CSM::Consed::Consed - A perl module for handling phred-phrap-Consed datafiles

Chad S. Matsalla

Agriculture & Agri-Food Canada, Saskatoon Research Centre

25 June 2001

A. Rationale and requirements

The Saskatoon Research Centre does extensive, dual-pass EST sequencing with ABI3700 sequencers. Given a bunch of folders some mechanism was needed to:

1. Align bunches of sequences
2. Classify contigs of aligned sequences based on how many reads are in each aligned contig
    a) we want there to be two reads in each contig: the forward read and the reverse read
    b) contigs containing more then two reads are either ignored or people will manually perform some action on them
    c) singletons and singlets need to be delt with separately.
3. Name contigs containing two reads where possible (see below for an explanation)
4. Perform quality trimming based on the consensus sequence of any given contig by some arbitrary quality trimming mechanism
5. Provide mechanisms for visualisation of the sequence alignments
6. Provide mechanisms for staff to manually change quality trimming and alignment decisions
7. Provide a persistent data structure for contig data so mechanisms can go in and asynchronously harvest sequences matching arbitrary criteria

Strategy:

Our staff is accustomed to using Consed to work with sequence assemblies. Would there be a way for them to continue to use Consed so I don't have to develop a whole new way for them to do things?

Strategies organised by requirements from section A (above)

1. I decided to use phred to call bases, phrap to assemble the sequences, consed for visualising the assemblies, and a custom program to parse the ace- family of files. This way changes that the staff make to the project in Consed (example: dissolve contigs) can be easily reflected when the acefile is parsed out.

2. If a contig contains two reads the names of the reads will be examined. If the names of the reads are the same except for an arbitrary forward and reverse designator then the contigs will be named and the consensus sequence will be used to represent that EST. for

example: Contig 1 contains two reads: chad408R and chad408F. This designator has not been the same over the duration of this project so a mechanism must be provided to change it from, for example, _F and _R to _f and _r to _Forward and _Reverse.

3. I chose to build a sliding window-average type of quality determination. This is, IMHO, the weakest part of the module. Can somebody help?

4. Staff can use Consed to see the decisions that phrap made based on the chromats. I wrote a GUI for the staff to browse the additional decisions that CSM::Consed::Consed made based on the qualities of the contigs. Despite the fact that I feel my quality trimming mechanisms are highly inefficient and weak at best the decisions that it makes quite accurately reflect the decisions that the staff would have made had they done the trimming manually so there you are.

5. I wrote a GUI for browsing the CSM::Consed::Consed object in Perl/TK.

6. I decided to use Ilya's freeze-thaw mechanism for serializing the CSM::Consed::Consed object. The collection robot simply recursively searches directories, thaws the frozen files, and inserts the {sequence_trimmed} parts of the object. This pleases the staff because what they see in the {sequence_trimmed} part of the GUI is exactly what goes into the databases.

C. Contig Classes

Each contig parsed out of the ace- family of files is assigned to a "class" as follows:

1. singlet

A "contig" of one. To quote the Consed documentation: "If a read does not have a significant match (with Smith-Waterman score exceeding minscore) to any other read, that read is not included in the ace file. Instead, that read is put in the '.singlets' file. That read will not appear in Consed."

- the singlets file is parsed into a CSM::Consed::Consed object by the subroutine $o_consed->set_singlets();

2. singleton

A contig of one. To quote the Consed documentation: "If a read does have a significant match to any other read, then it will appear in the ace file and be shown by consed. [...] Such a read would end up in a contig all of its own."

3. doublet

A contig of two is a doublet when the following applies:

a) when the forward_designator is subtracted from the end of one read and the reverse_designator is subtracted from the end of the other read the names of the reads are the same. forward_designator and reverse_designator are 'set'able on instantiation of the CSM::Consed::Consed object and are used by the subroutine set_doublets()

b) example: Contig1 contains two reads: Chad408_R and Chad408_F. The forward designator was set to "_F" and the reverse designator was set to "_R". Thus, the {name} of Contig1 will be set to "Chad408" and the class of Contig1 will be set to "doublet".

c) this is of especial interest to me, doing dual-pass EST sequencing. I want a lot of doublets because they are the closest to what a want: a metric bunchload of doublets of aligned reads.

