`).

Based on our analysis of the high quality discrepancies, we have successfully defined the boundary of where the data for the gap between contigs 6 and 7 could have been buried. We will look for evidence of buried data from 2,362 (beginning of Duplicate 1) to 4,042 (the base before the first genuine high quality discrepancy between contigs 6 and 7) in Contig7.

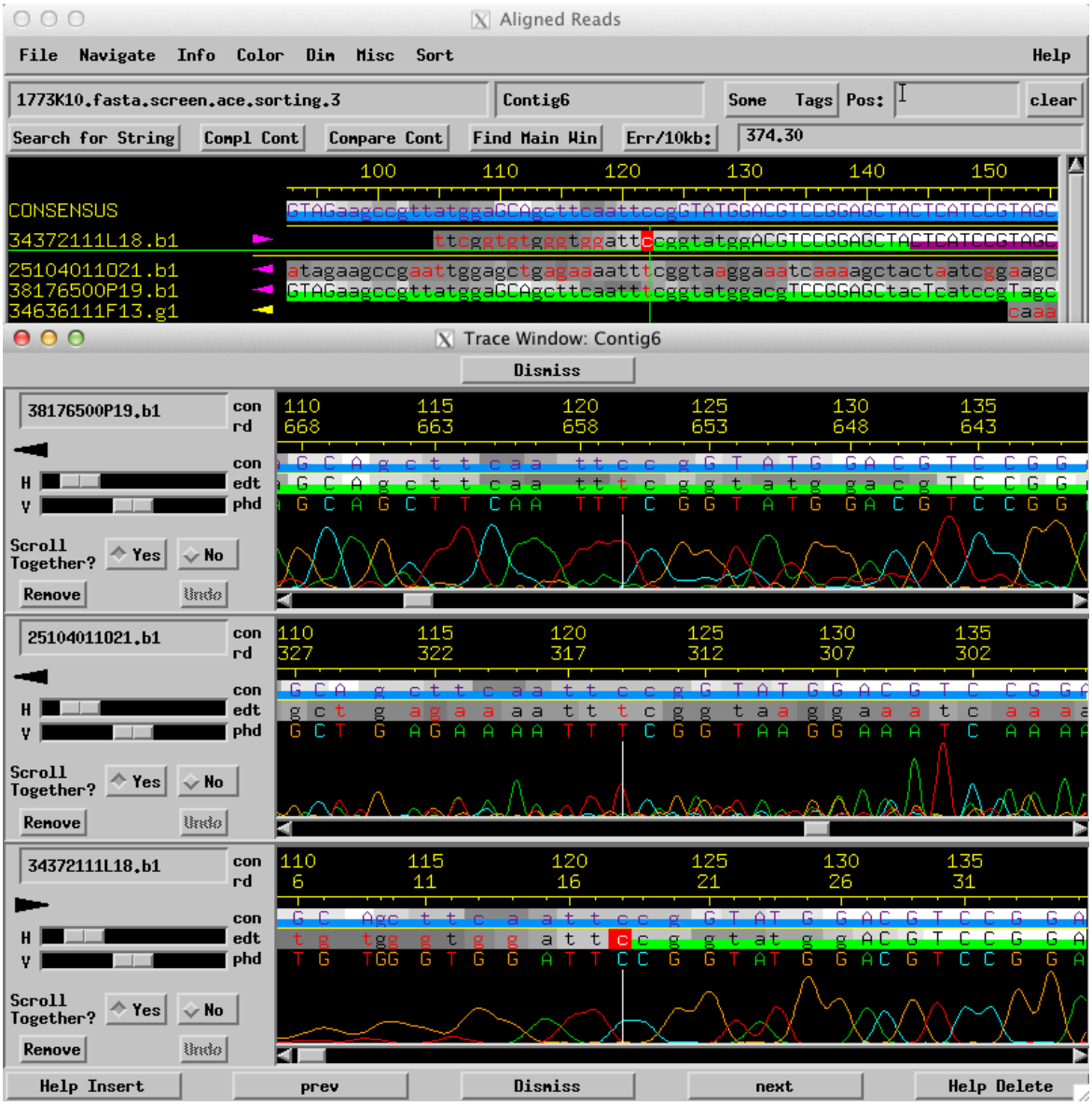


Figure 25 Incorrect discrepant consensus base C caused by unclipped vector in 3437211L18.b1

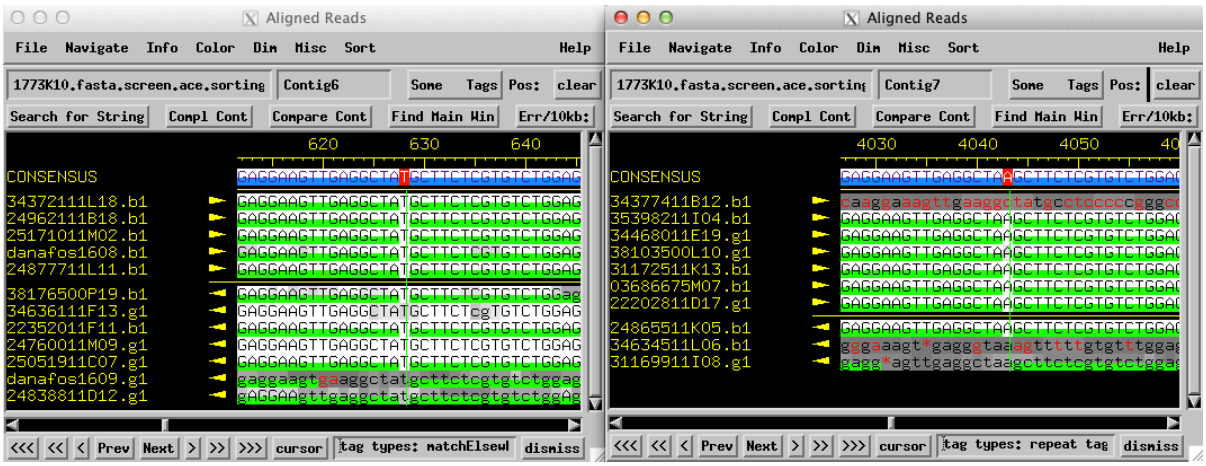


Figure 26 Discrepant base in Contig6 (T, left) versus Contig7 (A, right) within the duplication.

We will use the inconsistent forward/reverse mate pairs between contigs 6 and 7 to help us identify reads that were misplaced. In Assembly View, highlight the red "v" lines between contigs 6 and 7 within the duplication. A new "Clicked Forward/Reverse Pairs" window will appear with a list of inconsistent mate pairs (Figure 27).

Figure 27 Inconsistent mate pairs between contigs 6 and 7.

Notice that the reads are inconsistent because they are too far from end of Contig7. Most of these reads fall within the region where we are looking for buried data (i.e. 2,362-4,042). The only read on this list of inconsistent reads that extends beyond this range is `34377411B12.b1`, which terminates at 4,084. It is safe to move this read because the region that extends beyond the buried data range is very low quality (Figure 28).

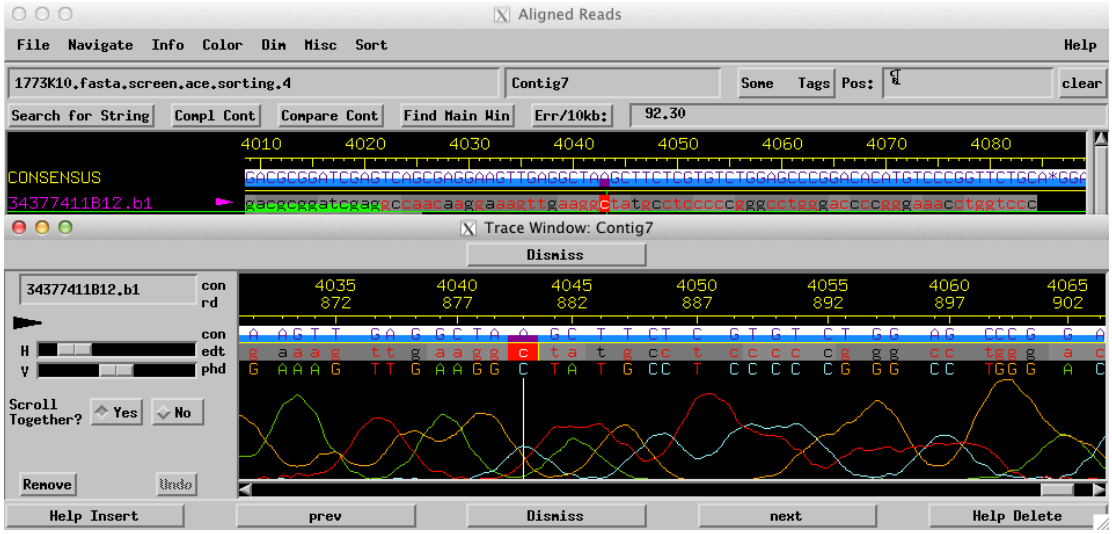


Figure 28 We can pull the read `34377411B12.b1` out of its current location. The region of the read that extends beyond the buried data range is unreliable because it is very low quality.

Extract and Reassemble the Buried Reads

Collectively, our analysis suggests that the inconsistent mate pairs in Contig7 might have been placed in the wrong copy of the duplication. We will pull these reads out of their current location using Assembly View. In the "Clicked Forward/Reverse Pairs" dialogue box, click on "Pull Out Reads". This will bring up a new window "Put Reads Into Their Own Contigs". Select all the reads in Contig7 by left clicking on the first Contig7 read on the list (`24882211E19.b1`). Then scroll to the last Contig7 read, press and hold down the shift key and select the last inconsistent mate pair read (`34377411B12.b1`) in Contig7 (Figure 29). Click on the "Remove Highlighted Reads" button to pull out these reads. Save the assembly (`1773K10.fasta.screen.ace.sorting.5`).

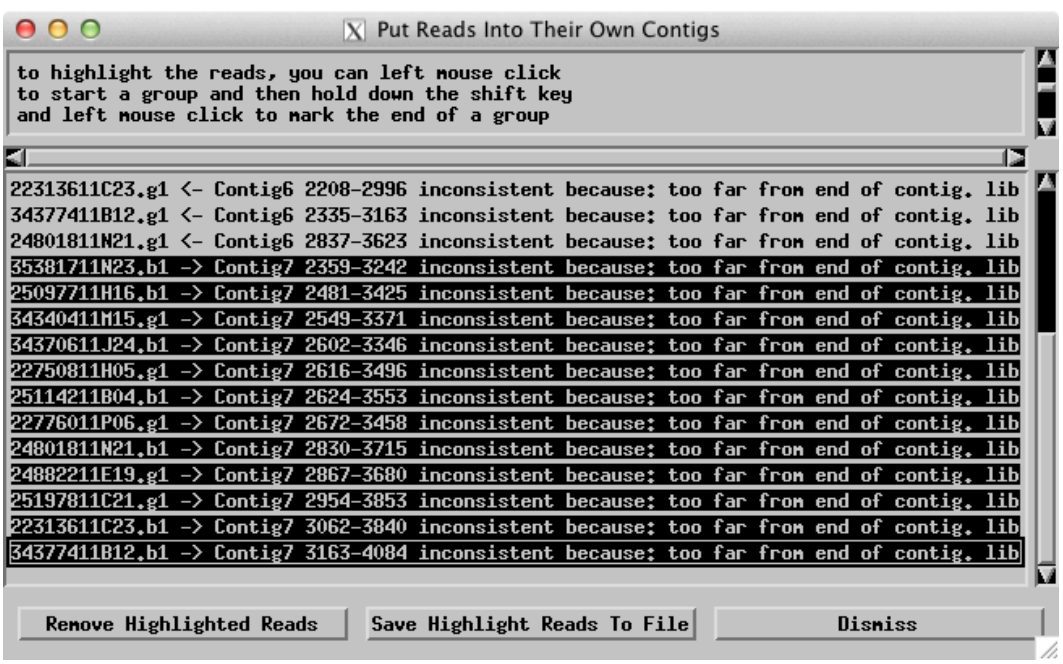


Figure 29 Remove all the inconsistent mate pair reads on Contig7.

Next, we will try to reassemble the twelve reads that we just pulled out of Contig7 using Miniassembly. Go back to the Consed Main Window and click on the “Miniassembly” button. This will bring up the “Reassemble Some Contigs” window. Verify that the twelve reads (contigs 10-21) we have just pulled out of Contig7 are listed in the “Contigs to Reassemble” box (Figure 30). Click on the “Reassemble” button.

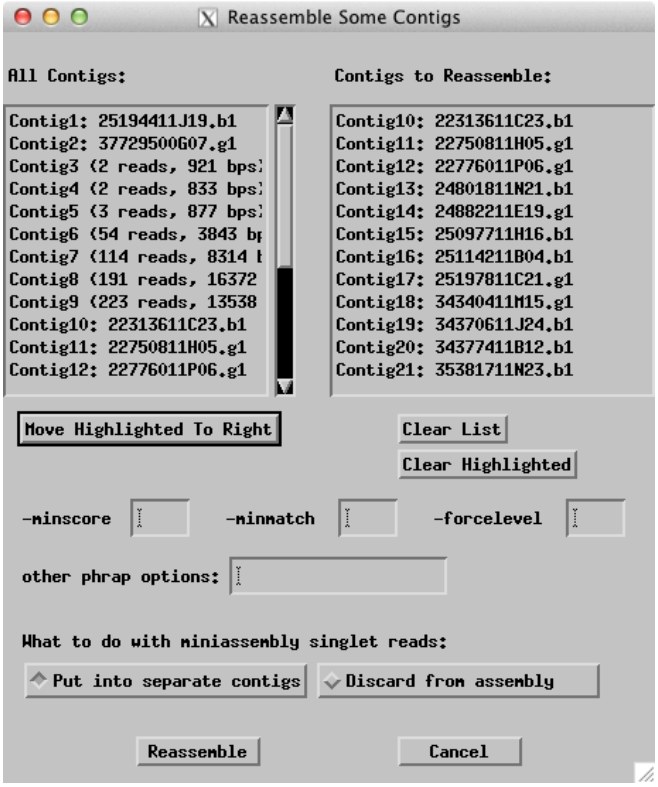


Figure 30 Reassemble the twelve reads we have just pulled out of Contig7.

After Miniassembly has been completed, a dialogue box will appear asking: “Are you finished mini-assembling these contigs?” Click on 'Yes' to dismiss this dialog and the “Reassemble Some Contigs” window. There should be a new navigation box, which shows the new contig (Contig10, Figure 31). Save the assembly (`1773K10.fasta.screen.ace.sorting.6`).

Figure 31 Miniassembly assembled the twelve reads misplaced in Contig7 into a single contig.

Resolve the Duplication Using the Newly Assembled Contig

In the “New Contigs” window, double click on Contig10 and then navigate to a high quality area at the beginning of the contig. Search for String using the consensus sequence from positions 100-150. There should be three matches (Figure 32).

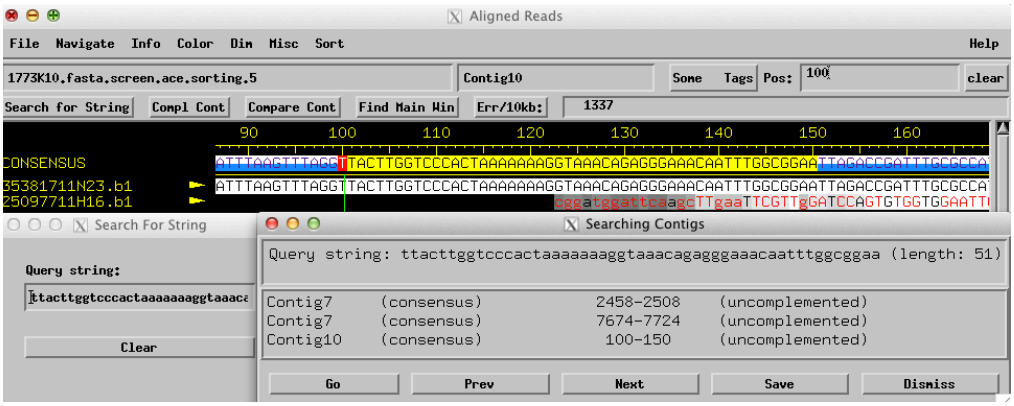


Figure 32 Two regions in Contig7 matches Contig10 from 100 to 150.