4. pair

A contig of two that is not a doublet. That is to say:

a) when the forward_designator was subtracted from the end of one read and the reverse_designator was subtracted from the end of the other read the names of the reads were not the same.

b) example: Contig4 contains two reads: Chad455_R and Chad233_F. These are of no interest to me and represent alignment problems (see 3(c), immediately above)

5. multiplet

A contig containing more then two reads. useless to me and probably requires manual manipulation by staff.

D. The internal structure of a CSM::Consed::Consed object

This is an example of a CSM::Consed::Consed object. It is broken down and explained in section E.

```
CSM::Consed::Consed=HASH(0x80cd538)
    'fh' => FileHandle=GLOB(0x80d1bdc)
        -> *Symbol::GEN0
                        FileHandle({*Symbol::GEN0}) => fileno(3)
    'filename' => 't/acefile.ace.1'
    'forward_designator' => 'R'
    'o_trim' => CSM::Consed::Trim=HASH(0x80d1d14)
        'f_designator' => 'F'
        'r_designator' => 'R'
    'path' => 't/'
    'reverse_designator' => 'F'
    'verbose' => 0
    'contigs' => HASH(0x8256698)
    1 => HASH(0x8180704)
        'bottom_complement' => 'U'
        'bottom_name' => 'ML2749F'
        'bottom_phreds' => undef
        'bottom_start' => 21
        'class' => 'doublet'
        'consensus' => 'TTTTTTTTTTTTTT...'
        'contig_direction' => 'U'
        'end_point' => 469
        'member_array' => ARRAY(0x817b394)
                0  'ML2749R'
                1  'ML2749F'
        'name' => 'ML2749'
        'num_members' => 2
```

'quality' => '44 56 56 56 56 56 56 56...'
'sequence_trimmed' => 'TTTTTTTTTTTTTT...'
'start_point' => 0
'top_complement' => 'C'
'top_name' => 'ML2749R'
'top_phreds' => undef
'top_start' => '-243'
2 => HASH(0x822ba44)
...

E. An explanation of the different parts of a CSM::Consed::Consed object

First, a quick explanation of the terms "bottom" and "top" is necessary. CSM::Consed::Consed was designed to be sensitive to contigs containing two reads and to treat everything else with some amount of indifference. "top" refers to the first read in a contig and "bottom" refers to the second read in a contig. This is basically irrelevant for singlets, singletons, pairs, and mutiplets.

CSM::Consed::Consed=HASH(0x80cd538)
This is a CSM::Consed::Consed object.
'fh' => FileHandle=GLOB(0x80d1bdc)
-> *Symbol::GEN0
FileHandle({*Symbol::GEN0}) => fileno(3)
A FileHandle object for taking in the acefile.
'filename' => 't/acefile.ace.1'
The path to the acefile.
'forward_designator' => 'R'
The forward_designator that will be used to try to name the doublets.
'o_trim' => CSM::Consed::Trim=HASH(0x80d1d14)
'f_designator' => 'F'
'r_designator' => 'R'
A CSM::Consed::Trim object that will be used to trim sequences based on quality.
'path' => 't/'
The path to the acefile relative to the current working directory.
'reverse_designator' => 'F'
The reverse_designator that will be used to try to name the doublets.
'verbose' => 0
The verbosity level of the output from CSM::Consed::Consed
'contigs' => HASH(0x8256698)
A hash representing the contigs in the ace- family of files.
1 => HASH(0x8180704)
1 is the key for this example contig in the CSM::Consed::Consed object
'bottom_complement' => 'U'

Defined by consed. Is the bottom strand in this contig complemented or not? this affects how CSM::Consed::Consed determines the final quality values for certain parts of the consensus sequence. This is really only relevant for contigs of class "doublet". I don't care about much else.

'bottom_name' => 'ML2749F'

The name of the second read in this contig.

'bottom_phreds' => undef

A placeholder for the phred values of the bottom reads. Used by CSM::Consed::Consed to determine the quality in certain parts of the consensus sequence and then set as undef to speed up the FreezeThaw process.

'bottom_start' => 21

Defined by consed. This read starts 21 bases into the consensus sequence.

'class' => 'doublet'

The class of this contig. See above for an explanation of the concept of contig classes.

'consensus' => 'TTTTTTTTTTTTTT...'

Defined by consed. The consensus sequence of this contig. In this example the sequence is truncated.

'contig_direction' => 'U'

Defined by consed. Is this contig reversed and complemented?

'end_point' => 469

Defined by consed. This read ends 469 bases into the consensus sequence.

'member_array' => ARRAY(0x817b394)

0 'ML2749R'

1 'ML2749F'

Defined by consed. The names of the reads in this contig.

'name' => 'ML2749'

Defined by CSM::Consed::Consed according to the strategy describes elsewhere in this document. set_doublets() does this setting for contigs and set_singlets() does it for singlets.

'num_members' => 2

Defined by CSM::Consed::Consed. The number of reads in this contig.

'quality' => '44 56 56 56 56 56 56 56...'

Defined by consed. The quality for the consensus sequence of this contig.

'sequence_trimmed' => 'TTTTTTTTTTTTTT...'

Defined by CSM::Consed::Consed in conjunction with CSM::Consed::Trim. This is the final trimmed sequence. The contig harvester will come in and integrate *this* sequence into our sequence databases.

'start_point' => 0

Defined by consed. The start point of this consensus sequence.

'top_complement' => 'C'

Defined by consed. Is the top read in this contig reversed and complemented?

'top_name' => 'ML2749R'

Defined by CSM::Consed::Consed. The name of the first read in this contig.

'top_phreds' => undef

A placeholder for the phred values of the top reads. Used by CSM::Consed::Consed to determine the quality in certain parts of the consensus sequence and then set as undef to speed up the FreezeThaw process.

'top_start' => '-243'

Defined by consed. This read starts this many bases into the consensus sequence.

F. Documentation for subroutines

Important note and source of much confusion for outlanders:
- keyname is a key for the above hash. For singletons, doublets, pairs, and multiplets this would be "ContigN" where N ranges from 1 to the number of contigs in the acefile. This value is assigned by consed.
- {name} is a value assigned to a contig by CSM::Consed::Consed based on the names of the reads in that contig. it is not a key.
- in the code the variable $contig is normally used to refer to the key and the name is usually found as $self->$contigs{$contig}{name}.

sub new($filename) {

Description: Instantiate a new CSM::Consed::Consed object.

Requires: A filename of an acefile. If a full path is not specified "./" is prepended to the filename and used from instantiation until destruction.

Returns: A reference to a CSM::Consed::Consed object

Notes:

sub set_verbose($scalar) {

Description: Set the verbosity level for debugging messages. On instantiation of the CSM::Consed::Consed object the verbosity level is set to 0 (quiet).

The verbosity levels are:
0 - quiet
1 - noisy
2 - noisier
3 - annoyingly noisy

Requires: An integer from 0 to 3.

Returns: 1 or 0

Notes: The different verbosity levels are not implemented the way I would like. At this point messages are either on or off. Help?

sub get_filename() {

Description: Returns the name of the acefile being used by the object

Requires: Nothing
Returns: A scalar containing the name of a file

sub get_contigs() {
Description: Return the keys to the CSM::Consed::Consed object.
Requires: Nothing.
Returns: An array containing the keynames in the CSM::Consed::Consed object.
Notes: This would normally be used to get the keynames for some sort of iterator. These keys are worthless in general day-to-day use because in the Consed acefile they are simply Contig1, Contig2, ...

sub get_quality_array($contig_keyname) {
Description: Returns the quality for the consensus sequence for the given contig as an array. See get_quality_scalar to get this as a scalar.
Requires: The keyname of a contig. Note: This is a **keyname**. The key would normally come from get_contigs.
Returns: An array containing the quality for the consensus sequence with the given keyname.
Notes: Returns an array, not a reference. Is this a bug? <thinking> No.

sub get_quality_scalar($contig_keyname) {
Description: Returns the quality for the consensus sequence for the given contig as a scalar. See get_quality_array to get this as an array.
Requires: The keyname of a contig. Note this is a _keyname_. The key would normally come from get_contigs.