Because we just pulled the Contig10 reads out of the first copy of the duplication in Contig7 (i.e. the match at 2,458-2,508), we will try to join Contig10 with the second copy of the duplication in Contig7, that at 7,674-7,724.

Navigate to the Contig10 match and click on "Compare Cont". Open the Contig7 match at 7,674-7,724 and click on "Compare Cont". Then click on "Align" in the Compare Contigs window. We need to examine the alignment carefully before we join the contigs together. Specifically, we will look for high quality discrepancies in the alignment because they allow us to distinguish different copies of a repeat from each other. Because there are no high quality discrepancies in the alignment, we can join the two contigs together by clicking on the "Join Contigs" button. The new contig is Contig11.

Open the Aligned Reads Window for Contig11. Click the triple arrow button ">>>" to navigate to the right end of the contig. Use the single left arrow button "<" to scroll to the left until you reach a high quality region. Search for String from position 9,000 to 9,030. There should again be three matches (Figure 33).

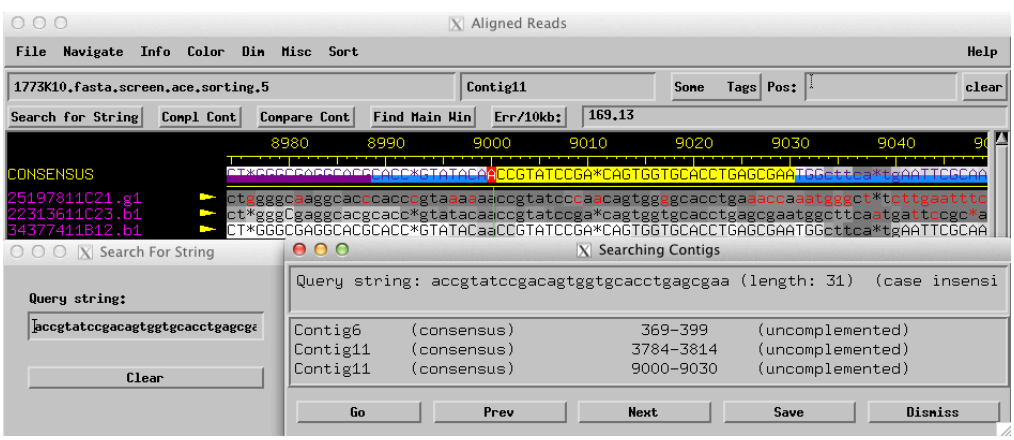


Figure 33 Search for String using the end of the new Contig11 results in three matches.

To close the gap, we will join the end of Contig11 (at 9,000-9,030) with the beginning of Contig6 (369-399). Compare the two regions using the "Compare Cont" button and the Compare Contigs window, examine the alignment and join the two regions together. The new contig is called Contig12. Save the assembly (`1773K10.fasta.screen.ace.sorting.7`).

To verify that we have resolved the duplication, we will examine the region in Assembly View. Open Assembly View and then run `crossmatch`. Click on the large duplication at the beginning of Contig12. We see that the total size of the duplication is ~3.4kb (Figure 34).

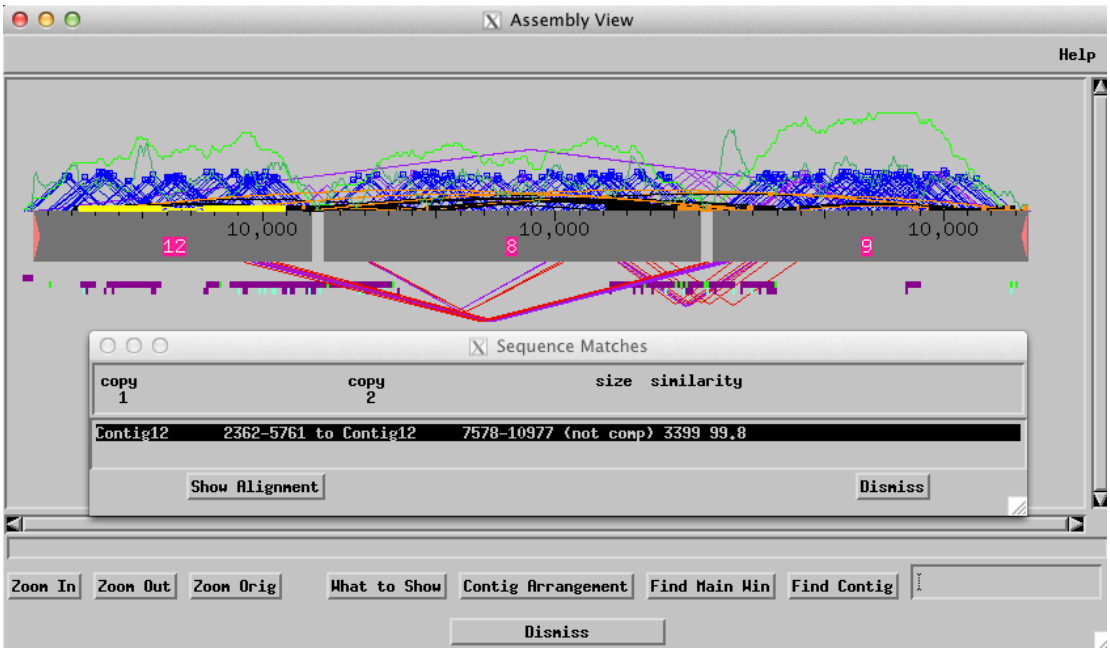


Figure 34 The beginning of the fosmid contains a ~3.4kb duplication with 99.8% similarity.

Sorting the Large Inverted Repeat

The black boxes and lines in Assembly View indicate there is a large inverted repeat that involves all of the major contigs (12, 8, and 9). The `crossmatch` results also show a direct repeat between the end of Contig8 and the beginning of Contig9. We will resolve the misassembly in this region first because there could be a potential join.

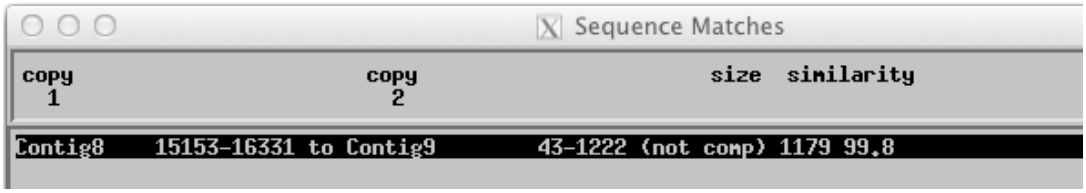


Figure 35 Sequence match at the end of Contig8 and beginning of Contig9.

Resolving the Misassembly in Contig9

Click on the direct (orange) repeat at the end of Contig8 in Assembly View. Select the match between the end (15,153-16,331) of Contig8 and the beginning (43-1,222) of Contig9 in the Sequence Matches window and then click on "Show Alignment" (Figure 35). Using the same strategy as we have described above, we will look for and carefully examine any high quality discrepancies within this alignment. Click on the "Next Discrepancy" button to navigate to the first discrepant position between the two repeats (Figure 36).

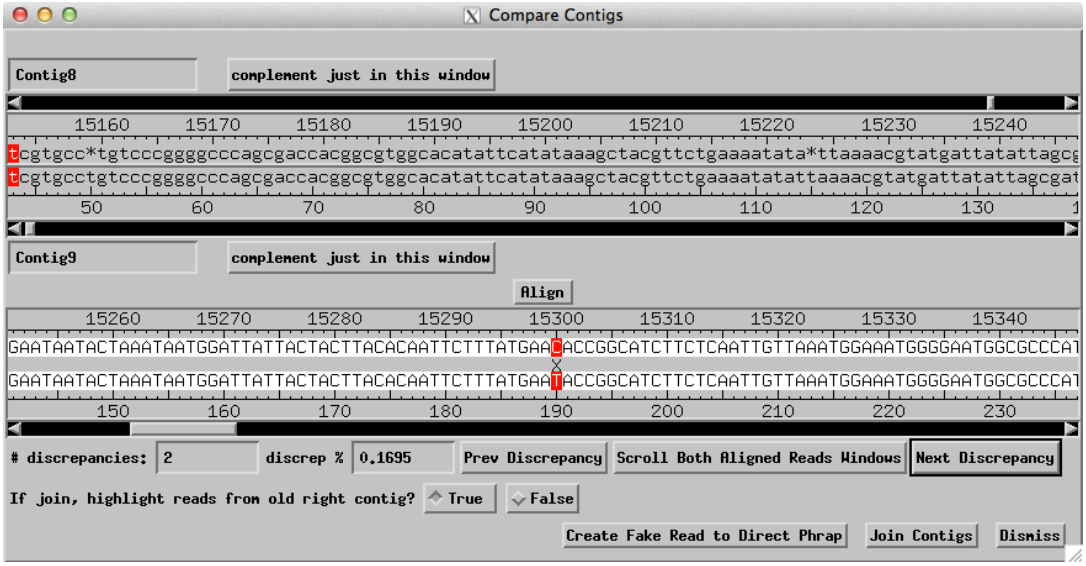


Figure 36 First high quality discrepancy between the repeats in contigs 8 and 9.

The first genuine mismatch is found in Contig8 at 15,300 and Contig9 at 190. In the Aligned Reads window of both contigs 8 and 9, create a comment tag on the mismatched base with the comment "base pair change". Because both the direct repeat and the presence of multiple consistent forward/reverse pairs indicates that there is a potential join between contigs 8 and 9, we will examine the subclones that contain these base discrepancies more closely (Figure 37).

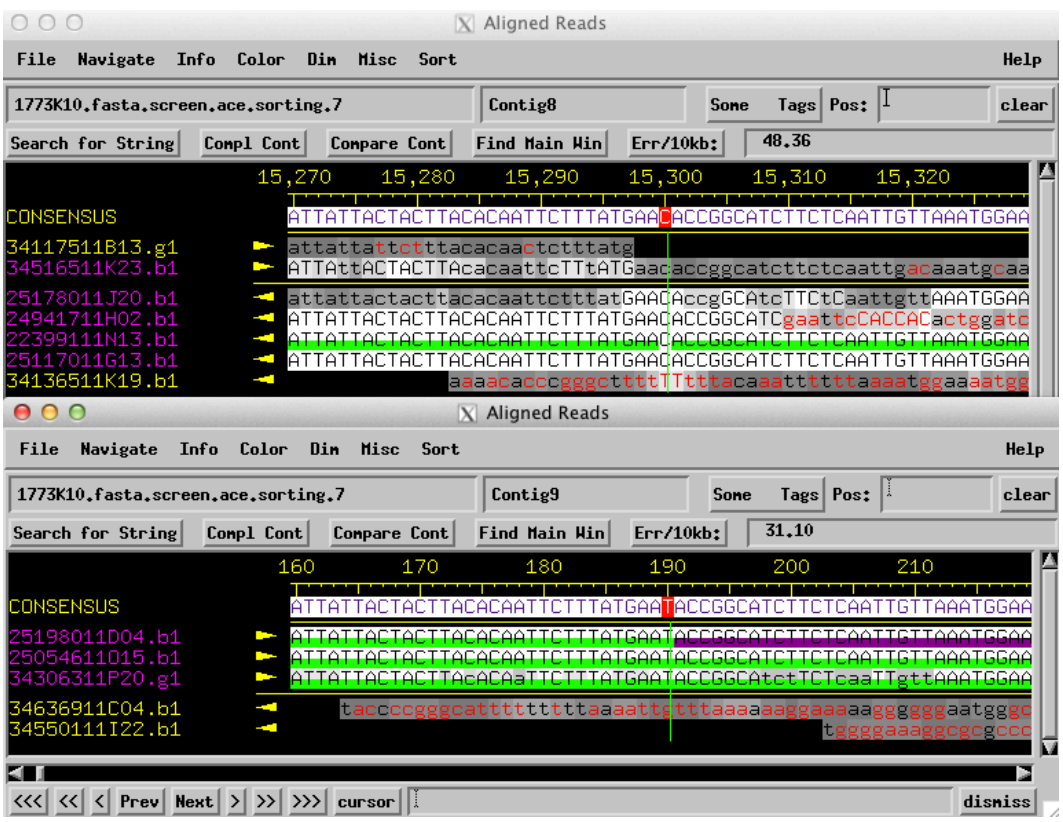


Figure 37 Reads with a discrepant base in contigs 8 and 9 are highlighted in purple.

For each discrepant read, we will try to locate its mate pair in the current assembly. Left click on the read name on the Aligned Reads window to copy it and then middle click to paste the name into the "Find 1st read above starting with:" textbox on the Consed Main Window. Because the mate pair has the same prefix as the read you have selected, the mate pair read will either be just above (for selected reads with g1 suffix) or below (for selected reads with b1 suffix) the highlighted read in reads list box (Figure 38).

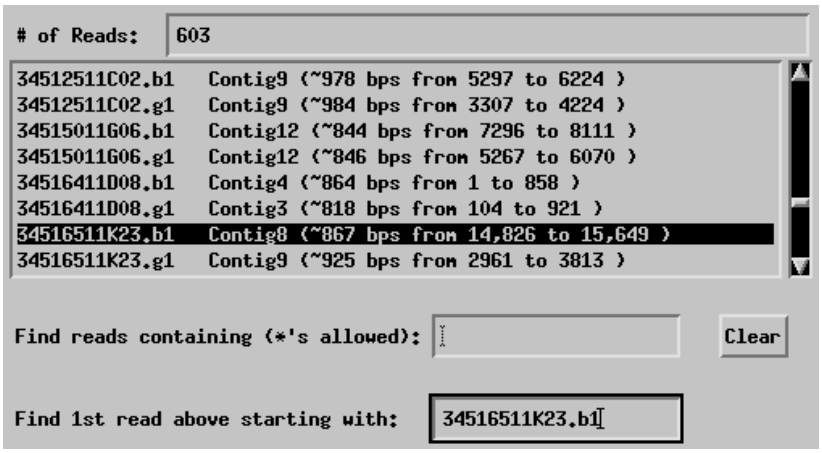


Figure 38 Using the "Find 1st read above starting with" field to locate the mate pair read.

The placement of the discrepant reads and their mate pairs is summarized in Table 1:

Table 1 Placement of the discrepant reads and their mate pairs

Discrepant read	Location	Mate pair read	Mate pair location
25178011J20.b1	Contig8:14518-15513	25178011J20.g1	Contig8:11793-12787
24941711H02.b1	Contig8:14536-15352	24941711H02.g1	Contig8:11803-12742
22399111N13.b1	Contig8:14755-15591	22399111N13.g1	Contig8:11319-12234
34516511K23.b1	Contig8:14826-15649	34516511K23.g1	Contig9:2961-3813
25117011G13.b1	Contig8:14898-15758	25117011G13.g1	Contig8:11496-12444
25198011D04.b1	Contig9:1-812	25198011D04.g1	Contig12:10367-11274
25054611O15.b1	Contig9:55-937	25054611O15.g1	Contig12:10641-11477
34306311P20.g1	Contig9:89-991	34306311P20.b1	Contig12:9956-10498

Connect Contig8 with Contig9 Connect Contig9 with Contig12

Four out of five of the discrepant reads on Contig8 have consistent mate pairs that are also placed on Contig8. The only exception is the mate pair for `34516411K23.b1`, which is found on Contig9. Examination of the mate pair of `34516411K23` shows that they are in the correct relative orientation, thereby supporting the suspected join between these two contigs.

In contrast, the three reads that are discrepant at position 190 on Contig9 (`25198011D04.b1`, `25054611O15.b1`, `34306311P20.g1`) have mate pairs that are placed on Contig12. Because the beginning of Contig9 have inconsistent mate pairs with both Contig8 and Contig12, we know that this end of the repeat copy on Contig9 is misassembled and needs to be torn out. However, we have not yet determined the extent of the misassembly at the beginning of

Contig9. Hence our next step is to use the high quality discrepancies to help us determine precisely how much of Contig9 has been incorrectly assembled.

To do this, go back to the Compare Contigs window and click "Next Discrepancy." Consed will navigate to the next base pair change between contigs 8 and 9. Create a comment tag at each of the discrepant positions with the comment "base pair change". Repeat the same process and create comment tags for all the remaining genuine discrepancies. Save the assembly (`1773K10.fasta.screen.ace.sorting.8`).