sub freeze_hash() {
Description: Use Ilya's FreezeThaw module to create a persistent data object for this CSM::Consed::Consed data structure. In the case of AAFC, we use CSM::Consed::Consed to pre-process bunches of sequences, freeze the structures, and send in a harvesting robot later to do database stuff.
Requires: nothing
Returns: 0 or 1;
Notes: This probably requires more checking to see if the print FREEZE was successful. Fix this.

sub get_members($contig_keyname) {
Description: Return the names of the reads in this contig.
Requires: The keyname of a contig. Note this is a **keyname**. The keyname would normally come from get_contigs.
Returns: An array containing the names of the reads in this contig.

sub get_members_by_name($some_arbitrary_name) {
Description: Return the names of the reads in a contig. This is the name given to $contig{key} based on what is in the contig. This is different from the keys retrieved through get_contigs().

Requires: The name of a contig. **Not** a key, but a name.
Returns: An array containing the names of the reads in the contig with this name.
Notes: highly inefficient. use some other method if possible.

sub get_contig_number_by_name($some_arbitrary_name) {
Description: Returns the key for a given contig with this name.
Requires: The name of a contig (not the key!)
Returns: The key of a contig.
Notes: This fails if more then one contig has the same name. Only the first contig in key order will be returned. This sub is rarely used so this was never fixed. It should probably be fixed to resemble get_members_by_name, I would think.

sub get_sequence($contig_keyname) {
Description: Returns the consensus sequence for a given contig.
Requires: The keyname of a contig. Note this is a **key**. The key would normally come from get_contigs.
Returns: a scalar containing a sequence

sub set_final_sequence($some_sequence) {
Description: Provides a manual way to set the sequence for a given key in the contig hash. Rarely used.
Requires: The keyname of a contig.
Returns: 1 or 0.

sub _read_file() {
Description: An internal subroutine used to read in an acefile and parse it into a CSM::Consed::Consed object
Requires: Nothing. uses the filename from the new() subroutine
Returns: 0 or 1.
Notes: Refer to internal documentation for procedures used in this subroutine. A little complicated.

sub set_reverse_designator($some_string) {
Description: Set the designator for the reverse read of contigs in this CSM::Consed::Consed object
Requires: a scalar containing a string that will be used in naming the contigs
Returns: The value of $o_consed->{reverse_designator} so you can check to see that it was set properly

sub set_forward_designator($some_string) {
Description: Set the designator for the forward read of contigs in this CSM::Consed::Consed object
Requires: a scalar containing a string that will be used in naming the contigs
Returns: The value of $o_consed->{forward_designator} so you can check to see that it was set properly

sub set_designator_ignore_case {
Description: Deprecated
Requires: Deprecated
Returns: Deprecated
Notes: Deprecated

sub set_trim_points_singlets_and_singletons {
Description: set the trim points for singlets and singletons based on quality. Uses the CSM::Consed::Trim object.
Requires: nothing. acts on all contigs of class /singlet|singleton/
Returns: nothing.
Notes: how can I throw errors out of this process?

sub set_trim_points_doublets() {
Description: set the trim points for doublets based on quality. Uses the CSM::Consed::Trim object.
Requires: nothing
Returns: nothing
Notes: how can I throw errors out of this process? Control over things is completely passed to the CSM::Consed::Trim object.

sub get_trimmed_sequence_by_name($name) {
Description: Returns the trimmed_sequence of a contig with {name} eq $name
Requires: the {name} of a contig
Returns: the trimmed_sequence
Notes: I am really not sure why this sub is here. Deprecated, I am sure.

sub set_dash_present_in_sequence_name("yes") {
Description: Deprecated. Part of an uncompleted thought.
Requires: "yes" to set {dash_present_in_sequence_name} to 1
Returns: nothing
Notes: What does this do?

sub set_doublets() {
Description: Find pairs that have similar names and mark them as doublets and set the {name}.
Requires: Nothing.
Returns: 0 or 1.
Notes: A complicated subroutine that iterates over the CSM::Consed::Consed looking for contigs of 2. If the forward and reverse designator are removed from each of the reads in {member_array} and the remaining reads are the same, {name} is set to that name and the contig's class is set as "doublet". If any of those cases fail the contig is marked as a "pair".