The last base pair change in Contig9 occurs at position 499 (Figure 39). Given that the alignment terminates at the end of Contig8 and there are no additional discrepancies beyond this position, the misassembly is likely limited to this part of Contig9. Consequently, we will use the discrepant position (C versus *) at position 499 to dictate whether each read should be placed on the new left or new right contig when we tear this contig apart. All the reads with a high quality C at position 499 in Contig9 will be sent to the new left contig when we tear the Contig9 at this position.

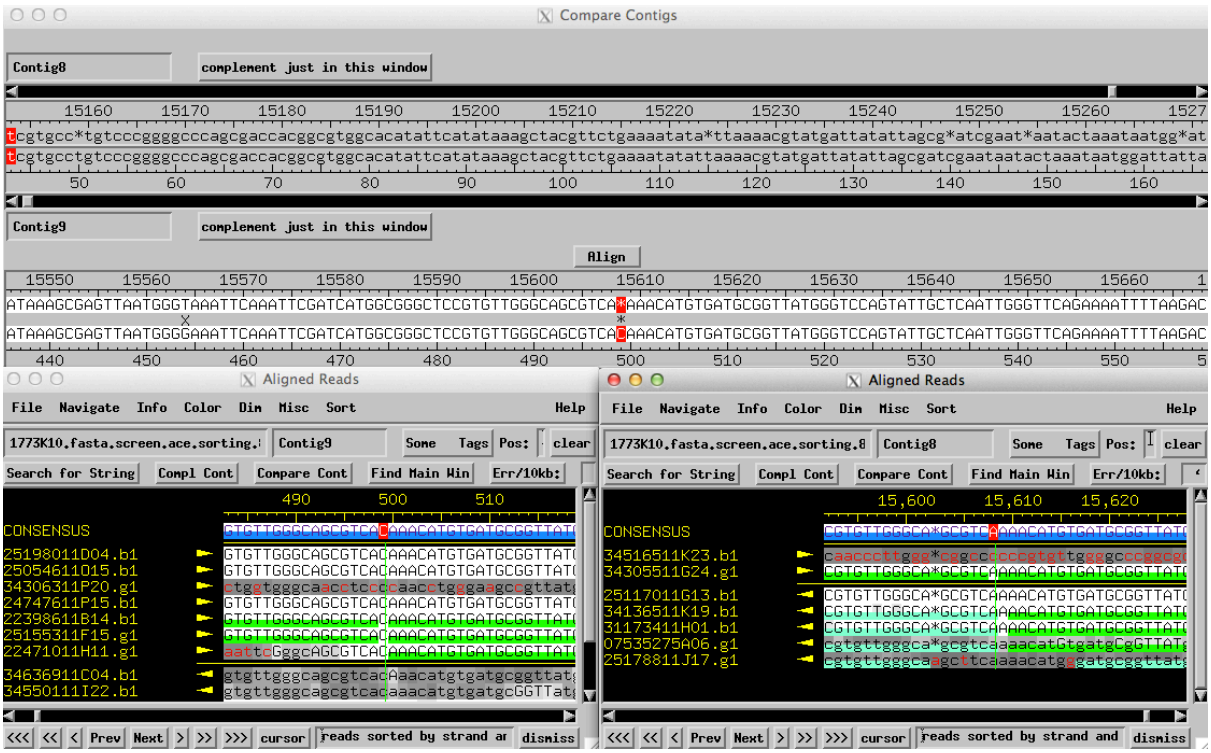


Figure 39 Use the last discrepant position between contigs 8 and 9 to determine which of the reads should be sent to the new left contig when tearing Contig9 apart.

On Contig9 at position 499, click on the "C" on the consensus, click the right mouse button and select "Tear contig at this consensus position". We need to highlight all the reads with the "C" at consensus position 499 to send them to the left. Highlight the names of all the subclones except for `31171111G07.g1`, `25194411J19.g1`, and `24884311G11.g1` (Figure 40). Click on the "Do Tear" button on the Tear Contig window to tear Contig9 into two new contigs, 13 and 14. Save the new assembly (`1773K10.fasta.screen.ace.sorting.9`).

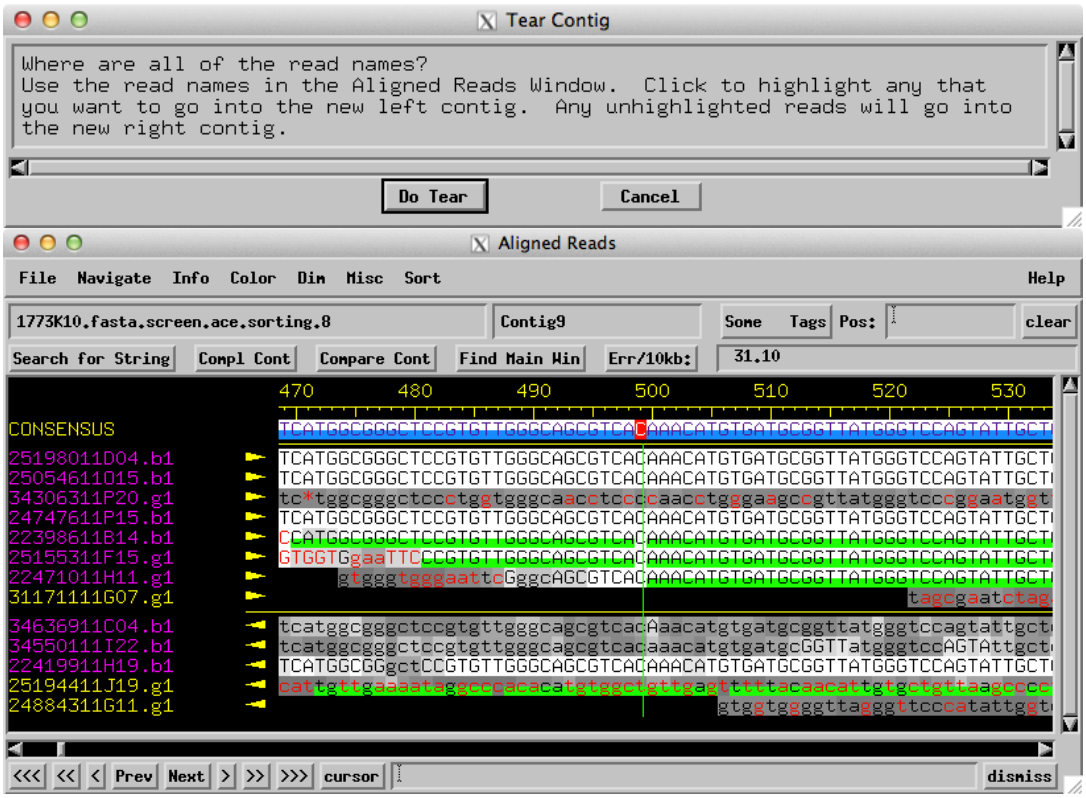


Figure 40 Send all the reads except `31171111G07.g1`, `25194411J19.g1`, and `24884311G11.g1` to the new left contig.

Open Assembly View to examine the new assembly (Figure 41). The small Contig13 (which contains the reads from the beginning of Contig9) is now placed between contigs 12 and 8 based on the mate pair information. Because Contig13 has a "c" after its name, we will complement this contig first so that it is in the orientation suggested by Assembly View. We will then look for potential joins at the ends of Contig13.

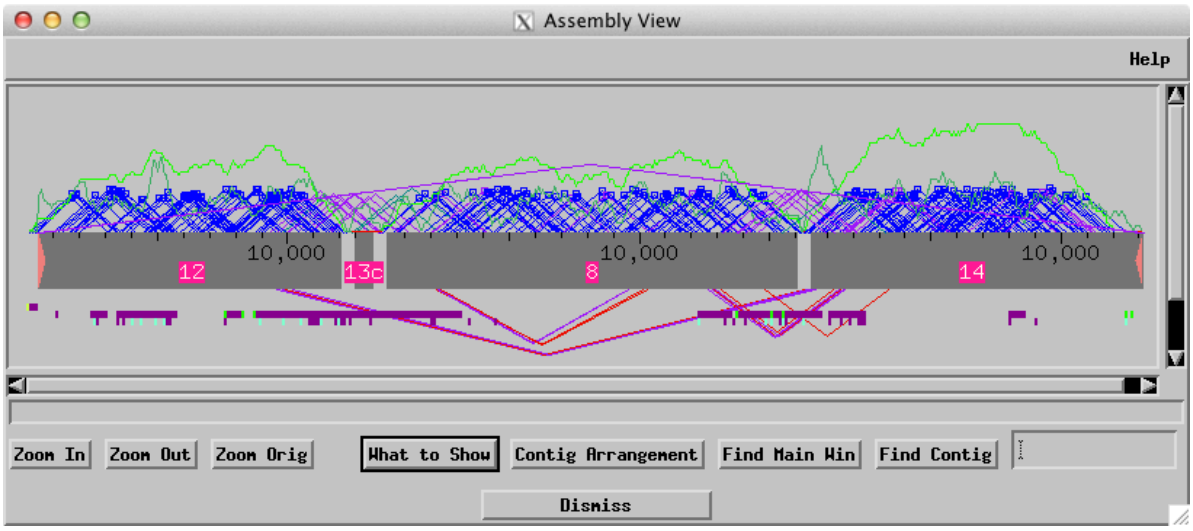


Figure 41 New Assembly View after tearing Contig9.

Open the Aligned Reads window for Contig13, complement the contig and navigate to the end of the high quality region. Then Search for String using the sequence from position 1,300 to 1,350. There should be three matches (Figure 42).

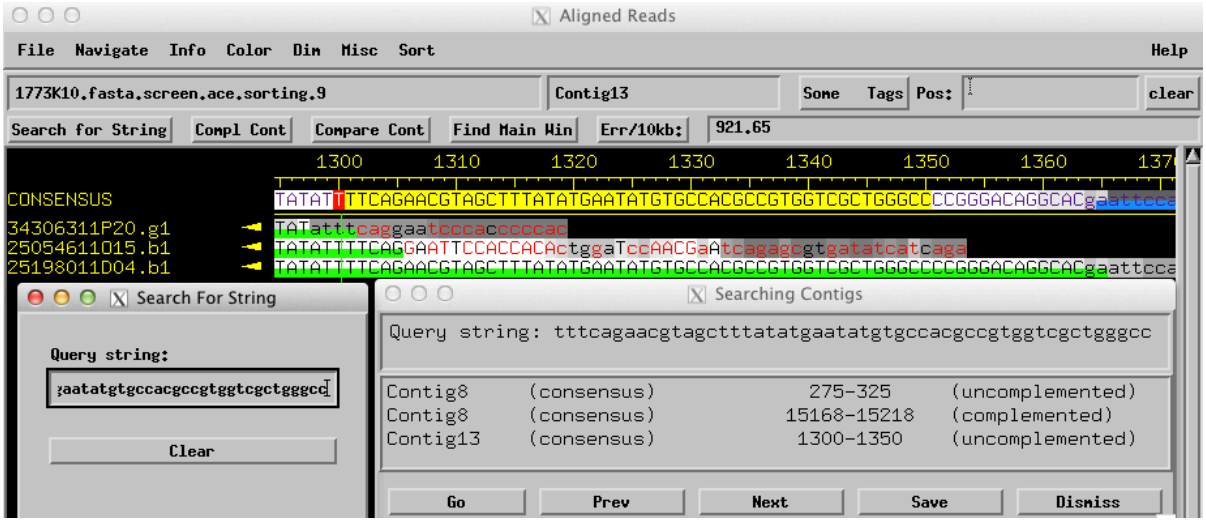


Figure 42 Two matches in Contig8 to Contig13 from 1300 to 1350.

Based on the consistent mate pairs (e.g. from subclones `34636911C04` and `34550111I22`) between the end of Contig13 and the beginning of Contig8, we will try to join the sequence match at the beginning of Contig8 (275-325, uncomplemented) with the end of Contig13 (1300-1350, uncomplemented). Compare these two regions and examine the alignment.

The high quality discrepancies at the beginning of the alignment can be attributed to unclipped vector at the beginning of Contig8 (Figure 43). (See page 12 for detailed instructions on how to recognize unclipped vector sequences.) The rest of the alignment looks good so we will join the two contigs together. The new contig is Contig15. Save the assembly (`1773K10.fasta.screen.ace.sorting.10`).

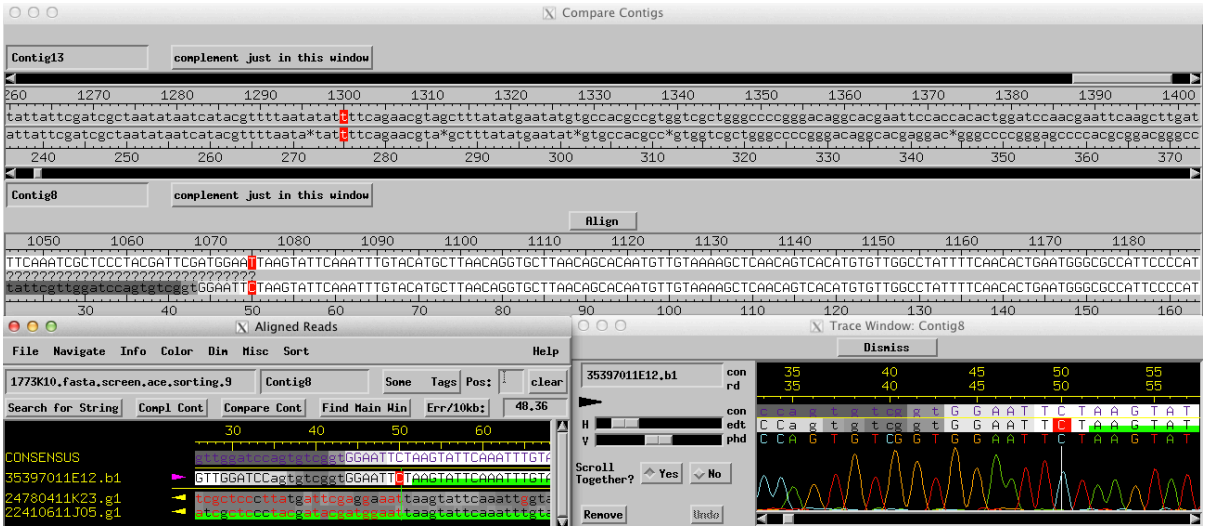


Figure 43 Discrepancies at the beginning of Contig8 are caused by unclipped vector.

Closing the Gap between Contig15 and Contig14

Open Assembly View and run `crossmatch` (Figure 44). This brings up a complex Assembly View that makes it difficult to search for potential joins between adjacent contigs.

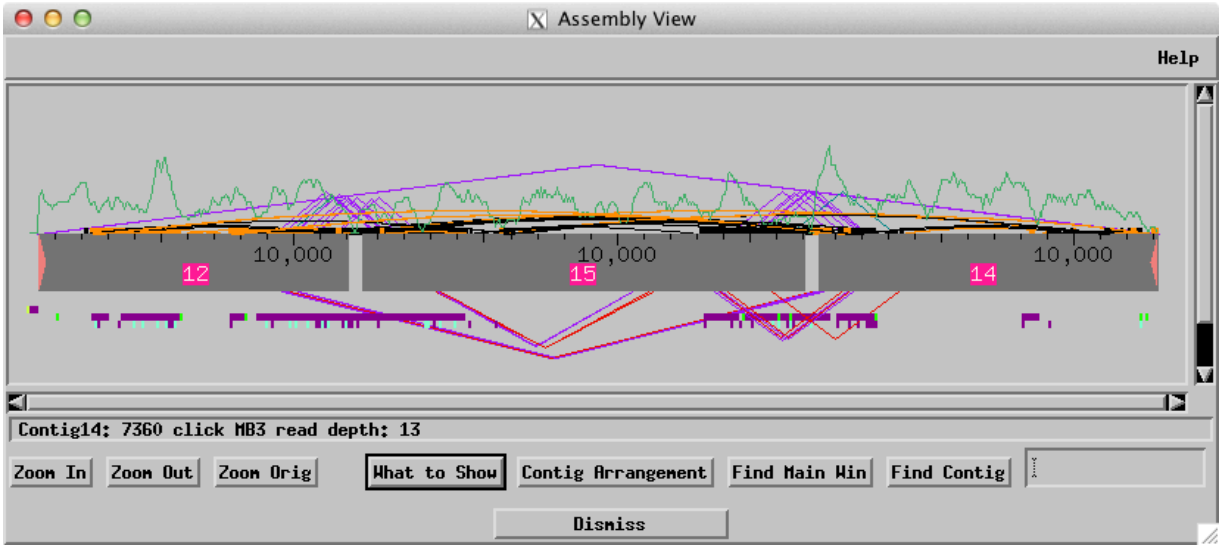


Figure 44 Assembly View after joining contigs 8 and 13.

In order to see whether we can join the end of Contig15 with the beginning of Contig14, we would like to focus on the subset of direct sequence matches between the two contigs. To filter the rest of the sequence matches, go back to the "Which Sequence Matches to Show in Assembly View" dialogue box and turn off the following options:

- ok to show sequence matches within contigs
- ok to show inverted sequence matches

Click on the "Apply" button in the lower left corner of this window to update Assembly View (Figure 45). The new assembly will now only show the direct sequence matches between contigs. We see that there is a large direct repeat between the end of Contig15 and the beginning of Contig14.

Some of the mate pairs between the end of Contig15 and the beginning of Contig14 are inconsistent because they are too far apart. The consistent mate pairs between the two contigs and the direct repeat identified by `crossmatch` in this region lend further credence to the hypothesis that we should be able to join these two contigs together.

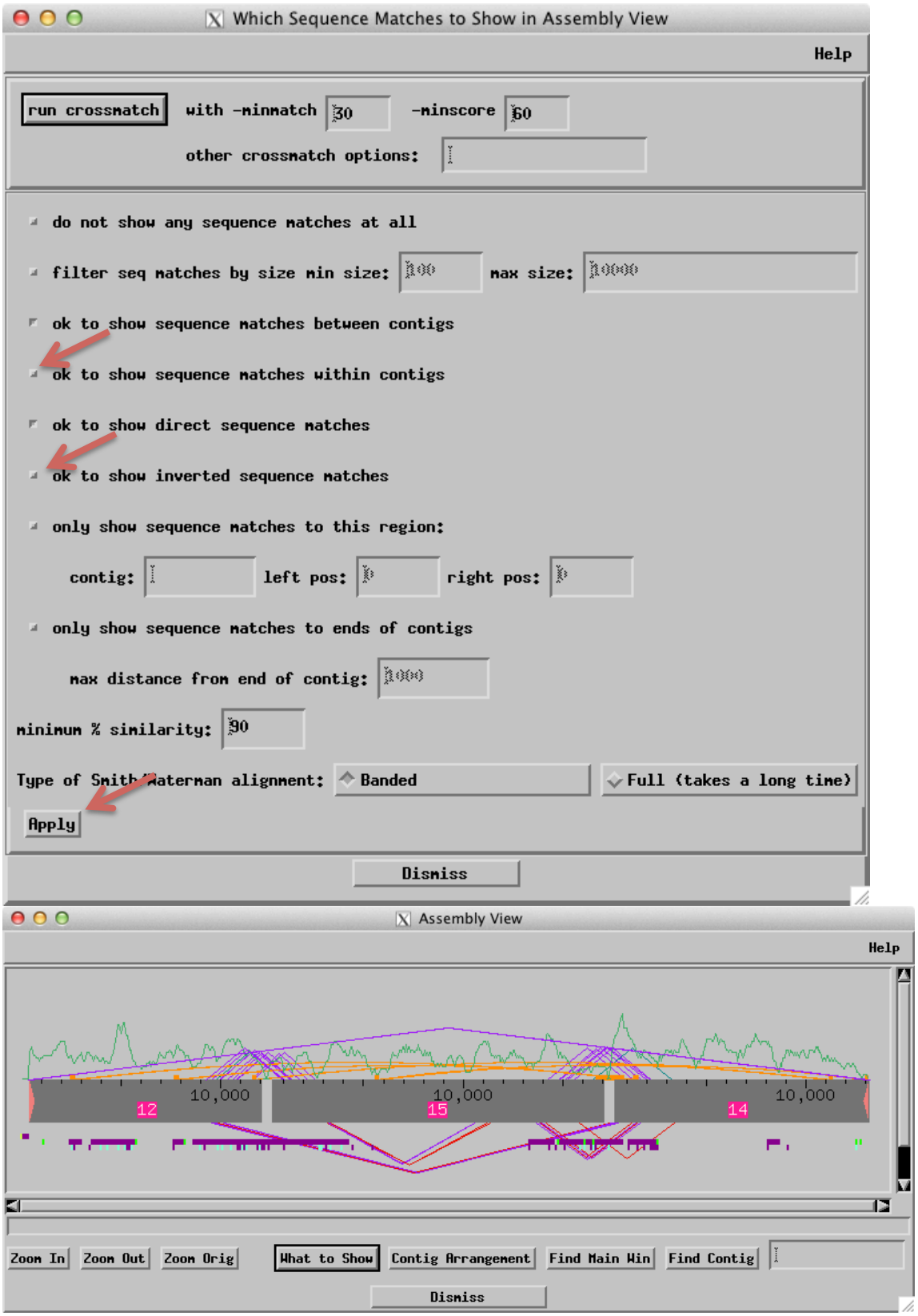


Figure 45 Change the crossmatch filter options to show only direct sequence matches between contigs.

To see if we can join these two regions together, click on the direct repeat (orange boxes) between the end of Contig15 and the beginning of Contig14. Select the 657 base pair match and then click on the "Show Alignment" button to examine the alignment (Figure 46).

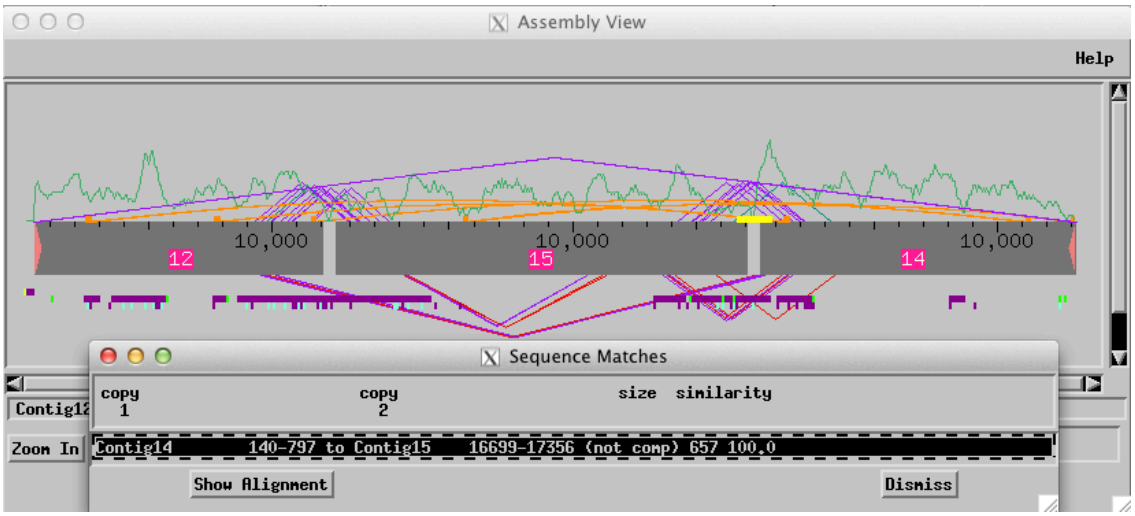


Figure 46 Perfect repeat between the end of Contig15 and the beginning of Contig14.

The high quality discrepancies at the beginning and at the end of the alignment can again be attributed to unclipped vector sequences and can be ignored (Figure 47). Join the two contigs together and save the assembly (`1773K10.fasta.screen.ace.sorting.11`).

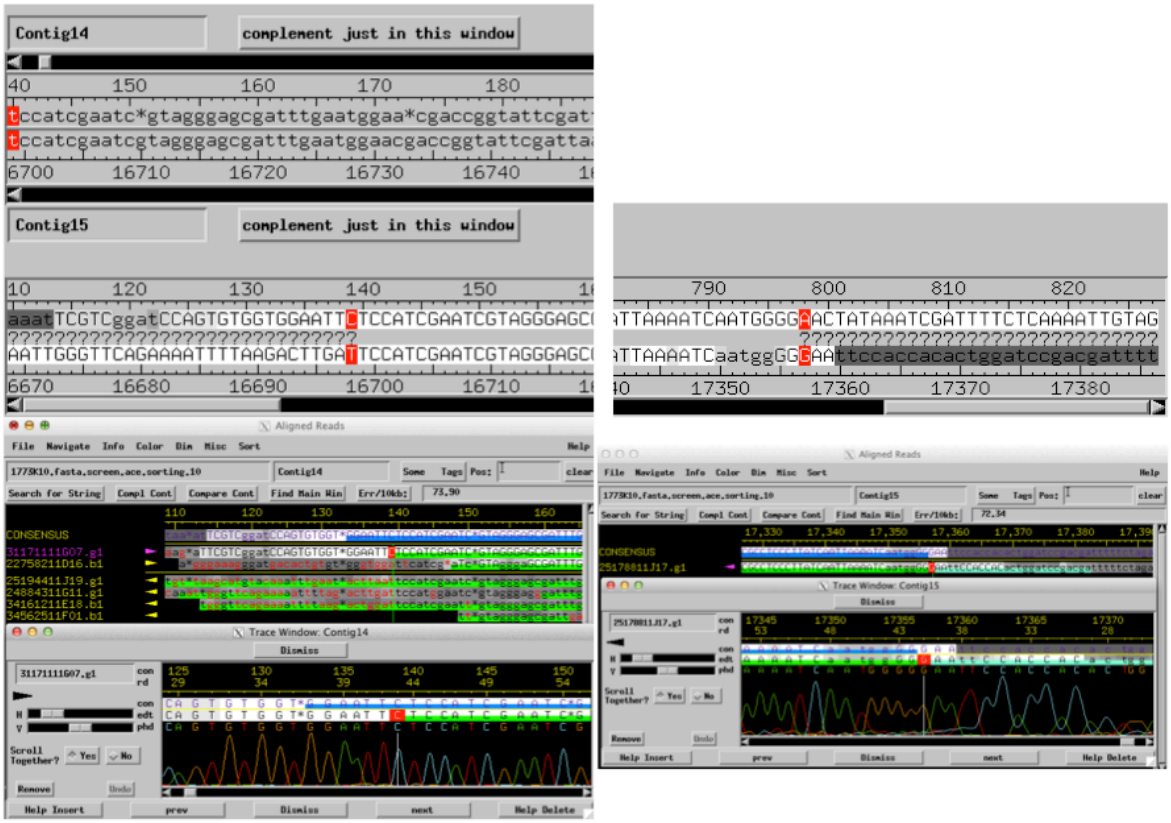


Figure 47 High quality discrepancies between contigs 14 and 15 can be attributed to unclipped vector.

Resolving the Gap between Contig12 and Contig16

Open Assembly View and run `crossmatch`. In order to show all the repeats, we will toggle back on the "ok to show sequence matches within contigs" and "ok to show inverted sequence matches" options. Then click on the "Apply" button.

Notice the inconsistent forward/reverse pairs between contigs 12 and 16 are part of a large (black) inverted repeat (Figure 48). This is good news because the data for the gap between contigs 12 and 16 might have simply been buried within the Contig16 copy of the inverted repeat. We will pull out the inconsistent mate pairs that are placed in the Contig16 copy of the inverted repeat using the strategy described above.

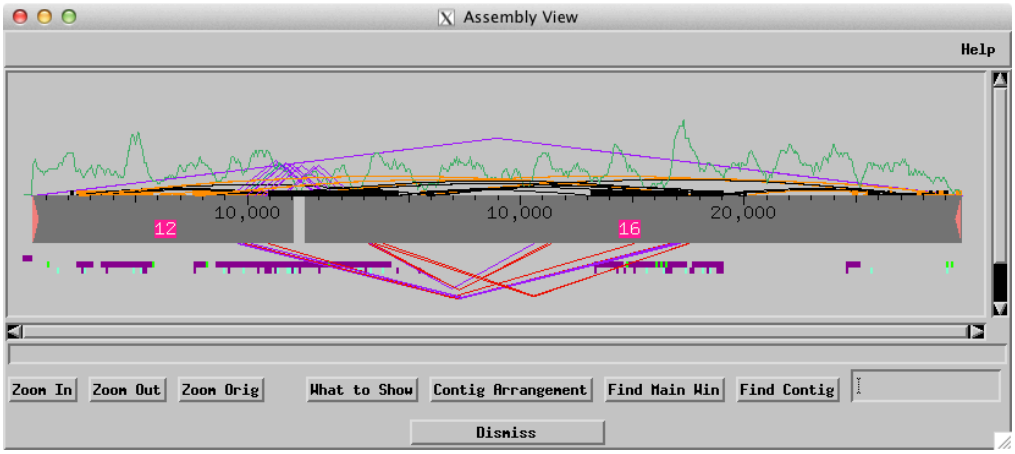


Figure 48 Inconsistent mate pairs between contigs 12 and 16 suggest that the missing data in the gap might have been buried in Contig16.

Click on the red lines (inconsistent forward/reverse pairs) between contigs 12 and 16 in the "Clicked Forward/Reverse Pairs" box then click on "Pull Out Reads" (Figure 49).

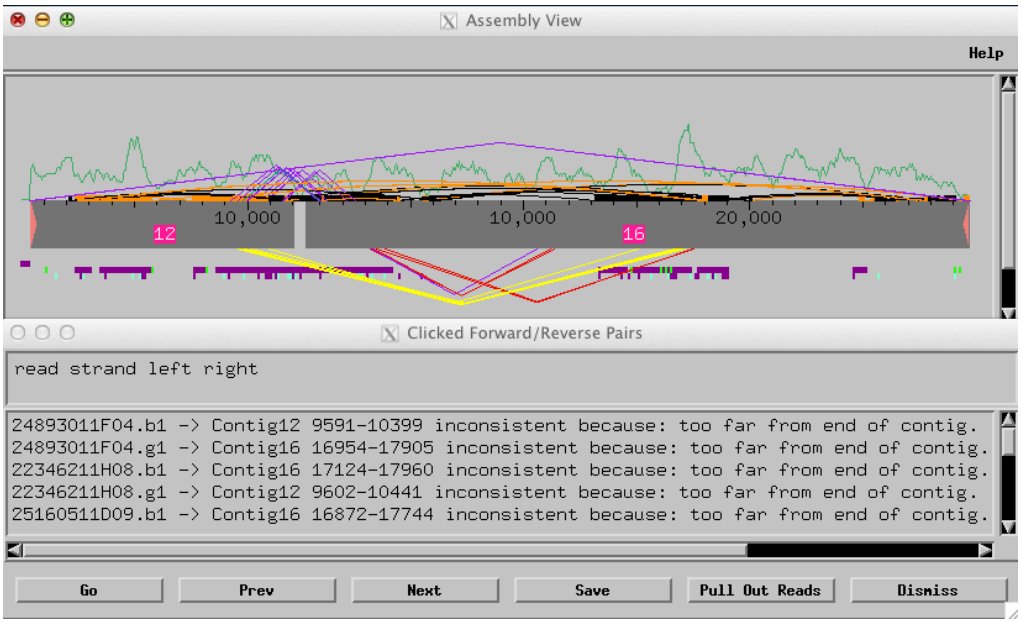


Figure 49 Select the inconsistent mate pairs between contigs 12 and 16 from Assembly View.

In the "Put Reads Into Their Own Contigs" window, select all the reads that are placed on Contig16 and then click on "Remove Highlighted Reads" (Figure 50). Save the new assembly (`1773K10.fasta.screen.ace.sorting.12`).