sub set_singlets($verbosely) {

Description: Read in a singlets file and place them into the CSM::Consed::Consed object
Requires: Nothing. If $verbosely has a value the singlets will be set verbosely.
Returns: Nothing.

sub get_singlets() {
Description: Return the keynames of the singlets.
Requires: Nothing.
Returns: Returns an array containing the keynames of all CSM::Consed::Consed sequences in the class "singlet"

sub set_quality_by_name($name,$quality) {
Description: Deprecated. Make the contig with {name} have {quality} $quality. Probably used for testing.
Requires: the {name} of a contig and $quality to put into{quality}
Returns: Nothing.

sub set_singlet_quality() {
Description: For each singlet, go to the appropriate file in phd_dir and read in the phred quality for that read and place it into {quality}
Requires: Well, a phd_dir containing appropriate phreds.
Returns: 0 or 1.
Notes: This is the next subroutine that will receive substantial revision in the next little while. It really should eval the creation of CSM::Phred objects and go from there.

sub set_contig_quality() {
Description: Deprecated.
Requires: Deprecated.
Returns: Deprecated.

sub get_multiplets() {
Description: Return the keynames of the multiplets.
Requires: Nothing.
Returns: Returns an array containing the keynames of all CSM::Consed::Consed sequences in the class "multiplet"

sub get_all_members() {
Description: Return a list of all of the read names in the CSM::Consed::Consed object.
Requires: Nothing.
Returns: An array containing all of the elements in all of the {member_array}'s
Notes: A possible alternative to the sum_lets()-style of statistical insurance

sub sum_lets($total_only) {

Description: Provide numbers for how many sequences were accounted for in the CSM::Consed::Consed object.
Requires: Nothing, but the optional argument will return the total only.
Returns: If a scalar is present, returns the total number of sequences accounted for in all classes. If no scalar passed then returns a string that looks like this:
Singt/singn/doub/pair/mult/total : 2,0,1(2),0(0),0(0),4
This example means the following:
There were 1 singlets.
There were 0 singletons.
There were 1 doublets for a total of 2 sequences in this class.
There were 0 pairs for a total of 0 sequences in this class.
There were 0 multiplets for a total of 0 sequences in this class.
There were a total of 4 sequences accounted for in the CSM::Consed::Consed object.

sub write_stats() {
Description: Write a file called "statistics" containing numbers similar to those provided in sum_lets()
Requires: Nothing.
Returns: Nothing. Write a file in $o_consed->{path} containing something like this:
0,0,50(100),0(0),0(0),100
Where the numbers provided are in the format described in the documentation for sum_lets().

sub get_singletons() {
Description: Return the keynames of the singletons.
Requires: Nothing.
Returns: Returns an array containing the keynames of all CSM::Consed::Consed sequences in the class "singleton"

sub get_pairs() {
Description: Return the keynames of the pairs.
Requires: Nothing.
Returns: Returns an array containing the keynames of all CSM::Consed::Consed sequences in the class "pair"

sub get_name($contig_keyname) {
Description: Return the {name} for $contig_keyname
Requires: A contig keyname
Returns: A string. ({name})

sub _get_contig_name($r_array_containing_reads) {
Description: The logic for the set_doublets subroutine
Requires: A reference to an array containing read names
Returns: The name for this contig

sub get_doublets() {
Description: Return the keynames of the doublets.
Requires: Nothing.
Returns: Returns an array containing the keynames of all CSM::Consed::Consed sequences in the class "doublet"

sub dump_hash() {
Description: Use dumpvar.pl to dump out the CSM::Consed::Consed object to STDOUT
Requires: Nothing
Returns: Nothing.
Notes: Really useful for debugging.