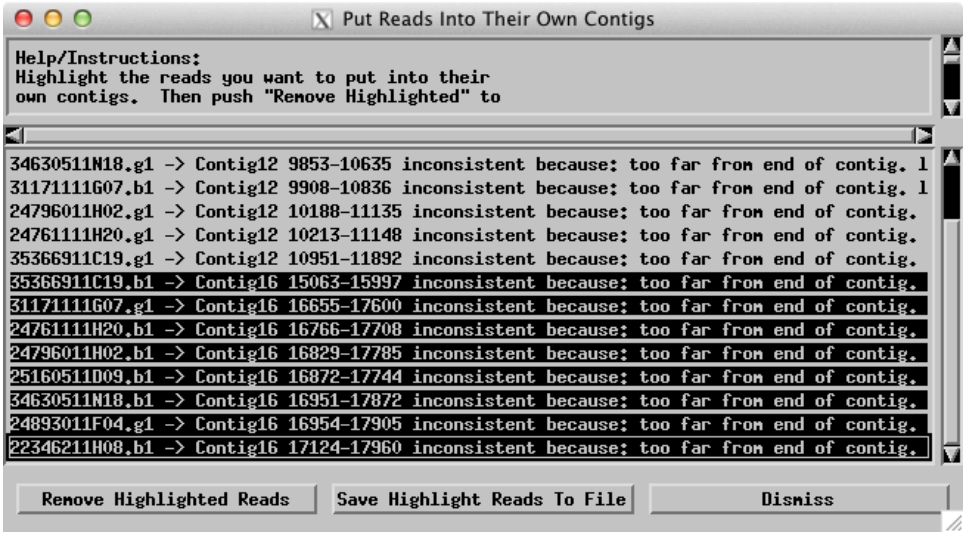


Figure 50 Remove inconsistent mate pairs that were misplaced in the Contig16.

Now we will run Miniassembly using the eight reads we have just pulled out. Click on "Miniassembly" on the Consed Main Window, verify that the eight reads are listed under the "Contigs to Reassemble" section and then click on "Reassemble" (Figure 51).

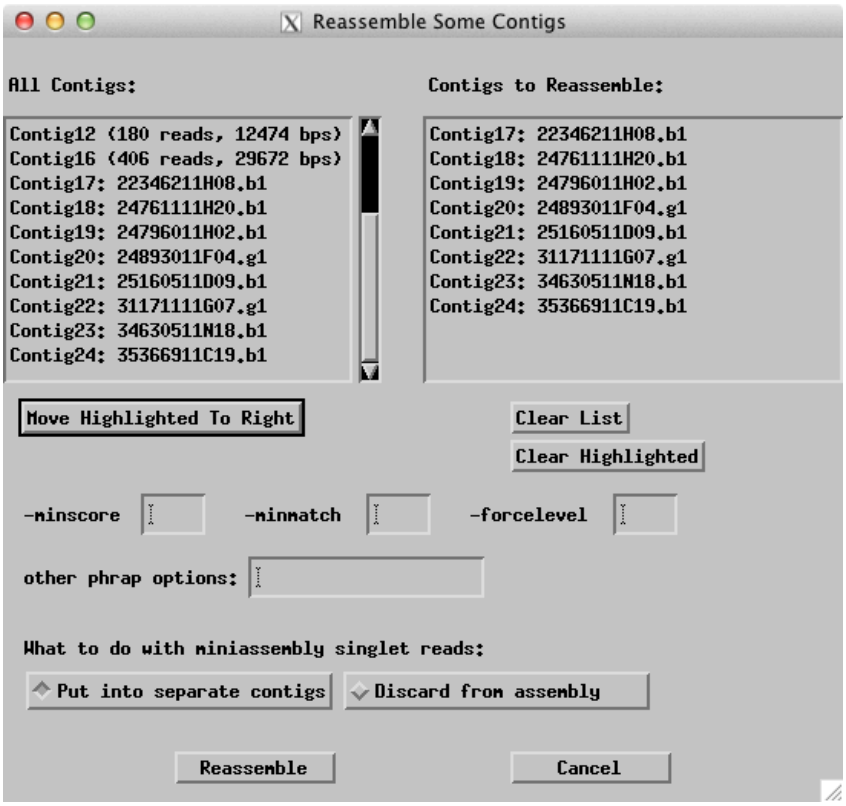


Figure 51 Reassemble the eight reads we have just pulled out of Contig16 using Miniassembly.

There should be two new contigs after Miniassembly is complete: Contig25 (which consists of a single read `35366911C19.b1`) and Contig26 (with the other seven reads). Save the assembly (`1773K10.fasta.screen.ace.sorting.13`).

Open Assembly View. Notice that Contig26 is in the complemented orientation relative to the other contigs in Assembly View so we need to complement this contig in the Aligned Reads Window so that it is in the orientation suggested by Assembly View (Figure 52).

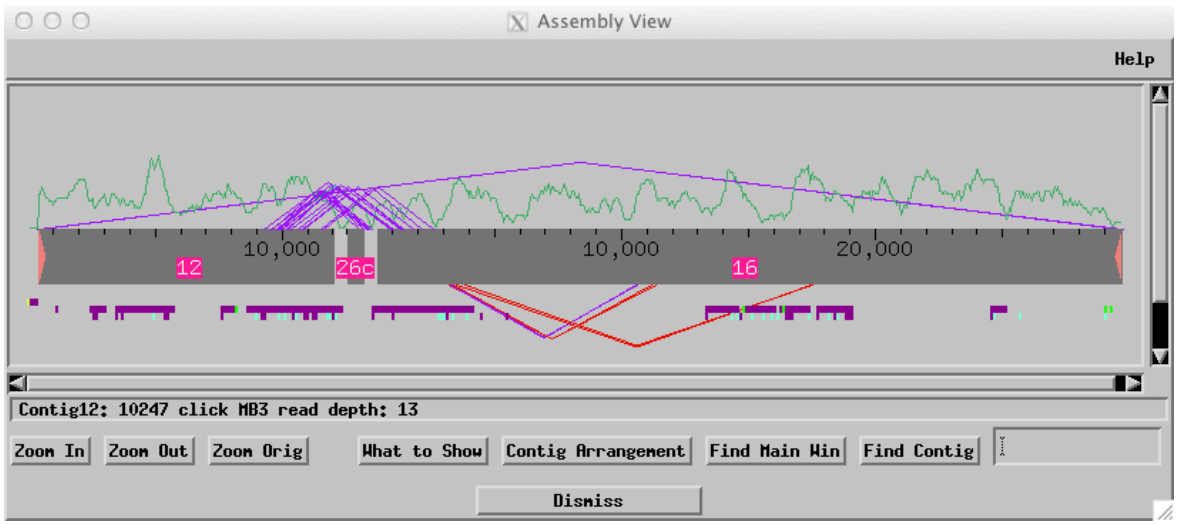


Figure 52 Miniassembly created Contig26, which is in the reverse complemented orientation relative to the rest of the scaffold.

To see if we can join Contig26 with contigs 12 and 16, open the Aligned Reads window for Contig12 and click on the ">>>" button to navigate to the right end of the contig. Then use the "<" button to scroll to the left until we reach a region with high quality data. Perform Search for String using the consensus sequence from 12,230-12,260 in Contig12. There should be three matches (Figure 53).

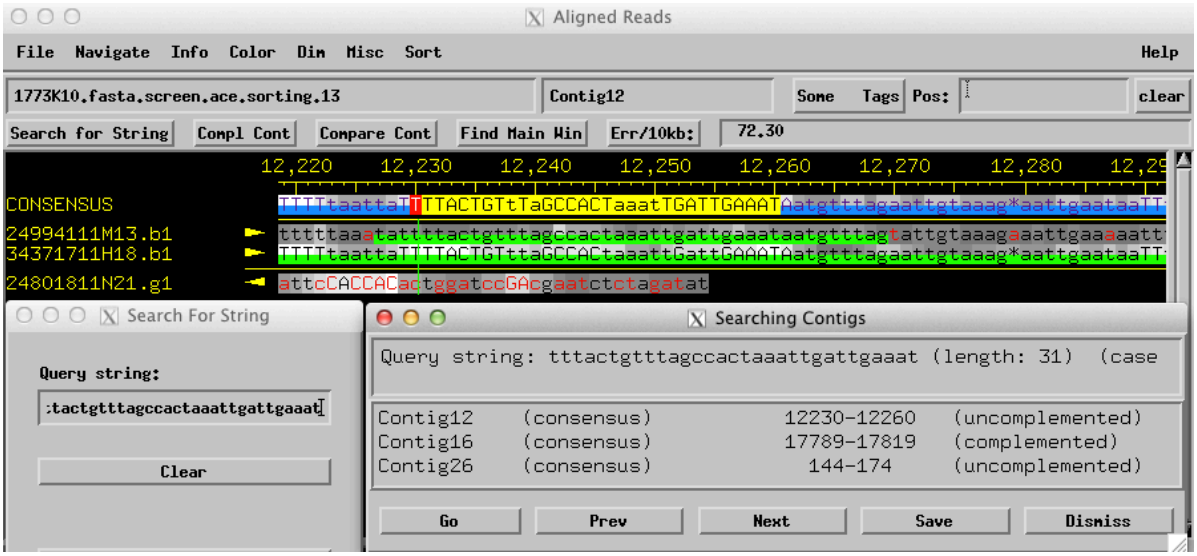


Figure 53 Search for String using the end of Contig12 from 12,230-12,260.

Compare the sequence matches between contigs 12 and 26. Examine the alignment and then join these two contigs together. This will create Contig27.

Next we will try to join the end of Contig27 with the beginning of Contig16. Open the Aligned Reads window for Contig27 and navigate to the end of the contig. Notice that the end of the consensus is based on a single read (`31171111G07.g1`). Part of this read is tagged with a dark green chimera tag (Figure 54).

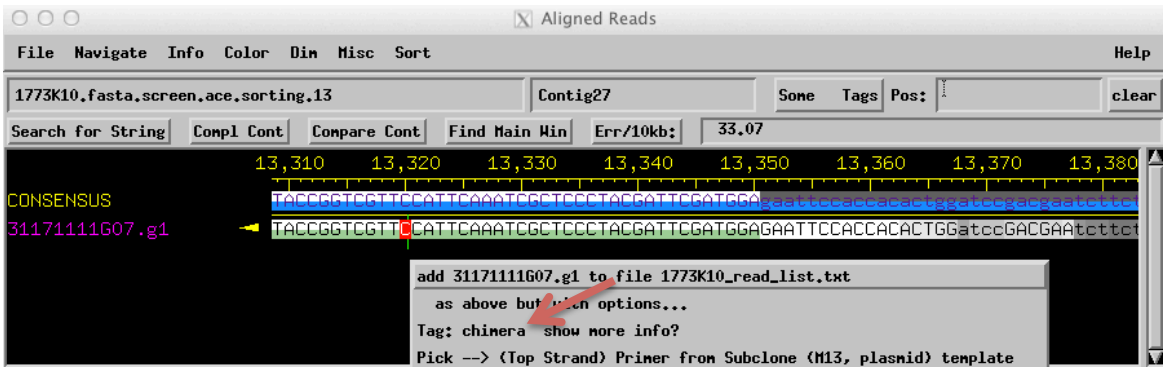


Figure 54 Chimera tag on the read `31171111G07.g1` at the end of Contig27.

This chimera tag was added by `phrap` during the assembly process. It indicates that the read might contain two distinct DNA inserts that were joined together during cloning. This tag is likely spurious because of the major misassemblies found in this project. However, to ameliorate the concern that the consensus sequence in this region might be unreliable, we will navigate to the left until we are outside of this chimera tag. Search for String using the consensus position from 13,200-13,230. There should be three matches (Figure 55).

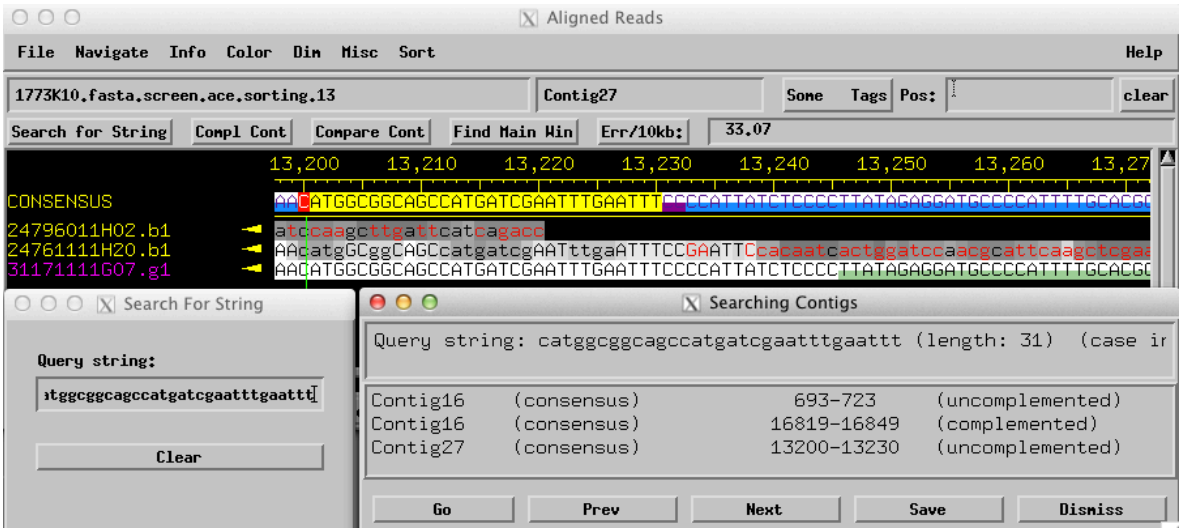


Figure 55 Two regions in Contig16 match the end of Contig27.

Compare the beginning of Contig16 (693-723) with the end of Contig27 (13,200-13,230). Align these two regions, examine the alignment and join the two contigs together to create Contig28. Save the assembly (`1773K10.fasta.screen.ace.sorting.14`). Open Assembly View and run `crossmatch`. The duplication has been successfully resolved (Figure 56).

Although the assembly is now contiguous, the inverted repeats still have some inconsistent mate pairs that require additional sorting.

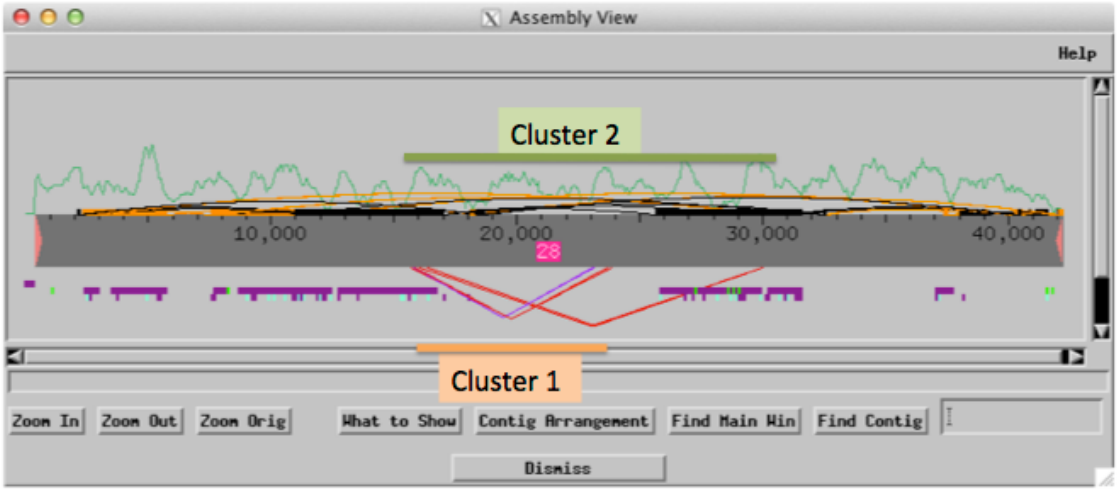


Figure 56 The duplication has been successfully resolved after sorting. However, there are still two clusters of inconsistent mate pairs remaining in Contig28.

Resolve the Remaining Misassembly within the Inverted Repeat

Assembly View shows that there are two clusters of inconsistent mate pairs in Contig28 (Figure 56). For Cluster 1, one member of the mate pair is placed at ~15kb while its partner is at ~23kb. For Cluster 2, one member is placed at ~15kb while its partner is at ~30kb.

According to the `crossmatch` results, the reads placed at ~23kb are in a unique region, which suggests that reads at ~15kb in Cluster 1 are placed in the wrong copy of the inverted repeat. To fix these inconsistent mate pairs, click on the inconsistent mate pairs in Cluster 1 in Assembly View and put the four reads that were placed at ~15kb into their own contigs (Figure 57). Save the assembly (`1773K10.fasta.screen.ace.sorting.15`).

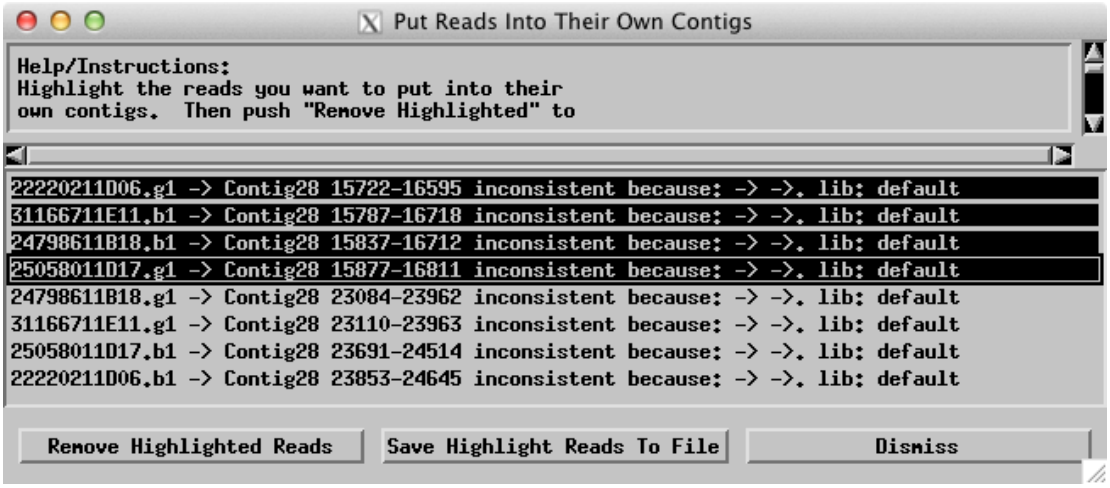


Figure 57 Pull the reads in Cluster 1 that are placed at ~15kb in Contig28 into their own contigs.

Run Miniassembly on the four reads we have just pulled out. This creates Contig29 with all four reads, indicating that these reads can be placed together as a single group. Save the assembly (`1773K10.fasta.screen.ace.sorting.16`).

We will now try to join the new Contig29 with the other inverted repeat copy that spans from ~25-30kb. Open the Aligned Reads window for Contig29 and navigate to a high quality region at the beginning of the contig. Perform a Search for String using the consensus sequence from 100-150. There should be three matches (Figure 58).

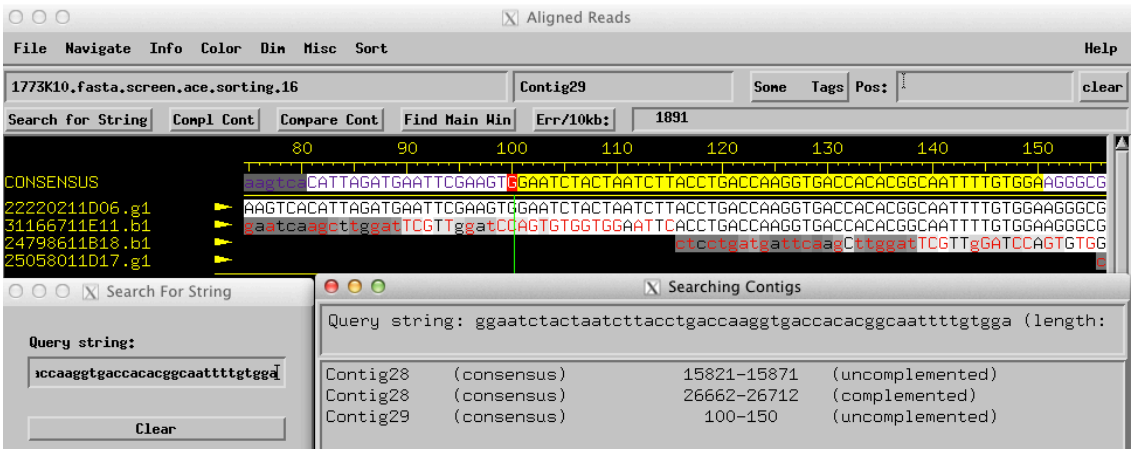


Figure 58 Match between Contig29 and Contig28 at 26662-26712 is in the complemented orientation.

Because the reads in Contig29 were pulled out of Contig28 at ~15kb, we know that the reads in Contig29 should not go back to this region. Instead, we will join Contig29 with Contig28 at 26,662-26,712. However, the match is in the complemented orientation (which makes sense, since we are dealing with an inverted repeat). Hence we need to complement Contig29 first and then perform Search for String again.

Complement Contig29 and Search for String using the consensus position 250-300. There should again be three matches but Contig29 is now in the same orientation as the inverted repeat at ~26kb (Figure 59). Compare the two regions, examine the alignment and join the contigs to create Contig30. Save the assembly (`1773K10.fasta.screen.ace.sorting.17`).

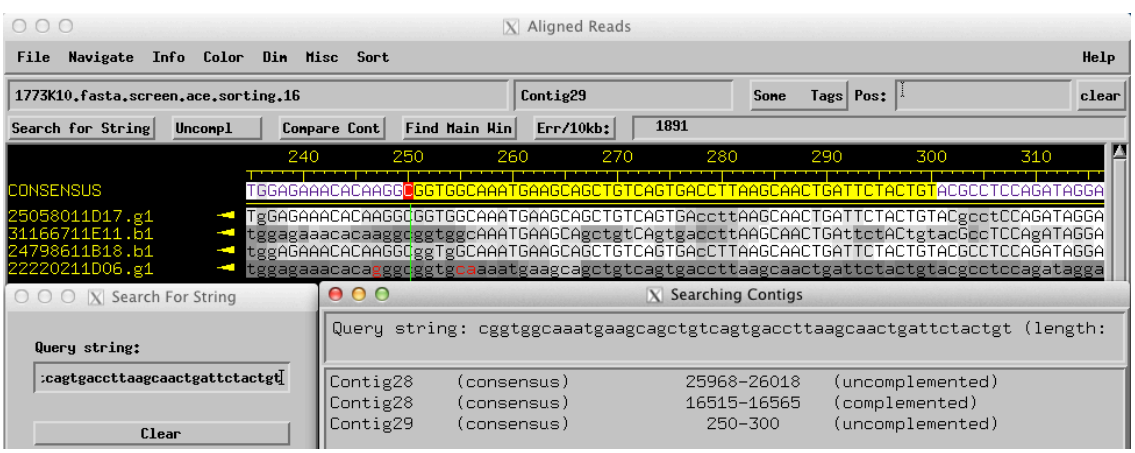


Figure 59 Contig29 is now in the same orientation as the inverted repeat at ~25kb in Contig28.

Open Assembly View and run `crossmatch`. Click on the other cluster of inconsistent mate pairs (Cluster 2) on Contig30. There should be two inconsistent mate pairs (from the subclones `34562511F01` and `34161211E18`). The two mate pairs in Cluster 2 are placed in two different copies of the inverted repeat. The mate pairs are inconsistent because they are pointing in the same direction. This means that both mate pairs should actually be placed within the same copy of the inverted repeat (Figure 60). Consequently, we need to determine the best copy of the inverted repeat in which to place both mate pairs.

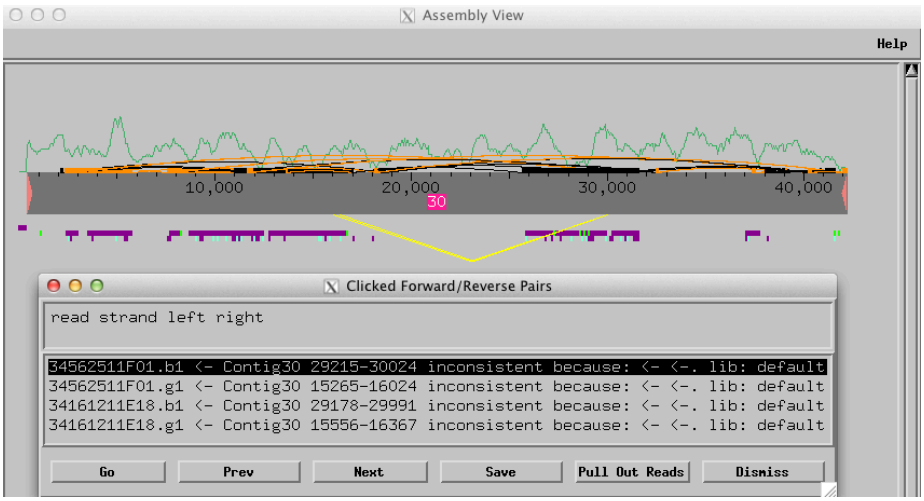


Figure 60 The inconsistent mate pairs in Cluster 2 should be placed in the same copy of the inverted repeat.

Click on the inconsistent mate pairs and pull all four reads out of Contig30. Save the assembly (`1773K10.fasta.screen.ace.sorting.18`). Then run Miniassembly, which will create two new contigs (Contig31 and Contig32, Figure 61). Save the assembly (`1773K10.fasta.screen.ace.sorting.19`).

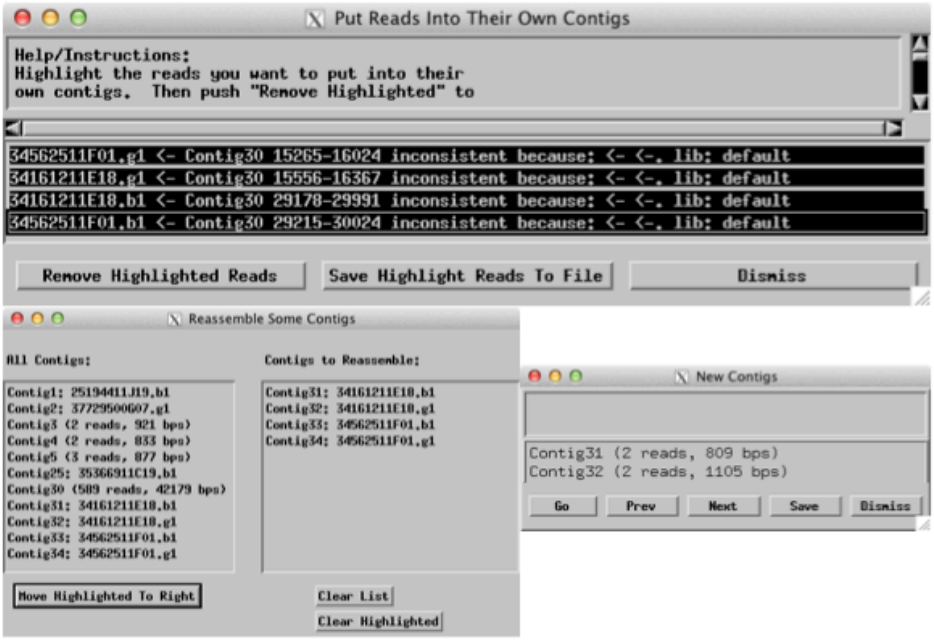


Figure 61 Pull out the inconsistent mate pairs and reassemble the four reads using Miniassembly.

Open Assembly View. By default, Assembly View will exclude contigs that have 10 or fewer reads or contigs that are less than 1kb in length. To see all the contigs in Assembly View, select "What to Show" and then click on "In/exclude Contigs". Change both the "exclude contig if this many reads or less" and "exclude contig if this many bases or less" fields to **0**. Click on "Apply and Restart Assembly View" (Figure 62).

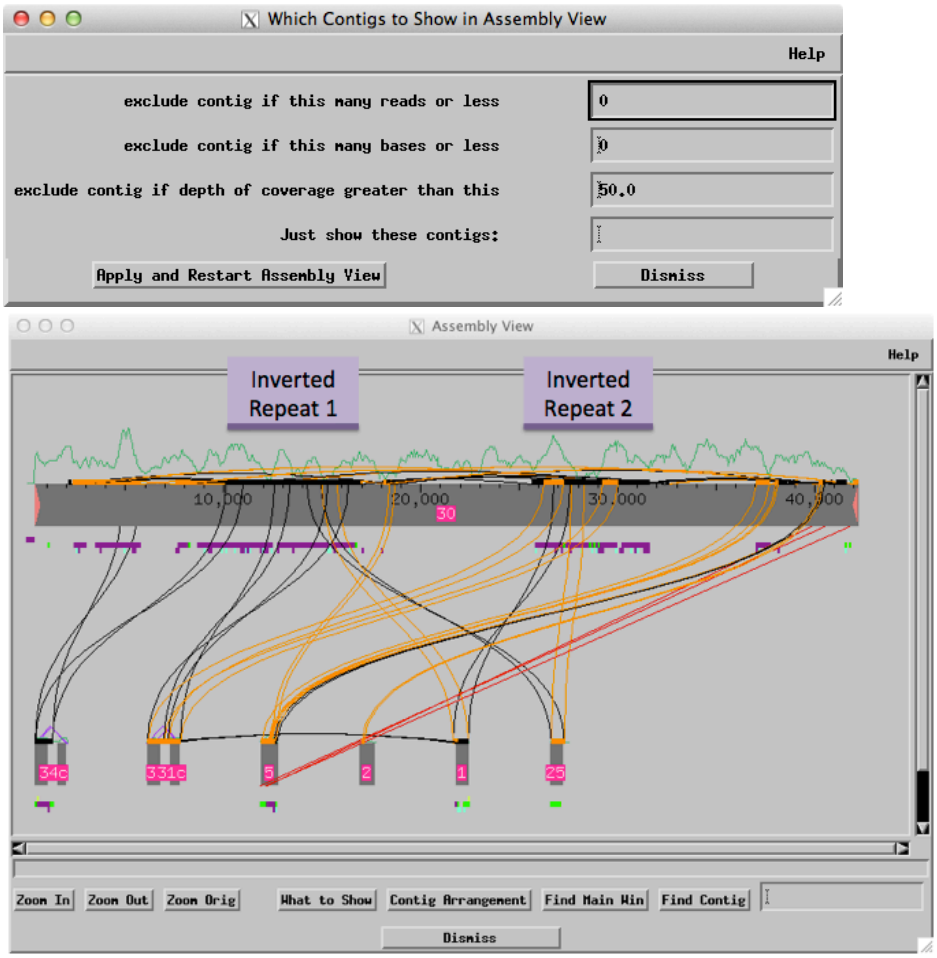


Figure 62 Change the "In/exclude Contigs" option to show the smaller contigs in Assembly View.

Run `crossmatch` so that we can determine the degree of sequence similarity among contigs 31, 32 and the two copies of the inverted repeat on Contig30. Note that, as expected, Assembly View has complemented Contig31 based on its relationship with Contig32. We will refer to the first inverted repeat copy at ~10-16kb of Contig30 as Inverted Repeat 1 and the other copy at ~25-31kb as Inverted Repeat 2 in the rest of this walkthrough.

Click on the two black inverted repeats shared among contigs 30, 31, 32 (Inverted Repeat 1, Figure 63). We see that Contig31 matches with 99.9% similarity to Contig30 at 12,552-13,339. While there are two matches between contigs 30 and 32, the first match (15,266-16,299) actually encompasses the entire region covered by the second match (15,270-16,299) with 0.1% lower similarity (98.9 versus 98.9%). Examination of the alignments reveals that all the discrepancies between Inverted Repeat 1 and contigs 31 and 32 are low quality.

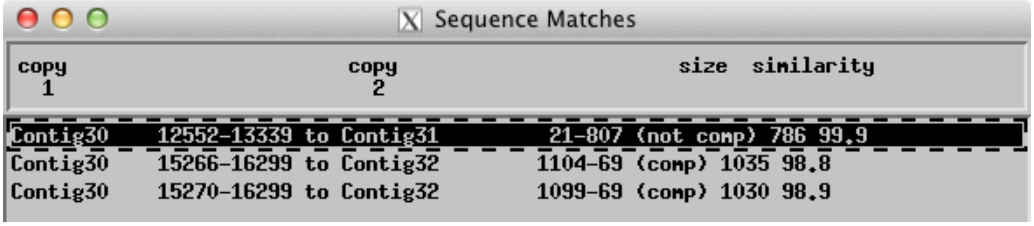


Figure 63 Matches between contigs 31 and 32 with the Inverted Repeat 1 in Contig30.