sub dump_hash_compact() {
Description: Dump out the CSM::Consed::Consed object in a compact way.
Requires: Nothing.
Returns: Nothing.
Notes: Cleaner then dumpValue(), dumpHash()

sub get_phreds() {
Description: For each doublet in the CSM::Consed::Consed hash, go and get the phreds for the top and bottom reads. Place them into {top_phreds} and {bottom_phreds}
Requires: Nothing.
Returns: Nothing.
Notes: Requires parse_phd() and reverse_and_complement(). I realize that it would be much more elegant to pull qualities as required but there were certain "features" in the acefile that required a bit more detailed work be done to get the qualities for certain parts of the consensus sequence. In order to make _sure_ that this was done properly I wrote things to do all steps and then I used dump_hash() and checked each one to ensure expected bahavior. I have never changed this, so there you are.

sub parse_phd($read_name) {
Description: Suck in the contents of a .phd file.
Requires: The name of a read.
Returns: A reference to an array containing the quality values for the read.
Notes: This is a significantly weak subroutine because it was always intended that these functions, along with the finctions provided by get_phreds() be put into the CSM::Phred module. This is done now but the CSM::Consed::Consed module has not be rewritten to reflect this change.

sub reverse_and_complement($r_array) {
Description: A stub for the recursive routine reverse_recurse()
Requires: A reference to an array of phred data.
Returns: A reference to a reversed and complemented array of phred data.

Notes:

sub reverse_recurse($r_source,$r_destination) {

Description: A recursive routine to reverse and complement an array of phred data.
Requires: A reference to a source array and a reverence to a destination array.
Returns: A reference to an array containing reversed phred data.
Notes: Recursion is kewl.

sub show_missing_sequence() {

Description: Used by set_trim_points_doublets() to fill in quality values where consed (phrap?) set them to 0 at the beginning and/or end of the consensus sequences.
Requires: Nothing.
Returns: Nothing.
Notes: Acts on doublets only. Really very somewhat quite ugly. A disgusting kludge. <insert pride here> It was written stepwise with no real plan because it was not really evident why consed (phrap?) was doing this.

sub show_doublets() {

Description: Deprecated
Requires: Deprecated
Returns: Deprecated
Notes: Deprecated

sub show_contigs_complete() {

Description: Deprecated
Requires: Deprecated
Returns: Deprecated
Notes: Deprecated

G. A Sample Script

This is taken from t/03test_project.t and cleaned up for readability. Note that most of the methods being used in this sample script are non-parameter requiring (global) methods.

```
#!/usr/bin/perl -w

use strict;
require 'dumpvar.pl';
use CSM::Consed::Consed;

my($o_consed,@singlets,@singletons,@pairs,@doublets,@multiplets,$invoker);

        # instantiate a new object
$o_consed = CSM::Consed::Consed-
>new("t/test_project/edit_dir/test_project.fasta.screen.ace.1");

        # set the verbosity to 0
my $verbosity = $o_consed->set_verbose(0);
```

```
        # set the singlets
$invoker = $o_consed->set_singlets("verbosely");

        # set the singlet quality
$o_consed->set_singlet_quality();

        # get a list of the keynames of the contigs
@singlets = $o_consed->get_singlets();

        # set the forward and reverse designators
$invoker = $o_consed->set_forward_designator("F");
$invoker = $o_consed->set_reverse_designator("R");

        # name the doublets and assign classes
$invoker = $o_consed->set_doublets();

        # retrieve a list of the doublets, pairs, singletons, and multiplets
@doublets = $o_consed->get_doublets();
@pairs = $o_consed->get_pairs();
@multiplets = $o_consed->get_multiplets();
@singletons = $o_consed->get_singletons();

        # make sure that the number of reads in the consed project matches the number of
        # reads that _should_ be in there
$invoker = $o_consed->count_sequences_with_grep();
my $total_grep_sequences = $invoker;

        # get some statistics from the CSM::Consed::Consed object
$invoker = $o_consed->sum_lets("total_only");
my $total_object_sequences = $invoker;
print("($total_object_sequences)\n");
print("Match?\n");
ok ($total_object_sequences == $total_grep_sequences);

        # print out some statistics
print("These are the statistics. Look right? ".$o_consed->sum_lets()."\n");

print("Dumping out the hash in a compact way...\n");
$o_consed->dump_hash_compact();

# print("Dumping out the hash in an ugly way...\n");
# $o_consed->dump_hash();
```