Next we will examine the direct repeat shared among contigs 30, 31, and 32 (Inverted Repeat 2, Figure 64). The match between Contig31 and Inverted Repeat 2 in Contig30 has the same length and sequence similarity as the match to Inverted Repeat 1.

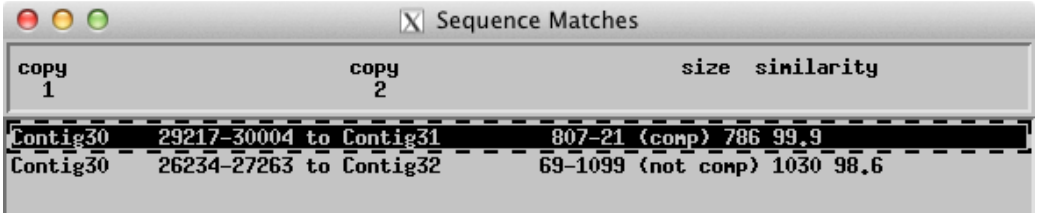


Figure 64 Matches between contigs 31 and 32 with Inverted Repeat 2 in Contig30.

However, the match between Contig32 and Inverted Repeat 2 in Contig30 has lower sequence similarity than the corresponding 1,030 bp match in Inverted Repeat 1 (98.6% versus 98.9%). Examination of the alignment between these two regions reveals multiple high quality discrepancies in the alignment (e.g. at position 26,884 in Contig 30, Figure 65). Based on this analysis, we will join contigs 31 and 32 with Inverted Repeat 1 in Contig30.

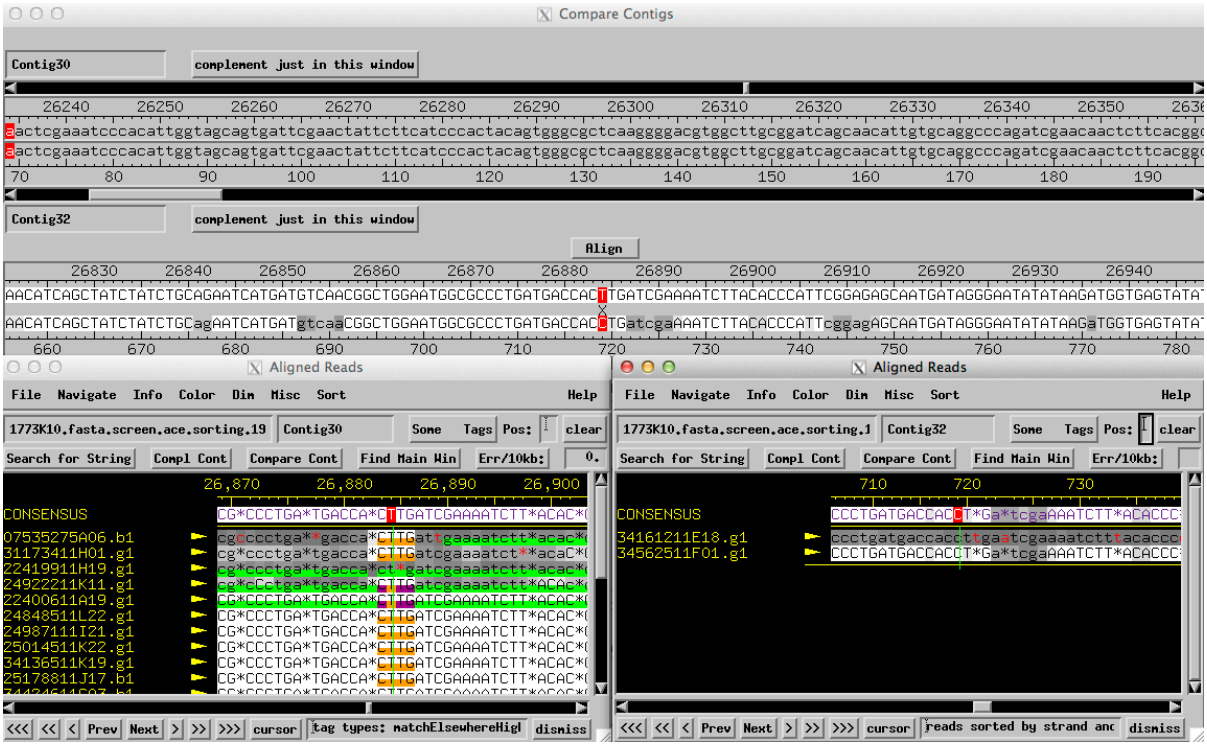


Figure 65 A high quality discrepancy between Contig32 and Inverted Repeat 2 in Contig30.

Before we can join these contigs together, we need to change the orientations of contigs 31 and 32 so that they are in the same orientation as Inverted Repeat 1 in Contig30. This means we need to complement Contig32.

In Assembly View, go to "Contig Arrangement," then select "Reorient Contigs". Select "32-31c" under the "Select a scaffold" section; select the option "Flip scaffold, and then do above." Click on "Apply and Restart Assembly View" (Figure 66). Save the new assembly (`1773K10.fasta.screen.ace.sorting.20`) and then run `crossmatch`. Both contigs 31 and 32 should now be in the same orientation as Inverted Repeat 1.

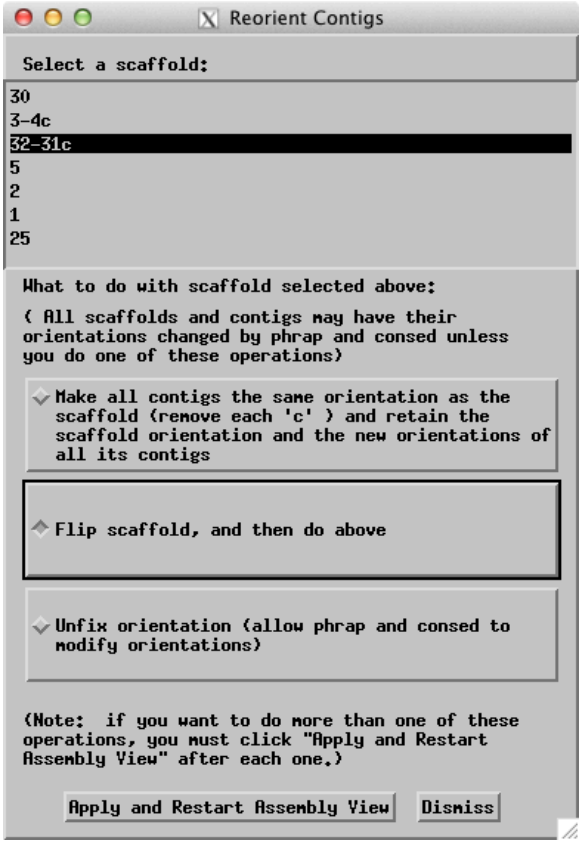


Figure 66 Flip the scaffold 32-31c so that it is in the same orientation as Inverted Repeat 1 in Contig30.

Click on the direct repeats shared between Contig31 and Inverted Repeat 1 in Contig30. Select the match in the Sequence Matches window and click on "Show Alignment". Examine the alignment and then join the two contigs together. Save the new assembly (`1773K10.fasta.screen.ace.sorting.21`).

Open Assembly View and run `crossmatch`. Repeat the same procedure to join Contig32 with Inverted Repeat 1 in Contig30 (using the longer 98.8% match). Save the assembly (`1773K10.fasta.screen.ace.sorting.22`). The inconsistent mate pairs in Cluster 2 have now all been resolved and the new main contig is called Contig34 (Figure 67).

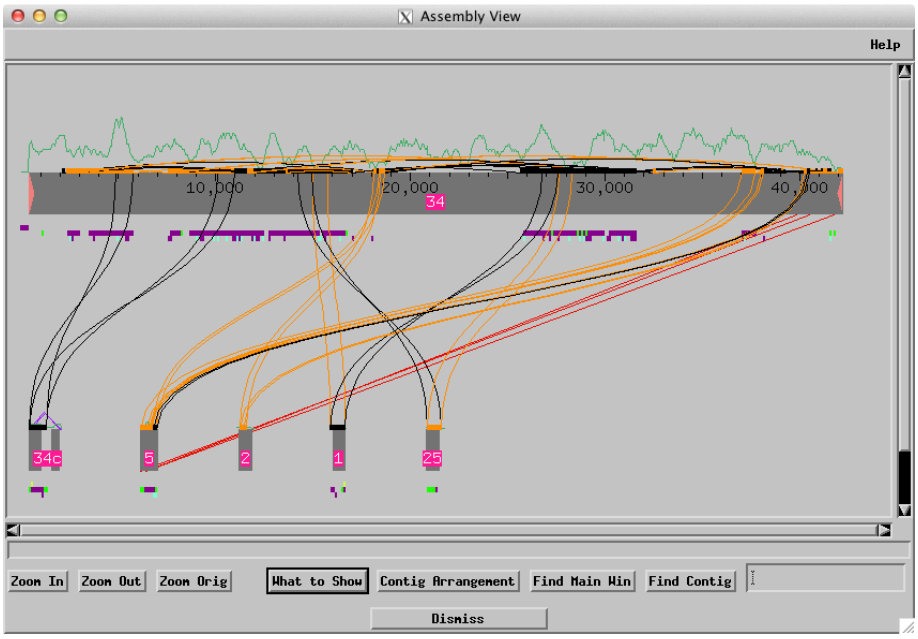


Figure 67 Assembly View after joining contigs 31 and 32 with Inverted Repeat 1 in Contig30.

Incorporate the Smaller Contigs into the Assembly

Open Assembly View and run `crossmatch`. Since the entire Contig25 (created by Miniassembly earlier on page 31) matches with Contig34, we will try to join this contig with Contig34. Open the Aligned Reads window for Contig25 and perform a Search for String using the high quality data at the beginning of the read from 50-100. There should be three matches (Figure 68).

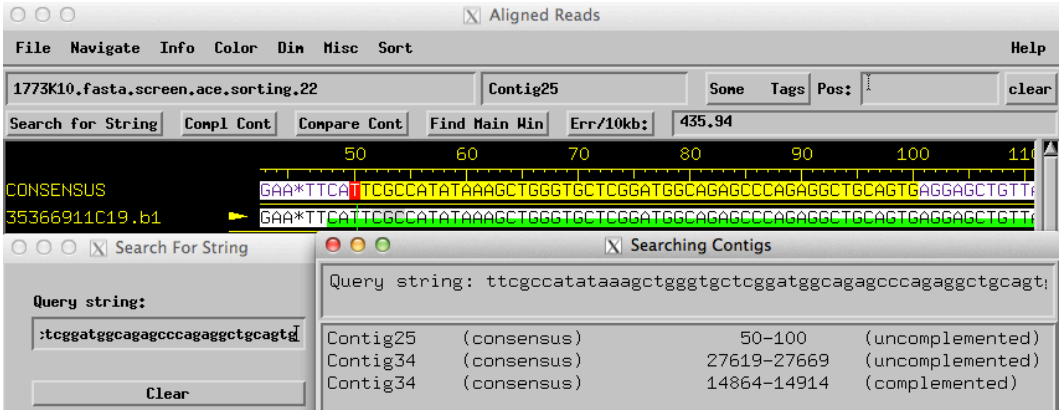


Figure 68 Two matches between the beginning of Contig25 and Contig34.

We will use the mate pair information to help us determine which of the two matches in Contig34 should be used to join Contig25 with Contig34. Highlight the read name `35366911C19.b1` to copy it onto the clipboard. Go to the Consed Main Window and search for the mate pair using the "Find 1st read above starting with:" field as we have previously described. The mate pair of this read (`35366911C19.g1`) is placed at Contig34 from 10,951-11,892 (Figure 69). Hence the match at 14,864-14,914 on Contig34 is a better candidate for the join. However, we need to complement Contig25 first before we can join the two contigs together.

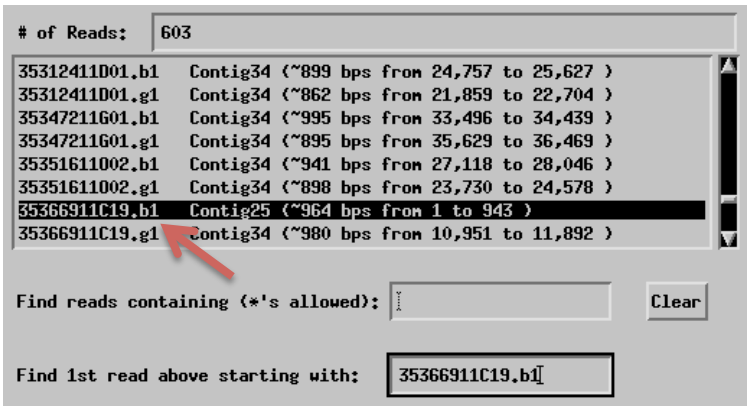


Figure 69 The Mate pair of `35366911C19.b1` is placed in Contig34 from 10,951 to 11,892.

Complement Contig25 and Search for String using the sequence from 844 to 894 (Figure 70). Compare Contig25 with the match at 14,864 to 14,914 on Contig34. Align the two regions and examine the alignment. Note that the high quality discrepancies at the end of the alignment are caused by unclipped vector at the end of Contig25 and can be ignored (Figure 71). Join the contigs to create Contig35. Save the assembly (`1773K10.fasta.screen.ace.sorting.23`).

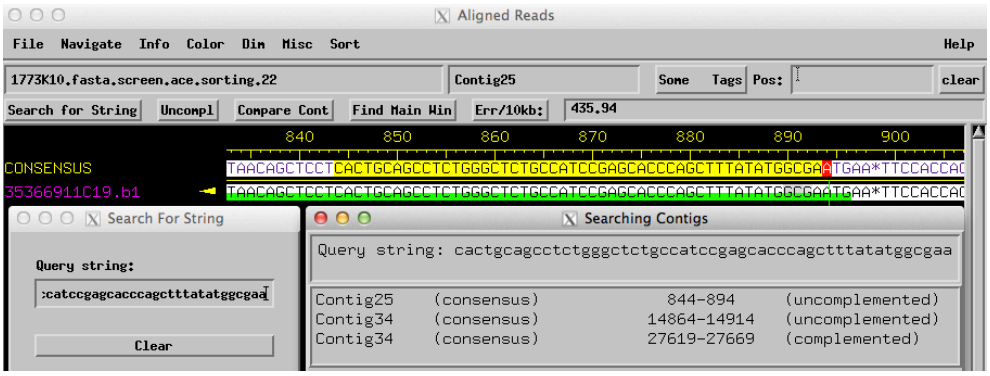


Figure 70 Search for String with the complemented Contig25 results in two matches in Contig34.

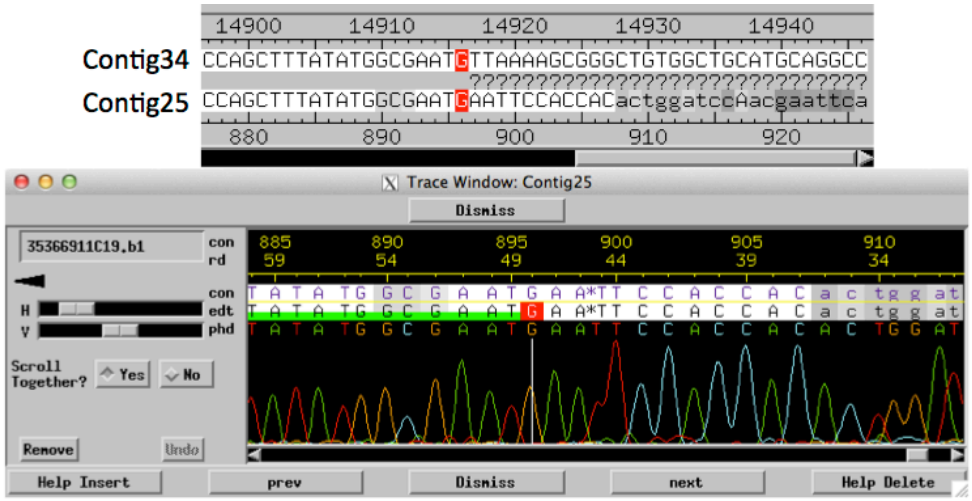


Figure 71 Unclipped vector sequence in Contig25 causes the high quality discrepancies at the end of the alignment between Contig34 (top) and Contig25 (bottom).

Open Assembly View and run `crossmatch`. The last issue we need to address is Contig5. The inconsistent forward reverse mate pairs suggest that Contig5 should be incorporated into the end of Contig35. Click on the large orange direct repeat at the beginning of Contig5. The `crossmatch` result shows that most of Contig5 (677bp) matches very well (99.7% similarity) to Contig35 at around 37kb (Figure 72).

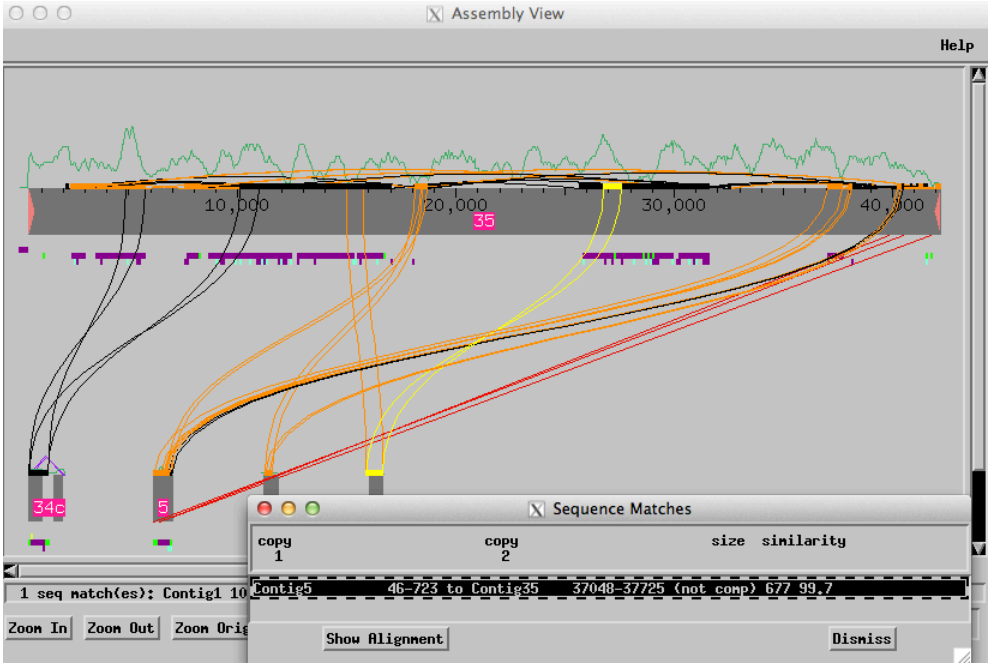


Figure 72 Sequence match between Contig5 and Contig35.

Select the sequence match and then click on the "Show Alignment" button so that we can determine why `phrap` decided not to incorporate Contig5 into Contig35. The alignment between the two contigs looks good until it reaches base 716 on Contig 5 (Figure 73). Place your cursor on the discrepant base at position 716 of Contig5 in the Compare Contigs window and click on "Scroll Both Aligned Reads Window". This will bring up the Aligned Reads window for both Contig5 and Contig35 at this discrepant position.

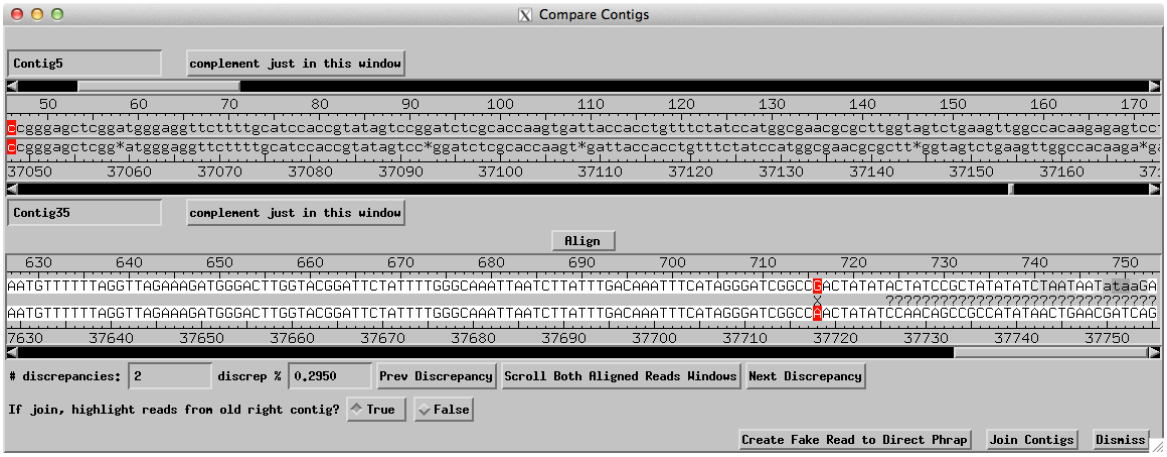


Figure 73 Discrepancies between Contig5 and Contig35 that kept the two contigs apart.

Open the three traces in Contig5 by middle clicking on each read at position 716 (Figure 74). We find that there is an unusually high G peak in all three reads at this position.

Figure 74 Tall G peak in all three reads at position 716 in Contig5.

Increase the vertical scale to maximum and look underneath the "G" peak. Notice that there is a small "A" peak in all three cases, which would be consistent with the A seen in Contig35 in the corresponding aligned position. Perform a Search for String using the region from 710 to 730 in Contig5. This should bring up three matches (Figure 75).

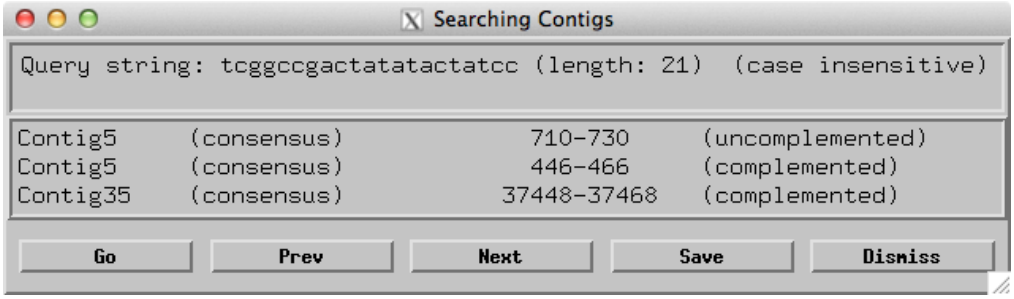


Figure 75 Search for String using 710-730 in Contig5 shows a complemented match to an earlier part (446-466) of Contig5.

Notice the complemented matches (710-730 with 446-466) on Contig5, which suggests that there is a small inverted repeat in the sequence. However, this small inverted repeat is not real and is caused by a phenomenon called "BigDye fold-back" that is often characterized by the anomalous G peaks shown above. The next step is to change all the bases in the fold-back region (starting from position 716 to the end of the read) to N's.

In the Trace Window, put the cursor on the edit (edt) line of the top trace. Middle click and select "Change to n's to right". Right click on the tag we have just created in the Trace Window and a dialogue box for the edit tag will appear. Enter the comment "Data changed to N because of BigDye fold-back" in the comment text box (Figure 76). Then click on "Save Changes." Repeat this process for the other two traces.

Figure 76 Change the BigDye fold-back portion of the trace (`25128611P05.b1`) to N's.

Once we have changed the consensus beyond position 716 in Contig5 to N's, click on "Align" again in the Compare Contigs window and join Contig5 with Contig35. This will create Contig36. Save the assembly (`1773K10.fasta.screen.ace.sorting.24`).

Open Assembly View to verify that the inverted repeat and duplication have been resolved successfully (Figure 77).

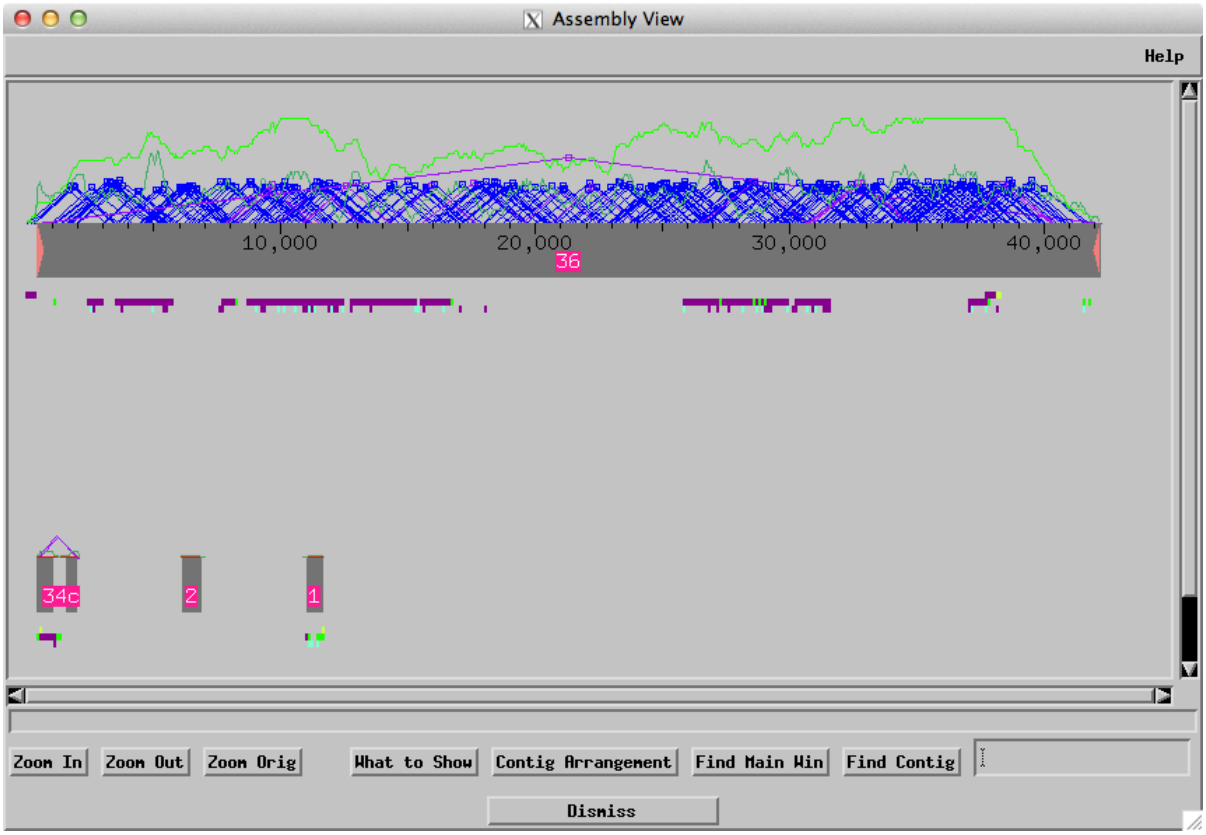


Figure 77 Final Assembly View with the duplication and large inverted repeat successfully resolved.

Verify the final assembly with restriction digests

We should also use the restriction digests data to confirm the veracity of the final assembly. Go back to the Consed Main Window and click on the "Digests" button. Recall from our earlier analysis that the left end of the fosmid insert is at position 389 of Contig36 (see page 8 for details). Consequently, we should run the *in-silico* digest with "Just Part of Clone" for Contig36 from 389 to the end of the contig at 42,179 (Figure 78).

Since all four *in-silico* digests match their corresponding real digests, we are confident that the misassembly has been successfully resolved (Figure 79).

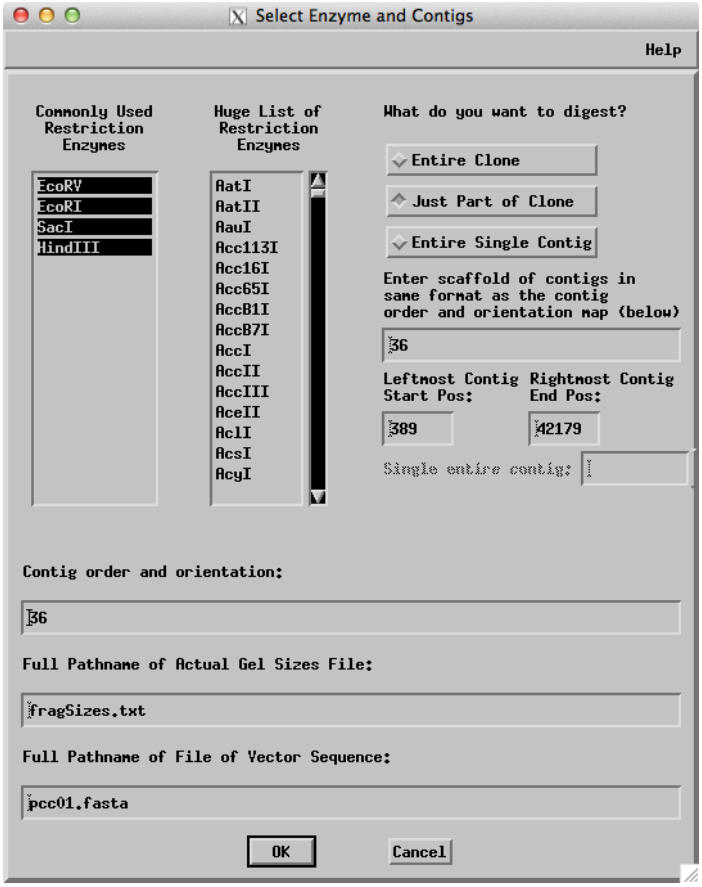
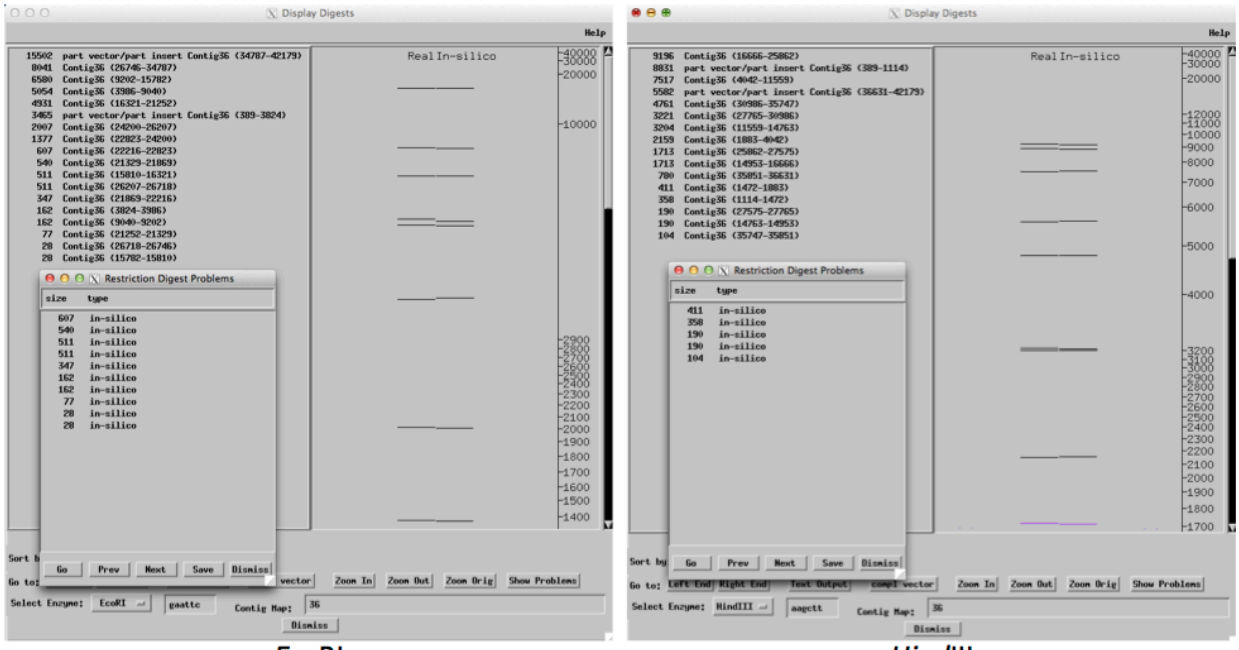


Figure 78 Tell consed to digest Contig36 from 389 to 42,179 to verify the final assembly.



*Eco*RI *Hind*III

Figure 79 *Eco*RI and *Hind*III in-silico digests match the real digests, indicating that the misassembly has been resolved successfully. (*Eco*RV and *Sac*I digests were also consistent, not shown).